Эндрю Питоньяк
Перевод Заимских А. Н.

OpenOffice.org pro

автоматизация работы

ДМК Пресс
издательство

Москва, 2008

УДК 004.4
ББК 32.973.26018.2
П32

П32 Эндрю Питоньяк (Andrew Pitonyak)
OpenOffice.org pro. Автоматизация работы. М: ДМК Пресс, 2008.
512 с., ил. + CD
Перевод с английского — Заимских А. Н.

ISBN 978-5-94074-441-2

Эта книга — первое выходящее на русском языке издание по программированию в OpenOffice.org, в основе которого лежит статья Эндрю Питоньяка «Макросы».

Эндрю Питоньяк признанный мастер по написанию макросов. Его книги программисты разбирают на цитаты. Приведённые примеры с описанием позволяют сделать сложные вещи.

Теперь эта книга доступна и вам. На прилагаемом CD вы найдёте все листинги из этой книги, а так же OpenOffice.org pro, который совершенно легально можете установить у себя дома и на всех компьютерах в офисе.

УДК 004.4
ББК 32.973.26018.2

GNU FDL. Все права защищены. Любая часть этой книги МОЖЕТ быть воспроизведена в любой форме и любыми средствами в любых целях без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но, поскольку вероятность технических ошибок всё равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несёт ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-5-94074-441-2
© Текст, обложка «Инфра-Ресурс», 2008
© Издание, «ДМК Пресс», 2008

Краткое оглавление

Об этой книге....1

Программирование в OOo....9

Глава I. Начало: концепция....11
Глава II. Примеры....25
Глава III. Разные примеры....35
Глава IV. Макросы электронных таблиц Calc....133
Глава V. Макросы текстового документа (Writer)....167
Глава VI. Текстовые таблицы....241
Глава VII. Форматирование макроса....259
Глава VIII. Формы....275
Глава IX. Пример по инвестициям (Investment)....301
Глава X. Обработчики (Handlers) и перехватчики (Listeners)....305
Глава XI. Язык (Language)....319
Глава XII. Операторы и их старшинство Operators и Precedence....349
Глава XIII. Действия со строками....351
Глава XIV. Работа с числами....357
Глава XV. Работа с датами....359
Глава XVI. Работа с файлами....361
Глава XVII. Операторы в выражениях, операторы программы, функции....363
Глава XVIII. Другие языки....461

Приложения....465

Приложение 1. Дополнения переводчика....467
Приложение 2. Горячие клавиши....469
Приложение 3. Особенности сборки OpenOffice.org pro....479
Приложение 4. Алфавитный указатель (Index)....483

Подробное оглавление

От редакции....IX
От переводчика....IX

Об этой книге....1
Благодарности....3
Заявление....3
Предисловие автора....3
Контактная информация....4
Сокращения и обозначения....5
Доступные ссылки....6
Ссылки на сайты....7

Программирование в OOo....9
Глава I. Начало: концепция....11
I.1. Мой первый макрос: «Hello World»....11
I.2. Группировка текста программ....12
I.3. Отладка....12
I.4. Переменные, константы, строки и числовые типы....12
I.5. Обращение к объектам и создание объектов (Objects) в OpenOffice....13
I.6. Что же такое UNO?....15
I.7. Итоги....24
Глава II. Примеры....25
II.1. Отладка и проверка макросов....25
II.2. Средство X-Ray....27
II.3. Диспетчер OOo (Dispatch): использование универсального сетевого объекта Universal Network Objects (UNO)....28
II.4. Перехват выполнения команд меню с использованием Basic....31
Глава III. Разные примеры....35
III.1.1. Вывод текста в строку состояния....35
III.2. Вывод всех стилей текущего документа....35
III.3. Перебрать все открытые документы....36
III.4. Вывести шрифты и другую информацию об экране....36
III.5. Установка шрифта по умолчанию с использованием ConfigurationProvider....38
III.6. Печать текущего документа....38
III.7. Информация о конфигурации....40
III.8. Открыть и закрыть документы (и рабочий стол)....41
III.9. Загрузка / Вставка графики в документ....50
III.10. Установка полей....55
III.11. Установка размера бумаги....56
III.12. Вызов внешней программы (Internet Explorer) с помощью OLE....56
III.13. Использовать команду Shell для файлов, содержащих пробелы....57
III.14. Чтение и Запись числа в файл....57
III.15. Создать стиль числового формата....58
III.16. Посмотреть существующие числовые стили формата....59
III.17. Возвращает массив чисел Фибоначчи....60
III.18. Вставка текста в месте закладки (Bookmark)....61
III.19. Сохранение и экспорт документа....61

III.20. Поля пользователя....63
III.21. Информация о документе....63
III.22. Текстовые поля....64
III.23. Мастер-поля (Master)....65
III.24. Удаление текстовых полей....70
III.25. Вставка адреса URL в ячейку OOo Calc....70
III.26. Добавление текстового поля — формулы (SetExpression TextField)....71
III.27. Типы данных, определённые пользователем....72
III.28. Проверка орфографии, переносы и толковый словарь — тезаурус....72
III.29. Изменение указателя мыши....74
III.30. Установка фона страницы (Page Background)....75
III.31. Работа с буфером (clipboard)....76
III.32. Установка локали (языка)....81
III.33. Установки локали для выделенного текста....82
III.34. Автотекст (Auto Text)....84
III.35. Десятичные футы и дроби....86
III.36. Преобразовать число в слова....90
III.37. Отправка электронного письма....94
III.38. Библиотеки макросов....97
III.39. Установка размеров точечного рисунка (Bitmap)....101
III.40. Извлечение файла Zip....106
III.41. Выполнить макрос по строке с его именем....111
III.42. Использование «приложения по умолчанию» для открытия файла....113
III.43. Распечатка перечня шрифтов....113
III.44. Получить для документа: адрес URL, имя файла и папку....114
III.45. Получить и установить текущую папку (directory)....114
III.46. Запись файла....117
III.47. Логический разбор синтаксиса (Parsing) XML....117
III.48. Работа с датами Dates....121
III.49. Встроен ли OpenOffice в Веб-браузер?....123
III.50. Активизировать (поставить на первый план) новый документ....123
III.51. Каков тип документа (основываясь на адресе URL)....123
III.52. Соединиться с удаленным сервером OOo с использованием Basic....124
III.53. Панели инструментов....125
Глава IV. Макросы электронных таблиц Calc....133
IV.1. Является ли этот документ электронной таблицей?....133
IV.2. Вывести для ячейки электронной таблицы значение (value), строковое значение (string) и формулу (formula)....133
IV.3. Установить для ячейки электронной таблицы значение (value), формат (format), текстовое значение (string) и формулу (formula)....134
IV.4. Ссылка на ячейку в другом документе....134
IV.5. Очистка ячейки....134
IV.6. Выделенный (Selected) текст — что это?....135
IV.7. Простой пример обработки выделенных ячеек....136
IV.8. Получить активную ячейку игнорировать остальное....138
IV.9. Выделить (Select) ячейку....139
IV.10. Удобный для чтения (Human readable) адрес ячейки....140
IV.11. Вставить форматированную дату в ячейку....141

IV.12. Более короткий путь для этого....142
IV.13. Вывести выделенный интервал ячеек (selected range) в окно сообщения 142
IV.14. Заполнить выделенный интервал ячеек заданным текстом....143
IV.15. Некоторые данные и статистика о выделенном интервале ячеек....143
IV.16. (Именованный) Интервал ячеек базы данных....144
IV.17. Определить выбранные ячейки в качестве (именованного) интервала ячеек базы данных....145
IV.18. Удалить (именованный) интервал ячеек базы данных....146
IV.19. Границы таблицы....146
IV.20. Интервал ячеек для сортировки....147
IV.21. Вывести все данные столбца....149
IV.22. Использование методов объединения (Outline, Grouping)....149
IV.23. Защита данных....150
IV.24. Установка текстов верхнего и нижнего заголовков....150
IV.25. Копирование листа целиком в новый документ....151
IV.26. Выделить именованный интервал ячеек (named range)....152
IV.27. Выделить столбец целиком....154
IV.28. Выделить строку целиком....154
IV.29. Преобразовать данные из столбца определенного вида в строки....154
IV.30. Включить/Выключить автоматический пересчет....156
IV.31. Какие ячейки листа используются?....157
IV.32. Поиск в электронной таблице Calc....158
IV.33. Напечатать интервал ячеек электронной таблицы....161
IV.34. Объединена ли эта ячейка (с другими)?....162
IV.35. Написать свою функцию электронной таблицы Calc....162
Глава V. Макросы текстового документа (Writer) 167
V.1. Что такое выделенный текст?....167
V.2. Что такое текстовые курсоры (Text Cursors)?....169
V.3. Работа с текстом (Andrew's Selected Text Framework)....173
V.4. Замена выделенного пробела с использованием строк (Strings). Вставка специальных символов....192
V.5. Установка атрибутов текста....203
V.6. Вставить текст....204
V.7. Поля....205
V.8. Вставка новой страницы....206
V.9. Установить стиль страницы в документе....207
V.10. Включение и выключение верхних и нижних заголовков....208
V.11. Вставить OLE-объект....208
V.12. Установка стиля параграфа (Paragraph)....209
V.13. Создать свой собственный стиль....210
V.14. Поиск и замена....210
V.15. Изменение строчных букв на прописные и наоборот (Case) в словах....215
V.16. Перебор параграфов (поведение текстового курсора)....217
V.17. Где находится курсор дисплея (Display Cursor)?....228
V.18. Вставка индекса или оглавления (table of contents)....232
V.19. Вставка адреса URL в документ OOo Writer....233
V.20. Сортировка текста....233

V.21. Нумерация структур (Outline).....234
V.22. Вставить оглавление (table of contents =TOC) или другой индекс.....236
V.23. Текстовые секции (sections).....237
V.24. Сноски на странице (Footnotes) и сноски в конце текста (Endnotes).....239
Глава VI. Текстовые таблицы.....241
VI.1. Поиск текстовых таблиц.....241
VI.2. Перебор ячеек в текстовой таблице.....244
VI.3. Извлечение данных из простой текстовой таблицы.....250
VI.4. Курсоры таблицы и интервалы ячеек.....251
VI.5. Интервалы ячеек (Cell ranges).....251
VI.6. Данные диаграммы (Chart data).....252
VI.7. Ширина столбцов.....252
VI.8. Установка оптимальной ширины столбца.....253
VI.9. Насколько широка текстовая таблица?.....254
VI.10. Курсор в текстовой таблице.....254
VI.11. Создание текстовой таблицы.....257
VI.12. Таблица без рамок границы (borders).....258
Глава VII. Форматирование макроса.....259
VII.1. Утилиты для строк и массивов.....259
VII.2. Утилиты для поиска разделов/секций с кодами макросов.....266
VII.3. Основной модуль макроса.....268
Глава VIII. Формы.....275
VIII.1. Введение.....275
VIII.2. Диалоги.....276
VIII.3. Получение элементов управления (Controls).....280
VIII.4. Выбор файла с использованием диалога (File Dialog).....283
VIII.5. Центрировать диалог на экране.....284
VIII.6. Установить перехватчик события (event listener) для элемента управления.....284
VIII.7. Управление диалогом я не создал.....286
Глава IX. Пример по инвестициям (Investment).....301
IX.1. Внутренняя ставка возврата средств (Internal Rate of Return = IRR) 301
Глава X. Обработчики (Handlers) и перехватчики (Listeners).....305
X.1. xKeyHandler пример.....305
X.2. Перехватчик (Listener), описанный автором Paolo Mantovani.....308
X.3. Что произошло с моим перехватчиком ActiveSheet?.....317
Глава XI. Язык (Language).....319
XI.1. Комментарии.....319
XI.2. Переменные.....319
XI.3. Функции (Functions) и процедуры (SubProcedures).....332
XI.4. Управление последовательностью выполнения программы (Flow Control) 335
XI.5. Разное.....347
Глава XII. Операторы и их старшинство Operators и Precedence.....349
Глава XIII. Действия со строками.....351
XIII.1. Удалить символы из строки.....352
XIII.2. Удалить текст из строки.....353
XIII.3. Печать кодов ASCII символов строки.....355

XIII.4. Удалить все экземпляры заданной подстроки из исходной строки.....355
Глава XIV. Работа с числами.....357
Глава XV. Работа с датами.....359
Глава XVI. Работа с файлами.....361
Глава XVII. Операторы в выражениях, операторы программы, функции.....363
XVII.1. Оператор вычитания (-).....363
XVII.2. Оператор умножения (*).....363
XVII.3. Оператор сложения (+).....364
XVII.4. Оператор возведения в степень (^).....364
XVII.5. Оператор деления (/).....365
XVII.6. Оператор AND 365
XVII.7. Функция Abs.....366
XVII.8. Функция создания массива Array 367
XVII.9. Функция Asc.....368
XVII.10. Функция ATN (арктангенс) 369
XVII.11. Оператор Beep.....370
XVII.12. Функция Blue 370
XVII.13. Ключевое слово ByVal 371
XVII.14. Ключевое слово Call 371
XVII.15. Функция CBool 372
XVII.16. Функция CByte.....373
XVII.17. Функция CDate 374
XVII.18. Функция CDateFromIso 374
XVII.19. Функция CDateToIso 375
XVII.20. Функция CDbl 375
XVII.21. Оператор ChDir — нежелателен.....376
XVII.22. Оператор ChDrive — нежелателен.....377
XVII.23. Функция Choose 377
XVII.24. Функция Chr 378
XVII.25. Функция CInt 379
XVII.26. Функция CLng 380
XVII.27. Оператор Close 380
XVII.28. Оператор константы Const 381
XVII.29. Функция ConvertFromURL 382
XVII.30. Функция ConvertToURL 382
XVII.31. Функция косинуса Cos 383
XVII.32. Функция создания диалога CreateUnoDialog 384
XVII.33. Функция CreateUnoService 385
XVII.34. Функция CreateUnoStruct 386
XVII.35. Функция CSng 386
XVII.36. Функция CStr 387
XVII.37. Функций CurDir 388
XVII.38. Функций Date 389
XVII.39. Функция DateSerial 389
XVII.40. Функция DateValue 390
XVII.41. Функция Day.....391
XVII.42. Оператор объявления Declare 392
XVII.43. Оператор DefBool 393

XVII.44. Оператор DefDate 393
XVII.45. Оператор DefDbl 394
XVII.46. Оператор DefInt 394
XVII.47. DefLng Statement....395
XVII.48. Оператор DefObj 396
XVII.49. Оператор DefVar 396
XVII.50. Оператор Dim....397
XVII.51. Функция DimArray 398
XVII.52. Функция Dir....399
XVII.53. Операторы Do...Loop 401
XVII.54. Оператор End 402
XVII.55. Функция Environ 403
XVII.56. Функция EOF 403
XVII.57. Функция EqualUnoObjects 404
XVII.58. Оператор EQV 405
XVII.59. Функция Erl 406
XVII.60. Функция Err 406
XVII.61. Оператор Error не работает так, как описано....407
XVII.62. Функция Error 408
XVII.63. Оператор Exit 408
XVII.64. Функция Exp....410
XVII.65. Функция FileAttr 410
XVII.66. Оператор FileCopy 412
XVII.67. Функция FileDateTime 412
XVII.68. Функция FileExists 413
XVII.69. Функция FileLen 413
XVII.70. Функция FindObject 414
XVII.71. Функция FindPropertyObject 415
XVII.72. Функция Fix 416
XVII.73. Конструкция For...Next 417
XVII.74. Функция Format 417
XVII.75. Функция FreeFile 420
XVII.76. Функция FreeLibrary 421
XVII.77. Оператор Function 421
XVII.78. Оператор Get 422
XVII.79. Функция GetAttr 423
XVII.80. Функция GetProcessServiceManager 425
XVII.81. Функция GetSolarVersion 425
XVII.82. Функция GetSystemTicks 426
XVII.83. Ключевое слово GlobalScope 427
XVII.84. Оператор GoSub 427
XVII.85. Оператор GoTo 428
XVII.86. Функция Green 429
XVII.87. Функция HasUnoInterfaces 430
XVII.88. Функция Hex 431
XVII.89. Функция Hour....432
XVII.90. Оператор If 432
XVII.91. Функция IIF....433

XVII.92. Оператор Imp 434
XVII.93. Оператор Input 435
XVII.94. Функция InputBox 436
XVII.95. Функция InStr 437
XVII.96. Функция Int 438
XVII.97. Функция IsArray 439
XVII.98. Функция IsDate 440
XVII.99. Функция IsEmpty 440
XVII.100. Функция IsMissing 441
XVII.101. Функция IsNull 442
XVII.102. Функция IsNumeric 442
XVII.103. Функция IsObject 443
XVII.104. Функция IsUnoStruct 444
XVII.105. Функция Kill 444
XVII.106. Функция LBound 445
XVII.107. Функция LCase 446
XVII.108. Функция Left 446
XVII.109. Функция Len 447
XVII.110. Ключевое слово Let 447
XVII.111. Оператор Line Input 448
XVII.112. Функция Loc 448
XVII.113. Функция Lof 449
XVII.114. Функция Log 450
XVII.115. Оператор Loop 451
XVII.116. Оператор LSet 451
XVII.117. Функция LTrim 452
XVII.118. Ключевое слово Private 453
XVII.119. Ключевое слово Public 454
XVII.120. Функция Red 454
XVII.121. Оператор RSet 455
XVII.122. Функция Shell 456
XVII.123. Функция UBound 458
XVII.124. Функция UCase....459
XVII.125. Имена файлов и адреса URL....460
Глава XVIII. Другие языки....461
XVIII.1. C#....461
XVIII.2. Программисты на Visual Basic 462

Приложения....465
Приложение 1. Дополнения переводчика....467
Пример макроса, исправляющего оглавление. 467
Приложение 2. Горячие клавиши....469
Введение....469
Назначение сочетаний клавиш....469
Общие сочетания клавиш для OpenOffice.org....471
Сочетания клавиш, использующие функциональные клавиши....473
Сочетания клавиш в Галерее....474
Выделение строк и столбцов в таблице....476
Сочетания клавиш для рисованных объектов....476

Приложение 3. Особенности сборки OpenOffice.org pro.....479
Общие улучшения.....479
Улучшения в программе Writer.....480
Улучшения в программе Calc.....480
Улучшения в программе Impress 480
Улучшения в сборке для Windows.....481
Улучшения в сборке для GNU/Linux и FreeBSD.....481
Приложение 4. Алфавитный указатель (Index).....483

От редакции

Книга, которую вы, уважаемый читатель, держите в руках является первым переводом на русский язык одной из работ Эндрю Питоньяка (Andrew Pitonyak), которую он скромно называет «Статьёй». Однако объём данного труда далеко перешагнул «статейные» рамки, и именно по этому мы решили издать её в виде самостоятельной книги.

Оригинальное название «Статьи» на английском языке «Useful Macro Information For OpenOffice By Andrew Pitonyak». Её оригинал часто редактируется и доступен по адресу:

`http://www.pitonyak.org/AndrewMacro.odt`

В тексте Автор часто ссылается на свою книгу «OpenOffice.org Macros Explained». Она гораздо больше по объёму и глубже по охвату материала. К сожалению, она ещё не переведена на русский язык.

От переводчика

Взяться за перевод я рискнул после того, как поисковик в Интернете не нашёл ни одной стоящей статьи по макросам OpenOffice на русском языке.

Хотя я и сам бы с удовольствием прочёл хорошую книгу на эту тему, а не перевод новичка, но все же это лучше, чем ничего. Надеюсь, теперь, к моменту окончания работы над переводом, я уже не совсем новичок, поэтому надеюсь проверить в работе и дополнить многие макросы для последней на сегодня версии OpenOffice.org pro.

Примечания переводчика помечены словом «Прим. пер.». В остальных случаях «Я», «Моя» означает автора оригинала Эндрю Питоньяка.

С уважением, Заимских А. Н.

`http://buhcia.narod.ru`

Об этой книге

Благодарности

Благодарю мою жену Michelle Pitonyak за поддержку и вдохновение меня в написании этого документа. Благодарю Laurent Godard (и всю группу переводчиков на французский язык) за полезные идеи и упорную работу. Благодарю Hermann-Josef Beckers за перевод на немецкий язык. Вы, Robert Black Eagle, не знали усталости и представили превосходный обзор, не говоря уже о Вашем потрясающем основном документе «How To» («как сделать»). Вы, Kelvin Eldridge проявили большое понимание и помогли мне понять множество ошибок. Благодарю Jean Hollis Weber и Solveig Haugland за персональные ответы, когда я встречал проблемы с определёнными документами. Вы, Sasa Kelecevic и Hermann Kienlein, предоставили много работающих примеров, чтобы помочь мне. Благодарю Andreas Bregas за быстрые ответы, описания и исправления. Благодарю Mathias Bauer за объяснения и примеры, объясняющие глубинные секреты внутренней работы программ. Я выражаю большую благодарность всему обществу Open Source и рассылкам, откуда я получил много полезной информации и поддержки. Я должен поблагодарить также Jean Hollis Weber и C Robert Pearsall за согласие быть техническими редакторами моей книги при публикации её в августе 2004. Благодарю Christian Junker за прочтение и значительные исправления и дополнения к этому документу — он провёл много часов за выверкой текста и внесением изменений.

Заявление

Содержание этого документа не даёт никаких гарантий применимости, точности и надёжности. Используйте информацию и макросы из этого документа на свой риск. В маловероятном случае, если это приведёт к потере части или всех Ваших данных и (или) оборудования компьютера, ни я, ни кто-либо из участников не будет нести ответственности.

Предисловие автора

Эта статья «Andrew's Macro Document», свободно распространяемый при соблюдении условий лицензии документ, был начат раньше, чем я написал ныне опубликованную книгу[1]. В книге содержится превосходный материал для начинающих, с полностью проверенными примерами, ссылочным материалом и иллюстрациями. Она является справочником по поддерживаемым командам Basic.

Настоящая же статья содержит меньше команд, примеры могут содержать ошибки, описания не столь подробны. Другими словами, подумайте о приобретении моей книги «OpenOffice.org Macros Explained»: `http://www.pitonyak.org/book/`

[1] Ещё не издана на русском языке — прим. ред.

Когда я написал свой первый макрос для OpenOffice, я был увлечён сложностью этой задачи. Я начал составлять макросы, которые выполняют простые задачи, и одновременно с этим приступил к созданию документации по макросам, нужной сообществу пользователей. История моих поисков понимания макросов OOo превратилась в эту статью.

Оригинал документа часто редактируется и доступен по адресу:
`http://www.pitonyak.org/AndrewMacro.odt`

Дата последней редакции оригинала документа[1] July 22, 2007 at 23:0413:07:00 PM Document Revision: 1012

Моя Интернет-страница показывает также дату и время последнего обновления. Хотя шаблон, использованный для создания данного документа, не обязателен для чтения настоящей статьи, все же я поместил его на мой сайт.

Контактная информация

Andrew Pitonyak • 6888 Bowerman Street West • Worthington, OH 43085 • USA, домашний адрес: `andrew@pitonyak.org`, домашний телефон: 614-438-0190 • сотовый телефон: 614-937-4641

Позвольте представиться.

У меня две степени бакалавра, одна — по компьютерам и одна — по математике. У меня также две степени магистра, одна — по применению математики в промышленности и одна — по компьютерам. Я обучался в Оклендском университете (Oakland) в Рочестере, Мичиган. Государственный университет Огайо (Ohio State University) в Колумбусе, Огайо, и Дрезденском техническом университете в Дрездене, Германия.

Ссылка на лицензию публичной документации (Public Documentation License)

Содержание данной Документации распространяется только по Лицензии Публичной Документации версии 1.0 (Public Documentation License Version 1.0), далее по тексту — «Лицензия», Вы имеете право использовать эту Документацию только если Вы согласны с условиями этой Лицензии.

Копия этой лицензии доступна по ссылке
`http://www.openoffice.org/licenses/pdl.pdf`

Оригинал Документации находится по ссылке
`http://www.pitonyak.org/AndrewMacro.odt`

Обязательное приложение согласно лицензии:

Contribution	*Contributor*	*Contact*	*Copyright*
Original author	Andrew Pitonyak	`andrew@pitonyak.org`	2002-2007

[1] Которая была использована при переводе — прим. пер.

Contribution	Contributor	Contact	Copyright
French translation	The French Nativelang documentation project	doc@fr.openoffice.org	2003
German Translation	Hermann-Josef Beckers	hjb-rheine@t-online.de	2003
Macros	Hermann Kienlein	info@kienlein.com	2002-2003
Macros	Sasa Kelecevic	scat@teol.net	2002-2003
Editing and additions	Christian Junker	usboy2004@web.de	2004
Русский перевод Russian translation	Заимских Александр Николаевич	http://buhcia.narod.ru buhcia2006@yandex.ru	2007

Сокращения и обозначения

OpenOffice.org часто сокращают как OOo. OOo Basic — это имя макро-языка, включённого в OpenOffice. OOo Basic похож на Visual Basic, так что знание Visual Basic было бы очень полезно.

Таблица 1. Общие сокращения и обозначения

Abbreviation	Definition
GUI	Графический интерфейс пользователя (Graphical User Interface)
IDE	Объединённая среда разработки (Integrated Development Environment)
UNO	Универсальные сетевые объекты (Universal Network Objects)
IDL	Язык описания интерфейса (Interface Definition Language)
SDK	Набор программ для разработчика (Software Development Kit)
OLE	Связывание и внедрение объектов (Object Linking and Embedding)
COM	Объектная модель компонента (Component Object

Abbreviation	*Definition*
	Model)
OOo	OpenOffice.org

Доступные ссылки

Включённый материал

Использование помощи Help — OpenOffice.org Help для открытия страниц справки по OOo. OOo имеет справку для Writer, Calc, Base, Basic, Draw, Math, и Impress. Верхний левый угол справочной системы OOo содержит окошко с выпадающим меню, которое определяет, для какого компонента открывается справка. Чтобы посмотреть справку для Basic, в выпадающем меню нужна строка «Help about OpenOffice.org Basic».

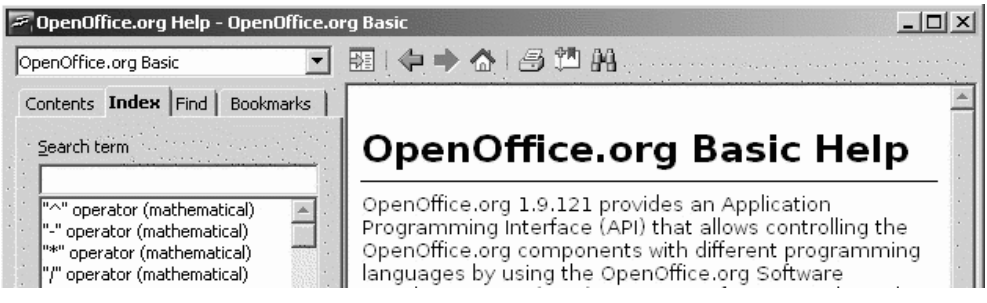


Рисунок 1: Страница помощи по OOo Basic.

Много замечательных макросов включено в OOo. Например, я нашёл макрос, который печатает свойства и методы для объекта. Я использовал эти методы до того, как я написал свой собственный перечень объектов. Используйте позиции Tools (инструменты) | Macros (макросы) | Organize Macros (управление макросами) | OpenOffice.org Basic (язык Basic) *(до версии 2.0 были только Tools | Macros | Macro)* для входа в диалог Macro. Расширяйте библиотеку инструментов Tools в разделе OpenOffice.org library. Попробуйте модуль Debug — несколько хороших примеров, включая WritedbgInfo (документ) и printdbgInfo (лист).

> ***Совет*** *До выполнения макроса должна быть загружена библиотека, содержащая этот макрос. Первая глава книги, где подробно обсуждаются библиотеки, доступна для свободного скачивания*

http://www.pitonyak.org/book/. Это большое введение для начинающих, и оно бесплатно.

Ссылки на сайты

Следующие ссылки помогут понять трудные поначалу понятия:

http://www.openoffice.org	главная
http://www.forum.i-rs.ru http://user.services.openoffice.org/en/forum	Если Вам нужна помощь по макросам, это — место, где можно спросить, возможно, это лучшие форумы в Интернете
http://api.openoffice.org/docs/common/ref/com/sun/star/module-ix.html	Это официальная ссылка IDL , здесь Вы найдёте описания почти всех команд
http://api.openoffice.org/DevelopersGuide/DevelopersGuide.html	Вторая официальная документация, описывает все детально на почти 1000 страниц!
http://www.pitonyak.org/AndrewMacro.odt	Последняя редакция оригинала настоящей статьи
http://www.pitonyak.org/book	Здесь можно купить изданную книгу[1], она подробнее!
http://www.alians-kniga.ru	Здесь можно купить данную книгу, которую вы держите в руках — Прим. ред.
http://docs.sun.com/app/docs	Фирма Sun написала книгу по программированию макросов. Ищите StarOffice
http://api.openoffice.org/basic/man/tutorial/tutorial.pdf	Превосходный учебный курс
http://docs.sun.com/	оригинальная документация фирмы Sun. Ищите StarOffice и документацию StarBasic
http://api.openoffice.org	Громоздкий, но полный сайт
http://documentation.openoffice.org	

[1] «OpenOffice.org Macros Explained» - Прим. ред.

http://documentation.openoffice.org/HOW_TO/various_topics/How_to_use_basic_macros.sxw	документ«How To»
http://udk.openoffice.org	дополнительная информация по UNO
http://udk.openoffice.org/common/man/tutorial/office_automation.html	примеры по OLE
http://ooextras.sourceforge.net/	примеры
http://disemia.com/software/openoffice/	примеры, в основном для Calculator
http://kienlein.com/pages/oo.html	примеры
http://www.darwinwars.com/lunatic/bugs/oo_macros.html	примеры
http://sourceforge.net/project/showfiles.php?group_id=43716	примеры
http://www.kargs.net/openoffice.html	примеры
http://www.8daysaweek.co.uk/	примеры и документация
http://ooo.ximian.com/lxr/	здесь можно покопаться в исходных кодах
http://homepages.paradise.net.nz/hillview/OOo/	множество превосходных макросов, включая исходные тексты макросов, макросы для клавиш и информацию по преобразованию из MS Office

Для конкретной подробной информации ищите в Руководстве для Разработчика Developer's Guide или поищите в Интернете что-то типа «cursor OpenOffice». Для более конкретного поиска используйте : «site:api.openoffice.org cursor».

Если я знаю наименование пакета, я обычно догадываюсь о его расположении в Интернете. Я просматриваю для информации об объектах сайт API:

http://api.openoffice.org/docs/common/ref/com/sun/module-ix.html

Программирование в OOo

Глава I. Начало: концепция

Первая глава моей книги, доступная для свободного скачивания, более полна для начинающего пользователя. Я считаю более полезным начать с неё:

http://www.pitonyak.org/book/

Макросы используются для автоматизации действий в OpenOffice.org. Макрос может автоматизировать такие действия, который иначе потребовали бы длительных ручных манипуляций с возможными ошибками. В настоящее время автоматизированные действия наиболее легко выполняются написанием макросов в OOo Basic. Новая среда для макросов в версии 2 OOo должна облегчить использование других языков, но Basic все ещё наиболее лёгкий в использовании. Вот несколько преимуществ использования языка OOo Basic для управления OOo:

- лёгок для изучения
- поддерживает объекты COM (ActiveX) и расширенные возможности GUI в OpenOffice
- есть сообщество пользователей в Интернет
- это — решение для нескольких платформ (Linux, Windows ...)

> ***Примечание*** *OpenOffice.org Basic известен также как StarBasic.*

I.1. Мой первый макрос: «Hello World»

Откройте новый документ OOo . Используйте меню Сервис — Макросы — Управление макросами — OpenOffice.org Бэйсик, чтобы начать диалог макросов Macro. С левой стороны окна диалога найдите документ, который Вы только что открыли. Новый документ, вероятно, назван «untitled1» или Безымянный1. Кликните (нажмите левую клавишу мыши) справа ниже от «untitled1» на слове «standard». Кликните кнопку Создать далеко справа для создания нового модуля.

Использование имени «Module1», вероятно, не лучшее решение. Когда у Вас открыто несколько документов и все они имеют модуль с именем «Module1», то становится трудно работать с ними. Лучше назовём Ваш первый модуль «MyFirstModule». Откроется среда редактирования и отладки макросов OOo Basic IDE . Введите (или скопируйте) текст, приведённый в Лист. 1.

Лист. 1: Ваш первый макрос, «Hello world».

```
Sub Main
  Print "Hello World"
End Sub
```

Кликните на кнопке с зелёным треугольничком («Выполнить Basic») в верхней панели для выполнения Вашего первого макроса OOo Basic.

I.2. Группировка текста программ

OOo Basic основан на процедурах и функциях, который задаются ключевыми словами Sub и Function — далее они будут называться процедурами (procedures, routines, subroutines) или соответственно функциями. Каждая процедура может вызывать другие процедуры. Разница между Sub и Function в том, что функция возвращает значение, а процедура — нет. Макрос на Лист. 2 получает текстовую строку от функции с именем HellowWorldString.

Лист. 2: «Hello world» с использованием процедуры и функции.

```
Sub HelloWorld
  Dim s As String
  s = HelloWorldString()
  MsgBox s
End Sub

Function HelloWorldString() As String
   HelloWorldString  = "Hello World"
End Function
```

Каждый модуль (module) содержит набор процедур (функций). Библиотека (Library) содержит набор модулей. Документ (document) может содержать библиотеку или несколько библиотек. Библиотека может существовать также на уровне Приложения (application level), такого как OOo Writer.

I.3. Отладка

Объединённая среда разработки IDE имеет средства отладки, такие как установка точек прерывания и наблюдаемых переменных. Вы можете также выполнять макрос по шагам, по оператору за один раз. Полезно установить точки прерывания перед подозрительным местом макроса и затем выполнять его по шагам, чтобы увидеть, как возникает ошибка.

I.4. Переменные, константы, строки и числовые типы

Переменная похожа на коробку, которая что-то содержит. Как коробка, некоторые переменные больше подходят для определённых типов содержимого. Тип переменной определяет, что она может содержать. Попытка сохранить в ней данные неверного типа часто вызывают ошибку, конкретнее — Exception. До использования переменной она должна получить имя. Простой пример:

Лист. 3: Определение простой переменной.

```
Dim <variablename> as <Type>
Dim i as integer
```

Переменные меняют свои значения, без этого Вы не можете сохранять данные. Как и содержимое коробки, содержимое переменной может меняться. У меня в комнате есть коробка, и только я могу изменить её содержимое. Теперь, я женился, так что моя жена тоже может менять содержимое коробки. Точнее говоря, область видимости переменной определяет, где доступна эта переменная. Вы можете определить переменную как доступную только для одной процедуры, или для целого модуля, или даже как глобальную — для всех библиотек. Ключевые слова «Global» (глобальная — для всех библиотек), «Public» (общая — для целого модуля), and «Private» (частная — только для одной процедуры или функции) используются для определения области видимости переменной. Местонахождение Вашего макроса, в котором определена эта переменная, также может изменять её область видимости.

Переменная, чьё значение не может быть изменено, называется константой (Constant). Константа определяется с использованием ключевого слова Const.

Лист. 4: Определение константы.

```
Const <constantname> = <constantvalue>
const pi=3.14159265358
```

OOo Basic поддерживает много различных типов данных. Строки — это простые текстовые значения, ограниченные двойными кавычками. Что касается чисел, то поддерживаются и целые числа, и числа с плавающей точкой. Подробнее о переменных см. главу XI «Язык (Language)» на стр. 319 а также справку OOo. Можно прочесть и мою книгу «OpenOffice.org Macros Explained».

I.5. Обращение к объектам и создание объектов (Objects) в OpenOffice

OpenOffice.org включает большое число сервисов и объектов (Services, Objects); обычно объекты легко доступны. Доступ к текущему документу и к рабочему столу использует глобальные переменные ThisComponent и StarDesktop соответственно — обе эти глобальные переменные являются объектами. Если у Вас есть документ, Вы можете получить доступ к нему (см. Лист. 5).

Лист. 5: Определение и использование некоторых переменных.

```
Sub Example
  Dim oDoc  As Variant        ' Ссылка на активный документ
  Dim oText As Variant        ' Ссылка на главный объект Text
                                    ' этого документа
  oDoc  = ThisComponent             ' Получить активный документ
  oText = oDoc.getText()      ' Получить объект — текст
                                    ' документа  TextDocument
End Sub
```

Чтобы создать экземпляр объекта, используйте глобальный метод createUnoService() как показано в Лист. 6. Он также показывает, как создать структуру (structure).

Лист. 6: Это старый способ отправки сообщений dispatch.

```
Sub PerformDispatch(vObj, uno$)
  Dim vParser    ' Будет ссылаться на URLTransformer.
  Dim vDisp      ' Результат отправки сообщения.
  Dim oUrl As New com.sun.star.util.URL   'создаёт структуру

  oUrl.Complete = uno$
  vParser = createUnoService("com.sun.star.util.URLTransformer")
  vParser.parseStrict(oUrl)

  vDisp = vObj.queryDispatch(oUrl,"",0)
  If (Not IsNull(vDisp)) Then vDisp.dispatch(oUrl,noargs())
End Sub
```

> ***Совет*** *Руководство разработчика Developer's Guide гласит, что тип Variant должен использоваться вместо типа object. Это обсуждается подробнее на стр.326.*

> ***Совет*** *Хотя Вы можете создать экземпляр объекта Desktop, но ниже будет показано, что Вам следовало бы вместо этого использовать глобальную переменную StarDesktop . Вам нужно создавать Desktop только на других языках, кроме StarBasic.*
>
> *createUnoService(«com.sun.star.frame.Desktop»)*

I.6. Что же такое UNO?

UNO (Universal Network Object, Универсальный сетевой объект) был создан, чтобы каждая среда (окружение) environment могла успешно взаимодействовать с каждой другой средой (окружением). Почему это нужно? Потому что разные языки программирования и разные среды (окружения) могут иметь разные способы представления одного и того же типа данных. Даже целые числа, наиболее простая вещь, могут быть представлены разными способами на разных компьютерах и разных языках программирования.

- UNO определяет множество базовых типов, таких как строки, целые числа и т. д. (поэтому они будут одними и теми же в разном окружении).
- объекты UNO могут иметь методы (метод может возвращать значение и может принимать аргументы).
- объекты UNO могут иметь свойства. Свойство может быть другим объектом UNO или может иметь простой тип. Свойство может также быть необязательным (опциональным).
- объекты UNO определены с использованием сложного (для большинства читателей) Языка Определения Интерфейса Interface Definition Language (UNO IDL). Хотя Вам может не понравиться изучение UNO IDL, именно он определяет свойства и методы, которые поддерживают объекты.
- Предположим, что я имею окружение UNO в языке Basic, и другое окружение в языке Java. Для того, чтобы окружение UNO использовало объект, все, что требуется — определение объекта на языке UNOIDL. Эти окружения могут легко передавать объекты туда и обратно.

Я использую OOo API для взаимодействия с OOo. Эта технология API может возвращать:

- внутренний тип UNO такой как число с плавающей точкой, целое число и т. д.
- константы — значения, обычно числовые, связанные с именем константы. Например, обычный текст имеет размер шрифта, определяемый константой com.sun.star.awt.FontWeight.NORMAL, которая равна 100.0.
- нумерации, использующие имя похожее на com.sun.star.awt.FontSlant.ITALIC, но они обычно связаны с целым числом, даже если значение нумератора не выводится на печать.

- Структуры — это объекты со свойствами, но не имеющие методов. Сам по себе факт, что объект не имеет методов, не обязательно означает, что это структура. Объект должен быть именно определён как структура.
- UNO объект с методами и (или) свойствами.

I.6.1. Структуры

Структуры, которые определены с помощью OpenOffice.org, могут быть проверены с использованием метода IsUNOStruct().

Лист. 7: Проверка на структуру UNO.

```
Sub ExamineStructures
  Dim oProperty As New com.sun.star.beans.PropertyValue
  With oProperty
    .name = "Joe"
    .value = 17
  End With
  Print oProperty.Name & " is " & oProperty.Value
  If IsUNOStruct(oProperty) Then
    Print "oProperty is an UNO Structure"
  End If
End Sub
```

Хотя Вы можете определить и использовать свои структуры, IsUNOStruct() не распознает их как структуры.

I.6.2. Интерфейсы

Почти все объекты OOo поддерживают сервисы (службы, services) и интерфейсы (interface). Когда здесь используется слово «интерфейс», связанное с каким-либо объектом, то это слово всегда означает набор методов, которые этот объект поддерживает. Например, если объект поддерживает интерфейс com.sun.star.frame.XStorable или короче XStorable, то объект поддерживает методы, перечисленные в Таблице 2.

Таблица 2. Методы, определённые интерфейсом XStorable.

Метод	***Описание***
hasLocation	Возвращает true если объект имеет сведения о его расположении, либо потому, что он был загружен оттуда, либо потому, что он был сохранен туда.
getLocation	Возвращает адрес URL , куда объект был сохранен.
isReadonly	*(только для чтения)* Если возвращает true, нельзя вызывать метод store().
store	Сохраняет данные по адресу URL , откуда они были

Метод	*Описание*
	загружены.
storeAsURL	Сохраняет объект по указанному адресу URL. Последующие вызовы метода store будут использовать этот адрес URL.
storeToURL	Сохраняет объекта по указанному адресу URL, но это не меняет адрес URL документа.

Макрос в Лист. 41 проверяет, поддерживает ли компонент интерфейс XStorable. Если да, то макрос использует методы hasLocation() и getLocation(). Проверка необходима потому, что, возможно, возвращаемый компонент не будет поддерживать интерфейс XStorable или данный документ не был сохранен и поэтому не имеет сведений о расположении, которые могли бы быть напечатаны.

I.6.3. Сервисы (Services)

Когда здесь используется слово «сервис» как связанное с объектом, то это слово означает набор интерфейсов, свойств и сервисов, которые поддерживает объект. Для конкретного интерфейса или сервиса можно найти его определение по ссылке:

`http://api.openoffice.org.`

Определение сервиса задаёт только объекты, непосредственно определённые этим сервисом. Например, сервис TextRange определён как поддерживающий сервис CharacterProperties, который определён как поддерживающий конкретные свойства, такие как CharFontName. Объект, которых поддерживает сервис TextRange будет поэтому поддерживать и свойство CharFontName, даже если оно не указано непосредственно в определении объекта; Вам нужно просмотреть все перечисленные интерфейсы, сервисы и свойства, чтобы получить все, что реально поддерживает данный объект.

I.6.4. Интерфейсы и сервисы (services)

Из вышесказанного следует, что, хотя доступны определения почти всех сервисов и интерфейсов, это не лучший способ быстро получить все сведения об объекте. BASIC IDE позволяет Вам поставить точки остановки в Вашем коде программы для просмотра значений переменных. Многие программисты просматривают объекты, используя бесплатную библиотеку просмотра объектов, называемую Xray. У меня есть мои собственные процедуры просмотра объектов, как написано в моей книге OpenOffice.org Macros Explained.

Если объект содержит интерфейс XserviceInfo, то Вы можете узнать, поддерживает ли объект конкретный сервис:

`http://api.openoffice.org/docs/common/ref/com/sun/star/lang/XServiceInfo.html`

Эта возможность обычно используется для выяснения, принадлежит ли документ к конкретному виду документов.

Лист. 8: Проверка, является ли документ текстовым документом.

```
Dim s As String
s = "com.sun.star.text.TextDocument"
If ThisComponent.supportsService(s) Then
  Print "Документ является текстовым документом"
 Else
  Print "Этот документ не является текстовым документом"
End If
```

I.6.5. Каков тип данного объекта?

Бывает полезно знать тип объекта, чтобы понять, что можно с ним делать. Вот краткий список способов проверки:

Таблица 3: Способы, используемые для проверки переменных.

Method	*Description*
IsArray	Является параметр массивом?
IsEmpty	Является параметр не инициализированной переменной типа variant?
IsNull	Является ли значение параметра пустым?
IsObject	Является параметр объектом OLE?
IsUnoStruct	Является параметр структурой UNO?
TypeName	Каково наименование типа параметра?

Имя типа переменной может также давать информацию о свойствах переменной.

Таблица 4: Значения, возвращаемые выражением TypeName()

Тип	*TypeName()*
Variant	«Empty» или имя содержащегося объекта
Object	«Object», даже если значение пустое. То же касается структуры
regular type	Имя обычного типа, такое как «String» (строка)
array	имя, заканчивающееся круглыми скобками «()»

Лист. 9 иллюстрирует выражения в Таблица 3,

Лист. 9: Проверка переменных.

```
Sub TypeTest
  Dim oSFA
  Dim aProperty As New com.sun.star.beans.Property
  oSFA = CreateUnoService( "com.sun.star.ucb.SimpleFileAccess" )
  Dim v, o As Object, s As String, ss$, a(4) As String
  ss = "Empty Variant: " & GetSomeObjInfo(v) & chr(10) &_
    "Empty Object: " & GetSomeObjInfo(o) & chr(10) &_
    "Empty String: " & GetSomeObjInfo(s) & chr(10)
  v = 4
  ss = ss & "int Variant: " & GetSomeObjInfo(v) & chr(10)
  v = o
  ss = ss & "null obj Variant: " & GetSomeObjInfo(v) & chr(10) &_
    "struct: " & GetSomeObjInfo(aProperty) & chr(10) &_
    "service: " & GetSomeObjInfo(oSFA) & chr(10) &_
    "array: " & GetSomeObjInfo(a())
  MsgBox ss, 64, "Type Info"
End Sub

REM  Функция возвращает основную информацию о типе полученного параметра
REM Она возвращает также размерности массива.
Function GetSomeObjInfo(vObj) As String
  Dim s As String
  s = "TypeName = " & TypeName(vObj) & CHR$(10) &_
    "VarType = " & VarType(vObj) & CHR$(10)
  If IsNull(vObj) Then
    s = s & "IsNull = True"
  ElseIf IsEmpty(vObj) Then
    s = s & "IsEmpty = True"
  Else
    If IsObject(vObj) Then
      On Local Error GoTo DebugNoSet
      s = s & "Implementation = " & _
              NotSafeGetImplementationName(vObj) & CHR$(10)
      DebugNoSet:
      On Local Error Goto 0
      s = s & "IsObject = True" & CHR$(10)
    End If
    If IsUnoStruct(vObj) Then s = s & "IsUnoStruct = True" & CHR$(10)
    If IsDate(vObj) Then s = s & "IsDate = True" & CHR$(10)
    If IsNumeric(vObj) Then s = s & "IsNumeric = True" & CHR$(10)
    If IsArray(vObj) Then
      On Local Error Goto DebugBoundsError:
      Dim i%, sTemp$
      s = s & "IsArray = True" & CHR$(10) & "range = ("
      Do While (i% >= 0)
        i% = i% + 1
        sTemp$ = LBound(vObj, i%) & " To " & UBound(vObj, i%)
        If i% > 1 Then s = s & ", "
        s = s & sTemp$
      Loop
      DebugBoundsError:
```

```
      On Local Error Goto 0
      s = s & ")" & CHR$(10)
    End If
  End If
  GetSomeObjInfo = s
End Function

REM  Функция помещает перехватчик ошибок, в котором можно получить
REM информацию о проблеме и в любом случае выдать хоть что-нибудь!
Function SafeGetImplementationName(vObj) As String
  On Local Error GoTo ThisErrorHere:
  SafeGetImplementationName = NotSafeGetImplementationName(vObj)
  Exit Function
ThisErrorHere:
  On Local Error GoTo 0
  SafeGetImplementationName = "*** Unknown ***"
End Function

REM Проблема в том, что данная функция вызывается, а параметр  vObj
REM   Не поддерживает вызов  getImplementationName(),
REM  поэтому я получаю ошибку вида "Object variable not set"
REM  при определении функции.
Function NotSafeGetImplementationName(vObj) As String
  NotSafeGetImplementationName = vObj.getImplementationName()
End Function
```

I.6.6. Какие методы, свойства, интерфейсы и сервисы поддерживаются?

И сервисы, и структуры имеют имя типа «Object». Для структуры нужно знать тип этой структуры, чтобы можно было найти поддерживаемые свойства — используйте Интернет-сайт API для проверки языка описания интерфейсов IDL. Объекты UNO, с другой стороны, обычно поддерживают ServiceInfo, который предоставляет информацию о сервисе. Метод объектов getImplementationName() возвращает полное имя (то есть со всеми вышестоящими объектами) данного объекта. Используйте полное имя объекта для поиска в Интернете или в Руководстве разработчика. Метод объекта getSupportedServiceNames() возвращает перечень всех интерфейсов, поддерживаемых объектом. Общий способ выяснить, что может делать данный объект, — это вызов следующих трёх методов:

Лист. 10: Что может делать этот объект?

```
MsgBox  vObj.dbg_methods              'Методы данного объекта
MsgBox  vObj.dbg_supportedInterfaces  'Интерфейсы, поддерживаемые этим
объектом
MsgBox  vObj.dbg_properties           'Свойства данного объекта
```

OOo включает макросы для вывода отладочной информации. Наиболее часто используются макросы PrintdbgInfo(объект) и ShowArray(объект). Макрос WritedbgInfo(объект) вставляет отладочную информацию в открытый документ OOo Writer.

I.6.7. Языки, отличные от Basic

StarBasic предоставляет множество возможностей, которые не дают другие языки программирования. В этом разделе упоминаются только некоторые из этих возможностей.

I.6.7.A. Метод CreateUnoService

Метод CreateUnoService() — это короткий путь вместо вызова менеджера глобального сервиса (global service manager) и затем вызова createInstance() для этого менеджера.

Лист. 11: Вызов менеджера глобального сервиса процессов.

```
oManager = GetProcessServiceManager()
oDesk = oManager.createInstance("com.sun.star.frame.Desktop")
```

В языке StarBasic этот процесс можно сделать в одну строку — если только Вам не нужно использовать createInstanceWithArguments().

Лист. 12: CreateUnoService даёт более короткий код программы, чем использование менеджера сервиса процессов.

```
oDesk = CreateUnoService("com.sun.star.frame.Desktop")
```

Другие языки, такие как Visual Basic, не поддерживают метод CreateUnoService().

Лист. 13: Создание сервиса UNO в языке Visual Basic.

```
'Visual Basic не поддерживает CreateUnoService().
'менеджер сервиса процессов всегда создается первым
REM   в Visual Basic.
Rem  Если OOo не запущен ещё, то он запустится сейчас
Set oManager = CreateObject("com.sun.star.ServiceManager")
Rem Создаем объект desktop
Set oDesk = oManager.createInstance("com.sun.star.frame.Desktop")
```

I.6.7.B. Глобальная переменная ThisComponent

В языке StarBasic, глобальная переменная ThisComponent ссылается на текущий документ или на документ, из которого был вызван данный макрос. Глобальная переменная ThisComponent получает своё значение один раз, когда макрос стартует, и затем не меняется, даже если этот макрос делает текущим другой документ — это включает закрытие ThisComponent. Даже со всеми проблемами ThisComponent очень удобен. В других языках, помимо Basic, общим способом является использование getCurrentComponent() для объекта рабочего стола (desktop). К сожалению, это не даст ссылку на объект документа, если текущим является окно Basic IDE или окно помощи, но это маловероятно, если используется другой язык.

Иногда Вам захочется использовать StarDesktop.getCurrentComponent() вместо того, чтобы просто использовать ThisComponent, но макрос выдаст ошибку, если он вызван из среды отладки Basic IDE.

I.6.7.C. Глобальная переменная StarDesktop

Язык OOo Basic определяет несколько глобальных переменных и предоставляет их для удобства программиста. Переменная StarDesktop ссылается на объект рабочего стола (desktop object), который, по существу, является основным приложением OOo. Это имя появилось, когда продукт назывался StarOffice, и оно означало объект главного рабочего стола, который содержал все остальные компоненты. Примерами компонентов, которые может содержать объект рабочего стола, являются все поддерживаемые документы, интегрированная среда разработки BASIC Integrated Development Environment (IDE), а также страницы помощи и поддержки (см. Рисунок 2).

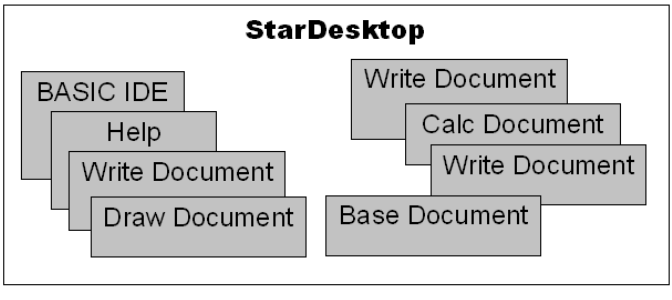


Рисунок 2: Компоненты Рабочего стола StarDesktop

Возвращаясь к макросу из Лист. 13, объект StarDesktop предоставляет доступ ко всем открытым в данный момент компонентам. Метод getCurrentComponent() возвращает компонент, активный в данный момент. Если макрос выполняется из BASIC IDE, то возвращается ссылка на BASIC IDE. Если макрос выполняется в то время, когда на экран остаётся выведен документ, вероятно выполняется посредством меню Сервис — Макросы — Выполнить, то oComp даст ссылку на текущий документ.

> ***Совет*** *Глобальная переменная ThisComponent ссылается на документ, активный в данный момент. Если активным является компонент другого типа, не документ, то ThisComponent ссылается на последний активный документ. В версии OOo 2.01 компоненты Basic IDE, страницы помощи, базы*

данных (Base documents) не вызывают изменения ThisComponent на ссылку на такой компонент.

I.6.8. Методы и свойства доступа к объектам в зависимости от языка

StarBasic автоматически делает методы и свойства, поддерживаемые объектом, доступными, причём иногда делает доступными свойства. которые недоступны для других методов. В других языках интерфейс, который определяет метод, который Вы хотели бы вызвать, должен быть извлечён (extracted) до его использования (см. Лист. 14).

Лист. 14: В языке Java Вы должны получить интерфейс до того, как Вы его сможете использовать.

```
XDesktop xDesk;
xDesk = (XDesktop) UnoRuntime.queryInterface(XDesktop.class, desktop);
XFrame xFrame = (XFrame) xDesk.getCurrentFrame();
XDispatchProvider oProvider = (XDispatchProvider)
UnoRuntime.queryInterface(XDispatchProvider.class, xFrame);
```

Если курсор на экране находится в текстовой секции, то свойство TextSection курсора содержит ссылку на эту текстовую секцию. В противном случае свойство TextSection имеет значение null. В языке StarBasic можно получить текстовую секцию следующим способом:

Лист. 15: OOo Basic позволяет получить прямой доступ к свойствам.

```
If IsNull(oDoc.CurrentController.getViewCursor().TextSection) Then
```

В других языках (не StarBasic) свойство CurrentController и свойство TextSection не доступны напрямую. Текущая позиция доступна с использованием метода «get», а текстовая секция доступна как значение свойства.

Лист. 16: Доступ к некоторым свойствам с использованием методов get.

```
oVCurs = oDoc.getCurrentController().getViewCursor()
If IsNull(oVCurs.getPropertyValue("TextSection")) Then
```

Код программы, использующий методы get и set, легче для переноса в другие языки программирования. StarBasic позволяет также обращаться с некоторыми свойствами как с массивом, даже если данное свойство не является массивом. Хорошим примером является свойство Sheets в документе Calc (электронной таблице). Для Calc оба следующих листинга выполняют одинаковые действия, но только второй из них может использоваться без StarBasic.

Лист. 17: OOo Basic позволяет получить доступ к некоторым свойствам как к массиву.

```
oDoc.sheets(1)
oDoc.getSheets().getByIndex(1)
```

I.7. Итоги

Написание макросов для OpenOffice.org (OOo) является сложной задачей, которой непросто научиться. Проблема не в базовом языке или окружении, а в интерфейсе прикладного программирования OOo API. Базовый язык содержит синтаксис и команды, которые не используются для взаимодействия с документом OOo.

Лист. 18: Простой макрос, который не обращается к OOo API.

```
Sub SimpleExample()
  Dim i As Integer
  i = 4
  Print "The value of i = " & i
End Sub
```

Большинство макросов написаны для взаимодействия с компонентами OOo и поэтому требуют OOo API. В этом смысле термин API содержит объекты и методы и свойства каждого объекта.

Лист. 19: Простой макрос, который использует OOo API для просмотра текущего компонента.

```
Sub ExamineCurrentComponent
  Dim oComp
  oComp = StarDesktop.getCurrentComponent()
  If HasUnoInterfaces(oComp, "com.sun.star.frame.XStorable") Then
    If oComp.hasLocation() Then
      Print "Текущий компонент OOo  имеет адрес URL: " & oComp.getLocation()
    Else
      Print "Текущий компонент не имеет адреса местоположения"
    End If
  Else
    Print "Текущий компонент не может быть сохранен"
  End If
End Sub
```

Хотя макрос из Лист. 19 достаточно простой, но требуется знать очень многое, чтобы его написать. Этот макрос начинается с описания переменной oComp, которая по умолчанию имеет тип Variant, потому что явного указания типа нет.

Глава II. Примеры

II.1. Отладка и проверка макросов

Может оказаться трудным определить, какие методы и свойства доступны для данного объекта. В этом разделе описаны способы, которые могут помочь.

II.1.1. Определить тип документа

В OOo основная мощь заключена в сервисах. Для определение типа документа достаточно посмотреть, какие сервисы он поддерживает. Приведённый ниже макрос использует этот способ. Это более надёжно, чем использование метода `getImplementationName()`.

Лист. 20: Определение большинства типов документов OpenOffice.org

```
'Author: Included with OpenOffice
'Modified by Andrew Pitonyak
Function GetDocumentType(oDoc)
  Dim sImpress$
  Dim sCalc$
  Dim sDraw$
  Dim sBase$
  Dim sMath$
  Dim sWrite$

  sCalc    = "com.sun.star.sheet.SpreadsheetDocument"
  sImpress = "com.sun.star.presentation.PresentationDocument"
  sDraw    = "com.sun.star.drawing.DrawingDocument"
  sBase    = "com.sun.star.sdb.DatabaseDocument"
  sMath    = "com.sun.star.formula.FormulaProperties"
  sWrite   = "com.sun.star.text.TextDocument"

  On Local Error GoTo NODOCUMENTTYPE
  If oDoc.SupportsService(sCalc) Then
    GetDocumentType() = "sCalc"
  ElseIf  oDoc.SupportsService(sWrite) Then
    GetDocumentType() = "sWriter"
  ElseIf  oDoc.SupportsService(sDraw) Then
    GetDocumentType() = "sDraw"
  ElseIf  oDoc.SupportsService(sMath) Then
    GetDocumentType() = "sMath"
  ElseIf  oDoc.SupportsService(sImpress) Then
       GetDocumentType() = "sImpress"
  ElseIf  oDoc.SupportsService(sBase) Then
    GetDocumentType() = "sBase"
  End If
  NODOCUMENTTYPE:
  If Err <> 0 Then
    GetDocumentType = ""
    Resume GOON
    GOON:
  End If
End Function
```

Лист. 21 возвращает имя фильтра экспорта PDF, основанное на типе документа.

Лист. 21: Использование типа документа для определения фильтра экспорта файла PDF

```
Function GetPDFFilter(oDoc)
  REM Author: Alain Viret [Alain.Viret@bger.admin.ch]
  REM Modified by Andrew Pitonyak
  On Local Error GoTo NODOCUMENTTYPE
  Dim sImpress$
  Dim sCalc$
  Dim sDraw$
  Dim sBase$
  Dim sMath$
  Dim sWrite$

  sCalc    = "com.sun.star.sheet.SpreadsheetDocument"
  sImpress = "com.sun.star.presentation.PresentationDocument"
  sDraw    = "com.sun.star.drawing.DrawingDocument"
  sBase    = "com.sun.star.sdb.DatabaseDocument"
  sMath    = "com.sun.star.formula.FormulaProperties"
  sWrite   = "com.sun.star.text.TextDocument"

  On Local Error GoTo NODOCUMENTTYPE
  If oDoc.SupportsService(sCalc) Then
    GetPDFFilter() = "calc_pdf_Export"
  ElseIf  oDoc.SupportsService(sWrite) Then
    GetPDFFilter() = "writer_pdf_Export"
  ElseIf  oDoc.SupportsService(sDraw) Then
    GetPDFFilter() = "draw_pdf_Export"
  ElseIf  oDoc.SupportsService(sMath) Then
    GetPDFFilter() = "math_pdf_Export"
  ElseIf  oDoc.SupportsService(sImpress) Then
    GetPDFFilter() = "impress_pdf_Export"
  End If
  NODOCUMENTTYPE:
  If Err <> 0 Then
    GetPDFFilter() = ""
    Resume GOON
    GOON:
  End If
End Function
```

II.1.2. Вывод всех интерфейсов, методов и свойств объекта

Следующая процедура выводит имена всех интерфейсов, методов или свойств, поддерживаемых объектом. Если вторым параметром является пустая строка, то имена методов выводятся в диалоге. Для других значений свойства и методы выводятся на печать. Так как выводимый список часто очень велик, список выводится в виде нескольких частей (по 16 штук), с которыми можно обращаться по отдельности.

Лист. 22: Вывод методов, интерфейсов и свойств объекта.

```
'Процедура вывода всех методов и свойство объекта, полученного в качестве
'параметра
'Первый параметр — объект; второй — тип вывода: m — методы,
's — интерфейсы, p — свойства
'Author: Tony Bloomfield [tonyb.lx@btinternet.com]
'Modified: hal@thresholddigital.com support services and verify obj.
'Modified: A Pitonyak to fix declaration errors
Sub DisplayMethods(oObj As Object, sWhat As String)
  Dim sMethodLIst As String, sMsgBox As String
  Dim fs As Integer, ep As Integer
  Dim i As Integer
  Dim EOL As Boolean
  If IsNull(oObj) Then
    Print "Объект не существует!"
  Else
    If sWhat = "m" Then
      sMethodList = oObj.DBG_Methods
    ElseIf sWhat = "s" Then
      sMethodList = oObj.DBG_SupportedInterfaces
    ElseIf sWhat = "p" Then
      sMethodLIst = oObj.DBG_Properties
    End If
    fs = 1
    EOL = FALSE
    While fs <= Len(sMethodList)
      sMsgBox = ""
      For i = 0 to 15
        ep = InStr(fs, sMethodList, ";")
        If ep = 0 then
          ep = Len(sMethodList)
        End If
        sMsgBox = sMsgBox & Mid$(sMethodList, fs, ep - fs) & Chr$(13)
        fs = ep + 1
      Next i
      MsgBox sMsgBox
    Wend
  End If
End Sub
```

II.2. Средство X-Ray

Bernard Marcelly написал программное средство X-Ray, которое выводит информацию об объекте в диалоге. Эта программа доступна по адресу:

`http://www.ooomacros.org/dev.php`

Очень рекомендую!

Я сам не использую X-Ray, так как написал своё средство раньше, чем X-Ray стало доступным. Моё средство — SimpleObjectBrowser — доступно в библиотеке Pitonyak на моём Интернет-сайте. Позже я создал «инспектора объектов» (Object inspector) для своей книги «OpenOffice.org Macros

Explained» — Вы должны будете купить эту книгу, чтобы иметь копию инспектора объектов. Заказать книгу можно как по приведённой ниже ссылке, так и на Amazon.com.

http://www.hentzenwerke.com/catalogpricelists/oome.htm

II.3. Диспетчер OOo (Dispatch): использование универсального сетевого объекта Universal Network Objects (UNO)

http://udk.openoffice.org

и руководство разработчика (Developer's Guide)

http://api.openoffice.org/DevelopersGuide/DevelopersGuide.html

— хорошие ссылки для понимания UNO. UNO — это модель компонента, предлагающая взаимодействие между различными языками программирования, моделями объектов, архитектурами компьютеров и между процессами.

На платформе Windows многие пакеты программ используют существующий COM. OOo, тем не менее, имеет собственную мульти-платформенную модель компонента-объекта. За счёт использования своей собственной модели функциональность OOo не ограничена платформой Windows. Во-вторых, OOo может обеспечить лучшую систему обработки ошибок, чем предоставляет COM. Тем не менее, COM все же можно использовать для управления OpenOffice в среде Windows. Более подробно:

http://api.openoffice.org/docs/DevelopersGuide/ProfUNO/ProfUNO.htm#1+4+4+Automation+Bridge.

Приводимый пример запускает на выполнение (dispatch) команду UNO для выполнения операции отмены изменений «undo» с использованием СТАРОГО механизма выполнения. **Желательно не использовать устаревшие методы!**

Лист. 23: Использование устаревшего метода запуска на выполнение (dispatch) для выполнения операции отмены изменений undo.

```
Sub UnoUndo
  PerformDispatch(ThisComponent.CurrentController.Frame, ".uno:Undo")
End Sub
' Вызываемая процедура
Sub PerformDispatch(oObj As Object, uno$)
  Dim oParser As Object
  Dim oUrl As New com.sun.star.util.URL
  Dim oDisp As Object
  REM  Сервис UNO представлен в форме адреса URL в параметре uno$
  oUrl.Complete = uno$

  REM Нужен синтаксический разбор адреса URL, сейчас назначенного oUrl
  oParser = createUnoService("com.sun.star.util.URLTransformer")
```

```
  oParser.parseStrict(oUrl)

  REM Смотрим, поддерживает ли текущий фрейм (Frame) данную команду UNO
  oDisp = oObj.queryDispatch(oUrl,"",0)
  If (Not IsNull(oDisp)) Then
    oDisp.dispatch(oUrl,noargs())
  Else
    MsgBox uno$ & " не найден"
  End If
End Sub
```

Начиная с версии OOo 1.1, это можно написать так

Лист. 24: Использование нового метода запуска на выполнение для операции отмены изменений undo.

```
Sub Undo
  Dim oDisp
  Dim oFrame
  oFrame = ThisComponent.CurrentController.Frame
  oDisp = createUnoService("com.sun.star.frame.DispatchHelper")
  oDisp.executeDispatch(oFrame,".uno:Undo", "", 0, Array())
End Sub
```

Здесь сложным является знание интерфейса UNO и каждого из параметров. Посмотрим следующие примеры, которые должны работать с новыми версиями OOo.

Лист. 25: Использование нового метода запуска на выполнение для экспорта документа OpenOffice Write в формат PDF.

```
Dim a(2) As New com.sun.star.beans.PropertyValue
a(0).Name = "URL"        : a(0).Value = "my_file_name_.pdf"
a(1).Name = "FilterName" : a(1).Value = "writer_pdf_Export"
oDisp.executeDispatch(oFrame, ".uno:ExportDirectToPDF", "", 0, a())
```

Лист. 26: Использование нового метода запуска на выполнение для перехода к ячейке, копирования её и вставки.

```
Dim a(1) As New com.sun.star.beans.PropertyValue
a(0).Name = "ToPoint" : a(0).Value = "$B$3"
oDisp.executeDispatch(oFrame, ".uno:GoToCell", "", 0, a())
oDisp.executeDispatch(oFrame, ".uno:Copy",     "", 0, Array())
oDisp.executeDispatch(oFrame, ".uno:Paste",    "", 0, Array())
```

II.3.1. Диспетчер (Dispatcher — Запуск на выполнение) изменён с версии 1.1 OOo

Работает ли диспетчер (dispatcher — запуск на выполнение) в версии 1.0.3? — Mathias Bauer ответил: «диспетчер (dispatcher — запуск на выполнение) использует некоторые возможности, которые отсутствуют в версии OOo 1.0 (например, dispatch helper). Скажем, программный код для запуска на выполнение с параметрами».

И ещё:

«Если макрос использует для диспетчера (dispatcher — запуск на выполнение) имена, а не числовые коды, то эти имена не будут меняться в более поздних версиях OOo».

II.3.2. Использование диспетчера (dispatcher — запуск на выполнение) требует пользовательского интерфейса (UI).

Имеет ли смысл использовать обычный вызов API вместо обращения к диспетчеру (dispatcher — запуск на выполнение) с именем функции? Ответ Mathias Bauer. Вызов диспетчера (dispatcher — запуск на выполнение) не будет работать с документом без UI (пользовательского интерфейса). Если OOo запущен в режиме реального «сервера» (это может произойти в версии OOo 2.0), когда документы могут загружаться и выполняться без любого GUI (графического пользовательского интерфейса), то будут работать только макросы, использующие «настоящий» API. При этом «настоящий» API намного мощнее и даёт лучшие возможности обращения к реальным объектам. По моему мнению, следует использовать обращение к диспетчеру (Dispatch API) только в двух случаях:

- Запись макроса (меню Сервис — Макросы — Записать макрос)
- Обходной путь, когда определённая задача не может быть выполнена «настоящим» API (он не существует или неработоспособен).

II.3.3. Изменение меню — пример диспетчера (dispatcher — запуск на выполнение)

Начиная с версии OpenOffice.org 2.0 появилась возможность назначить макрос позиции меню с использованием API:

`http://specs.openoffice.org/ui_in_general/api/ProgrammaticControlOfMenuAndToolbarItems.sxw`

Язык API IDL указывает, что существует интерфейс Xmenu и также перехватчик событий XMenuListener. Получение идентификатора меню (Menu ID) из XML-файла, который описывает это меню, уже возможно, а вот назначение макроса для этого идентификатора — нет, пока не будет версия OOo 2.0! Тем не менее, сейчас уже можно использовать DispatchProviderInterceptor, который может быть сравним с перехватчиком или с обработчиком событий.

Используя DispatchProviderInterceptor, мы можем ожидать и перехватывать события — диспетчер (dispatcher — запуск на выполнение). Можно перехватить почти любую команду. Используя идентификаторы позиций меню (slots), можно назначать макросы специальным командам запуска на выполнение (DispatchCommands), которые реализуют выполнение каждой позиции меню. Хотя нельзя запретить пользователю выбирать позицию меню, можно перехватить такую команду, когда она запускается на

выполнение. При этом должен быть установлен, загружен и зарегистрирован перехватчик запуска на выполнение DispatchInterceptor.

Следующий макрос ToggleToolbarVisibility, написанный Peter Biela на форуме по OOo — oooforum, включает/выключает видимость панелей инструментов.

Лист. 27: Включение/выключение видимости панели инструментов.

```
' Author: Peter Biela
' E-Mail: Peter.Biela@planet-interkom.de
' Modified: Andrew Pitonyak
Sub ToggleToolbarVisibility( )
  Dim oFrame
  Dim oDisp
  Dim a()
  Dim s$
  Dim i As Integer
  oFrame = ThisComponent.CurrentController.Frame
  a() = Array( ".uno:MenuBarVisible",   ".uno:ObjectBarVisible",      _
               ".uno:OptionBarVisible", ".uno:NavigationBarVisible", _
               ".uno:StatusBarVisible", ".uno:ToolBarVisible",       _
               ".uno:MacroBarVisible",  ".uno:FunctionBarVisible" )

  oDisp = createUnoService("com.sun.star.frame.DispatchHelper")
  For i = LBound(a()) to Ubound(a())
    oDisp.executeDispatch(oFrame, a(i), "", 0, Array())
  Next

  'для фреймов OOo Calc — CalcFrames
  s = ".uno:InputLineVisible"
  oDisp.executeDispatch(oFrame, s, "", 0, Array())
End Sub
```

II.4. Перехват выполнения команд меню с использованием Basic

Paolo Mantovani обнаружил, что возможно перехватить команды меню из языка Basic. Предполагалось, что это невозможно, так как этот язык не поддерживает создание пользовательских объектов UNO. Я рад отметить, это не первый раз, когда Paolo Mantovani удаётся выполнить то, что считалось невыполнимым.

Лист. 28: Перехват команд меню.

```
' Author: Paolo Mantovani
' Modifed: Andrew Pitonyak
Option Explicit

Global oDispatchInterceptor
Global oSlaveDispatchProvider
Global oMasterDispatchProvider
```

```
Global oFrame
Global bDebug As Boolean

Sub RegisterInterceptor
  Dim oFrame : oFrame = ThisComponent.currentController.Frame
  Dim s$     : s = "com.sun.star.frame.XDispatchProviderInterceptor"
  oDispatchInterceptor = CreateUnoListener("ThisFrame_", s)
  oFrame.registerDispatchProviderInterceptor(oDispatchInterceptor)
End Sub

Sub ReleaseInterceptor()
On Error Resume Next
  oFrame.releaseDispatchProviderInterceptor(oDispatchInterceptor)
End Sub

Function ThisFrame_queryDispatch ( oUrl As Object, _
      sTargetFrameName As String, lFlags As Long ) As Variant
  Dim oDisp
  Dim s$

  ' протокол slot вызывает ошибку OOo...
  If oUrl.protocol = "slot:" Then
    Exit Function
  End If

  If bDebug Then
    Print oUrl.complete, sTargetFrameName, lFlags
  End If

  s =  sTargetFrameName
  oDisp = oSlaveDispatchProvider.queryDispatch( oUrl, s, lFlags )

  'do your management here
  Select Case oUrl.complete
    Case ".uno:Save" ' предотвращаем выполнение команды save
      Exit Function
    'Case "..."  ' перехватываем другие нужные нам команды
      '...
    Case Else
      ' ничего не делаем
  End Select

  ThisFrame_queryDispatch = oDisp
End Function

Function ThisFrame_queryDispatches ( mDispArray ) As Variant
  ThisFrame_queryDispatches = mDispArray
End Function

Function ThisFrame_getSlaveDispatchProvider ( ) As Variant
  ThisFrame_getSlaveDispatchProvider = oSlaveDispatchProvider
End Function

Sub ThisFrame_setSlaveDispatchProvider ( oSDP )
  oSlaveDispatchProvider = oSDP
```

```
End Sub

Function ThisFrame_getMasterDispatchProvider ( )  As Variant
  ThisFrame_getMasterDispatchProvider = oMasterDispatchProvider
End Function

Sub ThisFrame_setMasterDispatchProvider ( oMDP )
  oMasterDispatchProvider = oMDP
End Sub

Sub ToggleDebug()
' внимательнее! Вы получите отладочное сообщение
' для каждого запуска на выполнение (dispatch)....
  bDebug = Not bDebug
End Sub
```

Глава III. Разные примеры

III.1.1. Вывод текста в строку состояния

Согласно Christian Erpelding [erpelding@ce-data.de], используя нижеприведённый код программы, можно только однажды изменить строку состояния, а затем все изменения строки состояния игнорируются. Используйте setText вместо start следующим образом.

Лист. 29: Вывод текста в строку состояния, при этом текст не будет меняться.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
' Можно использовать два метода для получения строки (индикатора) состояния
Function ProgressBar
  ProgressBar = ThisComponent.CurrentController.StatusIndicator
End Function

REM вывести текст в строку состояния — неверный код
Sub StatusText(sInformation as String)
  Dim iLen As Integer
  Dim iRest as Integer

  iLen = Len(sInformation)
  iRest = 270-iLen
  ProgressBar.start(sInformation+SPACE(iRest),0)
End Sub
```

Лист. 30: Вывод текста в строке состояния.

```
Sub StatusText(sInformation)
  Dim iLen as Integer
  Dim iRest As Integer
  iLen=Len(sInformation)
  iRest=350-iLen
  REM используется функция ProgressBar, приведенная выше!
  ProgressBar.setText(sInformation+SPACE(iRest))
End Sub
```

III.2. Вывод всех стилей текущего документа

Это не так трудно. Следующие семейства стилей присутствуют для текстового документа: CharacterStyles (стили символа), FrameStyles (стили врезок = фреймов), NumberingStyles (стили списка = нумерации), PageStyles (стили страницы), and ParagraphStyles (стили абзаца).

Лист. 31: Вывод всех стилей, используемых в текущем документе.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub DisplayAllStyles
  Dim mFamilyNames As Variant, mStyleNames As Variant
  Dim sMsg As String, n%, i%
  Dim oFamilies As Object, oStyle As Object, oStyles As Object
```

```
  oFamilies = ThisComponent.StyleFamilies
  mFamilyNames = oFamilies.getElementNames()
  For n = LBound(mFamilyNames) To UBound(mFamilyNames)
    sMsg = ""
    oStyles = oFamilies.getByName(mFamilyNames(n))
    mStyleNames = oStyles.getElementNames()
    For i = LBound(mStyleNames) To UBound (mStyleNames)
      sMsg=sMsg + i + " : " + mStyleNames(i) + Chr(13)
      If ((i + 1) Mod 21 = 0) Then
        MsgBox sMsg,0,mFamilyNames(n)
        sMsg = ""
      End If
    Next i
    MsgBox sMsg,0,mFamilyNames(n)
  Next n
End Sub
```

III.3. Перебрать все открытые документы

Лист. 32: Перебрать все открытые документы.

```
Sub Documents_Iteration( )
  Dim oDesktop As Object, oDocs As Object
  Dim oDoc As Object, oComponents As Object
  Dim i as Integer 'i подсчитывает, сколько окон открыто в OOo
  i = 0
   oComponents = StarDesktop.getComponents()
  oDocs = oComponents.createEnumeration()
  Do While oDocs.hasMoreElements()
    oDoc = oDocs.nextElement()
    i = i + 1 'Counter
  Loop
  MsgBox "" + i + " компонентов сейчас открыто"   ' исправлено переводчиком
End Sub
```

III.4. Вывести шрифты и другую информацию об экране

Благодаря Paul Sobolik, я получил, наконец, работающий пример. Сначала создаётся «набор инструментов для абстрактного окна», затем создаётся виртуальное устройство, совместимое с экраном. От этого устройства можно получать данные, такие как размеры экрана и информацию о шрифтах.

> ***Совет*** *Определяя шрифт, обычно создают версию для разных атрибутов вывода, таких как жирный или курсив. Когда выводится список поддерживаемых системой фондов, часто показывают все модификации. Windows содержит «Courier New*

Regular», «Courier New Italic», «Courier New Bold» и «Courier New Bold Italic».

На моем Интернет-сайте есть статья о шрифтах, гораздо более подробная, чем это описано здесь.

Лист. 33: Список доступных шрифтов.

```
'Author:  Paul Sobolik
'email:  psobolik@lycos.com
Sub ListFonts
  Dim oToolkit as Object
  oToolkit = CreateUnoService("com.sun.star.awt.Toolkit")

  Dim oDevice as Variant
  oDevice = oToolkit.createScreenCompatibleDevice(0, 0)

  Dim oFontDescriptors As Variant
  oFontDescriptors = oDevice.FontDescriptors

  Dim oFontDescriptor As Object

  Dim sFontList as String
  Dim iIndex as Integer, iStart As Integer
  Dim iTotal As Integer, iAdjust As Integer

  iTotal = UBound(oFontDescriptors) - LBound(oFontDescriptors) + 1
  iStart = 1
  iAdjust = iStart - LBound(oFontDescriptors)
  For iIndex = LBound(oFontDescriptors) To UBound(oFontDescriptors)
    oFontDescriptor = oFontDescriptors(iIndex)
    sFontList = sFontList & iIndex + iAdjust & ": " & _
                oFontDescriptor.Name & " " &_
                oFontDescriptor.StyleName & Chr(10)
    If ((iIndex + iAdjust) Mod 20 = 0) Then
      MsgBox sFontList, 0, "Шрифты от " & iStart & " до " & _
             iIndex + iAdjust & " из " & iTotal
      iStart = iIndex + iAdjust + 1
      sFontList = ""
    End If
  Next iIndex
  If sFontList <> "" Then
    Dim s$
    s = "Шрифты " & iStart & " по " & iIndex & " из " & iTotal
    MsgBox sFontList, 0, s
  End If
End Sub
```

Заметьте, что от Вашей операционной системы (и её настроек) зависит, какие шрифты поддерживаются.

III.4.1. Вывод поддерживаемых шрифтов

См. мою статью о шрифтах на Интернет-сайте
`http://www.pitonyak.org/AndrewFontMacro.odt`

Этот документ просматривает по очереди все шрифты и печатает сведения о каждом с примерами.

III.5. Установка шрифта по умолчанию с использованием ConfigurationProvider

Чтобы изменить шрифт по умолчанию, выполните следующий макрос и перезапустите OOo.

Лист. 34: Установка шрифта по умолчанию с использованием ConfigurationProvider.

```
'Author: Christian Junker
Sub DefaultFont_Change()
  Dim nodeArgs(0) As New com.sun.star.beans.PropertyValue
  Dim s$

  REM Свойства = Properties
  nodeArgs(0).Name = "nodePath"
  nodeArgs(0).Value = "org.openoffice.Office.Writer/DefaultFont"
  nodeArgs(0).State = com.sun.star.beans.PropertyState.DEFAULT_VALUE
  nodeArgs(0).Handle = -1  ' это означает без потока (no handle)!

  REM  необходимые сервисы конфигурации (Config Services)
  s = "com.sun.star.comp.configuration.ConfigurationProvider"
  Provider = createUnoService(s)
  s = "com.sun.star.configuration.ConfigurationUpdateAccess"
  UpdateAccess = Provider.createInstanceWithArguments(s, nodeArgs())

  REM  теперь устанавливаем нужный шрифт по умолчанию DefaultFont..
  UpdateAccess.Standard = "Arial"
  UpdateAccess.Heading = "Arial"
  UpdateAccess.List = "Arial"
  UpdateAccess.Caption = "Arial"
  UpdateAccess.Index = "Arial"
  UpdateAccess.commitChanges()
End Sub
```

III.6. Печать текущего документа

Я попробовал это и теперь могу печатать! Остановился на попытке напечатать лист A4 на принтере с бумагой размера Letter, пока нет на это времени.

Лист. 35: Печать текущего документа (здесь выводятся только возможные параметры принтера — Прим. пер.).

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub PrintCurrentDocument
  Dim mPrintopts1(), x as Variant

  'Здесь диапазон индексов от 0 до  0,  но если Вы хотите установить
и другие свойства,
  ' то укажите больший диапазон индексов....
  Dim mPrintopts2(0) As New com.sun.star.beans.PropertyValue
  Dim oDoc As Object, oPrinter As Object

  oDoc = ThisComponent

  ' Если Вы хотите выбрать конкретный принтер:
  'Dim mPrinter(0) As New com.sun.star.beans.PropertyValue
  'mPrinter(0).Name="Name"
  'mPrinter(0).value="Other printer"
  'oDoc.Printer = mPrinter()

  'Для упрощения печати документа выполните следующее:
  'oDoc.Print(mPrintopts1())

  ' Чтобы напечатать страницы 1-3, 7 и 9
  'mPrintopts2(0).Name="Pages"
  'mPrintopts2(0).Value="1-3; 7; 9"
  'oDoc.Printer.PaperFormat=com.sun.star.view.PaperFormat.LETTER
  'DisplayMethods(oDoc, "propr")
  'DisplayMethods(oDoc, "")
  oPrinter = oDoc.getPrinter()
  MsgBox "Параметры принтера  от " & LBound(oPrinter) & " по " &
UBound(oPrinter)
  sMsg = ""
  For n = LBound(oPrinter) To UBound(oPrinter)
    sMsg = sMsg + oPrinter(n).Name + Chr(13)
  Next n
  MsgBox sMsg,0,"Возможные параметры печати"

  'DisplayMethods(oPrinter, "propr")
  'DisplayMethods(oPrinter, "")

  'mPrintopts2(0).Name="PaperFormat"
  'mPrintopts2(0).Value=com.sun.star.view.PaperFormat.LETTER
  'oDoc.Print(mPrintopts2())
End Sub
```

III.6.1. Печать текущей страницы

Лист. 36: Печатается только текущая страница.

```
Dim aPrintOps(0) As New com.sun.star.beans.PropertyValue
oDoc = ThisComponent
oVCurs = oDoc.CurrentController.getViewCursor()
```

```
aPrintOps(0).Name = "Pages"
aPrintOps(0).Value = trim(str(oVCurs.getPage()))
oDoc.print(aPrintOps())
```

III.6.2. Другие параметры печати

Другой заслуживающий рассмотрения параметр — параметр Wait, значение которого устанавливается равным True. Это вызывает синхронную печать, и выход из процедуры не происходит до тех пор, пока печать не закончена. Это устраняет необходимость перехватывать события конца печати (предполагаю, что Вы захотели бы все же использовать такой перехватчик события). Заметьте, что имя Вашего принтера может потребоваться заключить в угловые скобки < > , а может и не потребоваться. Кажется, это было связано с сетевыми принтерами, но с тех пор могло измениться.

III.6.3. Ландшафтная печать

Лист. 37: Установить режим печати документа в режиме ландшафта (горизонтального расположения страницы).

```
Sub PrintLandscape()
  Dim oOpt(1) as new com.sun.star.beans.PropertyValue

  oOpt(0).Name = "Name"
  oOpt(0).Value = "<здесь введите имя Вашего принтера>"
  oOpt(1).Name = "PaperOrientation"
  oOpt(1).Value = com.sun.star.view.PaperOrientation.LANDSCAPE
  ThisComponent.Printer = oOpt()
End Sub
```

III.7. Информация о конфигурации

III.7.1. Версия OOo

К сожалению, часто функция GetSolarVersion не меняется, даже если меняется версия OOo. Версия OOo 1.0.3.1 возвращает «641», 1.1RC3 возвращает 645, а 2.01 RC2 возвращает 680, но это — недостаточная степень детализации.

Следующий макрос возвращает текущую версию OOo. *(для версии OOo 2.3.0 возвращается «2.3» — Прим. пер.)*

Лист. 38: Получение текущей версии OpenOffice.org

```
Function OOoVersion() As String
  ' Извлекает текущую версию  OOo
  'Author : Laurent Godard
  'e-mail : listes.godard@laposte.net
  '
  Dim oSet, oConfigProvider
```

```
  Dim oParm(0) As New com.sun.star.beans.PropertyValue
  Dim sProvider$, sAccess$
  sProvider = "com.sun.star.configuration.ConfigurationProvider"
  sAccess   = "com.sun.star.configuration.ConfigurationAccess"
  oConfigProvider = createUnoService(sProvider)
  oParm(0).Name = "nodepath"
  oParm(0).Value = "/org.openoffice.Setup/Product"
  oSet = oConfigProvider.createInstanceWithArguments(sAccess, oParm())
  OOOVersion=oSet.getByName("ooSetupVersion")
End Function
```

III.7.2. Язык (локаль) OOo

Лист. 39: Получение текущей локали OpenOffice.org

```
Function OOoLang() as string
  'Author : Laurent Godard
  'e-mail : listes.godard@laposte.net

  Dim oSet, oConfigProvider
  Dim oParm(0) As New com.sun.star.beans.PropertyValue
  Dim sProvider$, sAccess$
  sProvider = "com.sun.star.configuration.ConfigurationProvider"
  sAccess   = "com.sun.star.configuration.ConfigurationAccess"
  oConfigProvider = createUnoService(sProvider)
  oParm(0).Name = "nodepath"
  oParm(0).Value = "/org.openoffice.Setup/L10N"
  oSet = oConfigProvider.createInstanceWithArguments(sAccess, oParm())

  Dim OOLangue as string
  OOLangue= oSet.getbyname("ooLocale")     ' ru   вместо EN-en
  OOLang=lcase(Left(trim(OOLangue),2))     ' ru
End Function
```

III.8. Открыть и закрыть документы (и рабочий стол)

III.8.1. Закрыть OpenOffice и/или документы

Все документы и объекты-фреймы (сервисы) OpenOffice.org поддерживают интерфейс Xcloseable. Чтобы закрыть эти объекты, нужно вызвать close(bForce As Boolean). Вообще говоря, если bForce равен false, то объект может отказаться закрываться, в остальных случаях он закроется обязательно. «Вообще говоря» здесь упомянуто потому, что если что-нибудь зарегистрировалось для ожидания (прослушивания) событий закрытия, то оно может наложить запрет на закрытие документа. Когда Вы даёте команду напечатать документ, управление возвращается до того, как печать закончится. Если бы Вы могли закрыть документ в этот момент, то OOo, вероятно, завершился бы аварийно. См. Руководство разработчика для более полного описания интерфейса Xclosable, и, если Вы не уверены, используйте close(True).

Объект рабочего стола не поддерживает интерфейс Xcloseable по причинам наследования (legacy). Для этого используется метод terminate(). Этот метод вызывает событие queryTermination, которое является широковещательным для всех действующих перехватчиков этого события. Если ни одно указание о запрете закрытия TerminationVetoException не получено, то возникает событие notifyTermination и возвращается значение true. В противном случае возникает широковещательное событие abortTermination и возвращается значение false. Цитируя Mathias Bauer, «метод terminate() существовал уже давно до того, как мы открыли, что неправильно просто закрывать вручную документы или окна. Если бы этого метода не было, нам пришлось бы использовать XCloseable и для объекта рабочего стола.»[Bauer001]

Лист. 40: Правильный способ закрыть документ OpenOffice.org

```
If oDoc.supportsService("com.sun.star.frame.XModel")
  If HasUnoInterfaces(oDoc, "com.sun.star.util.XCloseable") Then
    oDoc.close(true)
  Else
    oDoc.dispose()
  End If
End If
```

Christian Junker добавляет:

Есть много способов правильно закрыть документ или прекратить процесс soffice. Метод StarDesktop.terminate() не прекращает этот процесс!).

Используйте oDoc.close(true), если только у Вас нет необычных фреймов.

Избегайте oDoc.dispose()! Документ в состоянии disposed может быть видимым на экране, но пользователь не может как-либо управлять им. Даже ссылка на такой документ с использованием ThisComponent может привести к ошибке, так как модель документа (Document's Model) в этом случае не существует.

Версия OOo 2.0 будет предоставлять улучшенные возможности Framework, особенно для методов закрытия.

III.8.1.A. Что, если документ (файл) изменён?

Здесь предполагается, что данный документ поддерживает метод close. Другими словами, используется новая версия OpenOffice.org. Попытаемся избежать лишних диалогов, для чего проверяем, был ли документ изменён. Если можно сохранить документ, то попытаемся сделать это.

Лист. 41: Закрытие (без сохранения) документа, который был изменён

```
oDoc = ThisComponent
If (oDoc.isModified) Then
  If (oDoc.hasLocation AND (Not oDoc.isReadOnly)) Then
    oDoc.store()
```

```
  Else
    oDoc.setModified(False)
  End If
End If
oDoc.close(True)
```

III.8.2. Загрузка документа по его адресу URL

Чтобы загрузить документ по заданному URL, используем метод LoadComponentFromURL() из объекта рабочего стола (desktop). Следующий макрос загружает компонент или в новый, или в существующий фрейм.

Синтаксис вызова метода:

```
loadComponentFromURL(
  string aURL,
  string aTargetFrameName,
  long nSearchFlags,
  sequence< com::sun::star::beans::PropertyValue > aArgs)
```

Возвращаемое значение:

```
com::sun::star::lang::XComponent
```

Параметры:

aURL: URL документа, который нужно загрузить. Чтобы создать новый документ, используйте «private:factory/scalc», «private:factory/swriter», и т. д.

aTargetFrameName: Имя фрейма, в котором будет содержаться документ. Если фрейм с таким именем существует, используется этот фрейм, в противном случае он создаётся. Значение этого параметра «_blank» создаёт новый фрейм, «_self» использует текущий фрейм, «_parent» использует родительский фрейм, а «_top» использует корневой фрейм из текущего пути (current path in the tree).

nSearchFlags: Используйте значения FrameSearchFlag для указания, как найти указанное имя фрейма aTargetFrameName. Обычно, просто используйте 0. См. подробнее
`http://api.openoffice.org/docs/common/ref/com/sun/star/frame/FrameSearchFlag.html`

Таблица 5: Флаги поиска фрейма.

код	*Имя*	*Описание*
0	Auto	Текущий фрейм и его дочерние SELF+CHILDREN
1	PARENT	Включает родительский фрейм
2	SELF	Включает текущий (стартовый) фрейм
4	CHILDREN	Включает дочерний фреймы текущего фрейма

код	*Имя*	*Описание*
8	CREATE	Создать новый фрейм, если не будет найден
16	SIBLINGS	Включает другие дочерние фреймы родительского фрейма (братья и сестры)
32	TASKS	Включает все фреймы всех задач в иерархии текущего фрейма
23	ALL	Включает все фреймы, но не других задач. 23 = 1+2+4+16 = PARENT + SELF + CHILDREN + SIBLINGS.
55	GLOBAL	Поиск во всей иерархии фреймов. 55 = 1+2+4+16+32 = PARENT + SELF + CHILDREN + SIBLINGS + TASKS.
63		GLOBAL + CREATE

aArgs: Указывает компонент или фильтр специфического поведения. «ReadOnly» с логическим значением указывает, открыть ли документ только для чтения. «FilterName» указывает тип компонента, который должен быть создан, и фильтр, который следует использовать, например: «scalc: Text — csv». См. подробнее:

`http://api.openoffice.org/docs/common/ref/com/sun/star/document/MediaDescriptor.html`

Лист. 42: Загрузить документ по заданному адресу given URL.

```
'Фрейм "MyName" будет создан, если ещё не существует
'потому что включен бит "CREATE" (в числе 63)
oDoc1 = StarDesktop.LoadComponentFromUrl(sUrl1, "MyName", 63, Array())
REM  Использовать существующий фрейм "MyName"
oDoc2 = StarDesktop.LoadComponentFromUrl(sUrl2, "MyName", 55, Array())
```

Совет *В версии OOo 1.1 фрейм включает loadComponentFromURL, поэтому можно использовать:*

```
oDoc =
oDesk.LoadComponentFromUrl(sUrl_1,
«_blank», 0, Noargs())
oFrame = oDoc.CurrentController.Frame
oDoc =
oFrame.LoadComponentFromUrl(sUrl_2, «»,
2, Noargs())
```

Обратите внимание на аргументы флага поиска и на аргументы имени фрейма.

Внимание *В версии OOo 1.1 Вы можете повторно использовать фрейм, только если знаете его имя.*

Лист. 43: Вставить документ в текущую позицию курсора.

```
Sub insertDocumentAtCursor(sFileUrl$, oDoc)
  Dim oCur  ' Создаваемый курсор
  Dim oVC   : oDoc.getCurrentController().getViewCursor()
  Dim oText : oText = oVC.getText()
  oCur=oText.createTextCursorByRange(oVC.getStart())
  oCur.insertDocumentFromURL(sFileURL, Array())
End Sub
```

Имейте в виду, что начиная с версии OOo 2.x в случае проблем с открытием документа произойдёт ошибка. В более ранних версиях возвращался пустой документ (NULL).

Лист. 44: Создать новый документ.

```
'- создать новый файл OOo Writer
Dim s$ : s = "private:factory/swriter"
oDoc=StarDesktop.loadComponentFromURL(s,"_blank",0,Array())
'- открыть существующий файл
oDoc=StarDesktop.loadComponentFromURL(sUrl,"_blank",0,Array())
```

III.8.2.A. Полный пример

Цель данного примера — в цикле обработать три документа с заголовками one, two и three. Каждый документ открывается в текущем фрейме.

Лист. 45: Загрузить документ в существующий фрейм.

```
Sub open_new_doc
  Dim mArgs(2) as New com.sun.star.beans.PropertyValue
  Dim oDoc
  Dim oFrame
  Dim s As String

  If (ThisComponent.isModified) Then
    If (ThisComponent.hasLocation AND (Not ThisComponent.isReadOnly)) Then
      ThisComponent.store()
    Else
      ThisComponent.setModified(False)
    End If
  End If

  mArgs(0).Name = "ReadOnly"
  mArgs(0).Value = True
```

```
  mArgs(1).Name = "MacroExecutionMode"
  mArgs(1).Value = 4

  mArgs(2).Name = "AsTemplate"
  mArgs(2).Value = FALSE

  REM  Выбрать следующий документ для загрузки
  If ThisComponent.hasLocation Then
    s = ThisComponent.getURL()
    If InStr(s, "one") <> 0 Then
      s = "file:///C:/tmp/two.oxt"
    ElseIf InStr(s, "two") <> 0 Then
      s = "file:///C:/tmp/three.oxt"
    Else
      s = "file:///C:/tmp/one.oxt"
    End If
  Else
      s = "file:///C:/tmp/one.oxt"
  End If

  REM  Получить фрейм этого документа и затем загрузить указанный документ
  REM  в текущий фрейм!
  oFrame = ThisComponent.getCurrentController().getFrame()
  oDoc = oFrame.LoadComponentFromUrl(s, "", 2, mArgs())

  If IsNull(oDoc) OR IsEmpty(oDoc) Then
    Print "Unable to load " & s
  End If
End Sub
```

III.8.3. Сохранить документ с паролем

Чтобы сохранить документ с паролем, нужно задать атрибут «Password».

Лист. 46: Сохранить документ с паролем.

```
Sub SaveDocumentWithPassword
  Dim args(0) As New com.sun.star.beans.PropertyValue
  Dim sURL$

  args(0).Name  ="Password"
  args(0).Value = "test"

  sURL=ConvertToURL("/andrew0/home/andy/test.odt")
  ThisComponent.storeToURL(sURL, args())
End Sub
```

> ***Внимание*** *В имени аргумента важен регистр, поэтому «password» не будет работать!*

III.8.4. Создание нового документа на основе шаблона

Чтобы создать новый документ на основе шаблона, используйте следующее:

Лист. 47: Создать новый документ на основе шаблона.

```
Sub NewDoc
  Dim oDoc
  Dim sPath$
  Dim a(0) As New com.sun.star.beans.PropertyValue
  a(0).Name = "AsTemplate"
  a(0).Value = true

  sPath$ = "file://~/Documents/DocTemplate.stw"
  oDoc = StarDesktop.LoadComponentFromUrl(sPath$, "_blank" , 0, a())
End Sub
```

Если Вы хотите отредактировать шаблон в качестве шаблона (to edit the template as a template), установите значение параметра «AsTemplate» равным «False».

> ***Внимание*** *После загрузки документа как скрытого, Вам не следует делать этот документ видимым, потому что не все требуемые сервисы будут инициализированы. Это приведёт к сбою OOo. Надеюсь, это будет исправлено в версии OOo 2.0 .*

III.8.5. Как разрешить макросы с помощью LoadComponentFromURL

Если документ загружен с помощью макроса, то содержащиеся в документе макросы будут запрещены к выполнению. Это требование безопасности. Начиная с версии OOo 1.1, можно разрешить выполнение макросов, когда документ уже загружен. Установите свойство «MacroExecutionMode» равным 2 или 4 и макросы будут работать. Благодарю Mikhail Voitenko [Mikhail.Voitenko@Sun.com] и участников рассылки:
http://www.openoffice.org/servlets/ReadMsg?msgId=782516&listName=dev

Краткое содержание ответа:

Свойство MacroExecutionMode объекта MediaDescriptor использует значения из констант com.sun.star.document.MacroExecMode. Если они не заданы, то поведением по умолчанию будет запрет выполнения макросов. Допустимые значения констант: NEVER_EXECUTE (никогда не выполнять), FROM_LIST (выполнять только из списка), ALWAYS_EXECUTE (всегда выполнять), USE_CONFIG (согласно заданной конфигурации), ALWAYS_EXECUTE_NO_WARN (всегда выполнять без предупреждений),

USE_CONFIG_REJECT_CONFIRMATION (использовать конфигурацию, прерываясь в случае запроса подтверждения)
и USE_CONFIG_APPROVE_CONFIRMATION (использовать конфигурацию, продолжая работу в случае запроса подтверждения).

Есть несколько подводных камней, требующих внимания. Если загружается документ «AsTemplate», то документ не открывается, а создаётся новый. Тогда нужно обрабатывать события, связанные с созданием документа («create document»), а не с открытием документа («open document»). Чтобы все предусмотреть, обрабатывайте оба события.

Лист. 48: Примеры установки свойств описателя носителя данных.

```
Dim oProp(1) As New com.sun.star.beans.PropertyValue
oProp(0).Name="AsTemplate"
oProp(0).Value=True
oProp(1).Name="MacroExecutionMode"
oProp(1).Value=4
```

Этот пример будет работать для макросов, сконфигурированных для события «OnNew» (создание документа), если загрузить шаблон (template) или загрузить sxw (но я не проверял это). Если использовать событие «OnLoad» (открытие документа), то нужно установить свойство «AsTemplate» в значение *False* (или использовать файл sxw, так как по умолчанию это свойство равно *False, но для* stw значение по умолчанию *True*).

В версии OOo 2.0 безопасность макросов будет расширена.

III.8.6. Обработка ошибок при загрузке документа

Если документ не удалось загрузить, то выводится сообщение с информацией о неудачной загрузке. Если документ загружался из C++, то, возможно, ни одна обработка ошибок не включается, поэтому не удаётся выяснить наличие и причины ошибки.

Mathias Bauer объяснил, что интерфейс XcomponentLoader не может устанавливать обработку произвольных исключений, поэтому нужно использовать концепцию «Interaction Handler». Когда документ загружен с помощью loadComponentFromURL, в массиве аргументов передаётся «InteractionHandler» (обработчик прерываний). Графический пользовательский интерфейс GUI обеспечивает основанный на пользовательском интерфейсе UI обработчик прерываний, который преобразует ошибки в доступный пользователю вид, такой как вывод на экран сообщения об ошибке или запрос пароля (несколько примеров есть в Руководстве Разработчика Developer's Guede). Если Обработчик прерываний Вами не указан, то используется обработчик по умолчанию. Обработчик по умолчанию перехватывает все прерывания и переназначает некоторые из них, которые могут быть обработаны из loadComponentFromURL. Хотя с помощью языка Basic невозможно вставить

Ваш собственный обработчик прерываний, для других языков есть несколько примеров в Руководстве разработчика.

III.8.7. Пример использования Mail Merge для объединения всех документов в папке

Утилита mail merge создаёт отдельный документ для каждой сливаемой записи. Приводимая здесь утилита извлекает все документы OpenOffice Writer из данной папки и создаёт единый выходной файл, который содержит все документы, объединённые в один документ. Я модифицировал первоначальный макрос так, что все переменные явно определены, и макрос работает даже в случае, когда первый найденный файл не является документом OpenOffice Write.

Лист. 49: Объединение всех документов из папки в один документ.

```
'author: Laurent Godard
'Modified by: Andrew Pitonyak
Sub MergeDocumentsInDirectory()
'  On Error Resume Next
  Dim DestDirectory As String
  Dim FileName As String
  Dim SrcFile As String, DstFile As String
  Dim oDesktop, oDoc, oCursor, oText
  Dim argsInsert()
  Dim args()
  'Удалите ниже символ комментария, чтобы выполнять в "скрытом" режиме
  'Dim args(0) As New com.sun.star.beans.PropertyValue
  'args(0).name="Hidden"
  'args(0).value=true

  ' В какую директорию (папку) поместить результат?
  DestDirectory=Trim(GetFolderName())

  If DestDirectory = "" Then
    MsgBox "Не выбрана папка для результата, выход",16,"Объединение
документов"
    Exit Sub
  End If

  REM  Принудительно ставим в конце обратный слэш \
  REM  Это нормально, так как используется запись адресов в виде URL
  If Right(DestDirectory,1) <> "/" Then
    DestDirectory=DestDirectory & "/"
  End If

  oDeskTop=CreateUnoService("com.sun.star.frame.Desktop")

  REM  Читаем первый файл!
  FileName=Dir(DestDirectory)
  DstFile = ConvertToURL(DestDirectory & "ResultatFusion.sxw")
  Do While FileName <> ""
    If lcase(right(FileName,3))="sxw" Then
```

```
      SrcFile = ConvertToURL(DestDirectory & FileName)
      If IsNull(oDoc) OR IsEmpty(oDoc) Then
        FileCopy( SrcFile, DstFile )
        oDoc=oDeskTop.Loadcomponentfromurl(DstFile, _
                                           "_blank", 0, Args())
        oText = oDoc.getText
        oCursor = oText.createTextCursor()
      Else
        oCursor.gotoEnd(false)
        oCursor.BreakType = com.sun.star.style.BreakType.PAGE_BEFORE
        oCursor.insertDocumentFromUrl(SrcFile, argsInsert())
      End If
    End If
    FileName=dir()
  Loop

  If IsNull(oDoc) OR IsEmpty(oDoc) Then
    MsgBox "Ни один исходный документ не обработан!",16,"Объединение
документов"
    Exit Sub
  End If

  ' Сохранить документ
  Dim args2()
  oDoc.StoreAsURL(DestDirectory & "ResultatFusion.sxw",args2())
  If HasUnoInterfaces(oDoc, "com.sun.star.util.XCloseable") Then
    oDoc.close(true)
  Else
    oDoc.dispose()
  End If

  'Повторно загружаем этот документ!
  oDoc=oDeskTop.Loadcomponentfromurl(DstFile,"_blank",0,Args2())
End Sub
```

III.9. Загрузка / Вставка графики в документ

Эту простую задачу трудно выполнить, если Вы не знаете, что вставляемый объект должен быть создан с помощью createInstance(«object») из данного документа. Графический объект вставляется как ссылка внутрь документа. Следующий макрос вставляет текстовый графический объект как ссылку в существующий документ. Это текстовый графический объект, который вставляется в позицию курсора. Хотя Вы можете задать размер графического объекта, Вам не требуется задавать его позицию.

Лист. 50: Вставка GraphicsObject (графического объекта) в документ как ссылки.

```
Sub InsertGraphicObject(oDoc, sURL$)
  REM Author: Andrew Pitonyak
  Dim oCursor
  Dim oGraph
  Dim oText
```

```
  oText = oDoc.getText()
  oCursor = oText.createTextCursor()
  oCursor.goToStart(FALSE)
  oGraph = oDoc.createInstance("com.sun.star.text.GraphicObject")

  With oGraph
    .GraphicURL = sURL
    .AnchorType = com.sun.star.text.TextContentAnchorType.AS_CHARACTER
    .Width = 6000
    .Height = 8000
  End With

  ' теперь вставляем изображение в текстовый документ
  oText.insertTextContent( oCursor, oGraph, False )
End Sub
```

Можно также вставить форму графического объекта (graphics object shape), которая вставляется в страницу рисунка (draw page), а не в позицию курсора. Поэтому обязательно нужно указать и расположение, и размер.

Лист. 51: Вставляем объект GraphicsObjectShape в страницу рисунка (draw page).

```
Sub InsertGraphicObjectShape(oDoc, sURL$)
  REM Author: Andrew Pitonyak
  Dim oSize As New com.sun.star.awt.Size
  Dim oPos As New com.sun.star.awt.Point
  Dim oGraph

  REM Сначала создаем форму графического объекта (graphic object shape)
  oGraph = oDoc.createInstance("com.sun.star.drawing.GraphicObjectShape")

  REM  Размер и положение этого графического объекта
  oSize.width=6000
  oSize.height=8000
  oGraph.setSize(oSize)

  oPos.X = 2540
  oPos.Y = 2540
  oGraph.setposition(oPos)

  REM  Предполагаем, что находимся в текстовом документе, и добавляем одну
страницу рисунка (draw page)
  oDoc.getDrawpage().add(oGraph)

  REM  Устанавливаем адрес URL  для этого графического объекта
  oGraph.GraphicURL = sURL
End Sub
```

> ***Совет*** *Вставленный графический объект может содержаться в документе, или находиться вне документа. В обоих случаях GraphicURL всегда ссылается на этот графический объект. Если*

объект не вставлен как ссылка, то URL описывается текстом (stars with the text) «vnd.sun.star.GraphiObject:» Графические объекты, вставленные с использованием API, вставляются как ссылки — они не включаются в сам документ. Danny Brewer, тем не менее, показал, как обойти это, сейчас это будет кратко описано.

III.9.1. Преобразование графического объекта как ссылки во встроенный графический объект.

Чтобы вставить встроенный графический объект в документ, сначала такой объект должен быть вставлен как ссылка и затем изменён на встроенный объект. К сожалению, я знаю, как это сделать для рисунка, но не для текстового графического объекта. Это обидно, так как я предпочитаю текстовую графику в документах OpenOffice Writer, с которой я могу создать якорь (установить перекрёстную ссылку) как свойство символа(ов). Следующий макрос используется в процессе перебора текстового содержимого, для преобразования каждого графического объекта «как ссылки» во встроенный графический объект.

Лист. 52: Вставить ссылку GraphicsObjectShape в страницу рисунка.

```
Sub EmbedLinkedGraphic(oGraph)
  REM Author: Andrew Pitonyak
  Dim sGraphURL As String
  Dim oGraph_2
  Dim oCurs
  Dim oText
  Dim oAnchor
  Dim s$

  If InStr(oGraph.GraphicURL, "vnd.sun") <> 0 Then
    REM  Игнорировать изображение, которое уже вставлено как встроенное
    Exit Sub
  End If
  s = "com.sun.star.drawing.GraphicObjectShape"
  If oGraph.supportsService(s) Then
    REM  Я знаю, как преобразовать только GraphicObjectShape.
    REM   Я не знаю, как преобразовать TextGraphicObject,
    REM  но, вероятно, это связано с атрибутом ImageMap.
    oAnchor = oGraph.getAnchor()
    oText = oAnchor.getText()

    oGraph_2 = ThisComponent.createInstance(s)
    oGraph_2.GraphicObjectFillBitmap = oGraph.GraphicObjectFillBitmap
    oGraph_2.Size = oGraph.Size
    oGraph_2.Position = oGraph.Position
```

```
    oText.insertTextContent(oAnchor, oGraph_2, False)
    oText.removeTextContent(oGraph)
  End If
End Sub
```

III.9.2. Danny Brewer вставляет встроенный графический объект

Danny Brewer объяснил, как загрузить точечный рисунок в документ и затем получить адрес URL загруженного рисунка. Сервис com.sun.star.drawing.BitmapTable загружает изображение.

Лист. 53: Вставка изображения во внутренний список точечных рисунков (internal bitmap table).

```
REM  Задан адрес URL внешнего графического источника,
REM   загружаем этот графический объект постоянно в данный документ рисунка
(drawing document),
REM   а затем возвращаем новый адрес URL  уже ногого встроенного
графического объекта.
REM  Этот новый адрес URL можно использовать вместо старого адреса URL.
Function LoadGraphicIntoDocument( oDoc As Object, cUrl$, cInternalName$ ) As
String
  Dim oBitmaps
  Dim cNewUrl As String

  '  Получаем  список точечных рисунков (BitmapTable) из данного документа
  '  рисунка  (drawing document) Это сервис, который создаёт список точечных
  '  изображений, которые встроены в документ.
  oBitmaps = oDoc.createInstance( "com.sun.star.drawing.BitmapTable" )

  '  Добавляем внешний графический объект к списку точечных рисунков
  '  (BitmapTable) данного документа
  oBitmaps.insertByName( cInternalName, cUrl )

  '  Теперь запрашиваем его обратно Now ask for it back.
  '  Обратно получаем другой адрес Url, который указывает на графический
  ' объект, который находится внутри данного документа и сохраняется вместе
с документом
  cNewUrl = oBitmaps.getByName( cInternalName )

  LoadGraphicIntoDocument = cNewUrl
End Function
```

Загрузка точечного изображения в документ не выводит рисунок на экран. Внутреннее точечное изображение может использоваться или согласно Лист. 54 или Лист. 55 для вставки графического объекта, который хранится внутри документа.

Следующий макрос вставляет заданный объект двумя разными методами. Оба метода работают, в зависимости от того, хотите ли Вы получить встроенный графический объект или ссылку. Заметьте, что oBitmaps.getByName(«DBGif») возвращает адрес URL встроенного

изображения в форме vnd.sun.star.GraphicObject:<big hex number (длинное 16-чное число)>.

Лист. 54: Вставить можно как внешнюю или как внутреннюю ссылку.

```
  Dim sURL$
  Dim oBitmaps
  Dim s$

  REM  Вставляем ссылку на внешний файл
  sURL = "file:///andrew0/home/andy/db.gif"
  InsertGraphicObject(ThisComponent, sURL)

  REM  Вставляем ссылку на встроенный графический объект
  S = "com.sun.star.drawing.BitmapTable"
  oBitmaps = ThisComponent.createInstance( s )
  LoadGraphicIntoDocument(ThisComponent, sURL, "DBGif")
  InsertGraphicObject(ThisComponent, oBitMaps.getByName("DBGif"))
```

III.9.3. Вставить графический объект, используя диспетчер (dispatcher — запуск на выполнение)

Диспетчер легко может встроить графический объект в документ.

Лист. 55: Встроить графический объект в документ.

```
Dim oFrame
Dim oDisp
Dim oProp(1) as new com.sun.star.beans.PropertyValue

oFrame = ThisComponent.CurrentController.Frame
oDisp = createUnoService("com.sun.star.frame.DispatchHelper")

oProp(0).Name = "FileName"
oProp(0).Value = "file:///<YOURPATH>/<YOURFILE>"
oProp(1).Name = "AsLink"
oProp(1).Value = False
oDisp.executeDispatch(oFrame, ".uno:InsertGraphic", "", 0, oProp())
```

III.9.4. Встроить графический объект прямым методом

Следующий метод требует версии OOo 2.0 или более поздней. Я не буду объяснять, как он работает, но он определённо полезен.

Лист. 56: Встроить графический объект в документ.

```
Sub EmbedGraphic(oDoc, sURL$)
  REM Author: Stephan Wunderlich
  Dim oShape
  Dim oGraph     'Заданный графический объект в текстовом содержимом
  Dim oProvider  ' Сервис GraphicProvider
  Dim oText
  Dim s$
```

```
  s = "com.sun.star.drawing.GraphicObjectShape"
  oShape = oDoc.createInstance(s)
  oGraph = oDoc.createInstance("com.sun.star.text.GraphicObject")

  oDoc.getDrawPage().add(oShape)

  oProvider = createUnoService("com.sun.star.graphic.GraphicProvider")

  Dim oProps(0) as new com.sun.star.beans.PropertyValue
  oProps(0).Name  = "URL"
  oProps(0).Value = sURL

  oShape.Graphic = oProvider.queryGraphic(oProps())
  oGraph.graphicurl = oShape.graphicurl
  oText= oDoc.getText()

  ' Вставляем в текущее положение курсора
  Dim oVC : oVC = oDoc.getCurrentController().getViewCursor()
  oText.insertTextContent(oVC, oGraph, false)

  ' Нам больше не нужен объект формы (shape object)
  oDoc.getDrawPage().remove(oShape)
End Sub
```

III.10. Установка полей

Следующий макрос предполагает текстовый документ OOo Writer. Документы типов Draw и Impress используют метод BorderLeft страницы рисунка (draw page).

Лист. 57: Установить размеры полей с помощью модификации стиля текста (конкретно, стиля страницы).

```
Sub Margins( )
  Dim oStyleFamilies, oFamilies, oPageStyles, oStyle
  Dim oVCurs, oPageStyleName
  Dim fromleft%, fromtop%, fromright%, frombottom%
  Dim oDoc
  oDoc = ThisComponent

  REM  Не обязательно использовать видимый курсор (view cursor),
  REM  подойдет любой текстовый курсор (TextCursor)
  oVCurs = oDoc.CurrentController.getViewCursor()
  oPageStyleName = oVCurs.PageStyleName
  oPageStyles = oDoc.StyleFamilies.getByName("PageStyles")
  oStyle  = oPageStyles.getByName(oPageStyleName)
  REM fromleft, fromtop, fromright, frombottom =  все, что Вам нужно
  oStyle.LeftMargin = fromleft
  oStyle.TopMargin = fromtop
  oStyle.RightMargin = fromright
  oStyle.BottomMargin = frombottom
End Sub
```

Чтобы удалить поля, которые были ранее изменены вручную, установите ещё свойства ParaFirstLineIndent и ParaLeftMargin равными нулю.

III.11. Установка размера бумаги

Установка размера бумаги автоматически устанавливает тип бумаги.

Лист. 58: Установка размера страницы с помощью свойств Width и Height.

```
Sub SetThePageStyle()
  Dim oStyle
  Dim sPageStyleName$
  Dim oDoc
  Dim s$
  Dim oVC

  oDoc = ThisComponent

  REM  Не обязателен видимый курсор, подойдет и любой текстовый курсор
  oVC = oDoc.getCurrentController().getViewCursor()
  sPageStyleName = oVC.PageStyleName
  DIM oPageStyles
  oPageStyles = oDoc.StyleFamilies.getByName("PageStyles")
  oStyle = oPageStyles.getByName(sPageStyleName)
  REM  Проверяем, если был размер бумаги Letter, то устанавливаем размер A4
  If oStyle.Width =  27940 Then
    Print "Устанавливаем размер A4"
    oStyle.Width  = 21000
    oStyle.Height = 29700
  Else
    Print "Устанавливаем размер Letter"
    oStyle.Width  = 21590
    oStyle.Height = 27940
  End If

  REM  Заметьте, что свойства Width и Height те же, что и сохраненные
  REM  в свойстве Size значения. Да, кажется глупым...
  REM  Установка одного из свойств width или height меняет их оба...
  s="Width = " & CStr(oStyle.Width / 2540) & " inches" & CHR$(10)
  s=s&"Height = "&CStr(oStyle.Height / 2540)&" inches" & CHR$(10)
  s=s&"Width = "&CStr(oStyle.Size.Width / 2540) & " inches"&CHR$(10)
  s=s&"Height = "&CStr(oStyle.Size.Height / 2540)&" inches" & CHR$(10)
  MsgBox s, 0, "Page Style " & sPageStyleName
End Sub
```

III.12. Вызов внешней программы (Internet Explorer) с помощью OLE

Используем команду Shell или OleObjectFactory (только для Windows).

Лист. 59: Использование OleObjectFactory для запуска внешнего приложения.

```
Sub using_IE( )
  Dim oleService
  Dim IE
  Dim s$

  s = "com.sun.star.bridge.OleObjectFactory"
  oleService = createUnoService(s)
  IE = oleService.createInstance("InternetExplorer.Application.1")
  IE.Visible = 1
  IE.Navigate("http://www.openoffice.org")
End Sub
```

III.13. Использовать команду Shell для файлов, содержащих пробелы

Смотрите секцию синтаксиса URL! Кратко говоря, используйте %20 везде, где должен быть пробел.

Лист. 60: Обязательно используйте синтаксис URL для пробелов в командах shell.

```
Sub ExampleShell
  Shell("file:///C|/Andy/My%20Documents/oo/tmp/h.bat",2)
  Shell("C:\Andy\My%20Documents\oo\tmp\h.bat",2)
End Sub
```

Методы ConvertToUrl и ConvertFromUrl удобны для преобразований из синтаксиса URL в синтаксис Вашей операционной системы и обратно — используйте эти методы для сбережения Вашего времени.

Третий аргумент функции Shell — это аргумент, передаваемый вызванной программе. Четвёртый аргумент, называемый bSync, определяет, будет ли команда shell ожидать, пока процесс завершится (bSync = True) или управление будет возвращено немедленно (bSync = False). Значение по умолчанию **False**. Если два последовательных оператора Shell не установят bSync равным True, то второй оператор Shell может выполняться до того, как первая команда завершится.

III.14. Чтение и Запись числа в файл

Здесь показано, как читать и записывать строку в текстовый файл. Строка преобразуется в число и увеличивается. Число затем записывается обратно в файл как строка.

Лист. 61: Читать и записывать число в файл.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub Read_Write_Number_In_File
  Dim CountFileName As String, NumberString  As String
```

```
  Dim LongNumber As Long, iNum As Integer

  CountFileName = "C:\Andy\My Documents\oo\NUMBER.TXT"
  NumberString  = "00000000"
  LongNumber = 0

  If FileExists(CountFileName) Then
    ON ERROR GOTO NoFile
    iNum = FreeFile

    OPEN CountFileName for input as #iNum
    LINE INPUT #iNum ,NumberString
    CLOSE #iNum
    MsgBox("Read " & NumberString, 64, "Read")

    NoFile: ' в случае, если произойдет ошибка, попадаем сюда..
    If Err <> 0 Then
      Msgbox("Не удалось прочесть  " & CountFileName, 64, "Ошибка")
      NumberString = "00000001"
    End If
    On Local Error Goto 0
  Else
    Msgbox(CountFileName & " НЕ существует", 64, "Предупреждение")
    NumberString  = "00000001"
  End If

  ON ERROR GOTO BadNumber
  LongNumber = Int(NumberString) ' возвращается одно целое число
  LongNumber = LongNumber + 1
BadNumber:
  If Err <> 0 Then
    Msgbox(NumberString & " не является числом", 64, "Ошибка")
    LongNumber = 1
  End If
  On Local Error Goto 0
  NumberString=Trim(Str(LongNumber))
  While LEN(NumberString) < 8
    NumberString="0"&NumberString
  Wend
  MsgBox("Число равно (" & NumberString & ")", 64, "Информационное
сообщение")
  iNum = FreeFile
  OPEN CountFileName for output as #iNum
  PRINT #iNum,NumberString
  CLOSE #iNum
End Sub
```

III.15. Создать стиль числового формата

Если Вам нужен определённый числовой формат, то Вы можете проверить, существует ли такой, и создать новый, если нет. Более подробно о числовых форматах см. Справку по теме «number formats; formats». Это может быть достаточно сложным.

Лист. 62: Создать стиль числового формата

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Function FindCreateNumberFormatStyle (_
  sFormat As String, Optional doc, Optional locale)
  Dim oDoc As Object
  Dim aLocale As New com.sun.star.lang.Locale
  Dim oFormats As Object
  Dim formatNum As Integer
  oDoc = IIf(IsMissing(doc), ThisComponent, doc)
  oFormats = oDoc.getNumberFormats()
  ' Если Вы выберете вариант запроса типов, то Вам нужно использовать тип:
  'com.sun.star.util.NumberFormat.DATE
  ' Я мог бы установить язык (локаль) из числа значений, которые можно
посмотреть здесь:
  'http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt
  'http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html
  ' Здесь я использую локаль NULL и позволяю макросу использовать то, что он
сам выберет
  ' Сначала посмотрим, существует ли числовой формат, полученный как
аргумент
  If ( Not IsMissing(locale)) Then
    aLocale = locale
  End If
  formatNum = oFormats.queryKey (sFormat, aLocale, TRUE)
  MsgBox "Текущий  числовой формат — это " & formatNum
  ' Если нужный числовой формат не существует, то добавляем его
  If (formatNum = -1) Then
    formatNum = oFormats.addNew(sFormat, aLocale)
    If (formatNum = -1) Then formatNum = 0
    MsgBox "Новый числовой формат — это " & formatNum
  End If
  FindCreateNumberFormatStyle = formatNum
End Function
```

III.16. Посмотреть существующие числовые стили формата

Следующий макрос перебирает существующие числовые стили формата. Номера стилей (коды) и их текстовое представление вставляются в текущий документ. Недостаток этой версии — идёт перебор стилей, основанных на локали (языке OOo). Первоначальная версия макроса просматривала коды стилей от 0 до 1000, игнорируя ошибки. Это делало возможным найти все форматы независимо от локали, но я считаю приведённый ниже макрос более понятным решением.

Лист. 63: Перебрать существующие числовые стили формата (дописывает их в текущую позицию видимого курсора — Прим. пер.).

```
Sub enumFormats()
  'Author : Laurent Godard
  'e-mail : listes.godard@laposte.net
  'Modified : Andrew Pitonyak
```

```
  Dim oText
  Dim vFormats, vFormat
  Dim vTextCursor, vViewCursor
  Dim iMax As Integer, i As Integer
  Dim s$
  Dim PrevChaine$, Chaine$
  Dim aLocale As New com.sun.star.lang.Locale

  vFormats = ThisComponent.getNumberFormats()
  '  RunSimpleObjectBrowser(vFormats)   ' ЭТОЙ ПРОЦЕДУРЫ НЕ НАШЕЛ НИГДЕ!!
работает и без нее!- прим.перев.
  oText = ThisComponent.Text
  vViewCursor = ThisComponent.CurrentController.getViewCursor()
  vTextCursor = oText.createTextCursorByRange(vViewCursor.getStart())
  Dim v
  v = vFormats.queryKeys(com.sun.star.util.NumberFormat.ALL, _
                         aLocale, False)
  For i = LBound(v) To UBound(v)
    vFormat=vFormats.getbykey(v(i))
    chaine=VFormat.FormatString
    If Chaine<>Prevchaine Then
      PrevChaine=Chaine
      chaine=CStr(v(i)) & CHR$(9) & CHR$(9) & chaine & CHR$(10)
      oText.insertString(vTextCursor, Chaine, FALSE)
    End If
  Next
  MsgBox "Закончено"
End Sub
```

III.17. Возвращает массив чисел Фибоначчи

Лист. 64: Вывести массив чисел Фибоначчи.

```
' http://disemia.com/software/openoffice/macro_arrays.html
' Возвращает последовательность чисел Фиббоначчи
' предполагаем, что параметр  nCount >=2  для упрощения программы
Function Fibonacci( nCount As Integer )
  If nCount < 2 Then nCount = 2

  Dim result( 1 to nCount) As Double
  Dim i As Integer

  result( 1) = 0
  result( 2) = 1

  For i = 3 to nCount
    result(i) = result( i — 2) + result( i — 1)
  Next i

  Fibonacci = result()
End Function
```

Как бы я ни писал фамилия Фибоначчи, мне говорили, что я ошибаюсь. Я исследовал это вопрос, и нашёл следующее. Фибоначчи сначала

именовался Леонардо из Пизы, один из величайших европейских математиков средневековья. Леонардо называл себя Фибоначчи — сокращение от латинского «filius Bonacci», что означает «сын Боначчио». Фибоначчи использовал варианты «Bonacci», «Bonaccii» и «Bonacij»; в последнем использовано латинское написание. По-английски сейчас чаще используется Fibonacci, но если взять написание на других языках или более древние тексты, то написание будет другим.

III.18. Вставка текста в месте закладки (Bookmark)

Лист. 65: Вставить текст в месте расположения закладки.

```
oBookMark = oDoc.getBookmarks().getByName("<имяВашейЗакладки>")
oBookMark.getAnchor.setString("То, что Вы хотели вставить")
```

III.19. Сохранение и экспорт документа

Сохранение документа — это просто. Следующий макрос сохранит документ, но сохранить можно только если: документ изменён, документ не «только для чтения» , уже задано расположение файла, куда сохранять этот документ..

Лист. 66: Сохранить документ, если он не изменён и его можно сохранять.

```
If (oDoc.isModified) Then
  If (oDoc.hasLocation and (Not oDoc.isReadOnly)) Then
    oDoc.store()
  End If
End If
```

Если документ нужно сохранить по другому адресу, то нужно установить некоторые свойства для указания, куда и как сохранить документ.

Лист. 67: Сохранить документ по другому адресу расположения файла.

```
Dim oProp(0) As New com.sun.star.beans.PropertyValue
Dim sUrl As String
sUrl = "file:///<Полный/путь/к/новому/документу>"

REM  Поменяйте на True, если хотите перезаписать документ
oProp(0).Name = "Overwrite"
oProp(0).Value = False
oDoc.storeAsURL(sUrl, oProp())
```

Вышеприведённый код программы не экспортирует документ в файл другого типа. Для такого экспортирования должен быть определён специальных фильтр экспорта и установлены все необходимые свойства. Нужно знать имя и расширение файла фильтра экспорта. Список таких фильтров экспорта и импорта находится по ссылке

http://framework.openoffice.org/files/documents/25/897/filter_description.html

и ещё много полезной информации здесь

http://oooconv.free.fr/engine/OOOconv.php.

Особые методы нужны для экспорта графики и всего остального. Чтобы экспортировать с использованием не-графического формата, используйте формы типа приведённого ниже образца макроса.

Лист. 68: Экспорт документа.

```
Dim args2(1) As New com.sun.star.beans.PropertyValue
args2(0).Name = "InteractionHandler"
args2(0).Value = ""
args2(1).Name = "FilterName"
args2(1).Value = "MS Excel 97"            REM  Меняем фильтр экспорта
REM  Используйте правильное расширение файла
oDoc.storeToURL("file:///c|/new_file.xls",args2())
```

Заметьте, что я использовал корректное расширение файла и указал соответствующий фильтр импорта. Графические документы обрабатываются иначе. Во-первых, нужно установить GraphicExportFilter и затем вызвать его для экспорта по одной странице за один раз.

Лист. 69: Экспорт документа с использованием GraphicExportFilter.

```
oFilter=CreateUnoService("com.sun.star.drawing.GraphicExportFilter")
Dim args3(1) As New com.sun.star.beans.PropertyValue
For i=0 to oDoc.drawpages.getcount()-1
  oPage=oDoc.drawPages(i)
  oFilter.setSourceDocument(opage)
  args3(0).Name = "URL"
  nom=opage.name
  args3(0).Value =  "file:///c|/"&oPage.name&".JPG"
  args3(1).Name = "MediaType"
  args3(1).Value = "image/jpeg"
  oFilter.filter(args3())
Next
```

> ***Совет*** *Интернет-ссылки на фильтры экспорта и импорта часто меняются, но, вероятно, можно найти их на Интернет-сайте* *http://framework.openoffice.org*

Они также доступны в формате XML в файле:

<ПапкаУстановкиOpenOffice>\share\registry\data\org\openoffice\Office\TypeDetection.xcu

III.20. Поля пользователя

Я много занимался полями пользователя, так что если я и не все знаю, то все же могу использовать их. Большинство пользователей предпочтёт использовать Мастера полей (Master Fields). Эти поля позволяют установить нужные Вам имена и значения.

III.21. Информация о документе

Это четыре поля с именами «Info 1», «Info 2», «Info 3», и «Info 4». Я сам ими не пользовался, но упоминаю их, потому что Вы можете их использовать. Вы можете присвоить им нужные Вам имена и значения..

Лист. 70: Вывод информации о документе.

```
' Доступ к пользовательским полям для получения информации
' о данном документе
vInfo = vDoc.getDocumentInfo()
vVal = oData.ElementNames
s = "=== Поля пользователя ==="
For i = 0 to vInfo.GetUserFieldCount() — 1
  sKey = vInfo.GetUserFieldName(i)
  sVal = vInfo.GetUserFieldValue(i)
  s = s & Chr$(13) & "(" & sKey & "," & sVal & ")"
Next i
'(Info 1,)
'(Info 2,)
'(Info 3,)
'(Info 4,)
MsgBox s, 0, "Пользовательские поля"
```

Общая информация о документе, такая как автор, дата создания и т. д. также доступна в информационном объекте документа.

Лист. 71: Вывод дополнительной информации о документе.

```
Sub InspectDocumentProperties
  REM Author: Andrew Pitonyak
  REM  Некоторые типы свойств документа не могут быть напрямую преобразованы
  REM  в строку, поэтому я использую On Error Resume Next. На дату 26 июля
  REM  2005 г.в версии OOo 2.0 beta, все свойства, которые не удалось
обработать, были структураами даты
  REM  поэтому они не очень необходимы сейчас
  On Error Resume Next

  Dim oPropValues()  ' Массив значений свойств
  Dim i%             ' Переменная индекса, глобальная (General)
  Dim s$             ' Переменная строковая, глобальная (General

  REM  получить значения свойств из объекта информации о документе
  REM  Это — массив значений свойств
```

```
  oPropValues() = ThisComponent.getDocumentInfo().getPropertyValues()
  For i=LBound(oPropValues()) To UBound(oPropValues())
    s = s & oPropValues(i).Name

    REM Значение свойства является структурой, поэтому, возможно, это
    REM  структура даты (Date)
    If IsUnoStruct(oPropValues(i).Value) Then
      REM  Если имя структуры содержит "Date",  то предполагаем, что это
объект даты
      If InStr(oPropValues(i).Name, "Date") > 0 Then
        With oPropValues(i).Value
          s = s & " : " & .Month & "/" & .Day & "/" & .Year & _
              " " & .Hours & ":" & .Minutes & ":" & .Seconds & _
              "." & .HundredthSeconds
        End With
      Else
        REM  Это — неизвестное значение свойства
        s = s & "?:?"
      End If
    Else
      REM  Свойство не является структурой. Данный оператор даст ошибку,
если
      REM  объект не преобразуется в строку автоматически
      on error resume next  ' добавлено переводчиком
      s = s  & ":" & oPropValues(i).Value
      on error goto 0   ' добавлено переводчиком
    End If
    s = s & CHR$(10)
  Next
  MsgBox s
End Sub
```

III.22. Текстовые поля

В следующем макросе (автор Heike Talhammer) показано, как перебрать все текстовые поля, содержащиеся в документе. Макрос устанавливает значения полей и затем использует обновление (refresh) для изменения их значений. Если Вы пропустили эти слова, я повторю: «Этот макрос меняет значение каждого поля, содержащегося в документе».

Лист. 72: Перебрать текстовые поля.

```
'Author: Heike Talhammer <info@bios-pro.de>
'Modified: Andrew Pitonyak
Sub EnumerateFields
  Dim vEnum
  Dim vVal
  Dim s1$, s2$
  Dim sFieldName$, sFieldValue$, sInstanceName$, sHint$, sContent$

  vEnum = thisComponent.getTextFields().createEnumeration()
  If Not IsNull(vEnum) Then
    Do While vEnum.hasMoreElements()
```

```
      vVal = vEnum.nextElement()
      If vVal.supportsService("com.sun.star.text.TextField.Input") Then
        sHint=vVal.getPropertyValue("Hint")
        sContent=vVal.getPropertyValue("Content")
        s1=s1 &"Hint:" & sHint & " — Content: " & sContent & chr(13)
        ' меняем содержимое (content)
        vVal.setPropertyValue("Content", "My new content")
        ThisComponent.TextFields.refresh()
      End If

      If vVal.supportsService("com.sun.star.text.TextField.User") Then
        sFieldName  =vVal.textFieldMaster.Name
        sFieldValue = vVal.TextFieldMaster.Value
        sInstanceName= vVal.TextFieldMaster.InstanceName
        s2 = s2 & sFieldName & " = " & sFieldValue & chr(13) &
"InstanceName: " & _
             sInstanceName & chr(13)
        ' новое значение для текстового поля (textfield)
        vVal.TextFieldMaster.Value=25
      End If
    Loop
    MsgBox s1, 0, "=== Поля ввода (Input) ==="
    MsgBox s2, 0, "===  Поля пользователя ==="
  End If
  ThisComponent.TextFields.refresh()
End Sub
```

III.23. Мастер-поля (Master)

Мастер-поля удобны, можно присвоить им нужные Вам значения, формулы или числовые значения. Здесь о них говорится кратко, но этого будет достаточно для начала самостоятельного изучения. Я нашёл пять типов мастер-полей: Illustration (иллюстрация), Table (таблица), Text (текст), Drawing (чертёж), и User (пользователь). Все имена этих полей начинаются с «com.sun.star.text.FieldMaster.SetExpression.» затем тип и, наконец, ещё точка и имя. Вот простой путь создать или изменить текстовое поле.

Лист. 73: Получить мастер-поле.

```
vDoc = ThisComponent
sName = "Author Name"
If
vDoc.getTextFieldMasters().hasByName("com.sun.star.text.FieldMaster.User." &
sName) Then
  vField =
vDoc.getTextFieldMasters().getByName("com.sun.star.text.FieldMaster.User." &
sName)
  vField.Content = "Andrew Pitonyak"
  'vField.Value = 2.3  REM  Это если Вы предпочитаете числовой код для поля!
Else
  vField = vDoc.createInstance("com.sun.star.text.FieldMaster.User")
  vField.Name = sName
```

```
  vField.Content = "Andrew Pitonyak"
  'vField.Value = 2.3  REM Это если Вы предпочитаете числовой код для поля!
End If
```

Следующий макрос выводит все мастер-поля документа.

Лист. 74: Вывести все мастер-поля.

```
Sub FieldExamples
  Dim vDoc, vInfo, vVal, vNames
  Dim i%, sKey$, sVal$, s$
  vDoc = ThisComponent
  Dim vTextFieldMaster
  Dim sUserType$
  sUserType = "com.sun.star.text.FieldMaster.User"

  vVal = vDoc.getTextFieldMasters()
  vNames = vVal.getElementNames()
  ' Получаем имена такие как:
  'com.sun.star.text.FieldMaster.SetExpression.Illustration
  'com.sun.star.text.FieldMaster.SetExpression.Table
  'com.sun.star.text.FieldMaster.SetExpression.Text
  'com.sun.star.text.FieldMaster.SetExpression.Drawing
  'com.sun.star.text.FieldMaster.User
  s = "=== Мастер-поля текстового документа ==="
  For i = LBound(vNames) to UBound(vNames)
    sKey = vNames(i)
    s = s & Chr$(13) & "(" & sKey
    vTextFieldMaster = vVal.getByName(sKey)
    If Not IsNull(vTextFieldMaster) Then
      s = s & "," & vTextFieldMaster.Name
      ' Я не проверял, что это нужный случай!
      If (Left$(sKey,Len(sUserType)) = sUserType) Then
        ' Типы User таже имеют свойство Value (double) и можно узнавать, что
они являются выражениями!
        'http://api.openoffice.org/docs/common/ref/com/sun/star/text/FieldMa
ster/User.html
        s = s & "," & vTextFieldMaster.Content
      End If
    End If
    s = s & ")"
  Next i
  MsgBox s, 0, "Мастер-поля текста документа (Text)"
End Sub
```

Следующие процедуры (автор Rodrigo V Nunes [rodrigo.nunes@net-linx.com]) показывают, что в OOo можно установить переменные документа (как и в документах MS Word).

Лист. 75: Установка значений переменных документа.

```
' CountDocVars – процедура используется для подсчета переменных документа,
' доступных для текущего документа
'  получает аргументы:
```

```
'  —  DocVars: массив для сохранения переменных текущего документа (имен),
' присутствующих в the ad
' —  DocVarValue: массив для сохранения переменных текущего документа
' (значений), присутствующих в ad
' возвращает значение — целое (integer), содержащее общее число найденных
' переменных  пользователя в документе
Function CountDocVars(DocVars , DocVarValue) As Integer
  Dim VarCount As Integer
  Dim Names as Variant

  VarCount = 0
  Names = thisComponent.getTextFieldMasters().getElementNames()
  For i%=LBound(Names) To UBound(Names)
    if (Left$(Names(i%),34) = "com.sun.star.text.FieldMaster.User") Then
      xMaster = ThisComponent.getTextFieldMasters.getByName(Names(i%))
      DocVars(VarCount) = xMaster.Name
      DocVarValue(VarCount) = xMaster.Value
      VarCount = VarCount + 1    '  это переменная документа, созданная
                                 ' пользователем
    End if
  Next i%

  CountDocVars = VarCount
End Function
'
' SetDocumentVariable — процедура, используемая для создания или установки
' значения переменной документа в списке текстовых полей пользователя
' в документе, без физической вставке содержимого списка в текст  активного
' документа.
' на входе параметры:
' strVarName: строка с именем переменной, которую нужно создать или
' установить значение
' aValue: строка со значением переменной документа
' возвращает значение —  булевское (flag), показывающее статус операции:
' TRUE=OK, FALSE=переменная не может быть установлена или создана
Function SetDocumentVariable(ByVal strVarName As String, ByVal aValue As
String ) As Boolean
Dim bFound As Boolean

 On Error GoTo ErrorHandler
 oActiveDocument = thisComponent
 oTextmaster = oActiveDocument.getTextFieldMasters()
 sName = "com.sun.star.text.FieldMaster.User." + strVarName
 bFound = oActiveDocument.getTextFieldMasters.hasbyname(sName)  ' проверяем,
существует ли переменная
  if bFound Then
    xMaster = oActiveDocument.getTextFieldMasters.getByName( sName )
    REM  значение MEMBER  используется для десятичных значений, CONTENT для
строк
    'xMaster.value = aValue
    xMaster.Content = aValue
  Else        ' переменная документа ещё не существует
    sService = "com.sun.star.text.FieldMaster.User"
    xMaster = oActiveDocument.createInstance( sService )
    xMaster.Name = strVarName
```

```
    xMaster.Content = aValue
  End If
  SetDocumentVariable = True      ' Успех
  Exit Function

ErrorHandler:
  SetDocumentVariable = False
End Function

' InsertDocumentVariable — процедура, используемая для вставки переменной
' документа в список текстовых полей пользователя для данного документа
'  и в текст активного документа в позиции курсора
' получает аргументы:
' — strVarName: строка с именем переменной, которую нужно вставить
' — oTxtCursor: текущий объект курсора, в чью позицию нужно вставить
' переменную документа
' Выводит- ничего
Sub InsertDocumentVariable(strVarName As String, oTxtCursor As Object)

  oActiveDocument = thisComponent
  objField =
thisComponent.createInstance("com.sun.star.text.TextField.User")
  sName = "com.sun.star.text.FieldMaster.User." + strVarName
  bFound = oActiveDocument.getTextFieldMasters.hasbyname(sName)  '
проверяем, существует ли переменная
  if bFound Then
    objFieldMaster =  oActiveDocument.getTextFieldMasters.getByName(sName)
    objField.attachTextFieldMaster(objFieldMaster)
    ' вставить текстовое поле
    oText = thisComponent.Text
    'oCursor = oText.createTextCursor()
    'oCursor.gotoEnd(false)
    oText.insertTextContent(oTxtCursor, objField, false)
  End If
End Sub

' DeleteDocumentVariable — процедура, используемая для удаления переменной
' документа из списка текстовых полей пользователя
' на входе параметры
' — strVarName: строка с именем переменной, которую нужно удалить
' Возвращает — ничего

Sub DeleteDocumentVariable(strVarName As String)
  oActiveDocument = thisComponent
  objField =
oActiveDocument.createInstance("com.sun.star.text.TextField.User")
  sName = "com.sun.star.text.FieldMaster.User." + strVarName
  bFound = oActiveDocument.getTextFieldMasters.hasbyname(sName)  '
проверяем, существует ли переменная
  if bFound Then
    objFieldMaster =  oActiveDocument.getTextFieldMasters.getByName(sName)
    objFieldMaster.Content = ""
    objFieldMaster.dispose()
  End If
End Sub
```

```
' SetUserVariable —  функция, используемая для создания или установки
' значения переменной пользователя в данном документе.
'  Эти переменные предназначены только для внутренних нужд системы  и НЕ
' будут доступны или пригодны для использования в приложениях  java (см.
'SetDocumentVariables' для создания или установки переменных общего доступа
' документа получает параметры:
'  — strVarName: строка с именем переменной, чье значение нужно установить.
'   Если переменная не существует, она будет создана.
'  — avalue:     значение типа Variant с новым содержимым (content)
' переменной, имя которой задано параметром strVarName
' возвращает значение: булевское (flag) со статусом операции: TRUE=OK,
FALSE=переменная не может быть создана или установлено значение

Function SetUserVariable(ByVal strVarName As String, ByVal avalue As
Variant) As Boolean

Dim aVar As Variant
Dim index As Integer                ' индекс имени существующей переменной
Dim vCount As Integer

  On Error GoTo ErrorHandler
  ' Посмотрим, существует ли уже переменная документа
  oDocumentInfo = thisComponent.Document.Info
  vCount = oDocumentInfo.getUserFieldCount()
  bFound = false
  For i% = 0 to (vCount — 1)
    If strVarName = oDocumentInfo.getUserFieldName(i%) Then
      bFound = true
      oDocumentInfo.setUserFieldValue(i%,avalue)
    End If
  Next i%
  If not bFound Then                ' Переменная документа ещё не существует
    oDocumentInfo.setUserFieldName(i,strVarName)
    oDocumentInfo.setUserValue(i,avalue)
  End If
  ' проверьте, не будет ли значение i больше, чем число переменных
пользователя !

  SetUserVariable = True      ' Успех
  Exit Function

ErrorHandler:
  SetUserVariable = False
End Function
```

Christian Junker написал следующий макрос для вызова и тестирования вышеприведённой процедуры:

Лист. 76: Использование процедуры из Лист. 75

```
'Это код программы для вызова вышеприведенных процедур:
Sub Using_docVariables()

 odoc = thisComponent
```

```
 otext = odoc.getText()
 ocursor = otext.createTextCursor()
 ocursor.goToStart(false)

 SetDocumentVariable("docVar1", "Value 1") ' создаем свою переменнную
документа
 InsertDocumentVariable("docVar1", ocursor) ' вставляем её в текущий
документ
End Sub
```

III.24. Удаление текстовых полей

Извлечь текстовое поле и затем аннулировать его (dispose) с использованием field.dispose() !

III.25. Вставка адреса URL в ячейку OOo Calc

По причинам, не зависящим от меня, следующие операции требовалось выполнить много раз. Я решил добавить этот пример, так как я устал выполнять его каждый раз. Макрос InsertURLIntoCell преобразует текст ячейки в адрес URL и вставляет этот адрес URL в текстовое поле в ячейке. Прочтите комментарии, чтобы понять, как это работает.

Лист. 77: Вставить адрес URL в ячейку Calc.

```
Sub InsertURLIntoCell
  Dim oText   ' Текстовый объект для текущего объекта
  Dim oField  ' Поле, которое нужно вставить
  Dim oCell   ' Полученная заданная ячейка

  Rem  Получить ячейку, пока можно любую ячейку. Здесь извлекается ячейка C3
  oCell = ThisComponent.Sheets(0).GetCellByPosition(2,2)

  REM  Создать переменную типа "адрес URL текстового поля"
  oField = ThisComponent.createInstance("com.sun.star.text.TextField.URL")

  REM  Это рекальный текст, который выводится для данного адреса URL
  REM  Это могло бы быть и таким
  REM oField.Representation = "My Secret Text"
  oField.Representation = oCell.getString()

  REM  Свойство URL  — это просто текстовая строка с самим адресом URL
  oField.URL = ConvertToURL(oCell.getString())

  REM  Это текстовое поле добавляется в качестве текстового содержимого
в данную ячейку
  REM  Если здесь Вы не установите для данной строки пустое значение, то
  REM   существующий текст останется и новый URL  текстового поля будет
  REM   добавлен в конец
  oCell.setString("")
  oText = oCell.getText()
         'Это не сработало у меня в OOo Calc версии 2.3.0  (Прим. пер.)
```

```
  'oText.insertTextContent(oText.createTextCursor(), oField, False)
' может быть, имелось в виду такое:
' oText.insertTextContent(oText.getEnd(), oField, False)
  'а вот это работает
  oCell.setString(ConvertToURL(oCell.AbsoluteName))
End Sub
```

> ***Внимание*** *Это не работает в документе OOo Writer, не пробуйте там!*

III.26. Добавление текстового поля — формулы (SetExpression TextField)

Я создаю мои собственные поля формул и использую их для нумерации моих текстовых таблиц, листингов программ, рисунков и любых других вещей, которые нужно пронумеровать последовательно. Следующий макрос предполагает, что Вы знаете, как добавить их вручную, и что уже существует числовое поле с именем Table. Используйте меню Вставка — Поля — Дополнительно (Insert | Fields | Other), чтобы открыть диалог Поля (можно Ctrl+F2). Во вкладке Переменные в окне Тип поля выберите Диапазон нумерации. Обычно я ввёл бы выражение «Table+1», но сейчас я хочу показать использование макроса для добавления.

Лист. 78: Добавить текстовое поле — формулу (SetExpression) в конец документа.

```
Sub AddExpressionField
  Dim oField             ' Это и будет поле-формула, которую будем вставлять
  Dim oMasterField       ' Мастер-поле для типа полей — формул
  Dim sMasterFieldName$ '  Это имя данного мастер-поля
  Dim oDoc               '  Документ, куда вставляется поле
  Dim oText              '  Текстовый объект документа

  oDoc = ThisComponent

  REM  Текстовое поле должно быть создано из того документа, в который оно
будет помещено
  oField = oDoc.createInstance("com.sun.star.text.TextField.SetExpression")

  REM  Задаем формулу
  oField.Content = "Table+1"

  REM  Обычно Вам пожет понадобиться создать или проверить числовой формат
  REM   Вообще-то "=4" — это хитрость, потому что я обычно знаю, что это
  REM    такое, и хочу написать макрос покороче. У меня есть и другие
примеры,
  REM    которые показали бы Вам, как получить перечень числовых форматов
  oField.NumberFormat = 4
  oField.NumberingType = 4
```

```
  REM  Это очень длинное имя. Все мастер-поля именуются подобным образом.
  REM  Используйте это имя для получения мастер-поля, которое мы преполагаем
существующим
  sMasterFieldName = "com.sun.star.text.FieldMaster.SetExpression.Table"
  oMasterField = oDoc.getTextFieldMasters().getByName(sMasterFieldName)

  REM  Добавляем текстовое поле к соответствующему мастер-полю (списку полей
данного вида)
  oField.attachTextFieldMaster(oMasterField)

  REM  Наконец, вставляем это поле в КОНЕЦ данного документа
  oText = oDoc.getText()
  oText.insertTextContent(oText.getEnd(), oField, False)
End Sub
```

III.27. Типы данных, определённые пользователем

Как и в версии OOo 1.1.1, можно определить Ваши собственные типы данных.

Лист. 79: Вы можете определить Ваши собственные типы данных.

```
Type PersonType  ' Задаем новый тип PersonType — структура из двух полей
  FirstName As String
  LastName As String
End Type

Sub ExampleCreateNewType 'Это пример создания экземпляра и присвоения
значений
  Dim Person As PersonType
  Person.FirstName = "Andrew"
  Person.LastName  = "Pitonyak"
  PrintPerson(Person)
End Sub

Sub PrintPerson(x)  ' пример процедуры, работающей с переменной этого типа
  Print "Person = " & x.FirstName & " " & x.LastName
End Sub
```

На конференции по OOo в Берлина 2004 г. я представил презентацию по созданию расширенных типов данных с использованием структур. Эти примеры есть в презентации, находящейся на моем сайте.

III.28. Проверка орфографии, переносы и толковый словарь — тезаурус

Выполнить проверку орфографии, переносы слов, просмотреть тезаурус очень легко. Эти части возвратят пустое (null) значение, если соответствующие части не сконфигурированы. При первоначальном тестировании процедура переноса слов (Hyphenation) всегда возвращала null до тех пор, пока я не настроил переносы через диалог.

(В версии 2.3.0 переносы по слогам русского текста работают, включается Формат — Абзац — Положение на странице и там признак Автоматический перенос; можно и вручную внести переносы в нужных местах: Сервис — Язык — Расстановка переносов..., см. Справка — Расстановка переносов. Проверка орфографии для русского языка, тезаурус для русского языка не работают — Прим. пер.)

Лист. 80: Проверка орфографии, переносы слов и использование тезауруса.

```
Sub SpellCheckExample
  Dim s() As Variant
  Dim vReturn As Variant, i As Integer
  Dim emptyArgs(0) As New com.sun.star.beans.PropertyValue
  Dim aLocale As New com.sun.star.lang.Locale
  'aLocale.Language = "en"
  'aLocale.Country = "US"
  aLocale.Language = "ru"
  aLocale.Country = "RU"
  's = Array("hello", "anesthesiologist",
"PNEUMONOULTRAMICROSCOPICSILICOVOLCANOCONIOSIS", "Pitonyak", "misspell")
  s = Array("Привет", "анестезиолог", "Пнев", "Питоньяк", "АэтоБудетОшибка")

  '*********Пример проверки орфографии — не работает для русского языка!
  'http://api.openoffice.org/docs/common/ref/com/sun/star/linguistic2/XSpell
Checker.html
  Dim vSpeller As Variant
  vSpeller = createUnoService("com.sun.star.linguistic2.SpellChecker")
  'Use vReturn = vSpeller.spell(s, aLocale, emptyArgs()) if you want
options!
  For i = LBound(s()) To UBound(s())
    vReturn = vSpeller.isValid(s(i), aLocale, emptyArgs())
    if vReturn then
     MsgBox "Проверка орфографии успешна для " & s(i) & " — возвратила " &
vReturn
    else
     MsgBox "Проверка орфографии неудачна для " & s(i) & " — возвратила " &
vReturn
    endif
  Next

  '******Пример расстановки переносов — работате и для русского!
  'http://api.openoffice.org/docs/common/ref/com/sun/star/linguistic2/XHyphe
nator.html
  Dim vHyphen As Variant
  vHyphen = createUnoService("com.sun.star.linguistic2.Hyphenator")
  For i = LBound(s()) To UBound(s())
    'vReturn = vHyphen.hyphenate(s(i), aLocale, 0, emptyArgs())
    vReturn = vHyphen.createPossibleHyphens(s(i), aLocale, emptyArgs())
    If IsNull(vReturn) Then
      'hyphenation is probablly off in the configuration
      MsgBox "Попытка расстановки перносов для " & s(i) & " возвратила null
(выключены переносы?)"
    Else
```

```
      MsgBox "Переносы расставлены для строки " & s(i) & " так " &
vReturn.getPossibleHyphens()
    End If
  Next

  '******Пример проверки по тезаурусу (толковому словарю) — не работает для
русского языка!
  'http://api.openoffice.org/docs/common/ref/com/sun/star/linguistic2/XThesa
urus.html
  Dim vThesaurus As Variant, j As Integer, k As Integer
  vThesaurus = createUnoService("com.sun.star.linguistic2.Thesaurus")
    s = Array("Привет", "печать", "холод", "АэтоОшибка")
  For i = LBound(s()) To UBound(s())
    vReturn = vThesaurus.queryMeanings(s(i), aLocale, emptyArgs())
    If UBound(vReturn) < 0 Then
      Print "В тезаурусе ничего не найдено для " & s(i)
    Else
      Dim sTemp As String
      sTemp = "Переносы расставлены для " & s(i)
      For j = LBound(vReturn) To UBound(vReturn)
        sTemp = sTemp & Chr(13) & "Meaning = " & vReturn(j).getMeaning() &
Chr(13)
        Dim vSyns As Variant
        vSyns = vReturn(j).querySynonyms()
        For k = LBound(vSyns) To UBound(vSyns)
          sTemp = sTemp & vSyns(k) & " "
        Next
        sTemp = sTemp & Chr(13)
      Next
      MsgBox sTemp
    End If
  Next
End Sub
```

III.29. Изменение указателя мыши

Если ответить кратко: это не поддерживается. Привожу дискуссию, которую я сам не проверял.

anindya@agere.com спрашивает: пытался изменить указатель мыши на «песочные часы» на время выполнения макроса. Что неправильно в следующем коде программы?

Лист. 81: Нельзя изменить указатель мыши в версии OOo 1.1.3.

```
oDoc = oDeskTop.loadComponentFromURL(fileName,"_blank",0,mArg())
oCurrentController = oDoc.getCurrentController()
oFrame = oCurrentController.getFrame()
oWindow = oFrame.getContainerWindow()
oPointer = createUnoService("com.sun.star.awt.Pointer")
oPointer.SetType(com.sun.star.awt.SystemPointer.WAIT)
oWindow.setPointer(oPointer)
```

Mathias Bauer, которого все мы любим, ответил так: Вы не можете изменить указатель мыши в окне документа с помощью UNO-API. VCL управляет указателем мыши, основываясь на окне, а не на главном окне. Любое окно VCL может иметь свой собственный вид указателя мыши. Если Вы хотите изменить указатель мыши для окна документа, нужно получить доступ к XwindowPeer окна документа (а не окна фрейма), но это недоступно в API. Другая проблема может быть в том, что OOo меняет указатель мыши внутренним механизмом и все Ваши настройки отменяются. (конец цитаты)

Berend Cornelius дал окончательный ответ: Ваша процедура работает правильно для любого под-окна Вашего документа.

Следующий код программы ссылается на управление (control) в документе:

Лист. 82: Сменить указатель мыши на «управление» (control).

```
Sub Main
  GlobalScope.BasicLibraries.LoadLibrary("Tools")
  oController = Thiscomponent.getCurrentController()
  oControl
=oController.getControl(ThisComponent.Drawpage().getbyIndex(0).getControl())
  SwitchMousePointer(oControl.getPeer(), False)
End Sub
```

Эта процедура меняет указатель мыши, когда ей передано управление, но когда указатель не находится внутри окна управления, он меняется обратно. Вы хотите «песочные часы» = функция «Wait», чтобы менять указатель на состояние ожидания, но это сейчас не поддерживается в API. (конец цитаты)

А по моему мнению, изменить указатель мыши можно, но не во всех случаях, так:

```
oDoc.getCurrentController().getFrame().getContainerWindow().se
tPointer(...)
```

III.30. Установка фона страницы (Page Background)

Лист. 83: Установить фон страницы.

```
Sub Main
  ' Сначала получим семейства стилей  Style Families
  oStyleFamilies= ThisComponent.getStyleFamilies()
  ' теперь получим семейство "стили страницы"  PageStyles
  oPageStyles= oStyleFamilies.getByName("PageStyles")
  ' теперь получим Ваш стиль страницы
  oMyPageStyle= oPageStyles.getByName("Standard")
  ' Наконец, устанавливаем Ваш фон (для страницы) с помощью oMyPageStyle
    .BackGraphicUrl= _
      convertToUrl( <pathToYourGraphic> )
```

```
    .BackGraphicLocation= _
      com.sun.star.style.GraphicLocation.AREA
  end with
End Sub
```

III.31. Работа с буфером (clipboard)

Доступ к буферу системы clipboard напрямую не лёгок Обычно доступ выполняется с использованием команд операционной системы (dispatch statements). Чтобы скопировать данные в буфер системы, нужно сначала выделить данные. Необязательный интерфейс Controller com.sun.star.view.XSelectionSupplier предоставляет возможность выбрать объекты и получить доступ к выбранным в данный момент объектам. OOo Writer добавляет методы getTransferable и insertTransferable, которые позволяют копировать выбранные объекты без использования буфера системы. Этот метод доступен, начиная с версии OOo Calc 2.3.

III.31.1. Копировать ячейки электронной таблицы с помощью буфера системы Clipboard

Первый пример, присланный мне, выбирает ячейки электронной таблицы и затем вставляет их в другую электронную таблицу.

Лист. 84: Скопировать и вставить интервал ячеек с использованием буфера системы clipboard.

```
'Author: Ryan Nelson
'email: ryan@aurelius-mfg.com
'Modified By: Christian Junker и Andrew Pitonyak
'Этот макрос копирует интервал и вставляет его в новую или существующую
электронную таблицу spreadsheet.
Sub CopyPasteRange()
  Dim oSourceDoc, oSourceSheet, oSourceRange
  Dim oTargetDoc, oTargetSheet, oTargetCell
  Dim oDisp, octl
  Dim sUrl As String
  Dim NoArg()

  REM Задаем исходный doc/currentController/frame/sheet/range.
  oSourceDoc=ThisComponent
  octl = oSourcedoc.getCurrentController()
  oSourceframe = octl.getFrame()
  oSourceSheet= oSourceDoc.Sheets(0)
  oSourceRange = oSourceSheet.getCellRangeByPosition(0,0,100,10000)

  REM Создаем сервис DispatcherService
  oDisp = createUnoService("com.sun.star.frame.DispatchHelper")

  REM выделяем исходный интервал range
  octl.Select(oSourceRange)

  REM копируем текущее выделение в буфер системы clipboard.
```

```
  oDisp.executeDispatch(octl, ".uno:Copy", "", 0, NoArg())

  REM открываем новую электронную таблицу spreadsheet.
  sURL = "private:factory/scalc"
  oTargetDoc = Stardesktop.loadComponentFromURL(sURL, "_blank", 0, NoArg())
  oTargetSheet = oTargetDoc.getSheets.getByIndex(0)

  REM Можеб быть, Вы захотите сначала очистить интервал, куда будем
вставлять,
  REM если он содержит данные и форматирование.
  REM Передвигаем фокус на (активируем) ячейку 0,0.
  REM Это гарантирует, что перед вставкой активной является ячейка  "0,0"
  REM Вы могли бы сделать активной любую другую ячейку.
  REM Если Вы не выполните такую активацию ячейки, то вставка будет в
  REM позицию ячейки, которая была активной в то время, когда лист был
открыт последний раз.
  oTargetCell = oTargetSheet.getCellByPosition(0,0)
  oTargetDoc.getCurrentController().Select(oTargetCell)

  REM вставляем из буфера системы clipboard в текущую позицию.
  oTargetframe = oTargetDoc.getCurrentController().getFrame()
  oDisp.executeDispatch(oTargetFrame, ".uno:Paste", "", 0, NoArg())
End Sub
```

III.31.2. Скопировать ячейки электронной таблицы без использования буфера системы

Можно скопировать, вставить, переместить и удалить ячейки внутри одного документа OOo Calc без использования буфера системы, и даже между разными листами одного документа. См. подробнее:

`http://api.openoffice.org/docs/common/ref/com/sun/star/sheet/XCellRangeMovement.html`

Следующий код программы получен из рассылки devapi.

Лист. 85: Скопировать и вставить интервал ячеек без использования буфера системы.

```
' Author:  Oliver Brinzing
' email:   OliverBrinzing@t-online.de
Sub CopySpreadsheetRange
  REM Получаем лист 1, исходный, и 2, который станет его копией.
  oSheet1 = ThisComponent.Sheets.getByIndex(0)
  oSheet2 = ThisComponent.Sheets.getByIndex(1)

  REM Получаем интервал, откуда копировать, и интервал, куда копировать.
  oRangeOrg = oSheet1.getCellRangeByName("A1:C10").RangeAddress
  oRangeCpy = oSheet2.getCellRangeByName("A1:C10").RangeAddress

  REM Позиция для вставки
  oCellCpy = oSheet2.getCellByPosition(oRangeCpy.StartColumn,_
    oRangeCpy.StartRow).CellAddress

  REM Выполняем копирование
```

```
  oSheet1.CopyRange(oCellCpy, oRangeOrg)
End Sub
```

К сожалению, этот способ не копирует форматирование и др.

III.31.3. Получение типа содержимого (content-type) буфера системы Clipboard

Я создал следующий пример для демонстрации, как получить прямой доступ к буферу системы clipboard. Вообще-то это непрактично для чего-то другого, кроме прямых операций с текстом.

Лист. 86: Операции с буфером системы clipboard.

```
Sub ConvertClipToText
  REM Author: Andrew Pitonyak
  Dim oClip, oClipContents, oTypes
  Dim oConverter, convertedString$
  Dim i%, iPlainLoc%
  Dim sClipService As String

  iPlainLoc = -1
  sClipService = "com.sun.star.datatransfer.clipboard.SystemClipboard"
  oClip = createUnoService(sClipService)
  oConverter = createUnoService("com.sun.star.script.Converter")

  'Print "Clipboard name = " & oClip.getName()
  'Print "Implemantation name = " & oClip.getImplementationName()
  oClipContents = oClip.getContents()
  oTypes = oClipContents.getTransferDataFlavors()

  Dim msg$, iLoc%, outS
  msg = ""
  iLoc = -1
  For i=LBound(oTypes) To UBound(oTypes)
    If oTypes(i).MimeType = "text/plain;charset=utf-16" Then
      iPlainLoc = i
      Exit For
    End If
    'msg = msg & "Mime type = " & x(ii).MimeType
    'msg = msg & " normal = " & x(ii).HumanPresentableName & Chr$(10)
  Next
  If (iPlainLoc >= 0) Then
    Dim oData
    oData = oClipContents.getTransferData(oTypes(iPlainLoc))
    convertedString = oConverter.convertToSimpleType(oData, _
                                        com.sun.star.uno.TypeClass.STRING)
    MsgBox convertedString
  End If
End Sub
```

III.31.4. Сохранение строки в буфер системы clipboard

Вот интересный пример (автор ms777 с форума oooforum), который демонстрирует, как записать строку в буфер системы.

Лист. 87: Записать строку в буфер системы.

```
Private oTRX

Sub Main
  Dim null As Object
  Dim sClipName As String

  sClipName = "com.sun.star.datatransfer.clipboard.SystemClipboard"
  oClip = createUnoService(sClipName)
  oTRX = createUnoListener("TR_", "com.sun.star.datatransfer.XTransferable")
  oClipContents = oClip.setContents(oTRX, null)
End Sub

Function TR_getTransferData( aFlavor As com.sun.star.datatransfer.DataFlavor
) As Any
  If (aFlavor.MimeType = "text/plain;charset=utf-16") Then
    TR_getTransferData = "From OO with love ..."
  EndIf
End Function

Function TR_getTransferDataFlavors() As Any
  Dim aF As New com.sun.star.datatransfer.DataFlavor
  aF.MimeType =    "text/plain;charset=utf-16"
  aF.HumanPresentableName = "Unicode-Text"
  TR_getTransferDataFlavors = Array(aF)
End Function

Function TR_isDataFlavorSupported( aFlavor As
com.sun.star.datatransfer.DataFlavor ) As Boolean

  'Моя Windows XP сигнализирует звуком (beep), показывая, что эта процедура
вызывается каждые 2 секунды
  'call MyPlaySoundSystem("SystemAsterisk", true)
  TR_isDataFlavorSupported = (aFlavor.MimeType =
"text/plain;charset=utf-16")
End Function
```

III.31.5. Просмотреть буфер системы clipboard как текст

Большинство получают доступ к буферу системы с использованием команд системы (dispatch) через UNO. Тем не менее, иногда нужно получить прямой доступ к буферу системы.. Лист. 88 показывает, как получит доступ к буферу системы как текст.

Лист. 88: Просмотреть буфер системы как текст.

```
Sub ViewClipBoard
  Dim oClip, oClipContents, oTypes
  Dim oConverter, convertedString$
  Dim i%, iPlainLoc%

  iPlainLoc = -1
```

```
  Dim s$ : s$ = "com.sun.star.datatransfer.clipboard.SystemClipboard"
  oClip = createUnoService(s$)
  oConverter = createUnoService("com.sun.star.script.Converter")

  'Print "Clipboard name = " & oClip.getName()
  'Print "Implemantation name = " & oClip.getImplementationName()
  oClipContents = oClip.getContents()
  oTypes = oClipContents.getTransferDataFlavors()

  Dim msg$, iLoc%, outS
  msg = ""
  iLoc = -1
  For i=LBound(oTypes) To UBound(oTypes)
    If oTypes(i).MimeType = "text/plain;charset=utf-16" Then
      iPlainLoc = i
      Exit For
    End If
    'msg = msg & "Mime type = " & x(ii).MimeType & " normal = " & _
    '      x(ii).HumanPresentableName & Chr$(10)
  Next
  If (iPlainLoc >= 0) Then
    convertedString = oConverter.convertToSimpleType( _
         oClipContents.getTransferData(oTypes(iPlainLoc)), _
         com.sun.star.uno.TypeClass.STRING)
    MsgBox convertedString
  End If
End Sub
```

III.31.6. Альтернатива буферу системы clipboard — перемещаемое содержимое (transferable content)

Когда-нибудь OOo Writer введёт getTransferable() и insertTransferable(), которые действуют как «внутренний» clipboard. Следующий макрос использует диспетчер (dispatch) чтобы выбрать документ целиком, создаёт новый документ OOo Writer и затем копирует все содержание текста в новый документ.

Лист. 89: Скопировать текстовый документ с использованием перемещаемого контента.

```
oFrame = ThisComponent.CurrentController.Frame
dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")
dim noargs()
dispatcher.executeDispatch(frame, ".uno:SelectAll", "", 0, noargs())

obj = frame.controller.getTransferable()
sURL = "private:factory/swriter"
doc = stardesktop.loadcomponentfromurl(sURL,"_blank",0,noargs())
doc.currentController.insertTransferable(obj)
```

> ***Примечание*** *Перемещаемое содержимое поддерживается в OOo Calc с версии 2.3.*

III.32. Установка локали (языка)

В OOo символы содержат локаль (locale), которая определяет язык и страну. Я использую стили для форматирования моих примеров кода программ. Я устанавливаю локаль (locale) в стиле кода макроса на неизвестную, так что их орфография не проверяется — если локаль неизвестна, то OOo не знает, какой словарь использовать. Чтобы сообщить OOo, что слово на французском языке, нужно установить локаль этих символов на French. Меня спрашивали, как установить локаль для всего текста документа в заданное значение. Это кажется очевидным поначалу. Курсор поддерживает свойства символов, что позволяет установить локаль. Я создал курсор, выделил документ целиком и установил локаль. Я получил ошибку выполнения. Я обнаружил, что свойство локаль является опцией — оно может быть пустым, если IsEmpty(oCurs.CharLocale) равно true. Хотя моя следующая попытка сработала для моего документа, Вам следует выполнить больше проверок с таблицами и другими объектами. Более надёжным будет использовать перебор, так как при переборе можно просматривать секции, которые все используют те же самые значения свойства, так что Вы можете всегда установить локаль.

Лист. 90: Установить локаль документа.

```
Sub SetDocumentLocale
  Dim oCursor
  Dim aLocale As New com.sun.star.lang.Locale
  aLocale.Language = "fr"
  aLocale.Country = "FR"

  REM Предполагаем, что документ текстовый text document
  REM Получаем компоненту  Text из данного документа
  REM Создаем текстовый курсор Text cursor
  oCursor = ThisComponent.Text.createTextCursor()
  REM Переходим к началу документа
  REM Теперь передвигаемся к концу документа, выбирая так ВЕСЬ текст
  oCursor.GoToStart(False)
  Do While oCursor.gotoNextParagraph(True)
    oCursor.CharLocale = aLocale
    oCursor.goRight(0, False)
  Loop
  Msgbox "успешно установлена французская локаль"
End Sub
```

Желательно предусмотреть строку «On Local Error Resume Next», но я не пробовал это и это может скрыть какие-то ошибки во время первоначального тестирования.

Вы также можете установить локаль для выделенного текста или текста, который найден встроенной процедурой поиска.

III.33. Установки локали для выделенного текста

Следующий макрос демонстрирует немного другой метод установки локали для выделенного текста или для всего документа. Я удалил большинство комментариев. См. секцию о работе с выделенным текстом в текстовом документе. Макрос в Лист. 90 просматривает весь документ, используя выделение параграфов. Я отказался от использования выделения параграфов в Лист. 91, так как выделенный текст может не включать целые параграфы. Основной целью применения этого метода является то, что просмотр документа по одному символу может занять много времени..

Лист. 91: Установить локаль для выделенного текста (или целого документа).

```
Sub MainSetLocale
  Dim oLoc As New com.sun.star.lang.Locale
  'oLoc.Language = "fr"  : oLoc.Country = "FR"
  oLoc.Language = "en"  : oLoc.Country = "US"
  SetLocaleForDoc(ThisComponent, oLoc)
End sub

Sub SetLocaleForDoc(oDoc, oLoc)
  Dim oCurs()
  Dim sPrompt$
  Dim i%

  sPrompt = "Установить язык (locale) равным (" & oLoc.Language & ", " &
oLoc.Country & ")?"
  If NOT CreateSelTextIterator(oDoc, sPrompt, oCurs()) Then Exit Sub
  For i = LBound(oCurs()) To UBound(oCurs())
    SetLocaleForCurs(oCurs(i, 0), oCurs(i, 1), oLoc)
  Next
End Sub

Sub SetLocaleForCurs(oLCurs, oRCurs, oLoc)
  Dim oText

  If IsNull(oLCurs)  OR IsNull(oRCurs)  Then Exit Sub
  If IsEmpty(oLCurs) OR IsEmpty(oRCurs) Then Exit Sub

  oText = oLCurs.getText()

  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub
  oLCurs.goRight(0, False)
  Do While oLCurs.goRight(1, True) and _
           oText.compareRegionEnds(oLCurs, oRCurs) >= 0
    oLCurs.CharLocale = oLoc
    oLCurs.goRight(0, False)
  Loop
End Sub

Function CreateSelTextIterator(oDoc, sPrompt As String, oCurs()) As Boolean
```

```
  Dim lSelCount As Long 'Количество выделенных секций sections.
  Dim lWhichSel As Long 'Текущий элемент выделения.
  Dim oSels   'Все элементы выделения selections
  Dim oLCurs As Object  'Курсор на левом краю текущего выделения.
  Dim oRCurs As Object  'Курсор на правом краю текущего выделения.

  CreateSelTextIterator = True
  If Not IsAnythingSelected(oDoc) Then
    Dim i%
    i% = MsgBox("Нет выделенного текста!" + Chr(13) + sPrompt, _
      1 OR 32 OR 256, "Warning")
    If i% = 1 Then
      oLCurs = oDoc.getText().createTextCursor()
      oLCurs.gotoStart(False)
      oRCurs = oDoc.getText().createTextCursor()
      oRCurs.gotoEnd(False)
      oCurs = DimArray(0, 1)
      oCurs(0, 0) = oLCurs
      oCurs(0, 1) = oRCurs
    Else
      oCurs = DimArray()
      CreateSelTextIterator = False
    End If
  Else
    oSels = oDoc.getCurrentSelection()
    lSelCount = oSels.getCount()
    oCurs = DimArray(lSelCount - 1, 1)
    For lWhichSel = 0 To lSelCount - 1
      GetLeftRightCursors(oSels.getByIndex(lWhichSel), oLCurs, oRCurs)
      oCurs(lWhichSel, 0) = oLCurs
      oCurs(lWhichSel, 1) = oRCurs
    Next
  End If
End Function

Function IsAnythingSelected(oDoc) As Boolean
  Dim oSels  'Все элементы выделения
  Dim oSel         'Отдельный элемент выделения
  Dim oCursor      'Временный курсор

  IsAnythingSelected = False
  If IsNull(oDoc) Then Exit Function
  oSels = oDoc.getCurrentSelection()
  If IsNull(oSels) Then Exit Function
  If oSels.getCount() = 0 Then Exit Function

  REM If there are multiple selections, then certainly something is selected
  If oSels.getCount() > 1 Then
    IsAnythingSelected = True
  Else
    oSel = oSels.getByIndex(0)
    oCursor = oSel.getText().CreateTextCursorByRange(oSel)
    If Not oCursor.IsCollapsed() Then IsAnythingSelected = True
  End If
End Function
```

```
Sub GetLeftRightCursors(oSel, oLeft, oRight)
  Dim oCursor
  If oSel.getText().compareRegionStarts(oSel.getEnd(), oSel) >= 0 Then
    oLeft  = oSel.getText().CreateTextCursorByRange(oSel.getEnd())
    oRight = oSel.getText().CreateTextCursorByRange(oSel.getStart())
  Else
    oLeft  = oSel.getText().CreateTextCursorByRange(oSel.getStart())
    oRight = oSel.getText().CreateTextCursorByRange(oSel.getEnd())
  End If
  oLeft.goRight(0, False)
  oRight.goLeft(0, False)
End Sub
```

III.34. Автотекст (Auto Text)

Я не проверял код программы, но меня уверяли, что он работает. Вы не сможете использовать этот код без изменений, так как он требует диалога, которого у Вас нет, но сама техника работы будет полезна. Вот некоторые ссылки на тему, которые я нашёл:

http://api.openoffice.org/docs/common/ref/com/sun/star/text/XAutoTextGroup.html

http://api.openoffice.org/docs/common/ref/com/sun/star/text/AutoTextGroup.html

http://api.openoffice.org/docs/common/ref/com/sun/star/text/AutoTextContainer.html

http://api.openoffice.org/docs/common/ref/com/sun/star/text/XAutoTextContainer.html

Лист. 92: Использование автотекста.

```
'Author: Marc Messeant
'email: marc.liste@free.fr
'Копировать один AutoText из группы в другой (AutoText)
'ListBox1 : Исходная группа
'ListBox2 : Группа, куда копировать
'ListBox3 : Элемент исходной группы, который копируем
'ListBox4 : Элемент группы, куда копировать (только для сведения)

Dim ODialog as object
Dim oAutoText as object

' Эта процедура открывает диалог и инициализирует списки групп

Sub OuvrirAutoText
  Dim aTableau() as variant
  Dim i as integer
  Dim oListGroupDepart as object, oListGroupArrivee as object

  oDialog = LoadDialog("CG95","DialogAutoText")
  oListGroupDepart = oDialog.getControl("ListBox1")
  oListGroupArrivee = oDialog.getControl("ListBox2")
  oAutoText = createUnoService("com.sun.star.text.AutoTextContainer")
  aTableau = oAutoText.getElementNames()
  oListGroupDepart.removeItems(0,oListGroupDepart.getItemCount())
  oListGroupArrivee.removeItems(0,oListGroupArrivee.getItemCount())
```

```
  For i = LBound(aTableau()) To UBound(aTableau())
    oListGroupDepart.addItem(aTableau(i),i)
    oListGroupArrivee.addItem(aTableau(i),i)
  Next
  oDialog.Execute()
End Sub

'Эти 3 процедуры вызываются, когда пользователь выбирает одну группу для
'инициализации списков элементов  AutoText для каждой группы
Sub ChargerList1()
  ChargerListeGroupe("ListBox1","ListBox3")
End Sub
Sub ChargerList2()
  ChargerListeGroupe("ListBox2","ListBox4")
End Sub

Sub ChargerListeGroupe(ListGroupe as string,ListElement as string)
  Dim oGroupe as object
  Dim oListGroupe as object
  Dim oListElement as object
  Dim aTableau() as variant
  Dim i as integer

  oListGroupe = oDialog.getControl(ListGroupe)
  oListElement = oDialog.getControl(ListElement)
  oGroupe = oAutoText.getByIndex(oListGroupe.getSelectedItemPos())
  aTableau = oGroupe.getTitles()
  oListElement.removeItems(0,oListElement.getItemCount())
  For i = LBound(aTableau()) To UBound(aTableau())
    oListElement.addItem(aTableau(i),i)
  Next
End Sub

'Эта процедура переносит один элемент из одной группы в другую
Sub TransfererAutoText()
  Dim oGroupDepart as object,oGroupArrivee as object
  Dim oListGroupDepart as object, oListGroupArrivee as object
  Dim oListElement as object
  Dim oElement as object
  Dim aTableau() as string
  Dim i as integer

  oListGroupDepart = oDialog.getControl("ListBox1")
  oListGroupArrivee = oDialog.getControl("ListBox2")
  oListElement = oDialog.getControl("ListBox3")
  i =oListGroupArrivee.getSelectedItemPos()
  If oListGroupDepart.getSelectedItemPos() = -1 Then
    MsgBox ("Не указан исходная группа, откуда копировать элемент")
    Exit Sub
  End If
  If oListGroupArrivee.getSelectedItemPos() = -1 Then
    MsgBox ("Не указано, куда копировать")
    Exit Sub
  End If
  If oListElement.getSelectedItemPos() = -1 Then
```

```
    MsgBox ("Нет элемента для копирования в исходной группе")
    Exit Sub
  End If
  oGroupDepart = oAutoText.getByIndex(oListGroupDepart.getSelectedItemPos())
  oGroupArrivee =
oAutoText.getByIndex(oListGroupArrivee.getSelectedItemPos())
  aTableau = oGroupDepart.getElementNames()
  oElement = oGroupDepart.getByIndex(oListElement.getSelectedItemPos())
  If oGroupArrivee.HasByName(aTableau(oListElement.getSelectedItemPos()))
Then
    MsgBox ("Элемент уже существует")
    Exit Sub
  End If
  oGroupArrivee.insertNewByName(aTableau(oListElement.getSelectedItemPos()),
_
    oListElement.getSelectedItem(),oElement.Text)
  ChargerListeGroupe("ListBox2","ListBox4")
End Sub
```

III.35. Десятичные футы и дроби

Меня попросили преобразовать некоторые макросы Microsoft Office в макросы OOo. Я решил улучшить их. Первый получает десятичное число футов и преобразует его в футы и дюймы с дробями. Я решил создать некоторые общие процедуры и игнорировать существующий код. Это также обходит несколько ошибок, которые я нашёл в старом коде макросов. Наиболее быстрый метод, который я знаю для преобразования дробей, это найти наибольший общий делитель GCD (Greatest Common Divisor). . Макрос обработки дробей вызывает GCD для упрощения дроби.

Лист. 93: Вычисление наибольшего общего делителя GCD

```
'Author: Olivier Bietzer
'e-mail: olivier.bietzer@free.fr
'Используется алгоритм Эвклида  Euclide и очень быстро работает!
Function GCD(ByVal x As Long, ByVal y As Long) As Long
  Dim pgcd As Long, test As Long

  ' Нужно обеспечить x >= y и значения больше нуля
  x=abs(x)
  y=abs(y)
  If (x < y) Then
   test = x : x = y : y = test
  End If
  If y = 0 Then Exit Function

  ' Euclide создал такой алгоритм ....
  pgcd = y        ' по определению, PGCD = наименьший
  test = x MOD y  ' остаток от деления
  Do While (test) ' то есть пока test не равен 0
   pgcd = test    ' pgcd равен остатку
   x = y          ' обмен значениями x,y и текущего pgcd
```

```
   y = pgcd
   test = x MOD y ' снова test приравниваем остатку
  Loop
  GCD = pgcd  ' pgcd равен последнему ненулевому остатку! Магия ...
End Function
```

Следующий макрос определяет дробь. Если x отрицательный, то и числитель (numerator), и возвращаемое значение x отрицательные на выходе. Заметьте, что параметр x изменяется.

Лист. 94: Преобразование числа с двойной точностью double в дробь.

```
'n: в конце работы содержит числитель numerator
'd: в конце работы содержит знаменатель denominator
'x: Сначала введенное значение для преобразования в дробь, в конце работы
целая часть
'max_d: Максимальный знаменатель Maximum denominator
Sub ToFraction(n&,  d&, x#, ByVal max_d As Long)
  Dim neg_multiply&, y#
  n = 0 : d = 1 : neg_multiply = 1 : y = Fix(x)
x=x-y:n=abs(Fix(x*max_d))  '  добавлено переводчиком
  If (x < 0) Then
    x = -x : neg_multiply = -1
  End If

  REM Только для случая, когда x не содержит дробной части
  If (n <> 0) Then
    d = GCD(n, max_d)
    n = neg_multiply * n / d
    d = max_d / d
    x = y
  End If
  x = y
End Sub
```

Для проверки процедуры я создал тестовый код.

Лист. 95: Проверка процедуры

```
Sub FractionTest
  Dim x#, inc#, first#, last#, y#, z#, epsilon#
  Dim d&, n&, max_d&
  first = -10 : last  = 10 : inc = 0.001
  max_d = 128
  epsilon = 1.0 / CDbl(max_d)
  For x = first To last Step inc
    y = x
    ToFraction(n, d, y, max_d)
    z = y + CDbl(n) / CDbl(d)
    If abs(x-z) > epsilon Then Print "Неверно преобразовано " & x & " в " &
z
  Next
End Sub
```

Хотя я полностью игнорировал старый код, я хотел сохранить форматы аргументов и результата из старого макроса, даже если мой и старый макросы не имеют ничего общего.

Лист. 96: Преобразовать десятичное число футов в строку.

```
'[-]feet'-inches n/d"
'Ни одна часть этого формата вывода не формируется, если она нулевая.
Function DecimalFeetToString64(ByVal x#) As String
  'я использую только 64, так как это было в оригинале
  DecimalFeetToString64 = DecimalFeetToString(x, 64)
End Function

Function DecimalFeetToString(ByVal x#, ByVal max_denominator&) As String
  Dim numerator&, denominator&
  Dim feet#, decInch#, s As String

  s = ""
  If (x < 0) Then
    s = "-"
    x = -x
  End If

  feet = Fix(x)    'Целое число футов Feet
  x = (x — feet) * 12  'Число дюймов Inches
  ToFraction(numerator, denominator, x, max_denominator)
  REM Выполняется некоторое округление
  If (numerator = denominator and numerator <> 0) Then
    numerator = 0
    x = x + 1
  End If

  If feet = 0 and x = 0 and numerator = 0Then
    s = s & "0'"
  Else
    If feet <> 0 Then
      s = s & feet & "'"
      If x <> 0 OR numerator <> 0 Then s = s &  "-"
    End If
    If x <> 0 Then
      s = s & x
      If numerator <> 0 Then s = s &  " "
    End If
    If numerator <> 0 Then s = s & numerator & "/" & denominator
    If x <> 0 OR numerator <> 0 Then s = s &  """
  End If
  DecimalFeetToString = s
End Function

Function StringToDecimalFeet(s$) As Double
  REM Максимальный пеоечень элементов синтаксиса:
  REM <feet><'><-><inches><space><numerator></><denominator><">
  REM Первым элементом синтаксиса ДОЛЖНО быть число!
  Dim tokens(8) As String '0 to 8
```

```
Dim i%, j%, num_tokens%, c%
Dim feet#, inches#, n#, d#, leadingNeg#
feet = 0 : inches = 0 : n = 0 : d = 1 : i = 1 : leadingNeg = 1.0
s = Trim(s)  ' Удаляем ведущие и замыкающие пробелы
If (Len(s) > 0) Then
  If Left(s,1) = "-" Then
    leadingNeg = -1.0
    s = Mid(s, 2)
  End If
End If

num_tokens = 0 : i = 1
Do While i <= Len(s)
  Select Case Mid(s, i, 1)
    Case "-", "0" To "9"
      j = i
      If Left(s, i, 1) = "-" Then j = j + 1
      c = Asc(Mid(s, j, 1))
      Do While (48 <= c and c <= 57)
        j = j + 1
        If j > Len(s) Then Exit Do
        c = Asc(Mid(s, j, 1))
      Loop
      tokens(num_tokens) = Mid(s, i, j-i)
      num_tokens = num_tokens + 1
      i = j
    Case "'"
      feet = CDbl(tokens(num_tokens-1))
      tokens(num_tokens) = "'"
      num_tokens = num_tokens + 1
      i = i + 1
      If (i <= Len(s)) Then
        If Mid(s,i,1) = "-" Then i = i + 1
      End If
    Case """", "/", " "
      tokens(num_tokens) = Mid(s, i, 1)
      i = i + 1
      Do While i < Len(s)
        If Mid(s, i, 1) <> tokens(num_tokens) Then Exit Do
        i = i + 1
      Loop
      If tokens(num_tokens) = "/" Then
        n = CDbl(tokens(num_tokens-1))
        num_tokens = num_tokens + 1
      ElseIf tokens(num_tokens) = " " Then
        Inches = CDbl(tokens(num_tokens-1))
      ElseIf tokens(num_tokens) = """" Then
        If num_tokens = 1 Then
          Inches = CDbl(tokens(num_tokens-1))
        ElseIf num_tokens > 1 Then
          If tokens(num_tokens-2) = "/" Then
            d = CDbl(tokens(num_tokens-1))
          Else
            Inches = CDbl(tokens(num_tokens-1))
          End If
```

```
          End If
        End If

      Case Else
        'Сюда попадаем только в случае ошибки
        i = i + 1
        Print "Попали на ветку else"
    End Select
  Loop

  If d = 0 Then d = 1
  StringToDecimalFeet = leadingNeg * (feet + (inches + n/d)/12)
End Function
```

III.36. Преобразовать число в слова

Преобразование целого числа со значениями от 0 до 999 в слова достаточно просто. Тем не менее, нужна несколько другая обработка для языков, помимо английского.

Лист. 97: Преобразовать целое от 0 до 999 в слова

```
Function SmallIntToText(ByVal n As Integer) As String
  REM by Andrew D. Pitonyak
  Dim sOneWords()
  Dim sTenWords()
  Dim s As String

  If n > 999 Then
    Print "Предупреждение: n = " & n & " слишком велико!"
    Exit Function
  End If

  sOneWords() = Array("zero", _
     "one",      "two",       "three",     "four",      "five", _
     "six",      "seven",     "eight",     "nine",      "ten", _
     "eleven",   "twelve",    "thirteen", "fourteen", "fifteen", _
     "sixteen", "seventeen", "eighteen", "nineteen", "twenty")

  sTenWords() = Array( "zero", "Ten", "twenty", "thirty", "fourty", _
                       "fifty", "sixty", "seventy", "eighty", "ninety")

  s = ""
  If n > 99 Then
    s = sOneWords(Fix(n / 100)) & " hundred"
    n = n MOD 100
    If n = 0 Then
      SmallIntToText = s
      Exit Function
    End If
    s = s & " "
  End If
```

```
  If (n > 20) Then
    s = s & sTenWords(Fix(n / 10))
    n = n MOD 10
    If n = 0 Then
      SmallIntToText = s
      Exit Function
    End If
    s = s & " "
  End If

  SmallIntToText = s & sOneWords(n)
End Function
```

Преобразование чисел вне интервала 0-999 требует несколько большей работы. Я вставил имена, как они используются в США, Англии и Германии. Я не обрабатывал дробные числа и отрицательные числа. Это всего лишь макрос, с которого Вы можете начать!

Лист. 98: Преобразовать большие числа в слова

```
Function NumberToText(ByVal n) As String
  REM by Andrew D. Pitonyak
  Dim sBigWordsUSA()
  Dim sBigWordsUK()
  Dim sBigWordsDE()

  REM 10^100 называется Гугль googol (10 duotrigintillion)
  REM Этот макрос работает до 10^303, нужно ли Вам больше?
  sBigWordsUSA = Array( "", _
     "thousand", "million", "billion", "trillion", "quadrillion", _
     "quintillion", "sexillion", "septillion", _
     "octillion", "nonillion", "decillion", "undecillion", "duodecillion", _
     "tredecillion","quattuordecillion", "quindecillion", "sexdecillion", _
     "septdecillion", "octodecillion", "novemdecillion", "vigintillion", _
     "unvigintillion", "duovigintillion", "trevigintillion", _
     "quattuorvigintillion", "quinvigintillion", "sexvigintillion", _
     "septvigintillion", "octovigintillion", "novemvigintillion", _
     "trigintillion", "untrigintillion", "duotrigintillion", _
     "tretrigintillion", "quattuortrigintillion", "quintrigintillion", _
     "sextrigintillion", "septtrigintillion", "octotrigintillion", _
     "novemtrigintillion", "quadragintillion", "unquadragintillion", _
     "duoquadragintillion", "trequadragintillion", _
     "quattuorquadragintillion", _
     "quinquadragintillion", "sexquadragintillion", "septquadragintillion",
_
     "octoquadragintillion", "novemquadragintillion", "quinquagintillion", _
     "unquinquagintillion", "duoquinquagintillion", "trequinquagintillion",
_
     "quattuorquinquagintillion", "quinquinquagintillion", _
     "sexquinquagintillion", "septquinquagintillion", _
     "octoquinquagintillion", _
     "novemquinquagintillion", "sexagintillion", "unsexagintillion", _
     "duosexagintillion", "tresexagintillion", "quattuorsexagintillion", _
     "quinsexagintillion", "sexsexagintillion", "septsexagintillion", _
```

```
     "octosexagintillion", "novemsexagintillion", "septuagintillion", _
     "unseptuagintillion", "duoseptuagintillion", "treseptuagintillion", _
     "quattuorseptuagintillion", "quinseptuagintillion", _
     "sexseptuagintillion", _
     "septseptuagintillion", "octoseptuagintillion",
"novemseptuagintillion", _
     "octogintillion", "unoctogintillion", "duooctogintillion", _
     "treoctogintillion", "quattuoroctogintillion", "quinoctogintillion", _
     "sexoctogintillion", "septoctogintillion", "octooctogintillion", _
     "novemoctogintillion", "nonagintillion", "unnonagintillion", _
     "duononagintillion", "trenonagintillion", "quattuornonagintillion", _
     "quinnonagintillion", "sexnonagintillion", "septnonagintillion", _
     "octononagintillion", "novemnonagintillion", "centillion" _
  )

  sBigWordsUK() = Array( "", "thousand", _
     "milliard", "billion", "billiard", "trillion", "trilliard", _
     "quadrillion", "quadrilliard", "quintillion", "quintilliard", _
     "sextillion", "sextilliard", "septillion", "septilliard", _
     "octillion", "octilliard", "nonillion", "nonilliard", "decillion", _
     "decilliard", "undecillion", "undecilliard", "dodecillion", _
     "dodecilliard", "tredecillion", "tredecilliard", "quattuordecillion", _
     "quattuordecilliard", "quindecillion", "quindecilliard",
"sexdecillion", _
     "sexdecilliard", "septendecillion", "septendecilliard",
"octodecillion", _
     "octodecilliard", "novemdecillion", "novemdecilliard", "vigintillion",
_
     "vigintilliard", "unvigintillion", "unvigintilliard",
"duovigintillion", _
     "duovigintilliard", "trevigintillion", "trevigintilliard", _
     "quattuorvigintillion", "quattuorvigintilliard", "quinvigintillion", _
     "quinvigintilliard", "sexvigintillion", "sexvigintilliard", _
     "septenvigintillion", "septenvigintilliard", "octovigintillion", _
     "octovigintilliard", "novemvigintillion", "novemvigintilliard", _
     "trigintillion", "trigintilliard", "untrigintillion", _
     "untrigintilliard", "duotrigintillion", "duotrigintilliard", _
     "tretrigintillion", "tretrigintilliard", "quattuortrigintillion", _
     "quattuortrigintilliard", "quintrigintillion", "quintrigintilliard", _
     "sextrigintillion", "sextrigintilliard", "septentrigintillion", _
     "septentrigintilliard", "octotrigintillion", "octotrigintilliard", _
     "novemtrigintillion", "novemtrigintilliard", "quadragintillion", _
     "quadragintilliard", "unquadragintillion", "unquadragintilliard", _
     "duoquadragintillion", "duoquadragintilliard", "trequadragintillion", _
     "trequadragintilliard", "quattuorquadragintillion", _
     "quattuorquadragintilliard", "quinquadragintillion", _
     "quinquadragintilliard", "sexquadragintillion", "sexquadragintilliard",
_
     "septenquadragintillion", "septenquadragintilliard", _
     "octoquadragintillion", "octoquadragintilliard", _
     "novemquadragintillion", "novemquadragintilliard", "quinquagintillion",
_
     "quinquagintilliard" _
  )
```

```
  sBigWordsDE() = Array( "", "Tausand", _
     "Milliarde", "Billion", "Billiarde", "Trillion", "Trilliarde", _
     "Quadrillion", "Quadrilliarde", "Quintillion", "Quintilliarde", _
     "Sextillion", "Sextilliarde", "Septillion", "Septilliarde", _
     "Oktillion", "Oktilliarde", "Nonillion", "Nonilliarde", _
     "Dezillion", "Dezilliarde", "Undezillion", "Undezilliarde", _
     "Duodezillion", "Doudezilliarde", "Tredezillion", _
     "Tredizilliarde", "Quattuordezillion", "Quattuordezilliarde", _
     "Quindezillion", "Quindezilliarde", "Sexdezillion", _
     "Sexdezilliarde", "Septendezillion", "Septendezilliarde", _
     "Oktodezillion", "Oktodezilliarde", "Novemdezillion", _
     "Novemdezilliarde", "Vigintillion", "Vigintilliarde", _
     "Unvigintillion", "Unvigintilliarde", "Duovigintillion", _
     "Duovigintilliarde", "Trevigintillion", "Trevigintilliarde", _
     "Quattuorvigintillion", "Quattuorvigintilliarde", "Quinvigintillion", _
     "Quinvigintilliarde", "Sexvigintillion", "Sexvigintilliarde", _
     "Septenvigintillion", "Septenvigintilliarde", "Oktovigintillion", _
     "Oktovigintilliarde", "Novemvigintillion", "Novemvigintilliarde", _
     "Trigintillion", "Trigintilliarde", "Untrigintillion", _
     "Untrigintilliarde", "Duotrigintillion", "Duotrigintilliarde", _
     "Tretrigintillion", "Tretrigintilliarde", "Quattuortrigintillion", _
     "Quattuortrigintilliarde", "Quintrigintillion", "Quintrigintilliarde",
_
     "Sextrigintillion", "Sextrigintilliarde", "Septentrigintillion", _
     "Septentrigintilliarde", "Oktotrigintillion", "Oktotrigintilliarde", _
     "Novemtrigintillion", "Novemtrigintilliarde", "Quadragintillion", _
     "Quadragintilliarde", "Unquadragintillion", "Unquadragintilliarde", _
     "Duoquadragintillion", "Duoquadragintilliarde", "Trequadragintillion",
_
     "Trequadragintilliarde", "Quattuorquadragintillion", _
     "Quattuorquadragintilliarde", "Quinquadragintillion", _
     "Quinquadragintilliarde", "Sexquadragintillion", _
     "Sexquadragintilliarde", "Septenquadragintillion", _
     "Septenquadragintilliarde", "Oktoquadragintillion", _
     "Oktoquadragintilliarde", "Novemquadragintillion", _
     "Novemquadragintilliarde", "Quinquagintillion", "Quinquagintilliarde" _
  )

  Dim i As Integer
  Dim iInt As Integer
  Dim s As String
  Dim dInt As Double


  REM Отсекаем дробную часть.
  dInt = Fix(n)
  If (dInt < 1000) Then
    NumberToText = SmallIntToText(CInt(dInt))
    Exit Function
  End If

  REM i = индекс в массиве sBigWords
  i = 0
  s = ""
  Do While dInt > 0
```

```
    iInt = CInt(dInt - Fix(dInt / 1000) * 1000)
    If iInt <> 0 Then
      If Len(s) > 0 Then s = " " & s
      s = SmallIntToText(iInt) & " " & sBigWordsUSA(i) & s
    End If
    i = i + 1
    dInt = Fix(dInt / 1000)
    'Print "s = " & s & " dInt = " & dInt
  Loop
  NumberToText = s
End Function
```

Следующий макрос является примером преобразования чисел в текст. Форма исходного числа «$123,453,223.34». Ведущий символ доллара удаляется, все запятые удаляются, точка отделяет доллары от центов.

Лист. 99: Преобразовать сумму в долларах в слова.

```
Function USCurrencyToWords(s As String) As String
  Dim sDollars As String
  Dim sCents As String
  Dim i%

  If (s = "") Then s = "0"
  If (Left(s, 1) = "$") Then s = Right(s, Len(s) - 1)

  If (InStr(s, ".") = 0) Then
    sDollars = s
    sCents = "0"
  Else
    sDollars = Left(s, InStr(s, ".")-1)
    sCents = Right(s, Len(s) - InStr(s, "."))
  End If

  Do While (InStr(sDollars, ",") > 0)
    i = InStr(sDollars, ",")
    sDollars = Left(sDollars, i-1) & Right(sDollars, Len(sDollars) - i)
  Loop

  If (sDollars = "") Then sDollars = "0"
  If (sCents = "") Then sCents = "0"
  If (Len(sCents) = 1) Then sCents = sCents & "0"

  USCurrencyToWords = NumberToText(sDollars) & " Dollars and " & _
                      NumberToText(sCents) & " Cents"
End Function
```

III.37. Отправка электронного письма

OOo предоставляет возможность отправки электронного письма (sending email), но это должно быть правильно настроено, особенно для Linux. OOo использует существующую программу почтового клиента вместо прямой отправки по протоколам электронной почты. При первоначальной установке

OOo должен знать, как использовать обычные почтовые клиенты, такие как Mozilla/Netscape, Evolution, и K-Mail. В среде Windows OOo использует MAPI, поэтому все совместимые cl MAPI клиенты должны работать. Вам нужно использовать «com.sun.star.system.SimpleSystemMail». Здесь SimpleCommandMail использует инструменты командной строки операционной системы для отправки письма, но я применял этот сервис при работе под Linux. Следующий пример представлен автором Laurent Godard.

Лист. 100: Послать электронное письмо (email).

```
Sub SendSimpleMail()
  Dim vMailSystem, vMail, vMessage
  'vMailSystem = createUnoService( "com.sun.star.system.SimpleCommandMail" )
  vMailSystem=createUnoService("com.sun.star.system.SimpleSystemMail")
  vMail=vMailSystem.querySimpleMailClient()
  'Посмотреть, что еще можно сделать с этим, можно по ссылке
  'http://api.openoffice.org/docs/common/ref/com/sun/star/system/XSimpleMail
Message.html
  vMessage=vMail.createsimplEmailMessage()
  vMessage.setrecipient("andrew@pitonyak.org")
  vMessage.setsubject("This is my test subject")

  'Приложения к письму заданы последовательностью, которая в языке basic
означает массив array
  'Я мог бы использовать ConvertToURL() для создания адреса URL!
  Dim vAttach(0)
  vAttach(0) = "file:///c:/macro.txt"
  vMessage.setAttachement(vAttach())

  'DEFAULTS означает текущие настройки почтового клиента операционной
системы.
  'NO_USER_INTERFACE Не показывать интерфейс пользователю, только послать
письмо!
  'NO_LOGON_DIALOG Не открывать диалог входа logon, но это приведет к
ошибке, если такой диалог потребуется.
  vMail.sendSimpleMailMessage(vMessage, _
              com.sun.star.system.SimpleMailClientFlags.NO_USER_INTERFACE)
End Sub
```

Ни сервис SimpleSystemMail, ни сервис SimpleCommandMail не могут послать текстовое тело электронного письма (send an email text body). Согласно Mathias Bauer, целью этих сервисов является отправка документа в качестве приложения к письму. Можно использовать адрес URL «mailto» для отправки электронного письма с текстовым телом письма, но это не будет содержать приложенных к письму файлов. Идея заключается в том, чтобы позволить операционной системе передать mailto URL объекту по умолчанию, который может, я надеюсь, сможет разобрать синтаксис всего текста. Поддержка этого метода зависит от операционной системы и установленных программ..

Лист. 101: Послать электронное письмо, используя URL.

```
Dim noargs()
email_dispatch_url = "mailto:demo@someplace.com?subject=Test&Body=Text"
dispatcher = createUnoService( "com.sun.star.frame.DispatchHelper")
dispatcher.executeDispatch( StarDesktop,email_dispatch_url, "", 0, noargs())
```

Согласно Daniel Juliano [daniel.juliano@rainhail.com], размер сообщения ограничен операционной системой. Для Windows 2000 лимит близок к 500 символам. Если этот размер превышен, то письмо не отправляется и ошибка НЕ происходит. (Andrew Pitonyak подозревает, что размер сообщения ограничен потому, что оно посылается из командной строки операционной системы. Другие интерпретаторы командной строки поддерживают другой размер командной строки. Например, 4NT, вероятно, поддерживает более длинные командные строки, чем Microsoft.)

При выполнении под Windows с использованием Outlook можно легко отправить текстовое тело письма и приложения, которые Вам нужны.

Лист. 102: Послать электронное письмо, используя Microsoft Outlook

```
Sub UseOutlook( )
  Dim oOLEService
  Dim oOutlookApp
  Dim oOutlookMail

  oOLEService = createUnoService("com.sun.star.bridge.OleObjectFactory")
  oOutlookApp = oOLEService.createInstance("Outlook.Application")
  oOutlookMail = oOutlookApp.CreateItem(0)

  REM Я могу явно задать получателей письма, установив свойство To
  oOutlookMail.To = "andrew@pitonyak.org"

  REM Я могу также добавить адрес к списку получателей, но в моих
экспериментах получалось обращение
  REM к почтовому ящику mail box и поэтому Outlook запрашивает меня,
разрешить ли это действие. Другими словами,
  REM требуется вмешательство пользователй. Вероятно, я мог бы в настройках
безопасности outlook
  REM просто разрешать это действие, но тогда эти действия стали бы
разрешены и для вирусов.
  'oOutlookMail.Recipients.Add("andrew@pitonyak.org")

  oOutlookMail.Subject = "Это - тема письма"
  oOutlookMail.Body = "Это - текст тела письма"

  REM Можно также добавить вложения к письму
  'oOutlookMail.Attachments.Add("C:\foo.txt")

  REM Я могу вывести на экран и отредактировать письмо
  'oOutlookMail.Display()

  REM Или я могу отослать письмо
```

```
  'oOutlookMail.send()
End Sub
```

III.38. Библиотеки макросов

Этот раздел показывает, как использовать и устанавливать библиотеки макросов. Russ Phillips является автором очень хорошей статьи «Macro Library How-To», доступной по ссылке

`http://www.ooomacros.org`

Этот документ «как сделать» (how-to) посвящён созданию и использованию библиотек из OOo GUI.

III.38.1. Словарь

Чтобы понять работу с библиотеками, нужно понять разницу между контейнером библиотек, библиотекой и модулем.

III.38.1.A. Контейнер библиотеки модулей

Контейнер библиотеки содержит библиотеки макросов. OOo содержит два контейнера библиотек, «OpenOffice.org Macros» и «My Macros». Используйте Сервис — Макросы — Управление макросами (**Tools** > **Macros** > **Organize Macros**) — **OpenOffice.org Basic** для просмотра доступных контейнеров библиотеки.

Макросы, распространяемые с OOo находятся в контейнере OpenOffice.org Macros и не следует модифицировать их. Все Ваши макросы следует сохранять в контейнере My Macros. Каждый документ также является контейнером библиотеки и виден как доступный контейнер библиотеки.

III.38.1.B. Библиотеки

Библиотека содержит модули. Библиотека используется для верхнего уровня группировки. Например, если я хочу сохранить группу взаимосвязанных макросов и затем распространять их, то, вероятно, я сохраню их всех в одной библиотеке.

Нельзя выполнить макрос, содержащийся в библиотеке, если библиотека ещё не загружена. Загрузить библиотеку можно используя GUI или из макроса.

Каждый контейнер библиотеки автоматически имеет библиотеку с именем Standard. Библиотека Standard всегда уже загружена. Чтобы гарантировать, что определённый макрос всегда загружен, сохраните макрос в библиотеке Standard. Например, я часто сохраняю макросы, вызываемые из формы управления, в библиотеке Standard. Эти макросы «обработки событий» (event handler) могут затем загружать другие библиотеки и вызывать макросы из других библиотек, если необходимо.

III.38.1.C. Модули

Модули содержат процедуры и функции (и диалоги) — макросы.

III.38.2. Где хранятся библиотеки?

Предположим, что OOo установлен в папку «C:\Program Files\OpenOffice». Тогда «OpenOffice.org Macros» сохраняются в «C:\Program Files\OpenOffice\user\basic\». Ваши макросы хранятся в папке, похожей на «C:\Documents and Settings\<user name>\Application Data\OpenOffice\user\basic\». Под Linux Ваши макросы хранятся вне Вашей домашней библиотеки в «.OpenOffice.org/user/basic».

Эта папка содержит файлы Script.xlc и Dialog.xlc, которые ссылаются на библиотеки, видимые в OOo. Если библиотека существует, но Вы её не видите, вероятно, проблема в одном из этих двух файлов.

Каждая библиотека представлена как папка с тем же именем, что и имя библиотеки. На каждую библиотеку есть ссылка в Script.xlc и Dialog.xlc. Каждая папка с библиотекой содержит модули с расширением .xba или .xdl, также как script.xlb и dialog.xlb, которые содержат список модулей, входящих в библиотеку. Каждая библиотека связана с конкретным документом или с приложением OOo.

III.38.3. Контейнер библиотек

До версии 1.0 контейнер библиотек LibraryContainer был доступен из Basic, но не из других языков. Сервис «com.sun.star.script.ApplicationScriptLibraryContainer» открывает библиотеки для других языков, но этот сервис не опубликован официально — считается плохой практикой использование неопубликованных интерфейсов и сервисов. См. интерфейс «com.sun.star.script.XLibraryContainer» для изучения, как использовать этот интерфейс.

Переменная BasicLibraries, доступная только из Basic, ссылается на библиотеки Basic, хранящиеся в ThisComponent. Аналогично, переменная DialogLibraries ссылается на библиотеки диалога, хранящиеся в ThisComponent. Библиотеки уровня приложений доступны с использованием GlobalScope.BasicLibraries и GlobalScope.DialogLibraries.

> ***Внимание*** *Библиотеки документов доступны также с использованием нежелательного метода getLibraryContainer() и его соответствующего свойства LibraryContainer. Это также единственный способ получить доступ к библиотекам из Basic, для документа, который НЕ является ThisComponent.*

Следующий пример демонстрирует, как обращаться с библиотеками, используя ApplicationScriptLibraryContainer.

Лист. 103: Использование ApplicationScriptLibraryContainer.

```
Sub LibContainer()
  REM Christian Junker
  Dim allLibs()
  Const newlib As String = "dummy" 'Имя Вашей новой библиотеки library
  Dim sService As String
  sService = "com.sun.star.script.ApplicationScriptLibraryContainer"
  oLibCont = createUnoService(sService)

  'создаем новую библиотеку library
  If (Not oLibCont.hasByName(newlib)) Then
    oLibCont.CreateLibrary(newlib)
  End If
  'проверяем, загружена ли эта библиотека, и если нет - загружаем!
  If (Not oLibCont.isLibraryLoaded(newlib)) Then
    oLibCont.loadLibrary(newlib)
  End If
  'устанавливаем пароль на библиотеку (она не должна быть "только для
чтения")
  oLibCont.setLibraryReadOnly(newlib, False)
  oLibCont.changeLibraryPassword(newlib, "", "password")
  MsgBox "The password: ""password"" was set for library " & newlib
  'показать все библиотеки, включая новую:
  allLibs = oLibCont.getElementNames()
  ShowArray(allLibs()) 'Эта функция есть в библиотеке  Tools
  'Удаляем библиотеку (она не должна быть "только для чтения")
  oLibCont.removeLibrary(newlib)
  'Показать все библиотеки снова, новая теперь отсутствует
  allLibs = oLibCont.getElementNames()
  ShowArray(allLibs())
End Sub
```

К сожалению, переименование библиотеки во время выполнения макроса не работает для этого примера.??

III.38.4. Предупреждение о неопубликованных сервисах

Сервис ApplicationScriptLibraryContainer ни опубликован, ни документирован. Согласно Jürgen Schmidt из фирмы Sun, вероятно, есть две серьёзных причины, почему этот сервис не опубликован. Хотя Вы найдёте и можете использовать этот сервис, он может измениться, потому что он не опубликован официально. В будущем мы документируем не опубликованные APIs, но они будут помечены и должны использоваться очень внимательно. Мы понимаем, что иногда лучше иметь некоторый опыт использования API и и получить отзывы до того, как этот API будет опубликован, так как «опубликован» означает «не подлежит изменению». Даже самый лучший проект может потребовать изменений.

III.38.5. Что означает загрузить (load) библиотеку?

Когда библиотека загружена, содержащиеся в ней макросы становятся видимыми из Basic. В этот момент файлы XML загружены и макросы скомпилированы. Другими словами, если библиотека не загружена, Вы не можете вызывать процедуры, функции и диалоги, которые в ней содержатся. Вы не захотите загружать все существующие библиотеки, так как обычно не используете все макросы из них и это будет лишними затратами памяти. Библиотека Standard, тем не менее, всегда загружена и доступна.

III.38.6. Распространение и развёртывание библиотеки

Добавление макроса в документ — это наиболее лёгкий путь предоставления библиотеки для общего использования. Однако если у Вас есть множество макросов, которые Вы хотите внедрить во все приложение, то предпочтительнее использование инструмента pkgchk. Этот инструмент pkgchk также используется для регистрации компонентов, которые Вы написали на других языках программирования, отличных от Basic. Простое объяснение — pkgchk (сокращение pkgchk означает packagecheck — проверка пакетов) упаковывает библиотеки в одну коллекцию, которая сохраняется как файл .zip в папке «C:\Program Files\OpenOffice\user\uno-packages\». Если параметр «--shared» используется, то эта коллекция сохраняется в другой папке «C:\Program Files\OpenOffice\share\uno-packages\» . Используйте следующие шаги для создания такого файла zip:

Скопируйте папку (папки) с библиотеками во временную папку.

Упакуйте эти библиотеки в архив Zip, используя Вашу любимую программу архивации. Убедитесь, что в архиве сохранилась структура папок.

Найдите программу pkgchk — она расположена в папке программ вне папки установки OpenOffice.org.

Для установки пакета «mymacros.zip» выполните «pkgchk -shared mymacros.zip» — Вам, вероятно, понадобится указать полный путь к файлу «mymacros.zip». Этот макрос должен быть установлен в общей папке пакетов UNO.

Один код программы (автор Sunil Menon) даёт пример такого процесса использования макроса [Andrew Pitonyak замечает: я делаю это другим способом в моей книге, используя BasicLibraries и т. п.]

Лист. 104: Распространение макроса с использованием ApplicationScriptLibraryContainer.

```
'author:  Sunil Menon
'email:  sunil.menon@itb-india.com
service_name = "com.sun.star.script.ApplicationScriptLibraryContainer"
Set oLibLoad = objServiceManager.createInstance(service _name)
If Not oLibLoad Is Nothing Then
```

```
  On Error Resume Next
  If oLibLoad.isLibraryLoaded("mymacros") Then
    oLibLoad.removeLibrary ("mymacros")
  End If
  spath = "file:///D|/StarOfficeManual/mymacros"
  slib = "mymacros"
  Call oLibLoad.CreateLibraryLink(slib, spath, False)
  oLibLoad.loadLibrary ("mymacros")
  oLibLoad = Nothing
End If
```

Если макрос уже существует, то он должен быть зарегистрирован (registered) снова до того, как новая библиотека будет видна. Это выполняется выгрузкой и повторной загрузкой этой библиотеки. Метод CreateLibraryLink создаёт ссылку на внешнюю библиотеку, доступную с использованием менеджера библиотек (library manager). Формат объекта StorageURL зависит от способа вставки макроса (The format of the StorageURL is implementation dependent). Булевый параметр — это флаг, доступный только для чтения.

III.39. Установка размеров точечного рисунка (Bitmap)

Если Вы загружаете изображение, его размер может оказаться не тем, какой Вам нужен. Первым указал мне на эту проблему Vance Lankhaar. Его первое решение создавало очень маленькое изображение.

Лист. 105: Вставить GraphicObjectShape.

```
'Author:  Vance Lankhaar
'email:  vlankhaar@linux.ca
Dim oDesktop As Object, oDoc As Object
Dim mNoArgs()
Dim sGraphicURL As String
Dim sGraphicService As String, sUrl As String
Dim oDrawPages As Object, oDrawPage As Object
Dim oGraphic As Object
sGraphicURL = "http://api.openoffice.org/branding/images/logonew.gif"
sGraphicService = "com.sun.star.drawing.GraphicObjectShape"
sUrl = "private:factory/simpress"
oDesktop = createUnoService("com.sun.star.frame.Desktop")
oDoc = oDesktop.loadComponentFromURL(sUrl,"_default",0,mNoArgs())
oDrawPages = oDoc.DrawPages
oDrawPage = oDrawPages.insertNewByIndex(1)
oGraphic = oDoc.createInstance(sGraphicService)
oGraphic.GraphicURL = sGraphicURL
oDrawPage.add(oGraphic)
```

Первое решение (автор Laurent Godard) устанавливает размер изображения в максимально допустимый размер.

Лист. 106: Установить для изображения максимальный поддерживаемый размер.

```
'Максимальный размер, пропорции не будут соблюдены.
Dim TheSize As New com.sun.star.awt.Size
Dim TheBitmapSize As New com.sun.star.awt.Size
Dim TheBitmap as object
Dim xmult as double, ymult as double

TheBitmap=oGraphic.GraphicObjectFillBitmap
TheBitmapSize=TheBitmap.GetSize

xmult=TwipsPerPixelX/567*10*100
ymult=TwipsPerPixelY/567*10*100

TheSize.width=TheBitmapSize.width*xmult
TheSize.height=TheBitmapSize.height*ymult

oGraphic.setsize(TheSize)
```

Vance Lankhaar дал окончательное решение, которое максимизирует размер, но сохраняет пропорции изображения.

Лист. 107: Установить максимальный размер изображения при сохранении пропорций.

```
oBitmap = oGraphic.GraphicObjectFillBitmap
aBitmapSize = oBitMap.GetSize
iWidth  = aBitmapSize.Width
iHeight = aBitmapSize.Height

iPageWidth  = oDrawPage.Width
iPageHeight = oDrawPage.Height
dRatio = CDbl(iHeight) / CDbl(iWidth)
dPageRatio = CDbl(iPageHeight) / CDbl(iPageWidth)

REM Это дает максимально возможный размер с сохранением пропорций
REM Замените < на > для минимально возможного размера с сохранением
пропорций
If (dRatio < dPageRatio) Then
  aSize.Width = iPageWidth
  aSize.Height = CInt(CDbl(iPageWidth) * dRatio)
Else
  aSize.Width = CInt(CDbl(iPageHeight) / dRatio)
  aSize.Height = iPageHeight
End If

aPosition.X = (iPageWidth – aSize.Width)/2
aPosition.Y = (iPageHeight – aSize.Height)/2

oGraphic.SetSize(aSize)
oGraphic.SetPosition(aPosition)
```

III.39.1. Вставить, изменить размер, указать позицию изображения в документе OOo Calc.

David Woody [dwoody1@airmail.net] должен был вставить графический объект в заданную позицию и с заданным размером. С небольшой помощью и большим объёмом работы он нашёл следующее решение:

Этот ответ занял некоторое время, так как я имел и другие проблемы, которые нужно было решить для установки корректных значений координат X и Y. Следующий код вставляет графический объект, устанавливает его размер, передвигает его в нужную позицию. Я должен добавить следующую сроку к коду программы из книги в секции об установки размеров точечного рисунка (bitmap).

```
Dim aPosition As New com.sun.star.awt.Point
```

Другой проблемой было то, что я должен был определить пропорции, которые нужны были для правильного отображения координат aPosition.X и aPosition.Y на изображении. На моем компьютере значения 2540 и для X, и для Y соответствуют одному дюйму (2,51 см) на экране. Нижеприведённые значения поместят изображение на 1 дюйм от верхнего края листа и на 1 дюйм от левого края.

Лист. 108: Вставить изображение и расположить его в документе OOo Calc.

```
Sub InsertAndPositionGraphic
  REM Получить лист sheet
  Dim vSheet
  vSheet = ThisComponent.Sheets(0)

  REM Добавляем графический объект graphics object
  Dim oDesktop As Object, oDoc As Object
  Dim mNoArgs()
  Dim sGraphicURL As String
  Dim sGraphicService As String, sUrl As String
  Dim oDrawPages As Object, oDrawPage As Object
  Dim oGraphic As Object
  sGraphicURL = "file:///OOo/share/gallery/bullets/blkpearl.gif"
  sGraphicService = "com.sun.star.drawing.GraphicObjectShape"
  oDrawPage = vSheet.getDrawPage()
  oGraphic = ThisComponent.createInstance(sGraphicService)
  oGraphic.GraphicURL = sGraphicURL
  oDrawPage.add(oGraphic)

  REM Задаем размер объекта
  Dim TheSize As New com.sun.star.awt.Size
  TheSize.width=400
  TheSize.height=400
  oGraphic.setsize(TheSize)

  REM Задаем позицию объекта
```

```
  Dim aPosition As New com.sun.star.awt.Point
  aPosition.X = 2540
  aPosition.Y = 2540
  oGraphic.setposition(aPosition)
End Sub
```

III.39.2. Экспортировать изображение, задав его размер

(автор Sven Jacobi [Sven.Jacobi@sun.com])

Хотя это не документировано в версии OOo 1.1 в Руководстве разработчика, можно экспортировать изображение с заданием его разрешения. Объект MediaDescriptor в каждом графическом фильтре поддерживает последовательность свойства «FilterData», которое устанавливает размер изображения в пикселях с использованием свойств PixelWidth (ширина пикселя) и PixelHeight (высота пикселя). Логический размер может быть установлен с шагом 1/100 мм с использованием свойств LogicalWidth (логическая ширина) и LogicalHeight (логическая высота).

Это использует GraphicExportFilter, который может только: экспортировать форму (shape), формы (shapes) и страницу рисунка (draw page). Приведённый ниже макрос получает выделенный объект, который нужно экспортировать. В документе OOo Writer, например, выбранный встроенный графический объект не является формой (shape); он является TextGraphicObject (текстовый графический объект).

Лист. 109: Экспортировать текущую страницу как графический объект указанного размера.

```
Sub ExportCurrentPageOrSelection
  REM Свойства фильтра, зависят от выбранного фильтра
  Dim aFilterData (4) As New com.sun.star.beans.PropertyValue
  Dim sFileUrl As String

  aFilterData(0).Name  = "PixelWidth"
  aFilterData(0).Value = 1000
  aFilterData(1).Name  = "PixelHeight"
  aFilterData(1).Value = 1000
  aFilterData(2).Name  ="LogicalWidth"
  aFilterData(2).Value = 1000
  aFilterData(3).Name  ="LogicalHeight"
  aFilterData(3).Value = 1000
  aFilterData(4).Name  ="Quality"
  aFilterData(4).Value = 60

  sFileUrl = "file:///d:/test2.jpg"

  REM При достаточном опыте можно применить это для документов  Draw или
Impress
  xDoc = ThisComponent
  xView = xDoc.currentController
  xSelection = xView.selection
  If isEmpty( xSelection ) Then
```

```
    xObj = xView.currentPage
  Else
    xObj = xSelection
  End If
  Export( xObj, sFileUrl, aFilterData() )
End Sub

Sub Export( xObject, sFileUrl As String, aFilterData )
  Dim  xExporter

  xExporter = createUnoService( "com.sun.star.drawing.GraphicExportFilter" )
  xExporter.SetSourceDocument( xObject )

  Dim aArgs (2) As New com.sun.star.beans.PropertyValue
  Dim aURL As New com.sun.star.util.URL

  aURL.complete = sFileUrl
  aArgs(0).Name  = "MediaType"
  aArgs(0).Value = "image/jpeg"
  aArgs(1).Name  = "URL"
  aArgs(1).Value = aURL
  aArgs(2).Name  = "FilterData"
  aArgs(2).Value = aFilterData
  xExporter.filter( aArgs() )
End Sub
```

III.39.3. Нарисовать линию в документе OOo Calc

David Woody [dwoody1@airmail.net] предоставил следующее:

Убедитесь, что переменные TheSize заданы относительно переменной aPosition, так что если Вы хотите x1 = 500 и x2 = 2000, то тогда TheSize.width = x2 — x1. Аналогично для координаты Y.

Лист. 110: Нарисовать линию в документе OOo Calc.

```
Sub DrawLineInCalcDocument
  Dim xPage as object, xDoc as object, xShape as object
  Dim aPosition As New com.sun.star.awt.Point
  Dim TheSize As New com.sun.star.awt.Size

  xDoc = thiscomponent
  xPage = xDoc.DrawPages(0)
  xShape = xDoc.createInstance( "com.sun.star.drawing.LineShape" )
  xShape.LineColor = rgb( 255, 0, 0 )
  xShape.LineWidth = 100
  aPosition.X = 2500
  aPosition.Y = 2500
  xShape.setPosition(aPosition)
  TheSize.width = 2500
  TheSize.height=5000
  xShape.setSize(TheSize)
  xPage.add( xShape )
End Sub
```

III.40. Извлечение файла Zip

Laurent Godard [listes.godard@laposte.net] снова поражает своим решением. С некоторыми изменениями вот его сообщение.

Чтобы заставить это работать, необходимо обрабатывать содержимое (content) входного потока, как это делает OOo's API: не обращать внимание, что это такое (don't care what it is)!

Для решения моей проблемы я установил OutputStream и записал туда мой входной поток InputStream. Это все. И это, кажется, работает (проверено на текстовом файле, но должно работать во всех случаях). Как я обещал, представляю мой макрос для разархивирования UNZIP заданного файла, упакованного в формат ZIP. Остаётся ещё много сделать с этим макросом, но и сейчас он может помочь. Laurent Godard.

Лист. 111: Разархивировать zip-файл.

```
Sub UnzipAFile(ZipURL as string, SrcFileName as string, DestFile as string)

  Dim bExists as boolean

  ozip=createUnoService("com.sun.star.packages.Package")

  Dim oProp(0)
  oProp(0)=ConvertToURL(ZipURL)
  ozip.initialize(oProp())

  'существует ли srcFile ?
  bExists=ozip.HasByHierarchicalName(SrcFileName)
  if not bExists then exit sub

  'retreive a Packagestream
  ThePackageStream=ozip.GetByHierarchicalName(SrcFileName)

  'Retreive the InputStream on SrcFileName
  MyInpuStream=ThePackageStream.GetInputStream()

  'Define the outpu
  oFile = createUnoService("com.sun.star.ucb.SimpleFileAccess")
  oFile.WriteFile(ConvertToURL(DestFile),MyInpuStream)

  'Its Done !!!

End Sub
```

III.40.1. Другой пример Zip-файла

Dan Juliano [daniel.juliano@rainhail.com djuliano@dmacc.edu] расширил пример, представленный автором Laurent Godard. Следующий пример извлекает все файлы из архива zip.

Лист. 112: Извлечь все файлы из zip-файла.

```
' сначала пробные вызовы следующих процедур
call unzipFileFromArchive("c:\test.zip", "test.txt", "c:\test.txt")
call unzipArchive("c:\test.zip", "c:\")

Sub unzipFileFromArchive( _
  strZipArchivePath As String, _
  strSourceFileName As String, _
  strDestinationFilePath As String)

  Dim blnExists           As Boolean
  Dim args(0)             As Variant
  Dim objZipService       As Variant
  Dim objPackageStream    As Variant
  Dim objOutputStream     As Variant
  Dim objInputStream      As Variant
  Dim i                   As Integer

  ' Разархивируем отдельный файл из архива. Нужно знать точное имя файла
  ' внутри этого архива до выполнения этой процедуры.
  ' strZipArchivePath = полный путь (папки и файл) к архивному файлу .zip
  ' strSourceFileName = имя извлекаемого из архива файла.
  ' strDestinationFilePath = полный путь (папки и файл), куда будет извлечен
файл source
  '                         file will be dumped.
  ' Создаем  handle для сервиса zip-файлов,
  objZipService = createUnoService("com.sun.star.packages.Package")
  args(0) = ConvertToURL(strZipArchivePath)
  objZipService.initialize(args())

  ' Существует ли исходный файл?
  If Not objZipService.HasByHierarchicalName(strSourceFileName) Then Exit
Sub

  ' Получить входной поток для исходного архивного файла.
  objPackageStream = objZipService.GetByHierarchicalName(strSourceFileName)
  objInputStream = objPackageStream.GetInputStream()

  ' Определяем вывод.
  objOutputStream = createUnoService("com.sun.star.ucb.SimpleFileAccess")
  objOutputStream.WriteFile(ConvertToURL(strDestinationFilePath),
objInputStream)
End Sub

Sub unzipArchive( _
  strZipArchivePath As String, _
  strDestinationFolder As String)

  Dim args(0)           As Variant
  Dim objZipService     As Variant
  Dim objPackageStream  As Variant
  Dim objOutputStream   As Variant
  Dim objInputStream    As Variant
  Dim arrayNames()      As Variant
```

```
  Dim strNames          As String
  Dim i                 As Integer

  ' Разархивируем целиком .zip архив в папку для результата.
  ' strZipArchivePath = полный путь (папки и файл) к архивному файлу .zip
  ' strDestinationFilePath = папка (только директория), куда исходные файлы
будут помещены.

  ' Создаем handle для сервиса работы с архивами zip
  objZipService = createUnoService("com.sun.star.packages.Package")
  args(0) = ConvertToURL(strZipArchivePath)
  objZipService.initialize(args())

  ' Получаем вывод пакета, содержащий архив целиком.
  objPackageStream = objZipService.GetByHierarchicalName("")

  ' Получаем список всех файлов в архиве
  arrayNames = objPackageStream.getElementNames()

  ' Выполняем для каждого файла в указанном архиве и выводим из архива файл
в указанную для вывода папку
  For i = LBound(arrayNames) To UBound(arrayNames)
    strNames = strNames & arrayNames(i) & Chr(13)

    ' Прочесть и вывести один файл за один раз в файловую систему
    objInputStream =
objZipService.GetByHierarchicalName(arrayNames(i)).GetInputStream()
    objOutputStream = createUnoService("com.sun.star.ucb.SimpleFileAccess")
    objOutputStream.WriteFile(ConvertToURL(strDestinationFolder &
arrayNames(i)),_
       objInputStream)
  Next
  MsgBox strNames
End Sub
```

III.40.2. Упаковать в формат Zip целые папки

Laurent Godard представил также следующий пример. Этот макрос упаковывает содержимое папки включая подпапки (subdirectories)

Лист. 113: Создать zip-файл.

```
sub ExempleAppel
  call ZipDirectory("C:\MesFichiers\Ooo\Rep","C:\resultat.zip")
end sub

REM Пути НЕ должны быть формата URL.
REM Предупреждение: созданный ZIP файл содержит две дополнительные
составляющие.
REM (1) Мета-информация, содержащая файл-содержание архива.
REM (2) Файл mimetype file нулевой длины.
sub ZipDirectory(sSrcDir As String, sZipName As String)
  'Author: Laurent Godard — listes.godard@laposte.net
  'Modified: Andrew Pitonyak
```

```
  Dim sDirs() As String
  Dim oUcb         ' com.sun.star.ucb.SimpleFileAccess
  Dim oZip         ' com.sun.star.packages.Package
  Dim azipper
  Dim args(0)      ' Инициализировать пакет сервисов для  zip файлов для
заданного имени архива.
  Dim argsDir(0)   ' Установить равным true, чтобы включить директории в
архив zip
  Dim sBaseDir$
  Dim i%
  Dim chaine$
  Dim decoupe      ' Каждый компонент директории в массиве.
  Dim repZip
  Dim RepPere
  Dim RepPereZip
  Dim sFileName$
  Dim oFile        ' File stream

  'Создать пакет!
  oZip=createUnoService("com.sun.star.packages.Package")
  args(0)=ConvertToURL(sZipName)
  oZip.initialize(args())

  'Создание структуры директорий для архива zip
  call Recursedirectory(sSrcDir, sDirs())

  argsDir(0)=true

  'on saute le premier --> repertoire contenant
  'Pourra etre une option a terme
  sBaseDir=sDirs(1)

  For i=2 To UBound(sDirs)
    chaine=mid(sDirs(i),len(sBaseDir)+2)
    decoupe=split(mid(sDirs(i),len(sBaseDir)+1),getPathSeparator())
    repZip=decoupe(UBound(decoupe))
    azipper=oZip.createInstanceWithArguments(argsDir())

    If len(chaine)<>len(repZip) then
      RepPere=left(chaine,len(chaine)-len(repZip)-1)
      RepPere=RemplaceChaine(reppere, getpathseparator, "/", false)
    Else
      RepPere=""
    Endif

    RepPereZip=oZip.getByHierarchicalName(RepPere)
    RepPereZip.insertbyname(repzip, azipper)
  Next i

  'insertion des fichiers dans les bons repertoires
  dim args2(0)
  args2(0)=false
  oUcb = createUnoService("com.sun.star.ucb.SimpleFileAccess")

  for i=1 to UBound(sDirs)
```

```
    chaine=mid(sDirs(i),len(sBaseDir)+2)
    repzip=remplacechaine(chaine, getpathseparator, "/", false)
    sFileName=dir(sDirs(i)+getPathSeparator(), 0)
    While sFileName<>""
      azipper=oZip.createInstanceWithArguments(args2())
      oFile = oUcb.OpenFileRead(ConvertToURL(sDirs(i)+"/"+sFileName))
      azipper.SetInputStream(ofile)
      RepPere=oZip.getByHierarchicalName(repZip)
      RepPere.insertbyname(sFileName, azipper)
      sFileName=dir()
    Wend
  next i

  'Valide les changements
  oZip.commitChanges()
  msgbox "Finished"
End Sub

REM Читать имена директорий
Sub RecurseDirectory(sRootDir$, sDirs As Variant)
  'Author: laurent Godard — listes.godard@laposte.net
  'Modified: Andrew Pitonyak
  Redim Preserve sDirs(1 to 1)

  Dim nNumDirs%    ' Отслеживает число директорий или файлов
  Dim nCurIndex%   ' Текущий индекс директорий или файлов
  Dim sCurDir$     ' Текущая директория.

  nNumDirs=1
  sDirs(1)=sRootDir
  nCurIndex=1
  sCurDir = dir(ConvertToUrl(sRootDir & "/"), 16)

  Do While sCurDir <> ""
    If sCurDir <> "." and sCurDir<> ".." Then
      nNumDirs=nNumDirs+1
      ReDim Preserve sDirs(1 to nNumDirs)
      sDirs(nNumDirs)=convertfromurl(sDirs(nCurIndex)+"/"+sCurDir)
    endif
    sCurDir=dir()

    Do While sCurDir = "" and nCurIndex < nNumDirs
      nCurIndex = nCurIndex+1
      sCurDir=dir(convertToURL(sDirs(nCurIndex)+"/"),16)
    Loop
  Loop
End Sub

Function RemplaceChaine(ByVal sSearchThis$, sFindThis$, dest$, bCase As
Boolean)
  'Auteurs: Laurent Godard & Bernard Marcelly
  ' fournit une sSearchThis dont toutes les occurences de sFindThis ont été
remplacées
  ' par dest
  ' bCase = true pour distinguer majuscules/minuscules, = false sinon
```

```
  Dim nSrcLen As Integer
  Dim i%          ' Текущий индекс.
  Dim nUseCase%   ' Аргумент для функции InStr, задает, нужно ли отличать
большие и малые буквы.
  Dim sNewString As String

  sNewString=""
  nUseCase = IIF(bCase, 0, 1)
  nSrcLen = len(sFindThis)
  i = instr(1, sSearchThis, sFindThis, nUseCase)

  REM Пока While nSearchThis содержит sFindThis
  Do While i<>0
    REM Если положение найдено после 32K байт, то удаляем первые 32000
символов.
    REM Это делается, чтобы избежать отрицательных величин.
    Do While i<0
      sNewString = sNewString & Left(sSearchThis,32000)
      sSearchThis = Mid(sSearchThis,32001)
      i=InStr(1, sSearchThis, sFindThis, nUseCase)
    Loop

    If i>1 Then
       sNewString = sNewString & Left(sSearchThis, i-1) & dest
    else
       sNewString = sNewString & dest
    endif
    ' raccourcir en deux temps car risque : i+src > 32767
    sSearchThis = Mid(sSearchThis, i)
    sSearchThis = Mid(sSearchThis, 1+nSrcLen)
    i = instr(1, sSearchThis, sFindThis, nUseCase)
  Loop
  RemplaceChaine = sNewString & sSearchThis
End Function
```

III.41. Выполнить макрос по строке с его именем

Заданное имя макроса — процедуры или функции — может быть вызвано с использованием диспетчера API. Это полезно, когда нужная процедура, которую потребуется вызвать, не может ещё быть определена во время написания макроса. Например, список процедур для вызова хранится во внешнем файле. Благодарности Paolo Mantovani за следующее решение:

Лист. 114: Выполнить макрос, заданный именем в виде строки.

```
Sub RunGlobalNamedMacro
  oDisp = createUnoService("com.sun.star.frame.DispatchHelper")
  sMacroURL = "macro:///Gimmicks.AutoText.Main"
  oDisp.executeDispatch(StarDesktop, sMacroURL, "", 0, Array())
End Sub
```

Заметьте, что рабочий стол desktop использован как объект, который обрабатывает задачу диспетчера (dispatch).

III.41.1. Выполнить макрос из командной строки

Выполнить макрос из командной строки по заданному имени::

```
soffice.exe macro:///standard.module1.macro1
```

В этом примере «standard» — имя библиотеки, «module1» — имя модуля, «macro1» — имя макроса

> ***Совет*** *Если макрос ничего не создаёт и не открывает внутри документа, то макрос встроенный и StarOffice после этого закрывается.*

III.41.2. Выполнить указанный макрос в документе

Все вышеприведенные примеры выполняли макросы, содержащиеся в глобальном контейнере объектов. Можно также выполнить макрос, который находится в документе.

Лист. 115: Выполнить макрос, находящийся в документе, по его имени в текстовой строке.

```
Sub RunDocumentNamedMacro
  Dim oDisp
  Dim sMacroURL As String
  Dim sMacroName As String
  Dim sMacroLocation As String
  Dim oFrame

  oDisp = createUnoService("com.sun.star.frame.DispatchHelper")

  REM Чтобы получить явно URL, добавьте кнопку и назначьте для нее
  REM исполнение нужного макроса.
  sMacroName = "vnd.sun.star.script:Standard.Module1.MainExternal"
  sMacroLocation = "?language=Basic&location=document"
  sMacroURL = sMacroName &  sMacroLocation

  REM Чтобы вызвать макрос текущего документа ThisComponent,
  REM нужно использовать фрейм frame из документа, содержащего этот макрос
  REM в качестве получателя для диспетчера dispatch
  oFrame = ThisComponent.CurrentController.Frame
  oDisp.executeDispatch(oFrame, sMacroURL, "", 0, Array())
  'oDisp.executeDispatch(StarDesktop, sMacroURL, "", 0, Array())
End Sub
```

По-моему, это можно сделать ещё проще... Я сохранил это в документе «delme.odt», так что я могу просто использовать следующий адрес URL, даже если я использую StarDesktop в качестве получателя для диспетчера (dispatch receiver):

```
sMacroURL = «macro://delme/Standard.Module1.MainExternal»
```

III.42. Использование «приложения по умолчанию» для открытия файла

Когда я использую компьютер под Windows, первым делом я устанавливаю 4NT от фирмы JP Software (`http://www.jpsoft.com`) потому что я использую командную строку. Когда я хочу открыть файл PDF, я просто набираю имя файла и нажимаю enter. Windows смотрит на расширение файла и затем автоматически открывает программу чтения PDF. Эквивалент в GUI — двойной клик на файле, и он открывается соответствующим приложением.

Вы можете делать то же самое с использванием OOo, применяя сервис SystemShellExecute. (Благодарю Russ Phillips [`avantman42@users.sourceforge.net`] за указание на результаты автора Erik Anderson `http://www.oooforum.org/forum/viewtopic.php?t=6657`)

Волшебство выполняется сервисом SystemShellExecute, который содержит один метод; выполните!

Лист. 116: Открыть файл в зависимости от приложения по умолчанию.

```
Sub LaunchOutsideFile()
  Dim oSvc as object
  oSvc = createUnoService("com.sun.star.system.SystemShellExecute")

  Rem File:
  ' oSvc.execute(ConvertToUrl("C:\sample.txt"), "", 0)
  Rem Folder:
  ' oSvc.execute(ConvertToUrl("C:\Program Files\OpenOffice.org1.1.0"), "",
0)
  Rem Web address:
  ' oSvc.execute("http://www.openoffice.org/", "", 0)
  Rem Email:
  ' oSvc.execute("mailto:anonymous@ftp.com", "", 0)
End Sub
```

III.43. Распечатка перечня шрифтов

Доступные шрифты известны контейнеру окна (window). Метод getFontDescriptors() возвращает массив структур AWT FontDescriptor, который содержим множество информации о шрифтах. Дескриптор шрифта может быть передан методу getFont(), который возвращает объект, который поддерживаент интерфейс AWT Xfont. Интерфейс Xfont предоставляет методы для определения метрик шрифтов и ширины отдельного символа или всей строки символов.

Лист. 117: Выдать список шрифтов.

```
Sub ListFonts
  Dim oWindow    'Контейнер окна window поддерживает интерфейс awt XDevice.
  Dim oDescript  'Массив структур awt FontDescriptor
  Dim s$         'Временная строковая переменная, содержащая все строковые
имена.
  Dim i%         'Переменная общего индекса

  oWindow =
ThisComponent.getCurrentController().getFrame().getContainerWindow()
  oDescript = oWindow.getFontDescriptors()
  s = ""

  For i = LBound(oDescript) to UBound(oDescript)
    s = s & oDescript(i).Name & ", "
  Next
  MsgBox s
End Sub
```

III.44. Получить для документа: адрес URL, имя файла и папку

Не пытайтесь получить адрес URL документа, если он не имеет URL. Если документ ещё не был сохранен, например. Прежде чем записывать мои собственные процедуры, я использую некоторые функции из модуля Strings, находящегося в библиотеке Tools.

Лист. 118: Извлечение информации о файле и пути из адреса URL.

```
'Author: Andrew Pitonyak
Sub DocumentFileNames
  Dim oDoc
  Dim sDocURL
  oDoc = ThisComponent
  If (Not GlobalScope.BasicLibraries.isLibraryLoaded("Tools")) Then
    GlobalScope.BasicLibraries.LoadLibrary("Tools")
  End If
  If (oDoc.hasLocation()) Then
    sDocURL = oDoc.getURL()
    Print "Директория = " & DirectoryNameoutofPath(sDocURL, "/")
    Print "Имя файла = " & FileNameoutofPath(sDocURL, "/")
  End If
End Sub
```

III.45. Получить и установить текущую папку (directory)

В OOo команды ChDir и ChDrive пока не делают ничего — это пока в проектах на будущее. Следующий пример основан на дискуссии между Andreas Bregas, Christian Junker, Paolo Mantovani и Andrew Pitonyak.

Первоначально операторы ChDir и ChDrive производили вызовы системы, но нои были переписаны с использованием уровня UCB — as was all file system related functionality. Это было сделано потому, что все команды теперь принимают также синтаксис URL. Поддержка текущей папки (directory) не поддерживается в UCB и используемой в ней sal/osl API, из-за внутренних проблем (ошибок) в среде множественных нитей (threaded). В Windows, например, диалог Открыть файл меняет текущую папку процесса, как и другие вызовы API. Вы ожидаете одного значения текущей папки, но уже другая нить (thread) изменила её. Paolo рекомендует использование сервиса PathSettings, который содержит множество значений пути (см. Таблицу Таблица 6).

Таблица 6. Свойства, поддерживаемые сервисом com.sun.star.util.PathSettings.

Свойство	***Описание***
Backup	Автоматическое копирование документов описано здесь.
Basic	Файлы Basic, использованные в AutoPilots, можно найти здесь. Значение может содержать несколько путей, разделенных точкой с запятой.
Favorite	Путь для сохранения папки с закладками (bookmarks)
Gallery	Расположение базы данных Галереи (Gallery) и мультимедийных файлов. Значение может содержать несколько путей, разделенный точкой с запятой.
Graphic	Эта папка (directory) выводится, когда вызывается диалог открытия графического объекта или для сохранения нового графического объекта.
Help	Путь к файлам помощи Office.
Module	Это путь к модулям.
Storage	Здесь хранится информация о почте, новостях и других (например, о FTP Сервере).
Temp	Базовый адрес URL для временных файлов office
Template	Исходные шаблоны берутся из этих папок и подпапок. Значение может содержать несколько путей, разделенных точкой с запятой.
UserConfig	Папка, которая содержит настройки пользователя.
Work	Рабочая папка пользователя, которая может быть изменена — используется диалогами Открыть и Сохранить.

Следующий код программ поясняет, как это работает:

Лист. 119: Использовать сервис PathSettings.

```
'Author: Paolo Mantovani
Function pmxCurDir() As String
  Dim  oPathSettings
  oPathSettings = CreateUnoService("com.sun.star.util.PathSettings")

  'Путь к рабочей папке можно изменить при необходимости.
  'Указанный здесь путь можно увидеть в диалогах Открыть или Сохранить.
  pmxCurDir = oPathSettings.Work
End Function

Function pmxChDir(sNewDir As String) As String
  Dim  oPathSettings
  oPathSettings = CreateUnoService("com.sun.star.util.PathSettings")
  oPathSettings.Work = ConvertToUrl(sNewDir)
  pmxChDir = oPathSettings.Work
End Function
```

Существует также сервис com.sun.star.util.PathSubstitution, который предоставляет доступ ко многим интересным значениям, связанным с путями. Переменные пути не чувствительны к регистру (большим и малым буквам) и всегда возвращают адрес URL, согласованый с UCB, например, «file:///c:/temp» или «file:///usr/install». Поддерживаемый список значений хранится в файле конфигурации Office (org/openoffice/Office/Substitution.xml). Переменные с предустановленными значениями следующие:

Таблица 7. Переменные, распознаваемые сервисом com.sun.star.util.PathSubstitution.

Переменная	***Описание***
$(inst)	Путь установки Office.
$(prog)	Путь к программам Office.
$(user)	Папка установки пользователя (The user installation directory).
$(work)	Рабочая папка пользователя. Под Windows это подпапка «MyDocuments». Под Unix это домашняя папка (home-directory)
$(home)	Домашняя папка пользователя. Под Unix это домашняя папка (home-directory). Под Windows это подпапка «Documents and Settings «.
$(temp)	Текущая папка для временных файлов.
$(path)	Значение переменной среды PATH
$(lang)	Код страны, используемый в Office, например, 01=english

Переменная	*Описание*
	(английский), 49=german (немецкий).
$(langid)	Код языка, используемый в Office, например, 0x0009=english(английский), 0x0409=english us (американский английский).
$(vlang)	Язык, используемый в Office в виде строки. Например, «german» для немецкого Office.

Лист. 120: Использовать сервис PathSubstitution.

```
oPathSubst = createUnoService("com.sun.star.util.PathSubstitution")
Print oPathSubst.getSubstituteVariableValue("$(inst)")
```

III.46. Запись файла

Методы, предоставляемые самим BASIC, имеют некоторые недостатки и интересное поведение. В моей книге это обсуждается в четырех секциях. Я недавно нашел интересный кусок у автора Christian Junker, которого мне иногда нехватало для изучения. Вы можете установить кодировку текста, используя метод setEncoding().

Лист. 121: Метод SimpleFileAccess позволяет увидеть формат вывода.

```
fileAccessService = createUnoService("com.sun.star.ucb.SimpleFileAccess")
textOutputStream = createUnoService("com.sun.star.io.TextOutputStream")
'открываем файл.
outputStream = fileAccessService.openFileWrite(<yourFileName>)
outputStream.truncate()
textOutputStream.setOutputStream(outputStream)
'Теперь записываем что-нибудь в этот файл
textOutputStream.writeString("Это формат utf-8 ")
'и не забывайте закрыть файл.
textOutputStream.closeOutput()
```

III.47. Логический разбор синтаксиса (Parsing) XML

Прекрасный пример разбора синтаксиса (parsing) XML предоставил автор DannyB на форуме oooforum

`http://www.oooforum.org/forum/viewtopic.php?t=4907`

Вам следует прочесть его до использования следующего макроса:

Лист. 122: Разбор синтаксиса (Parsing) XML.

```
Sub Main
   cXmlFile = "C:\TestData.xml"
```

```
   cXmlUrl = ConvertToURL( cXmlFile )

   ReadXmlFromUrl( cXmlUrl )
End Sub

' Процедура демонстрирует, как использовать из Universal Content Broker
'  сервис SimpleFileAccess для чтения из локального файла.
Sub ReadXmlFromUrl( cUrl )
   ' Сервис SimpleFileAccess позволяет открывать, читать, записывать файлы,
   '  а также просматривать оглавления папок.
   ' Преимущества этого метода по сравнению с неудачной реализацией
обращения с файлами в языке Basic
   '  в том, что этот способ работает в любом языке программирования.
   '  Более того, программа может выполняться на одном компьютере в то время
как сервис SimpleFileAccess
   '  обращается к файлам от имени компьютера, где исполняется OOo,
   '  а не компьютера, где исполняется по сети программа, например,  Java
или Python.
   oSFA = createUnoService( "com.sun.star.ucb.SimpleFileAccess" )

   ' Открываем исходный файл.
   oInputStream = oSFA.openFileRead( cUrl )

   ReadXmlFromInputStream( oInputStream )

   oInputStream.closeInput()
End Sub

Sub ReadXmlFromInputStream( oInputStream )
   ' Создаем обработчик синтаксиса Sax Xml parser.
   oSaxParser = createUnoService( "com.sun.star.xml.sax.Parser" )

   ' Создаем объект обработчика событий документа.
   ' Когда вызываются методы этого объекта, язык Basic выполняет
   '  вызов глобальных процедур (см.ниже).
   oDocEventsHandler = CreateDocumentHandler()

   ' Подключаем наш обработчик событий к обработчику синтаксиса parser.
   ' Так как обработчик синтаксиса (parser) выполняет чтение документа Xml,
то он вызывает методы
   '  этого объекта и, поэтому, вызываются глобальные процедуры (ниже),
   '  чтобы известить их, что просматривается в документе Xml.
   oSaxParser.setDocumentHandler( oDocEventsHandler )

   ' Создаем структуру InputSource.
   oInputSource = createUnoStruct( "com.sun.star.xml.sax.InputSource" )
   With oInputSource
      .aInputStream = oInputStream   ' подключаем поток входных данных input
stream
   End With

   ' Теперь синтаксический разбор документа.
   ' Здесь читается документ целиком.
   ' Методы объекта oDocEventsHandler вызываются,
```

```
    '  когда документ просматривается.
    oSaxParser.parseStream( oInputSource )
End Sub

'  Обработчик событий документа Xml Sax.

' Глобальные переменные, используемые нашим обработчиком событий документа
'
' Когда обработчик синтаксиса Sax parser передаст нам расположение документа
document locator,
'  переменная glLocatorSet устанавливается равной True,
'  и переменная goLocator будет содержать объект расположения locator
object.
'
' Методы объекта расположения locator object включают мощные методы,
'  которые могут сообщить, где внутри текущего документа Xml
'  происходил синтаксический разбор в тот момент, когда произошло событие
Sax.
' Объект расположения locator object включает com.sun.star.xml.sax.XLocator.
'
Private goLocator As Object
Private glLocatorSet As Boolean


' Здесь создается объект, который включает интерфейс
'  com.sun.star.xml.sax.XDocumentHandler.
' Обработчик событий документа doucment handler возвращается как результат
выполнения следующей функции.
Function CreateDocumentHandler()
    ' Используем функцию CreateUnoListener языка Basic.
    ' Язык Basic создает и возвращает объект, который включает конкретный
интерфейс.
    ' Когда вызываются методы этого объекта,
    '  язык Basic вызовет глобальные функции Basic с теми же именами, что и
имена этих методов,
    '  но с добавлением в начало имени определенного префикса.
    oDocHandler = CreateUnoListener( "DocHandler_",
"com.sun.star.xml.sax.XDocumentHandler" )

    glLocatorSet = False

    CreateDocumentHandler() = oDocHandler
End Function

'   Методы нашего обработчика событий документа
'    вызывают указанные глобальные функции.
'   Эти методы выглядят странно похожими на
'    обработчик событий SAX.  ;-)
'   Эти глобальные переменные вызываются обработчиком синтаксиса Sax
'    в то время, как он разбирает синтаксис документа XML.
'   Эти процедуры должны иметь имена с префиксом, за которым следует
'    имя интерфейса  com.sun.star.xml.sax.XDocumentHandler

Sub DocHandler_startDocument()
    Print "Начало документа"
```

```
End Sub

Sub DocHandler_endDocument()
'   Print "Конец документа"
End Sub

Sub DocHandler_startElement( cName As String, oAttributes As
com.sun.star.xml.sax.XAttributeList )
   Print "Начало элемента ", cName
End Sub

Sub DocHandler_endElement( cName As String )
'   Print "Конец элемента ", cName
End Sub

Sub DocHandler_characters( cChars As String )
End Sub

Sub DocHandler_ignorableWhitespace( cWhitespace As String )
End Sub

Sub DocHandler_processingInstruction( cTarget As String, cData As String )
End Sub

Sub DocHandler_setDocumentLocator( oLocator As com.sun.star.xml.sax.XLocator
)
   ' Сохраняем объект расположения locator object в глобальной переменной.
   ' Объект расположения имеет полезные методы, которые можно
   '  вызывать для определения
   goLocator = oLocator

   glLocatorSet = True
End Sub
```

DannyB рекомендует начинать со следующего маленького XML-файла для первоначальной проверки:

```
<Employees>
   <Employee id="101">
      <Name>
         <First>John</First>
         <Last>Smith</Last>
      </Name>
      <Address>
         <Street>123 Main</Street>
         <City>Lawrence</City>
         <State>KS</State>
         <Zip>66049</Zip>
      </Address>
      <Phone type="Home">785-555-1234</Phone>
   </Employee>
   <Employee id="102">
      <Name>
         <First>Bob</First>
         <Last>Jones</Last>
```

```
      </Name>
      <Address>
         <Street>456 Puke Drive</Street>
         <City>Lawrence</City>
         <State>KS</State>
         <Zip>66049</Zip>
      </Address>
      <Phone type="Home">785-555-1235</Phone>
   </Employee>
</Employees>
```

III.48. Работа с датами Dates

В моей книге полностью рассмотрена работа с датами, вместе со всеми их сложностями. Напомню, что дробная часть представляет время и целая часть представляет дни. Поэтому можно просто добавить число дней к объекту типа дата для увеличения текущей даты.

Лист. 123: Сложение двух дней (точнее, добавление числа дней к дате) — это легко.

```
Function addDays(StartDate as Date, nDays As Integer) As Date
  REM Чтобы добавить дни, просто выполните сложение
  addDay = StartDate + nDays
End Function
```

Хотя добавить заданное число дней к заданной дате — легкая задача, есть проблемы при добавлении лет или месяцев. Февраль содержит 29 дней в 2004 и 28 дней в 2005 году. Поэтому нельзя просто добавить единицу к году и спать спокойно; аналогичные проблемы существуют при добавлении месяцев. Вы можете решить, что это означает добавить единицу к году или к месяцу. Первый вариант процедуры предложил Eric VanBuggenhaut, но он не обрабатывал правильно подобные ситуации. Antoine Jarrige заметил некорректность поведения и предложил решение, но проблемы ещё оставались при добавлении 12 месяцев к дате в декабре.

Я переписал почти все, используя приемы, представленные в моей книге по макросам. Окончательный код программы добавляет и годы, и месяцы. При добавлении годов и месяцев первоначальная дата, которая начинается в последний день месяца, остается последней датой месяца. Когда это выполнено, добавляются дни. Если это не то, чего Вы хотели, то измените этот макрос.

Функция SumDate добавляет заданное число лет, месяцев и дней к указанной дате. Первый недостаток этой процедуры — то, что она отбрасывает время и неправильно обрабатывает даты с номером года меньше 100.

Лист. 124: Добавить годы, месяцы и дни к дате.

```
Function SumDate(StartDate As Date, nYears%, nMonths%, nDays%) As Date
  REM Author: Eric Van Buggenhaut [Eric.VanBuggenhaut@AdValvas.be]
  REM Modified By:
  REM         Antoine Jarrige [pierre-antoine.jarrige@laposte.net]
  REM Almost complete rewrite by Andrew Pitonyak

  Dim lDateValue As Long     ' Начальная дата – длинное целое
  Dim nDateDay   As Integer  ' День для начальной даты
  Dim nDateMonth As Integer  ' Месяц
  Dim nDateYear  As Integer  ' Год
  Dim nLastDay_1 As Integer  ' Последний день месяца для начальной даты
  Dim nLastDay_2 As Integer  ' Последний день месяца для даты-результата

  REM Определяем год, месяц и день
  nDateDay   = Day(StartDate)
  nDateMonth = Month(StartDate)
  nDateYear  = Year(StartDate)

  REM Находим последний день месяца
  If nDateMonth = 12 Then
    REM December always has 31 days
    nLastDay_1 = 31
  Else
    nLastDay_1 = Day(DateSerial(nDateYear, nDateMonth+1, 1)-1)
  End If

  REM Проблемой добавление года является для 29 февраля високосного года
  nDateYear  = nDateYear + nYears
  nDateMonth = nDateMonth + nMonths
  If nDateMonth > 12 Then
    nDateYear = nDateYear + (nDateMonth — 1) \ 12
    nDateMonth = (nDateMonth — 1) MOD 12 + 1
  End If

  REM Находим последний день месяца
  If nDateMonth = 12 Then
    REM Декабрь всегда содержит 31 день
    nLastDay_2 = 31
  Else
    nLastDay_2 = Day(DateSerial(nDateYear, nDateMonth+1, 1)-1)
  End If

  REM Обеспечиваем, что последний день месяца останется последним днем
  REM месяца. Не позволяем переход на следующий месяц
  REM Проблемой является добавление одного месяца к 31 января, чтобы
  REM не попасть в март.
  If nDateDay = nLastDay_1 OR nDateDay > nLastDay_2 Then
    nDateDay = nLastDay_2
  End If

  REM А при добавлении дней все эти ограничения не нужны.
  SumDate=CDate(DateSerial(nDateYear, nDateMonth, nDateDay)+nDays)
End Function
```

III.49. Встроен ли OpenOffice в Веб-браузер?

OpenOffice может открывать документ прямо в Вашем Веб-браузере. OpenOffice поддерживает недокументированное (и используемое внутри OOo) свойство isPlugged, которое показывает, подключен ли рабочий стол к браузеру.

```
Stardesktop.isPlugged()
```

Хотя это работает в версии OOo 1.1.2, юмор в том, что метод isPlugged будет удален в версии 2.0.

III.50. Активизировать (поставить на первый план) новый документ

Чтобы заставить документ, на который ссылается переменная oDoc2, стать активным (focused), используйте один из следующих двух методов:

Лист. 125: Сделать текущее окно активным.

```
oDoc2.CurrentController.Frame.ContainerWindow.toFront()
oDoc2.CurrentController.Frame.Activate()
```

Это не меняет значения объекта ThisComponent.

III.51. Каков тип документа (основываясь на адресе URL)

Christian Junker заметил, что можно использовать «глубокое» исследование для определения типа документа. Это означает, что корректный тип возвращается, даже если расширение файла не показывает правильного типа. Другими словами, этот способ сможет определить тип документ OOo Calc даже для документа с расширением .doc. Возвращаемая строка является внутренним именем формата.

Лист. 126: Определить тип документа.

```
Sub DetectDocType()
  Dim oMediaDescr(30) As new com.sun.star.beans.PropertyValue
  Dim s$ : s$ = "com.sun.star.document.TypeDetection"
  Dim oTypeManager

  oMediaDescr(0).Name = "URL"
  oMediaDescr(0).Value = ThisComponent.getURL()

  oTypeManager = createUnoService(s$)
  REM Perform a deep type detection
  REM not just based on filename extension.
  MsgBox oTypeManager.queryTypeByDescriptor(oMediaDescr(), True)
End Sub
```

III.52. Соединиться с удаленным сервером OOo с использованием Basic

Можно соединиться с удаленным OOo сервером, используя Basic.

Лист. 127: Соединиться с удаленным сервером.

```
Sub connectToRemoteOffice()
  REM Author: Christian Junker
  REM Author: Modified by Andrew Pitonyak
  Dim sURL$    ' Адрес URL для соединения с удаленным сервером.
  Dim sHost$   ' IP-адрес, где работает удаленный сервер.
  Dim sPort$   ' Порт, используемый удаленным сервером.
  Dim oRes     ' Обработчик адресов URL.
  Dim oRemote  ' Удаленный менеджер (клиент) для удаленного сервера.
  Dim oDesk    ' Объект рабочего стола Desktop удаленного сервера.
  Dim oDoc     ' Открытый документ.

  REM Устанавливаем адрес и порт работающего сервера. Сервер должен
  REM быть запущен для прослушивания указанного порта:
  REM Если не указать "host=0", сервер не будет принимать
  REM подключения по сети. Например, если стартовать
  REM soffice.exe на компьютере под  windows со следующими аргументами:
  REM "-accept=socket,host=0,port=8100;urp;StarOffice.ServiceManager"

  sHost = "192.168.0.5"
  sPort = "8100"
  sURL  = "uno:socket,host=" & sHost & _
          ",port=" & sPort & _
          ";urp;StarOffice.ServiceManager"
  oRes = createUNOService("com.sun.star.bridge.UnoUrlResolver")
  oRemote = oRes.resolve(sURL)
  oDesk = oRemote.createInstance("com.sun.star.frame.Desktop")

  REM Задаем, какой документ открыть!
  sURL = "private:factory/swriter"
  'sURL = "file:///home/andy/PostData.doc"
  oDoc = oDesk.loadComponentFromURL(sURL, "_blank", 0, Array())
  Print "Документ открыт"
End Sub
```

Вот ещё вопрос: посещали ли Вы сайт oood.py? Простой демон для OpenOffice.org.

`http://udk.openoffice.org/python/oood/`

III.53. Панели инструментов[1]

Панели инструментов имеет имена. Пользовательские панели инструментов все начинаются с «private:resource/toolbar/custom_». Стандартные имена панелей инструментов показаны в Лист. 128.

Лист. 128: Стандартные имена панелей инструментов.

```
Sub PrintStandardToolBarNames()
  MsgBox Join(GetStandardToolBarNames(), CHR$(10))
End Sub

Function GetStandardToolBarNames()
  GetStandardToolBarnames = Array ( _
            "private:resource/toolbar/alignmentbar", _
            "private:resource/toolbar/arrowshapes", _
            "private:resource/toolbar/basicshapes", _
            "private:resource/toolbar/calloutshapes", _
            "private:resource/toolbar/colorbar", _
            "private:resource/toolbar/drawbar", _
            "private:resource/toolbar/drawobjectbar", _
            "private:resource/toolbar/extrusionobjectbar", _
            "private:resource/toolbar/fontworkobjectbar", _
            "private:resource/toolbar/fontworkshapetypes", _
            "private:resource/toolbar/formatobjectbar", _
            "private:resource/toolbar/formcontrols", _
            "private:resource/toolbar/formdesign", _
            "private:resource/toolbar/formsfilterbar", _
            "private:resource/toolbar/formsnavigationbar", _
            "private:resource/toolbar/formsobjectbar", _
            "private:resource/toolbar/formtextobjectbar", _
            "private:resource/toolbar/fullscreenbar", _
            "private:resource/toolbar/graphicobjectbar", _
            "private:resource/toolbar/insertbar", _
            "private:resource/toolbar/insertcellsbar", _
            "private:resource/toolbar/insertobjectbar", _
            "private:resource/toolbar/mediaobjectbar", _
            "private:resource/toolbar/moreformcontrols", _
            "private:resource/toolbar/previewbar", _
            "private:resource/toolbar/standardbar", _
            "private:resource/toolbar/starshapes", _
            "private:resource/toolbar/symbolshapes", _
            "private:resource/toolbar/textobjectbar", _
            "private:resource/toolbar/toolbar", _
            "private:resource/toolbar/viewerbar", _
            "private:resource/menubar/menubar" _
  )
End Function
```

[1] Этот раздел ещё не дописан автором.

Используйте для фреймов метод LayoutManager для того, чтобы найти текущие панели инструментов. На следующем примере Лист. 129 показано, как вывести на экран панели управления, меню, строки состояния.

Лист. 129: Вывести на экран панели инструментов в текущем документе.

```
Sub SeeComponentsElements()
  Dim oDoc, oFrame
  Dim oCfgManager
  Dim oToolInfo
  Dim x
  Dim s$
  Dim iToolType as Integer

  oDoc = ThisComponent
  REM Это на самом деле целое число 3.
  iToolType = com.sun.star.ui.UIElementType.TOOLBAR

  oFrame = oDoc.getCurrentController().getFrame()
  oCfgManager = oDoc.getUIConfigurationManager()
  oToolInfo = oCfgManager.getUIElementsInfo( iToolType )
  For Each x in oFrame.LayoutManager.getElements()
    s = s & x.ResourceURL & CHR$(10)
  Next
  MsgBox s, 0, "Панели компонента OpenOffice.org Toolbars in Component"
End Sub
```

Используйте менеджер раскладки экрана для того, чтобы узнать, видима ли сейчас конкретная панель инструментов. Метод isElementVisible проверяет все типы элементов, не только панели инструментов.

Лист. 130: See if a specified toolbar is visible.

```
Sub TestToolBarVisible
  Dim s$, sName$
  For Each sName In GetStandardToolBarnames()
    s = s & IsToolbarVisible(ThisComponent, sName) & _
        " : " & sName & CHR$(10)
  Next
  MsgBox s, 0, "Панель видима Toolbar Is Visible"
End Sub

Function IsToolbarVisible(oDoc, sURL) As Boolean
  Dim oFrame
  Dim oLayout

  oFrame = oDoc.getCurrentController().getFrame()
  oLayout = oFrame.LayoutManager
  IsToolbarVisible = oLayout.isElementVisible(sURL)
End Function
```

Используйте методы hideElement и showElement для того, чтобы скрыть или показать панель инструментов. До версии 2.0 нужно было полагаться на диспетчеры (dispatches), чтобы менять видимость объекта. Например,

следующие диспетчеры были использованы: «.uno:MenuBarVisible» (видимость панели меню), «.uno:ObjectBarVisible» (видимость панели объекта), «.uno:OptionBarVisible» (видимость панели опций), «.uno:NavigationBarVisible» (видимость панели навигации), «.uno:StatusBarVisible» (видимость строки состояния), «.uno:ToolBarVisible» (видимость панели инструментов), «.uno:MacroBarVisible» (видимость панели макросов), «.uno:FunctionBarVisible» (видимость панели функций), и «.uno:InputLineVisible» (видимость входной строки).

Лист. 131: Включить/Выключить видимость панели инструментов.

```
Sub ToggleToolbarVisible(oDoc, sURL)
  Dim oLayout

  oLayout = oDoc.CurrentController.getFrame().LayoutManager
  If oLayout.isElementVisible(sURL) Then
    oLayout.hideElement(sURL)
  Else
    oLayout.showElement(sURL)
  End If
End Sub
```

III.53.1. Создание панели инструментов для типа компонентов

Можно создавать новую панель инструментов без программирования, используя дополнения (add-on). Создайте файл XML, который будет описывать эту панель инструментов, и затем установите его. Я ещё не занимался этим, поэтому не знаю как это сделано.

Используйте менеджер конфигурации документов для создания и сохранения панели инструментов для конкретного документа, вместо конкретного типа компонента.

Чтобы создать панель инструментов, связанную с типом компонента текстовый документ OOo Writer (электронная таблица OOo Calc, Basic IDE, и т.д.), извлеките модуль менеджера конфигурации интерфейса пользвателя и измените этот модуль в части панелей инструментов. (Благодарности Carsten Driesner за эту информацию и основные примеры).

III.53.2. Моя первая панель инструментов (toolbar)

Я буду создавать панель инструментов (toolbar), которая вызывает написанный мной макрос, который располагается в модуле UI, содержащемся в библиотеке PitonyakUtil.

Лист. 132:

```
Sub TBTest
 Print "Выполняется TBTest"
End Sub
```

Каждый элемент панели инструментов является массивом значений свойств.

Лист. 133: Создать простой элемент панели инструментов.

```
' com.sun.star.ui.ItemDescriptor является массивом значений свойств
'Этот пример не устанавливает значения всех поддерживаемых свойтств,
'таких как стиль "Style", который использует значения
'из com.sun.star.ui.ItemStyle. Для элементов меню обычно
'устанавливают также "ItemDescriptorContainer"
Function CreateSimpleToolbarItem( sCommand$, sLabel ) as Variant
  Dim oItem(3) As New com.sun.star.beans.PropertyValue

  oItem(0).Name = "CommandURL"
  oItem(0).Value = sCommand
  oItem(1).Name = "Label"
  oItem(1).Value = sLabel

  REM Другие поддерживаемые типы включают SEPARATOR_LINE,
  rem SEPARATOR_SPACE, и SEPARATOR_LINEBREAK.
  oItem(2).Name = "Type"
  oItem(2).Value = com.sun.star.ui.ItemType.DEFAULT

  oItem(3).Name = "Visible"
  oItem(3).Value = True

  CreateSimpleToolbarItem = oItem()
End Function
```

Создание панели инструментов — это просто.

Лист. 134: Добавить простую панель инструментов к Basic IDE.

```
Sub CreateBasicIDEToolbar
  Dim sToolbarURL$  ' URL пользовательской панели инструметов.
  Dim sCmdID$       ' Команда для отдельной кнопки панели инструментов .
  Dim sCmdLable     ' Метка для отдельной кнопки панели инструметов
  Dim sDocType$     ' Тип компонента, который будет содержать эту панель
  Dim sSupplier$    ' ModuleUIConfigurationManagerSupplier
  Dim oSupplier     ' ModuleUIConfigurationManagerSupplier
  Dim oModuleCfgMgr ' Module manager.
  Dim oTBSettings   ' Настройи, которые включают эту панель
  Dim oToolbarItem  ' Отдельная кнопка панели инструментов
  Dim nCount%

  REM Имя пользовательской панели инструментов; должно начинаться с
"custom_".
  sToolbarURL = "private:resource/toolbar/custom_test"

  REM Получаем менеджер модуля конфигурации
  REM из менеджера поддержки сентрального модуля конфигурации
  sSupplier = "com.sun.star.ui.ModuleUIConfigurationManagerSupplier"
  oSupplier = CreateUnoService(sSupplier)
```

```
  REM Указываем тип документа, ассоциировнный с этой панелью инструментов
  REM sDocType = "com.sun.star.text.TextDocument"
  sDocType = "com.sun.star.script.BasicIDE"

  REM Получить менеджер модуля конфигурации по типу документа
  REM *** См. подробнее com.sun.star.frame.ModuleManager
  oModuleCfgMgr = oSupplier.getUIConfigurationManager( sDocType )

  REM Для удаления панели инструментов можно использовать что-то такое:
  'If (oModuleCfgMgr.hasSettings(sToolbarURL)) Then
  '  oModuleCfgMgr.removeSettings(sToolbarURL)
  '  Exit Sub
  'End If

  REM Создаем контейнер настроек для определения структуры
  REM пользовательской панели инструментов.
  oTBSettings = oModuleCfgMgr.createSettings()

  REM *** Установить заголовок для новой пользовательской панели
инструментов custom toolbar
  oTBSettings.UIName = "My little custom toolbar"

  REM *** Создаем кнопку на новой пользовательсткой панели инструментов
  sCmdID   = "macro:///PitonyakUtil.UI.TBTest()"
  sCmdLable = "Test"
  nCount = 0
  oToolbarItem = CreateSimpleToolbarItem( sCmdID, sCmdLable )
  oTBSettings.insertByIndex( nCount, oToolbarItem )

  REM Чтобы добавить второ элемент, увеличиваем счетчик nCount,
  REM создаем новый элемент панели инструментов и вставляем его

  REM *** Установить настройки для новой панели инструментов
(заменить/вставить)  (replace/insert)
  If ( oModuleCfgMgr.hasSettings( sToolbarURL )) Then
    oModuleCfgMgr.replaceSettings( sToolbarURL, oTBSettings )
  Else
    oModuleCfgMgr.insertSettings( sToolbarURL, oTBSettings )
  End If
End Sub
```

Carsten Driesner предложил макрос, который добавляет кнопку к стандартной панели инструментов (standard toolbar) в текстовом документе Writer.

Лист. 135: Добавить кнопку на стандартную панель инструментов Writer.

```
'Этот макрос создает новую кнопку макроса basic
'на панели инструментов standard OOo Writer. Он не добавляет кнопку дважды.
'Макрос использует менеджер образов  image manager программы Ooo Writer для
'задания внешнего вида этой кнопки вызова макроса

Sub AddButtonToToolbar
  Dim sToolbar$ : sToolbar = "private:resource/toolbar/standardbar"
```

```
  Dim sCmdID$   : sCmdID   = "macro:///Standard.Module1.Test()"
  Dim sDocType$ : sDocType = "com.sun.star.text.TextDocument"
  Dim sSupplier$
  Dim oSupplier
  Dim oModuleCfgMgr
  Dim oImageMgr
  Dim oToolbarSettings
  Dim bHasButton As Boolean
  Dim nCount As Integer
  Dim oToolbarButton()
  Dim nToolbarButtonCount As Integer
  Dim i%, j%

  REM Извлекаем менеджер модуля конфигурации из
  REM менеджера поддержки центрального модуля конфигурации
  sSupplier = "com.sun.star.ui.ModuleUIConfigurationManagerSupplier"
  oSupplier = CreateUnoService(sSupplier)

  REM Извлекаем менеджер модуля конфигурации по типу документа module
identifier
  REM *** См. подробнее com.sun.star.frame.ModuleManager
  oModuleCfgMgr = oSupplier.getUIConfigurationManager( sDocType )
  oImageMgr = oModuleCfgMgr.getImageManager()

  oToolbarSettings = oModuleCfgMgr.getSettings( sToolbar, True )

  REM Ищем нашу кнопку с заданным свойством CommandURL
  bHasButton = False
  nCount = oToolbarSettings.getCount()
  For i = 0 To nCount-1
    oToolbarButton() = oToolbarSettings.getByIndex( i )
    nToolbarButtonCount = ubound(oToolbarButton())
    For j = 0 To nToolbarButtonCount
      If oToolbarButton(j).Name = "CommandURL" Then
        If oToolbarButton(j).Value = sCmdID Then
          bHasButton = True
        End If
      End If
    Next
  Next

  Dim oImageCmds(0)
  Dim oImages(0)
  Dim oImage
  REM *** Проверяем, добавлено ли уже изображение image
  If Not oImageMgr.hasImage( 0, sCmdID ) Then
    REM Пытаемся загрузить изображение по URL-адресу файла
    oImage = GetImageFromURL( "file:///tmp/test.bmp" )
    If Not isNull( oImage ) Then
      REM *** Вставляем новое изображение в менеджер изображений OOo Writer
      oImageCmds(0) = sCmdID
      oImages(0) = oImage
      oImageMgr.insertImages( 0, oImageCmds(), oImages() )
    End If
  End If
```

```
  If Not bHasButton Then
    sString = "My Macro's"
    oToolbarItem = CreateToolbarItem( sCmdID, "Standard.Module1.Test" )
    oToolbarSettings.insertByIndex( nCount, oToolbarItem )
    oModuleCfgMgr.replaceSettings( sToolbar, oToolbarSettings )
  End If
End Sub

Function GetImageFromURL( URL as String ) as Variant
  Dim oMediaProperties(0) As New com.sun.star.beans.PropertyValue
  Dim sProvider$ : sProvider = "com.sun.star.graphic.GraphicProvider"
  Dim oGraphicProvider

  REM Создаем экземпляр поставщика изображений graphic provider для загрузки
изображений из файлов.
  oGraphicProvider = createUnoService( sProvider )

  REM Устанавливаем свойство URL так, чтобы поставщик изображений graphic
provider смог загрузить это изображение
  oMediaProperties(0).Name = "URL"
  oMediaProperties(0).Value = URL

  REM Извлекаем экземпляр com.sun.star.graphic.XGraphic
  GetImageFromURL = oGraphicProvider.queryGraphic( oMediaProperties() )
End Function

Function CreateToolbarItem( Command$, Label$ ) as Variant
  Dim aToolbarItem(3) as new com.sun.star.beans.PropertyValue

  aToolbarItem(0).Name = "CommandURL"
  aToolbarItem(0).Value = Command
  aToolbarItem(1).Name = "Label"
  aToolbarItem(1).Value = Label
  aToolbarItem(2).Name = "Type"
  aToolbarItem(2).Value = 0
  aToolbarItem(3).Name = "Visible"
  aToolbarItem(3).Value = true

  CreateToolbarItem = aToolbarItem()
End Function
```

Позднее я добавлю пример макроса, который демонстрирует, как копировать пользовательскую панель инструментов из одного документа в другой документ. Пока не хватает времени.

Глава IV. Макросы электронных таблиц Calc

IV.1. Является ли этот документ электронной таблицей?

Электронная таблица состоит из набора листов (sheets). До того, как Вы сможете использовать специфические методы электронной таблицы, Вы должны иметь документ-электронную таблицу. Вы можете проверить это так:

Лист. 136: Является ли документ электронной таблицей: используется обработка ошибок.

```
Function IsSpreadsheetDoc(oDoc) As Boolean
  Dim s$ : s$ = "com.sun.star.sheet.SpreadsheetDocument"
  On Local Error GoTo NODOCUMENTTYPE
  IsSpreadsheetDoc = oDoc.SupportsService(s$)
  NODOCUMENTTYPE:
  If Err <> 0 Then
     IsSpreadsheetDoc = False
    Resume GOON
    GOON:
  End If
End Function
```

Если обработка ошибок неважна, потому что данная функция никогда не будет вызвана с пустым аргументом и данный объект будет всегда включать метод supportsService(), то Вы можете использовать такой вариант:

Лист. 137: Является ли документ электронной таблицей Calc, без обработки ошибок.

```
Function IsSpreadsheetDoc(oDoc) As Boolean
  Dim s$ : s$ = "com.sun.star.sheet.SpreadsheetDocument"
  IsSpreadsheetDoc = oDoc.SupportsService(s$)
End Function
'Вы можете проверить этот способ так:
Sub checking( )
  MsgBox IsSpreadsheetDoc(thisComponent)
End Sub
```

IV.2. Вывести для ячейки электронной таблицы значение (value), строковое значение (string) и формулу (formula)

Лист. 138: Доступ к ячейки в документе Calc.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
Sub ExampleGetValue
  Dim oDoc As Object, oSheet As Object, oCell As Object
  oDoc=ThisComponent
  oSheet=oDoc.Sheets.getByName("Sheet1")
  oCell=oSheet.getCellByposition(0,0) 'A1
  Rem содержимое ячейки может иметь один из следующих трех типов:
```

```
  Print oCell.getValue()
  'Print oCell.getString()
  'Print oCell.getFormula()
End Sub
```

IV.3. Установить для ячейки электронной таблицы значение (value), формат (format), текстовое значение (string) и формулу (formula)

Лист. 139: Присвоение ячейке таблицы значения, текстового значения и формулы в электронной таблице Calc

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
Sub ExampleSetValue
  Dim oDoc As Object, oSheet As Object, oCell As Object
  oDoc=ThisComponent
  oSheet=oDoc.Sheets.getByName("Sheet1")
  oCell=oSheet.getCellByPosition(0,0) 'A1
  oCell.setValue(23658)
  'oCell.NumberFormat=2   '23658.00
  'oCell.SetString("oops")
  'oCell.setFormula("=FUNCTION()")
  'oCell.IsCellBackgroundTransparent = TRUE
  'oCell.CellBackColor = RGB(255,141,56)
End Sub
```

IV.4. Ссылка на ячейку в другом документе

Из электронной таблицы можно получить доступ к ячейки другой электронной таблицы способом, похожим на следующее

```
file:///PATH/filename'#$Data.P40
```

Это можно сделать также в макросе путем присваивания ячейке формулы:

```
oCell = thiscomponent.sheets(0).getcellbyposition(0,0) ' A1
oCell.setFormula("=» &"'file:///home/USER/CalcFile2.sxc'#$Sheet2.K89")
```

IV.5. Очистка ячейки

Список вещей, которые могут быть очищены, можно найти по ссылке

```
http://api.openoffice.org/docs/common/ref/com/sun/star/sheet/CellFlags.html
```

Лист. 140: Очистка ячейки.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub ClearDefinedRange
  Dim oDoc As Object, oSheet As Object, oSheets As Object
```

```
  Dim oCellRange As Object
  Dim nSheets As Long

  oDoc = ThisComponent
  oSheets = oDoc.Sheets
  nSheets = oDoc.Sheets.Count
  REM получить третий лист, нумеруются они так 0, 1, 2
  oSheet = oSheets.getByIndex(2)
  REM Можно использовать интервал range в виде "A1:B2"
  oCellRange = oSheet.getCellRangeByName("<здесь_указать_интервал>")
  oCellRange.clearContents(_
    com.sun.star.sheet.CellFlags.VALUE OR _
    com.sun.star.sheet.CellFlags.DATETIME OR _
    com.sun.star.sheet.CellFlags.STRING OR _
    com.sun.star.sheet.CellFlags.ANNOTATION OR _
    com.sun.star.sheet.CellFlags.FORMULA OR _
    com.sun.star.sheet.CellFlags.HARDATTR OR _
    com.sun.star.sheet.CellFlags.STYLES OR _
    com.sun.star.sheet.CellFlags.OBJECTS OR _
    com.sun.star.sheet.CellFlags.EDITATTR)
End Sub
```

IV.6. Выделенный (Selected) текст — что это?

Выделенным текстом в электронной таблице могут быть несколько разных вещей; некоторые из них я понял, а некоторые ещё нет.

1. Выделена одна ячейка. Кликните (нажмите левую клавишу мыши) на ячейке один раз, затем держите нажатой клавишу shift и кликните на ту же ячейку ещё раз.
2. Выделена часть текста в одной ячейке. Двойной клик на одну ячейку и затем выделите какой-либо текст (из появившегося содержимого ячейки).
3. Ничего не выделено. Один клик на ячейки или передвижение между ячейками.
4. Выделено несколько ячеек. Один клин на ячейке и затем двигайте курсор.
5. Выделено несколько не связанных областей. Выделите несколько ячеек. Нажмите клавишу Ctrl и выделите ещё несколько.

До сих пор я не мог различить первые три случая. Если я смогу объяснить, как извлечь выделенный текст в случае 2, то я смогу решить эту проблему.

Лист. 141: Выделено ли что-нибудь в документе — электронной таблице Calc.

```
Function CalcIsAnythingSelected(oDoc As Object) As Boolean
  Dim oSels
  Dim oSel
  Dim oText
```

```
  Dim oCursor

  IsAnythingSelected = False
  If IsNull(oDoc) Then Exit Function
  ' Текущее выделение в текущем контроллере controller.
  'Если не существует текущего контроллера, возвращается  NULL.
  oSels = oDoc.getCurrentSelection()
  If IsNull(oSels) Then Exit Function
  If oSels.supportsService("com.sun.star.sheet.SheetCell") Then
    Print "Выделена одна ячейка = " & oSels.getImplementationName()
    MsgBox "getString() = " & oSels.getString()
  ElseIf oSels.supportsService("com.sun.star.sheet.SheetCellRange") Then
    Print "Выделен один интервал ячеек = " & oSels.getImplementationName()
  ElseIf oSels.supportsService("com.sun.star.sheet.SheetCellRanges") Then
    Print "Выделен множественный интервал ячеек Multiple Cell Ranges = " &
oSels.getImplementationName()
    Print "Количество интервалов в множественном интервале Count = " &
oSels.getCount()
  Else
    Print "Выделено что-то другое = " & oSels.getImplementationName()
  End If
End Function
```

IV.7. Простой пример обработки выделенных ячеек

Рассмотрим очень простой пример, который делит (арифметически) все выделенные ячейки на заданное число. Этот пример не содержит проверки ошибок. Без обработки ошибок деление ячейки на число очень просто.

Лист. 142: Деление одной ячейки на числовое значение..

```
Sub DivideCell(oCell, dDivisor As Double)
  oCell.setValue(oCell.getValue()/dDivisor)
End Sub
```

Деление прямоугольника ячеек (range) труднее. Массивы копируются по ссылке, а не по значению. Поэтому массив oRow() не нужно копировать обратно в массив oData().

Лист. 143: Деление каждой ячейки из прямоугольника (range) на одно и то же числовое значение.

```
Sub DivideRange(oRange, dDivisor As Double)
  Dim oData()
  Dim oRow()
  Dim i As Integer
  Dim j As Integer

  oData() = oRange.getDataArray()
  For i = LBound(oData()) To UBound(oData())
    oRow() = oData(i)
    For j = LBound(oRow()) To UBound(oRow())
      oRow(j) = oRow(j) / dDivisor
```

```
    Next
  Next
  oRange.setDataArray(oData())
End Sub
```

Следующий макрос предполагает, что текущим документом является электронная таблица Calc. Текущее выделение (selection) получено из текущего контроллера (current controller) и передано процедуре DivideRegions.

Лист. 144: Деление текущего выделения на числовое значение.

```
Sub DivideSelectedCells
  Dim dDivisor As Double
  Dim oSels

  dDivisor = 10
  oSels = ThisComponent.getCurrentController().getSelection()
  DivideRegions(oSels, dDivisor)
End Sub
```

Не поддавайтесь искушению вставить код программы из примера Лист. 145, где решается реальная задача, в процедуру Лист. 144. Преимущество отдельной процедуры станет ясным, когда выделены несвязанные наборы ячеек; другими словами, если выделение (selection) не является прямоугольником ячеек (range). Сложное выделение нескольких прямоугольников ячеек состоит из нескольких выделений SheetCellRange. Выделив из Лист. 145 его собственную процедуру, можно заставить её рекурсивно вызывать себя в случае сложного выделения.

Лист. 145: Первоначальный код макроса для деления ячеек на числовое значение.

```
Sub DivideRegions(oSels, dDivisor As Double)
  Dim oSel
  Dim i As Integer

  If oSels.supportsService("com.sun.star.sheet.SheetCell") Then
    DivideCell(oSels, dDivisor)
  ElseIf oSels.supportsService("com.sun.star.sheet.SheetCellRanges") Then
    For i = 0 To oSels.getCount() -1
      DivideRegions(oSels.getByIndex(i), dDivisor)
    Next
  ElseIf oSels.supportsService("com.sun.star.sheet.SheetCellRange") Then
    DivideRange(oSels, dDivisor)
  End If
End Sub
```

Отдельная ячейка SheetCell также является прямоугольником SheetCellRange, поэтому Лист. 142 реально не нужен, можно использовать Лист. 143 вместо него (см. Лист. 146).

Лист. 146: Первоначальный код макроса для деления ячеек на числовое значение.

```
Sub DivideRegions(oSels, dDivisor As Double)
  Dim oSel
  Dim i As Integer

  If oSels.supportsService("com.sun.star.sheet.SheetCellRanges") Then
    For i = 0 To oSels.getCount() -1
      DivideRegions(oSels.getByIndex(i), dDivisor)
    Next
  ElseIf oSels.supportsService("com.sun.star.sheet.SheetCellRange") Then
    DivideRange(oSels, dDivisor)
  End If
End Sub
```

Неуправляемый эксперимент с несколькими работающими процессами заставил меня поверить, что более эффективно обрабатывать одиночную ячейку как ячейку, а не как прямоугольник ячеек (поэтому Лист. 145 должен работать быстрее, чем Лист. 146). С другой стороны, если скорость важна, переупорядочите сравнения в тексте макроса так, чтобы более стандартные ситуации тестировались и обрабатывались первыми.

IV.8. Получить активную ячейку игнорировать остальное

Если Вам нужна только ячейка, где находится курсор (активная ячейка) и не нужно остальное, то Вы можете запросить контроллер (controller) выделить (select) пустой интервал ячеек, который был создан документом. Следующая процедура выполняет следующее:

1. Сохранить текущее выделение. Это полезно, если активны несколько ячеек.
2. Выделить (Select) пустой интервал ячеек (range), так что только ячейка, на которую установлен курсор, будет выделена. Эта ячейка выделена с рамкой вокруг нее, но она не полностью зачернена. Если Вы используете контроллер (controller) выделить интервал ячеек, то этот же способ может использоваться также для изменения выделения (selection) с полностью выделенной ячейки на просто активную ячейку.
3. Использовать сервис CellAddressConversion для получения адреса активной ячейки. Это новая возможность версии 1.1.1, часть включения в OOo «связанного управления» (linked controls).

Лист. 147: Найти активную ячейку.

```
'Author: Paolo Mantovani
'email: mantovani.paolo@tin.it
Sub RetrieveTheActiveCell()
  Dim oOldSelection 'Первоначальное выделение интервалов ячеек
  Dim oRanges        'Пустой интервал, созданный данным документом
  Dim oActiveCell   'Текущая активная ячейка
```

```
  Dim oConv          'Сервис преобразования адресов ячеек
  REM Сохраняем текущее выделение
  oOldSelection = ThisComponent.CurrentSelection

  oRanges =
ThisComponent.createInstance("com.sun.star.sheet.SheetCellRanges")
  ThisComponent.CurrentController.Select(oRanges)
  'получаем активную ячейку!
  oActiveCell = ThisComponent.CurrentSelection

  REM прекрасный сервис, который я недавно нашел!! :-)
  oConv =
ThisComponent.createInstance("com.sun.star.table.CellAddressConversion")
  oConv.Address = oActiveCell.getCellAddress
  Print oConv.UserInterfaceRepresentation
  Print oConv.PersistentRepresentation

  'восстанавливаем старое выделение (но теряем предыдущую активную ячейку)
  ThisComponent.CurrentController.Select(oOldSelection)
End Sub
```

IV.9. Выделить (Select) ячейку

После выделения ячейки становится выделенной вся ячейка — ячейка полностью черная. Это очень просто.

Лист. 148: Выделить одну ячейку, чтобы она стала черной.

```
  Dim oCell
  Dim oSheet

  REM Получаем первый лист sheet.
  oSheet = ThisComponent.getSheets().getByIndex(0)
  REM Получаем ячейку A2
  oCell = oSheet.GetCellbyPosition( 0, 1 )
  REM Перемещаем выделение на ячейку A2
  ThisComponent.CurrentController.Select(oCell)
```

Если Вы хотите, чтобы вся ячейка стала черной, но предпочтительнее чтобы была черная рамка вокруг нее, выделите (select) пустой интервал ячеек (range) ПОСЛЕ передвижения курсора на нужную ячейку.

Лист. 149: Выделить одну ячейку, чтобы она имела черную рамку вокруг.

```
Sub MoveCursorToCell
  Dim oCell
  Dim oSheet
  Dim oRanges

  REM Получаем первый лист
  oSheet = ThisComponent.getSheets().getByIndex(0)
  REM Получаем ячейку A2
  oCell = oSheet.GetCellbyPosition( 0, 1 )
```

```
  REM Перемещаем выделение на ячейку A2
  ThisComponent.CurrentController.Select(oCell)

  REM Выделяем пустой интервал ячеек.
  oRanges =
ThisComponent.createInstance("com.sun.star.sheet.SheetCellRanges")
  ThisComponent.CurrentController.Select(oRanges)
End Sub
```

IV.10. Удобный для чтения (Human readable) адрес ячейки

Сервис com.sun.star.table.CellAddressConversion может использоваться для получения удобного для чтения адреса ячейки в виде текстовой строки. Я не нашел какой-либо документации по этому сервису, но для версии OOo 1.1.1 он, кажется, работает достаточно хорошо. Следующий кусок текста программы предполагает, что текущий документ является электронной таблицей Calc и при этом выделена (selected) только одна ячейка.

Лист. 150: Cell Адрес ячейки в удобной для чтения форме с использованием CellAddressConversion.

```
  oActiveCell = ThisComponent.getCurrentSelection()
  oConv =
ThisComponent.createInstance("com.sun.star.table.CellAddressConversion")
  oConv.Address = oActiveCell.getCellAddress
  Print oConv.UserInterfaceRepresentation
  Print oConv.PersistentRepresentation
```

Я создал следующую функцию раньше, чем узнал о сервисе CellAddressConversion.

Лист. 151: Cell address in a readable form.

```
'По заданной ячейке извлечь удобный для чтения адрес ячейки
'Сначала извлекаем имя содержащего ее листа
'Затем получаем номер столбца и преобразуем его в букву
'Наконец, получаем строку. Нумерация строк начинается с 0, но выводим
нумерацию, начинающуюся с 1
Function PrintableAddressOfCell(the_cell As Object) As String
  PrintableAddressOfCell = "Unknown"
  If Not IsNull(the_cell) Then
    PrintableAddressOfCell = the_cell.getSpreadSheet().getName + ":" + _
      ColumnNumberToString(the_cell.CellAddress.Column) +
(the_cell.CellAddress.Row+1)
  End If
End Function

' Столбцы нумеруются начиная с 0, причем 0 соответствует A
' Столбцы обозначаются по порядку A-Z,AA-AZ,BA-BZ,...,IV
' По существу, вопрос в том, как преобразовать десятичное число в число
' по основанию 26.
' Заметьте, что столбец передается как параметр по значению!
```

```
Function ColumnNumberToString(ByVal the_column As Long) As String
  Dim s$
  'Сохраняем это, так как я НЕ изменил этот параметр
  '(из-за этой ошибки я потерял немало времени)
  Do while the_column >= 0
    s$ = Chr(65 +  the_column MOD 26) + s$
    the_column =  the_column \ 26 — 1
  Loop
  ColumnNumberToString = s$
End Function
```

IV.11. Вставить форматированную дату в ячейку

Вставьте дату в текущую ячейку. Будет выведена ошибка, если текущий документ не является электронной таблицей. Код программы предназначен для форматирования даты в указанном Вами формате, Вам нужно только удалить комментарии. Ещё предупреждение: этот макрос предполагает, что только одна ячейка выделена, и в противном случае выдаст ошибку. Если Вы хотите работать в случае, когда возможно выделение нескольких ячеек, то посмотрите раздел, в котором обсуждается выделение в электронной таблице Calc.

Лист. 152: Форматированная дата в ячейке.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
'uses:  FindCreateNumberFormatStyle
Sub InsertDateIntoCell
  Dim oSelection  'The currently selected cell
  Dim oFormats    'Available formats

  REM Проверяем, что это документ OOo Calc
  If ThisComponent.SupportsService("com.sun.star.sheet.SpreadsheetDocument")
Then
    oSelection = ThisComponent.CurrentSelection

    Rem Set the time, date, or date и time
    'oSelection.setValue(DateValue(Now()))   'Установить только дату
    'oSelection.setValue(TimeValue(Now()))   'Установить только время
    oSelection.setValue(Now())               'Установить и дату, и время

    Rem Можно было бы использовать FunctionAccess для установки даты и (или)
времени.
    'Dim oFunction   'используйте сервис FunctionAccess для вызова функции
Now
    'oFunction = CreateUnoService("com.sun.star.sheet.FunctionAccess")
    'oFunction.NullDate = ThisComponent.NullDate
    'oSelection.setValue(oFunction.callFunction("NOW", Array()))

    Rem Устанавливаем числовой формат даты по умолчанию
    oFormats = ThisComponent.NumberFormats
    Dim aLocale As New com.sun.star.lang.Locale
    oSelection.NumberFormat = oFormats.getStandardFormat(_
```

```
      com.sun.star.util.NumberFormat.DATETIME, aLocale)

    Rem Устанавливаем формат на что-то совсем другое
    'oSelection.NumberFormat = FindCreateNumberFormatStyle(_
    '  "YYYYMMDD.hhmmss",  doc)
  Else
    MsgBox "Этот макрос выполняется только для электронной таблицы
spreadsheet document"
  End If
End Sub
```

IV.12. Более короткий путь для этого

Рассмотрим следующие два метода (предоставленные Shez):

Лист. 153: Форматированная дата в ячейке более коротким способом.

```
Sub DateNow
  Dim here As Object
  here=ThisComponent.CurrentSelection
  here.setValue(DateValue(Now))
  here.NumberFormat=75
End sub
Sub TimeNow
  Dim here As Object
  here=ThisComponent.CurrentSelection
  here.setValue(TimeValue(Now))
  here.NumberFormat=41
End sub
```

Эти два способа предполагают, что документ — электронная таблица Calc, и задать нужно конкретный формат даты.

IV.13. Вывести выделенный интервал ячеек (selected range) в окно сообщения

Лист. 154: Вывести выделенный интервал ячеек.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
'Этот макрос получает текущее выделение и печатает
'окно сообщений, показывающее выбранный интервал и число выбранных ячеек
Sub SelectedCells
  oSelect=ThisComponent.CurrentSelection.getRangeAddress
  oSelectColumn=ThisComponent.CurrentSelection.Columns
  oSelectRow=ThisComponent.CurrentSelection.Rows

  CountColumn=oSelectColumn.getCount
  CountRow=oSelectRow.getCount

  oSelectSC=oSelectColumn.getByIndex(0).getName
  oSelectEC=oSelectColumn.getByIndex(CountColumn-1).getName
```

```
  oSelectSR=oSelect.StartRow+1
  oSelectER=oSelect.EndRow+1
  NoCell=(CountColumn*CountRow)

  If CountColumn=1 and CountRow=1 Then
    MsgBox("Cell " + oSelectSC + oSelectSR + chr(13) + "Cell No = " +
NoCell,, "SelectedCells")
  Else
    MsgBox("Интервал Range(" + oSelectSC + oSelectSR + ":" + oSelectEC +
oSelectER + ")" + chr(13) + "Количество ячеек  = " + NoCell,,
"SelectedCells")
  End If
End Sub
```

IV.14. Заполнить выделенный интервал ячеек заданным текстом

Этот простой макрос перебирает выделенные строки и столбцы, заполняя их текстом «OOPS».

Лист. 155: Перебрать ячейки и установить заданный текст.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
Sub FillCells
  oSelect=ThisComponent.CurrentSelection
  oColumn=oselect.Columns
  oRow=oSelect.Rows
  For nc= 0 To oColumn.getCount-1
    For nr = 0 To oRow.getCount-1
      oCell=oselect.getCellByPosition (nc,nr).setString ("OOOPS")
    Next nr
  Next nc
End Sub
```

Хотя эта техника циклов часто используется без проблем, все же может оказаться быстрее установить все значения за один раз, используя метод setDataArray().

IV.15. Некоторые данные и статистика о выделенном интервале ячеек

Лист. 156: Информация о выделенном интервале ячеек.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
'Напечатать сообщение, показывающее выделенный интервал и количество
' выделенных ячеек
Sub Analize
  sSum="=SUM("+GetAddress+")"
```

```
  sAverage="=AVERAGE("+GetAddress+")"
  sMin="=MIN("+GetAddress+")"
  sMax="=MAX("+GetAddress+")"
  CellPos(7,6).setString(GetAddress)
  CellPos(7,8).setFormula(sSum)
  CellPos(7,8).NumberFormat=2
  CellPos(7,10).setFormula(sAverage)
  CellPos(7,10).NumberFormat=2
  CellPos(7,12).setFormula(sMin)
  CellPos(7,12).NumberFormat=2
  CellPos(7,14).setFormula(sMax)
  CellPos(7,14).NumberFormat=2
End sub
Function GetAddress  'выделенной ячейки (ячеек)
  oSelect=ThisComponent.CurrentSelection.getRangeAddress
  oSelectColumn=ThisComponent.CurrentSelection.Columns
  oSelectRow=ThisComponent.CurrentSelection.Rows

  CountColumn=oSelectColumn.getCount
  CountRow=oSelectRow.getCount

  oSelectSC=oSelectColumn.getByIndex(0).getName
  oSelectEC=oSelectColumn.getByIndex(CountColumn-1).getName

  oSelectSR=oSelect.StartRow+1
  oSelectER=oSelect.EndRow+1
  NoCell=(CountColumn*CountRow)

  If CountColumn=1 and CountRow=1 then
    GetAddress=oSelectSC+oSelectSR
  Else
    GetAddress=oSelectSC+oSelectSR+":"+oSelectEC+oSelectER
  End If
End Function
Function CellPos(lColumn As Long,lRow As Long)
  CellPos= ActiveSheet.getCellByPosition (lColumn,lRow)
End Function
Function ActiveSheet
  ActiveSheet=StarDesktop.CurrentComponent.CurrentController.ActiveSheet
End Function
Sub DeleteDbRange(sRangeName As String)
  oRange=ThisComponent.DatabaseRanges
  oRange.removeByName (sRangeName)
End Sub
```

IV.16. (Именованный) Интервал ячеек базы данных

Я изменил вышеприведенный макрос, явно объявив переменные и добавив улучшенную проверку на ошибки. Я также создал приведенную ниже тестовую процедуру.

Лист. 157: Процедура проверки наличия имени интервала ячеек базы данных, добавления имени интервала ячеек базы данных и удаления имени интервала ячеек базы данных.

```
Sub TestDefineAndRemoveRange
  Dim s As String
  Dim oDoc
  Dim sNames()
  oDoc = ThisComponent
  s = "blah1"
  If oDoc.DatabaseRanges.hasByName(s) Then Print "Имя интервала ячеек базы
данных уже существует"
  sNames() = oDoc.DatabaseRanges.getElementNames()
  MsgBox Join(sNames(), CHR$(10)), 0, "Перед добавлением имени  " & s
  DefineDbRange(s)
  sNames() = oDoc.DatabaseRanges.getElementNames()
  MsgBox Join(sNames(), CHR$(10)), 0, "После добавления имени  " & s
  DeleteDbRange(s)
  sNames() = oDoc.DatabaseRanges.getElementNames()
  MsgBox Join(sNames(), CHR$(10)), 0, "После удаления имени интервала ячеек
базы данных  " & s
End Sub
```

IV.17. Определить выбранные ячейки в качестве (именованного) интервала ячеек базы данных

Лист. 158: Определить выбранные ячейки в качестве интервала ячеек базы данных.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
'Modified by : Andrew Pitonyak
Sub DefineDbRange(sRangeName As String) 'selected range
  Dim oSelect
  Dim oRange
  Dim oRanges
  On Error GoTo DUPLICATENAME
  oSelect=ThisComponent.CurrentSelection.RangeAddress
  oRanges =  ThisComponent.DatabaseRanges
  If oRanges.hasByName(sRangeName) Then
    MsgBox("Повторение имени " & sRangeName,,"INFORMATION")
  Else
    oRange= oRanges.addNewByName (sRangeName,oSelect)
  End If
DUPLICATENAME:
  If Err <> 0 Then
    MsgBox("Ошибка при добавлении имени интервала ячеек базы данных database
name " & sRangeName,,"INFORMATION")
  End If
End Sub
```

IV.18. Удалить (именованный) интервал ячеек базы данных

Лист. 159: Удалить интервал ячеек базы данных.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
'Modified by : Andrew Pitonyak
Sub DeleteDbRange(sRangeName As String)
  Dim oRanges
  oRanges=ThisComponent.DatabaseRanges
  If oRanges.hasByName(sRangeName) Then
    oRanges.removeByName(sRangeName)
  End If
End Sub
```

IV.19. Границы таблицы

Когда результатом сервиса является структура, передается копия структуры, а не ссылка на саму структуру. Это не дает Вам возможности напрямую модифицировать данные этой структуры. Вы можете вместо этого создать копию структуры, изменить её, а затем записать копию обратно (см. Лист. 160).

Лист. 160: Установить границу интервала ячеек Calc, используя временную структуру.

```
'Author: Niklas Nebel
'email:  niklas.nebel@sun.com
'установить границы интервала ячеек setting_borders_in_calc
oRange = ThisComponent.Sheets(0).getCellRangeByPosition(0,1,0,63)
aBorder = oRange.TableBorder

aBorder.BottomLine = lHor
oRange.TableBorder = aBorder
```

Niklas привел пример, который не работает, потому что модифицирует временную копию структуры.

Лист. 161: Это не работает, потому что структура TableBorder является копией.

```
' не работающий пример
lHor.Color = 0
lHor.InnerLineWidth = 0
lHor.OuterLineWidth = 150
Dim lHor As New com.sun.star.table.BorderLinelHor.LineDistance
oRange = ThisComponent.Sheets(0).getCellRangeByPosition(0,1,0,63)
oRange.TableBorder.BottomLine = lHor
```

А вот работающее решение автора David Woody [dwoody1@airmail.net]

Лист. 162: Другой пример установки границы интервала ячеек Calc.

```
'другой пример установки границы интервала ячеек setting_borders_in_calc
Sub Borders
  Dim aBorder, oRange, oDoc, oSheets
  Dim TableBorder As New com.sun.star.table.TableBorder
  Dim aTopLine As New com.sun.star.table.BorderLine

  oDoc = ThisComponent
  oSheets = oDoc.Sheets(0)
  oRange  = oSheets.getCellRangeByPosition(8,2,8,5)
  aBorder = oRange.TableBorder
  aTopLine.OuterLineWidth = 250
  aTopLine.InnerLineWidth = 0
  aTopLine.Color = 170000

  oRange.TableBorder.IsTopLineValid = 1
  aBorder.TopLine = aTopLine
  oRange.TableBorder = aBorder
End Sub
```

IV.20. Интервал ячеек для сортировки

Макрос из Лист. 163 выполняет сортировку по убыванию по столбцам. Другими словами, строки меняются местами.

Лист. 163: Сортировка по убыванию в электронной таблице.

```
'Author: Sasa Kelecevic
'email:  scat@teol.net
Sub SortRange
  Dim oSheetDSC,oDSCRange  As Object
  Dim aSortFields(0) As New com.sun.star.util.SortField
  Dim aSortDesc(0) As New com.sun.star.beans.PropertyValue

  'Установите нужное имя листа
  oSheetDSC = ThisComponent.Sheets.getByName("Sheet1")

  'установите адрес нужного интервала ячеек
  oDSCRange = oSheetDSC.getCellRangeByName("A1:L16")
  ThisComponent.getCurrentController.select(oDSCRange)

  aSortFields(0).Field = 0
  aSortFields(0).SortAscending = FALSE

  aSortDesc(0).Name = "SortFields"
  aSortDesc(0).Value = aSortFields()
  oDSCRange.Sort(aSortDesc())
End Sub
```

Предположим, я хочу отсортировать по второму и третьему столбцам, причем второй содержит текст, а третий должен сортироваться как числа. Я должен сортировать по двум полям, а не по одному.

Лист. 164: Сортировка

```
Sub Main
  Dim oSheetDSC As Object, oDSCRange As Object
  Dim aSortFields(1) As New com.sun.star.util.SortField
  Dim aSortDesc(0) As New com.sun.star.beans.PropertyValue

  'установите нужное имя листа
  oSheetDSC = THISCOMPONENT.Sheets.getByName("Sheet1")

  'установите адрес нужного интервала ячеек
  oDSCRange = oSheetDSC.getCellRangeByName("B3:E6")
  THISCOMPONENT.getCurrentController.select(oDSCRange)

  'Другой допустимый тип сортировки - это
  'com.sun.star.util.SortFieldType.AUTOMATIC
  'Помните, что поля нумеруются начиная с нуля, поэтому здесь
  'сортировка по столбцу B, а не по столбцу A
  aSortFields(0).Field = 1
  aSortFields(0).SortAscending = TRUE
  'Оказывается, нет необходимости указывать тип поля field type
  'для сортировки электронных таблиц, так как он игнорируется.
  'Электронная таблица уже знает этот тип.
  'aSortFields(0).FieldType = com.sun.star.util.SortFieldType.ALPHANUMERIC

  aSortFields(1).Field = 2
  aSortFields(1).SortAscending = TRUE
  aSortFields(1).FieldType = com.sun.star.util.SortFieldType.NUMERIC

  aSortDesc(0).Name = "SortFields"
  aSortDesc(0).Value = aSortFields()      ' aSortFields(0)
  oDSCRange.Sort(aSortDesc())             ' aSortDesc(0)
End Sub
```

Чтобы указать первую строку как строку заголовков, используйте другое свойство. Убедитесь, что размерность массива (здесь — aSortDesc) достаточна для размещения всех свойств.

```
aSortDesc(1).Name = «ContainsHeader"
aSortDesc(1).Value = True
```

Bernard Marcelly проверил, что свойство «Orientation» (ориентация) работает правильно. Чтобы установить ориентацию, используйте свойство «Orientation» и установите его значение одним из следующих:

```
com.sun.star.table.TableOrientation.ROWS
com.sun.star.table.TableOrientation.COLUMNS
```

Если сортировать с использованием GUI, то нельзя сортировать больше чем по трем строкам или по трем столбцам за один раз. Bernard Marcelly отметил, что нельзя обойти это ограничение с помощью макроса. Ограничение на три строки или три столбца существует.

IV.21. Вывести все данные столбца

Перебирая ячейки и печатая их значения, я хочу также печатать информацию об этих ячейках Вот как я это сделал.

Лист. 165: Печать информации о ячейках

```
Sub PrintDataInColumn (a_column As Integer)
  Dim oCells As Object, aCell As Object, oDoc As Object
  Dim oColumn As Object, oRanges As Object

  oDoc = ThisComponent
  oColumn = oDoc.Sheets(0).Columns(a_column)
  Print "Using column " + oColumn.getName
  oRanges = oDoc.createInstance("com.sun.star.sheet.SheetCellRanges")
  oRanges.insertByName("", oColumn)
  oCells = oRanges.Cells.createEnumeration
  If Not oCells.hasMoreElements Then Print "К сожалению, выводить нечего"
  While oCells.hasMoreElements
    aCell = oCells.nextElement
    'Нижеследующая функция определена в одном из предыдущих разделов!
    MsgBox PrintableAddressOfCell(aCell) + " = " + aCell.String
  Wend
End Sub
```

IV.22. Использование методов объединения (Outline, Grouping)

Ryan Nelson [ryan@aurelius-mfg.com] рассказал мне о возможности объединять ячейки в Calc и спросил, как это сделать в макросе. Здесь нужно иметь в виду две вещи. Во-первых, добавляет и отменяет объединение ячеек именно лист электронной таблицы (sheet), во-вторых, параметры должны быть корректными.

http://api.openoffice.org/docs/common/ref/com/sun/star/sheet/XSheetOutline.html

http://api.openoffice.org/docs/common/ref/com/sun/star/table/TableOrientation.html

Лист. 166: Объединение ячеек в Calc

```
Option Explicit
Sub CalcGroupingExample
  Dim oDoc As Object, oRange As Object, oSheet As Object
  oDoc = ThisComponent
  If Not oDoc.SupportsService("com.sun.star.sheet.SpreadsheetDocument") Then
    MsgBox "Этот макрос должен выполняться в электронной таблице spreadsheet
document", 64, "Error"
  End If
  oSheet=oDoc.Sheets.getByName("Sheet1")
  ' Параметры - номера строк/столбцов (лево, верх, право,низ)
  oRange = oSheet.getCellRangeByPosition(2,1,3,2)
  'Можно было бы вместо ROWS использовать COLUMNS
```

```
  oSheet.group(oRange.getRangeAddress(),
com.sun.star.table.TableOrientation.ROWS)
  Print "Интервал ячеек объединен"
  oSheet.unGroup(oRange.getRangeAddress(),
com.sun.star.table.TableOrientation.ROWS)
  Print "Интервал ячеек разгруппирован"
End Sub
```

IV.23. Защита данных

Сравнительно легко защитить электронную таблицу, только нужно брать лист (sheet) таблицы и защищать именно его. Мои эксперименты показывают, что хотя Вы не получаете ошибку при попытке защитить документ целиком, при этом документ не становится защищенным.

Лист. 167: Защита листа в Calc

```
Sub ProtectSpreadsheet
  Dim oDoc As Object, oSheet As Object
  oDoc = ThisComponent
  oSheet=oDoc.Sheets.getByName("Sheet1")
  oSheet.protect("password")
  Print "Protect value = " & oSheet.isProtected()
  oSheet.unprotect("password")
  Print "Protect value = " & oSheet.isProtected()
End Sub
```

IV.24. Установка текстов верхнего и нижнего заголовков

Этот макрос установит верхний заголовок для каждого листа в значение «Sheet: <sheet_name>». Верхние и нижние заголовки устанавливаются одинаковым способом, только нужно заменить «Header» на «Footer» при вызовах. Особые благодарности Oliver Brinzing [OliverBrinzing@t-online.de] за заполнение пробелов по вопросам, которые я не знал, особенно о том, что нужно записать заголовок обратно в документ.

Лист. 168: Установка заголовка листа

```
Sub SetHeaderTextInSpreadSheet
  Dim oDoc, oSheet, oPstyle, oHeader
  Dim oText, oCursor, oField
  Dim oStyles
  Dim sService$
  oDoc = ThisComponent
  ' Получаем стиль страницы  pagestyle для текущего активного листа
  oSheet = oDoc.CurrentController.getActiveSheet
  oStyles = oDoc.StyleFamilies.getByName("PageStyles")
  oPstyle = oStyles.getByName(oSheet.PageStyle)

  ' Включаем заголовки (колонтитулы) и затем делаем их общими!
  oPstyle.HeaderOn = True
```

```
  oPstyle.HeaderShared = True

  ' Кроме CenterText, есть еще RightText и LeftText
  oHeader = oPstyle.RightPageHeaderContent
  oText = oHeader.CenterText

  ' Теперь можно установить текстовый объект равным тому, что Вам нужно
  ' Используйте setSTring() для текстового объекта, чтобы задать простой
текст
  ' Используйте курсор cursor, чтобы вставить поле (такое как имя текущего
листа).
  ' Сначала очищаем существующий текст!
  oText.setString("")
  oCursor = oText.createTextCursor()
  oText.insertString(oCursor,  "Sheet: ", False)
  ' Здесь будет имя текущего листа!
  sService = "com.sun.star.text.TextField.SheetName"
  oField =  oDoc.createInstance(sService)
  oText.insertTextContent(oCursor, oField, False)

  ' а теперь для вышестоящего объекта можно
  ' занести обратно объект заголовка, так как до сих пор
  ' модифицировался временный объект
  oPstyle.RightPageHeaderContent = oHeader
End Sub
```

IV.25. Копирование листа целиком в новый документ

Следующий макрос копирует содержимое заданного листа во вновь созданный другой документ.

Лист. 169: Копирование листа в новый документ

```
'Author: Stephan Wunderlich [stephan.wunderlich@sun.com]
Sub CopySpreadsheet
  firstDoc = ThisComponent
  selectSheetByName(firstDoc, "Sheet2")
  dispatchURL(firstDoc,".uno:SelectAll")
  dispatchURL(firstDoc,".uno:Copy")
  secondDoc =
StarDesktop.loadComponentFromUrl("private:factory/scalc","_blank",0,dimArray
())
  secondDoc.getSheets().insertNewByName("inserted",0)
  selectSheetByName(secondDoc, "inserted")
  dispatchURL(secondDoc,".uno:Paste")
End Sub

Sub selectSheetByName(document, sheetName)
  document.getCurrentController.select(document.getSheets().getByName(sheetN
ame))
End Sub

Sub dispatchURL(document, aURL)
  Dim noProps()
```

```
  Dim URL As New com.sun.star.util.URL

  frame = document.getCurrentController().getFrame()
  URL.Complete = aURL
  transf = createUnoService("com.sun.star.util.URLTransformer")
 transf.parseStrict(URL)

  disp = frame.queryDispatch(URL, "",
com.sun.star.frame.FrameSearchFlag.SELF _
        OR com.sun.star.frame.FrameSearchFlag.CHILDREN)
  disp.dispatch(URL, noProps())
End Sub
```

IV.26. Выделить именованный интервал ячеек (named range)

Используйте меню Вставка — Названия — Определить (Ctrl+F3) (Use Insert | Names | Define), чтобы открыть окно диалога «Определить названия». Кнопка Дополнительно расширяет окно диалога, так что Вы можете указать больше информации об интервале ячеек. Именованный интервал ячеек — это не то же самое, что интервал ячеек базы данных, который определяется через меню Данные — Определить диапазон. Следующий макрос выводит все именованные интервалы ячеек из документа.

Лист. 170: Вывести все именованные интервалы ячеек из документа Calc.

```
oRanges = ThisComponent.NamedRanges
MsgBox "Текущие именованные интервалы ячеек = " & CHR$(10) & _
        Join(oRanges.getElementNames(), CHR$(10))
```

Именованный интервал ячеек поддерживает метод getReferencePosition(), который возвращает адрес ячейки, идентифицирующий верхний левый угол этого именованного интервала. Метод getContent() для именованного интервала ячеек возвращает текстовое представление этого именованного интервала ячеек. Макрос GetGlobalRangeByName автора Rob Gray упрощает портирование (перенос) макросов Excel VBA в макросы OpenOffice.org.

Лист. 171: Упрощение портирования

```
'Author: Rob Gray
'Email: robberbaron@optusnet.com.au
'Modified from a macro contained in Andrew Pitonyak's document.
'Этот макрос упрощает перенос глобальных интервалов ячеек VBA
'и отдельных параметров на лист Config sheet
Function GetGlobalRangeByName(rngname As String)
  Dim oSheet       'Лист, содержащий именованные интервалы
  Dim oNamedRange  'Именованный интервал ячеек
  Dim oCellAddr    'Адрес верхней левой ячейки в именованном интервале

  If NOT ThisComponent.NamedRanges.hasByName(rngname) Then
    MsgBox "К сожалению, именованный интервал !" & rngname & _
           "! не существует","MacroError",0+48
```

```
    Exit function
  End If

  REM Объект oNamedRange поддерживает интерфейс XNamedRange
  oNamedRange = ThisComponent.NamedRanges.getByName(rngname)
  'Print "Содержимое именованного интервала = " & oNamedRange.getContent()

  REM Получаем сервис com.sun.star.table.CellAddress
  oCellAddr = oNamedRange.getReferencePosition()

  REM Теперь получаем нужный лист!
  oSheet = ThisComponent.Sheets.getByIndex(oCellAddr.Sheet+1)
  GetGlobalRangeByName = oSheet.getCellRangeByName(rngname)

  REM Теперь можно использовать .GetCellByPosition(0,0).string  и т.д.
End function
```

Мой личный взгляд на именованные диапазоны заключается в выборе названия для диапазона. Я изменил макрос и создал SelectNamedRange.

Лист. 172: Мой взгляд на именованные диапазоны

```
Sub SelectNamedRange(rngname As String, Optional oDoc)
  REM Author: Andrew Pitonyak
  Dim oSheet       'Лист, содержащий именованный интервал
  Dim oNamedRange  'Объект именованного интервала named range object
  Dim oCellAddr    'Адрес верхней левой ячейки именованного интервала
  Dim oRanges      'Все именованные интервалы

  If IsMissing(oDoc) Then oDoc = ThisComponent
  oRanges = oDoc.NamedRanges
  If NOT oRanges.hasByName(rngname) Then
    MsgBox "К сожалению, именованный интервал " & rngname & _
           " не существует" &  CHR$(10) & _
           "Существующие именованные интервалы = " & CHR$(10) & _
           Join(oRanges.getElementNames(), CHR$(10))
    Exit Sub
  End If

  REM Объект oNamedRange поддерживает интерфейс XNamedRange
  oNamedRange = oRanges.getByName(rngname)
  'Print "Содержимое именованного интервала = " & oNamedRange.getContent()

  oCellAddr = oNamedRange.getReferencePosition()

  REM Теперь получаем нужный лист!
  oSheet = oDoc.Sheets.getByIndex(oCellAddr.Sheet)

  REM Можно было использовать текущий контроллер current controller
  REM для выделения того, что требуется выбрать.
  REM select ( VARIANT )
  REM setActiveSheet ( OBJECT )
  REM setFirstVisibleColumn ( LONG )
  REM setFirstVisibleRow ( LONG )
  oDoc.getCurrentController().setActiveSheet(oSheet)
```

```
  REM The sheet can return the range based on the name
  REM oSheet.getCellRangeByName(rngname)
  REM The sheet can also return a range by position, if you know it.

  REM This selects the ENTIRE range
  Dim oRange
  oRange = oSheet.getCellRangeByName(rngname)
  oDoc.getCurrentController().select(oRange)
End Sub
```

IV.27. Выделить столбец целиком

Выделите столбец целиком из заданного листа. Используйте текущий контроллер (current controller) для выделения этого столбца (автор макроса John Ward `http://digiassn.blogspot.com/`).

Лист. 173: Выделить третий столбец первого листа

```
Sub SelCol()
  Dim oSheet
  Dim oCol
  oSheet = ThisComponent.getSheets().getByIndex(0)
  oCol = oSheet.getColumns().getByIndex(2)
  ThisComponent.getCurrentController().select(oCol)
End Sub
```

IV.28. Выделить строку целиком

Выделите строку целиком из заданного листа. Используйте текущий контроллер (current controller) для выделения этой строки (Автор макроса John Ward `http://digiassn.blogspot.com/`).

Лист. 174: Выделить третью строку первого листа

```
Sub SelCol()
  Dim oSheet
  Dim oRow
  oSheet = ThisComponent.getSheets().getByIndex(0)
  oRow = oSheet.getRows().getByIndex(2)
  ThisComponent.getCurrentController().select(oRow)
End Sub
```

IV.29. Преобразовать данные из столбца определенного вида в строки

Вырезать и Вставить данные из веб-страницы, содержащей имя и информацию об адресе, расположенную в одном столбце в следующем порядке:

```
customer1
address1a
address1b

customer2
address2a
address2b
```

Нижеприведенный макрос преобразует эти записи в определенный вид по строкам, так что они могут быть использованы для вставки данных в форму письма.

```
Customer  address1  address2
------------------------------
customer1 address1a address1b
customer2 address2a address2b
```

Andy заметил, что следующий макрос предполагает, что данные расположены в первом листе и начинаются с первой строки в первом столбце. После преобразования данные также расположены начиная с первой строки первого столбца. Я всегда выполняю мои макросы с использованием опции Option Explicit (задавать все переменные явно), но переменные в этом макросе не определены явным способом. Хотя это и не лучший метод, но он определенно работает.

Лист. 175: Преобразование записей

```
'Транспонировать данные столбцов в строки.
'Author: David Kwok
'Email: dkwok@iware.com.au
Sub ColumnsToRows
  int_col = 0
  int_row = 0
  osheet = ThisComponent.Sheets.getByIndex(0)
  ThisComponent.CurrentController.Select(osheet.GetCellByPosition(0,0))
  Cellstring = Thiscomponent.getCurrentSelection.getstring
  loop_col = int_col
  loop_row = int_row
  row_cnt = int_row
  Do While Cellstring <> ""
    col_cnt = 1
    REM значение конца цикла, 3, зависит от числа полей в записи
    For xx = 1 To 3

      oRangeOrg = osheet.getCellByPosition(loop_col,loop_row).Rangeaddress
      oRangecpy = osheet.getCellByPosition(col_cnt,row_cnt).Rangeaddress
      oCellCpy = osheet.getCellByPosition(oRangecpy.StartColumn, _
        oRangecpy.StartRow).CellAddress
      osheet.MoveRange(oCellcpy, oRangeOrg)
      col_cnt = col_cnt + 1
      loop_row = loop_row + 1

    Next xx
```

```
    ThisComponent.CurrentController.Select(osheet.GetCellByPosition(loop_col
,loop_row))
    cellstring = ThisComponent.getCurrentSelection.getstring
    row_cnt = row_cnt + 1
  Loop
  osheet.Columns.removeByIndex(0,1)  'удаляем столбец
  osheet.Rows.insertByIndex(0,1)
  osheet.GetCellByPosition(0,0).string = "Customer"
  osheet.GetCellByPosition(1,0).string = "Address1"
  osheet.GetCellByPosition(2,0).string = "Address2"
  osheet.Columns(0).OptimalWidth = true  'автоподбор ширины
  osheet.Columns(1).OptimalWidth = true
  osheet.Columns(2).OptimalWidth = true
End Sub
```

IV.30. Включить/Выключить автоматический пересчет

Chris Clementson прислал макрос для включения/выключения автоматического пересчета электронной таблицы используя возможности диспетчера (dispatch). Я добавил только показ аргумента:

Лист. 176: Показ аргумента

```
  Dim oProp(0) As New com.sun.star.beans.PropertyValue
  oProp(0).Name = "AutomaticCalculation"
  oProp(0).Value = False
  dispatcher.executeDispatch(document, ".uno:AutomaticCalculation", "", 0,
oProp())
```

Электронные таблицы Calc включают интерфейс Xcalculatable, который определяет методы, показанные в следующей таблице.

Метод	***Описание***
calculate()	Пересчитать все требующие пересчета (dirty) ячейки.
calculate all()	Пересчитать все ячейки.
isAutomaticCalculationEnabled()	Возвращает True, если включен автоматический пересчёт
enableAutomaticCalculation(boolean)	Разрешить или запретить автоматический пересчёт

Следующий код программы запрещает и затем разрешает автоматический пересчёт:

Лист. 177: Пересчёт

```
ThisComponent.enableAutomaticCalculation(False)
ThisComponent.enableAutomaticCalculation(True)
```

IV.31. Какие ячейки листа используются?

SheetCellCursor включает методы gotoStartOfUsedArea(boolean) и gotoEndOfUsedArea(boolean), которые перемещают курсор на начало и на конец области, использованной в электронной таблице. Gerrit Jasper представил следующий пример (автор неизвестен):

Лист. 178: Область использования в таблице

```
'Author: Unknown
Sub testEndColRow
  Dim oSheet
  Dim oCell
  Dim nEndCol As Integer
  Dim nEndRow As Integer
  oSheet = ThisComponent.Sheets.getByIndex( 0 )
  nEndCol = getLastUsedColumn(oSheet)        'См. эти функции ниже
  nEndRow = getLastUsedRow(oSheet)

  REM Then do as you please, e.g.
  oCell = oSheet.GetCellByPosition( nEndCol + 1, nEndRow + 1 )

  oCell.String = "test"
  ThisComponent.CurrentController.Select(oCell)
End Sub

Function getLastUsedColumn(oSheet as Object) as Integer
  Dim oCell As Object
  Dim oCursor As Object
  Dim aAddress As Variant
  oCell = oSheet.GetCellbyPosition( 0, 0 )
  oCursor = oSheet.createCursorByRange(oCell)
  oCursor.GotoEndOfUsedArea(True)
  aAddress = oCursor.RangeAddress
  GetLastUsedColumn = aAddress.EndColumn
End Function

Function getLastUsedRow(oSheet as Object) as Integer
  Dim oCell As Object
  Dim oCursor As Object
  Dim aAddress As Variant
  oCell = oSheet.GetCellbyPosition( 0, 0 )
  oCursor = oSheet.createCursorByRange(oCell)
  oCursor.GotoEndOfUsedArea(True)
  aAddress = oCursor.RangeAddress
  GetLastUsedRow = aAddress.EndRow
End Function
```

IV.32. Поиск в электронной таблице Calc

Gerrit Jasper представил макрос для поиска в документе Calc, который сначала определяет интервал ячеек, которые используются, и затем получает каждую из ячеек и проверяет её на совпадение с заданной строкой. Я произвел некоторые изменения этого макроса:

Лист. 179: Поиск

```
'Возвращает ячейку, которая содержит заданный текст
Function uFindString(sString$, oSheet) As Variant
  Dim nCurCol As Integer
  Dim nCurRow As Integer
  Dim nEndCol As Integer
  Dim nEndRow As Integer
  Dim oCell As Object
  Dim oCursor As Object
  Dim aAddress As Variant
  Dim sFind As String

  oCell = oSheet.GetCellbyPosition( 0, 0 )
  oCursor = oSheet.createCursorByRange(oCell)
  oCursor.GotoEndOfUsedArea(True)
  aAddress = oCursor.RangeAddress
  nEndRow = aAddress.EndRow
  nEndCol = aAddress.EndColumn

  For nCurCol = 0 To nEndCol       'Просматриваем интервал столбец за
    For nCurRow = 0 To nEndRow     'столбцом, строку за строкой
      oCell = oSheet.GetCellByPosition( nCurCol, nCurRow )
      sFind = oCell.String          'Получаем содержимое ячейки
      If sFind = sString then
        uFindString = oCell
        Exit Function
      End If
    Next
  Next
End Function
```

В небольшом листе я мог найти ячейку, которая содержит заданный текст, примерно за 1184 тактов часов (clock ticks). Затем, я изменил макрос для использования массива данных. Использование массива уменьшает время примерно до 54 тактов часов — намного быстрее.

Лист. 180: Ускоренный поиск

```
'Возвращает ячейку, которая содержит заданный текст
Function uFindString_2(sString$, oSheet) As Variant
  Dim nCurCol As Integer
  Dim nCurRow As Integer
  Dim oCell As Object
  Dim oCursor As Object
```

```
  Dim oData
  Dim oRow

  oCell = oSheet.GetCellbyPosition( 0, 0 )
  oCursor = oSheet.createCursorByRange(oCell)
  oCursor.GotoEndOfUsedArea(True)
  oData = oCursor.getDataArray()
  For nCurRow = LBound(oData) To UBound(oData)
    oRow = oData(nCurRow)
    For nCurCol = LBound(oRow) To UBound(oRow)
      If (oRow(nCurCol) = sString$) Then
        uFindString_2 = oSheet.GetCellbyPosition( nCurCol, nCurRow )
        Exit Function
      End If
    Next
  Next
End Function
```

Поиск прямо в листе ещё быстрее — 34 такта!

Лист. 181: Ещё быстрее

```
'Найти первую ячейку, которая содержит заданный текст sString$
'Если bWholeWord равно True, то ячейка должна содержать ТОЛЬКО этот
'текст. Если bWholeWord равно False, то ячейка должна хотя бы содержать
'заданную строку.
Function SimpleSheetSearch(sString$, oSheet, bWholeWord As Boolean) As
Variant
  Dim oDescriptor
  Dim oFound
  REM Создаем дескриптор из просматриваемого документа.
  oDescriptor = oSheet.createSearchDescriptor()
  REM Установить текст, который нужно искать и др.
  REM
http://api.openoffice.org/docs/common/ref/com/sun/star/util/SearchDescriptor
.html
  With oDescriptor
    .SearchString = sString$
    REM По умолчанию все следующие свойства равны false
    REM А это значение SearchWords требует, чтобы ячейка содержала только
заданную строку
    .SearchWords = bWholeWord
    .SearchCaseSensitive = False
  End With
  REM Находим первую ячейку
  oFound = oSheet.findFirst(oDescriptor)
  SimpleSheetSearch = oFound

  REM Если нужно найти больше подходящих ячеек
  REM тогда можно продолжать поиск после найденной ячейки!
  'Do While Not IsNull(oFound)
  '  Print oFound.getString()
  '  oFound = oSheet.findNext( oFound, oDescriptor)
  'Loop
End Function
```

Как обычно, встроенные механизмы много быстрее, чем выполнение макроса. Тем не менее это демонстрирует, что часто лучше получить порцию данных, используя массив (array), чем обрабатывать по одной ячейке. Вот макрос, который я использовал для проверки времени выполнения:

Лист. 182: Проверка времени выполнения

```
Sub SimpleSheetSearchTest
  Dim nItCount As Integer
  Dim nMaxIt As Integer
  Dim lTick1 As Long
  Dim lTick2 As Long
  Dim oSheet
  Dim oCell
  Dim s As String

  nMaxIt = 10
  oSheet = ThisComponent.getSheets().getByIndex(0)
  lTick1 = GetSystemTicks()
  For nItCount = 1 To nMaxIt
    oCell = SimpleSheetSearch("hello", oSheet, True)
    'oCell = uFindString("hello", oSheet)
    'oCell = uFindString_2("hello", oSheet)
  Next
  lTick2 = GetSystemTicks()

  s = s & "Поиск занял " & (lTick2 - lTick1) & " тактов ticks для "  & _
          nMaxIt & " итераций " & CHR$(10) & _
          CStr((lTick2 - lTick1) / nMaxIt) & _
          " число тактов на итерацию" & CHR$(10)

  If IsEmpty(oCell) OR IsNull(oCell) Then
    s = s & "Текст не найден" & CHR$(10)
  Else
    s = s & "столбец = " & oCell.CellAddress.Column & _
            " строка  = " & oCell.CellAddress.Row  & CHR$(10)
  End If
  MsgBox s, 0, "Сравниваем время поиска"
End Sub
```

> ***Совет*** *По умолчанию встроенный механизм поиска ведет поиск в формулах ячеек (SearchType = 0). Установите свойство SearchType равным 1 для поиска по значениям ячейки.*

> ***Совет*** *Вы можете искать в листе, а также искать прямо в интервале ячеек листа.*

Gerrit Jasper отметил, что функция SimpleSheetSearch работает так же хорошо с интервалом ячеек, как и с листом — вид неудобства, который я не заметил, когда писал остаток этого раздела. Следующий пример демонстрирует эту возможность при поиске в заданном интервале ячеек.

Лист. 183: Поиск в заданном интервале ячеек

```
Sub SearchARange
  REM Author: Andrew Pitonyak
  Dim oSheet
  Dim oRange
  Dim oFoundCell
  oSheet = ThisComponent.getSheets().getByIndex(0)
  oRange = oSheet.getCellRangeByName("F7:H11")
  oFoundCell = SimpleSheetSearch("41", oRange, False)
End Sub
```

IV.33. Напечатать интервал ячеек электронной таблицы

Официальный метод печати интервала ячеек в документе Calc — установить область печати для каждого листа и затем использовать для документа в целом метод print. Хотя интерфейс XprintAreas, который используется для установки области печати, назван нежелательным на некоторое время, он все ещё официально санкционирован для печати частей документа Calc (Благодарности Niklas Nebel).

David French [dfrench@xtra.co.nz], модератор форума oooforum, указал, что когда печатается документ Calc, печатаются все области печати во всех листах[1]. Другими словами, когда Вы устанавливаете область печати, Вы должны сначала очистить области печати во всех листах. Cor Nouws предложил решение, показанное ниже (основано на идее David French). Он упоминает, что он получает листы, используя getByName(), потому что getByIndex() некорректно работает при наличии скрытых (hidden) листов.

Лист. 184: Печать из Calc

```
'Author: Cor Nouws <cno@nouenoff.nl>
Sub PrintSelectedArea (sSht$, nStC&, nStR&, nEndC&, nEndR&)
  Dim selArea(0) as new com.sun.star.table.CellRangeAddress
  Dim oDoc as object
  Dim oSheet as object
  Dim oSheets
  Dim i%
  oDoc = Thiscomponent
  oSheets = ThisComponent.Sheets
  For i = 0 to oSheets.getCount() - 1
    oSheet = ThisComponent.Sheets.getByIndex(i)
```

[1] Прим. ред. В настоящее время этот недостаток исправлен и по умолчанию печатается только область текущего листа.

```
    oSheet.setPrintareas(array())
  Next

  selArea(0).StartColumn = nStC
  selArea(0).StartRow = nStR
  selArea(0).EndColumn = nEndC
  selArea(0).EndRow = nEndR
  oSheet=ThisComponent.Sheets.getByName(sSht)
  oSheet.setPrintareas(selArea())
  oDoc.Print(Array())
End Sub
```

IV.34. Объединена ли эта ячейка (с другими)?

Когда ячейки объединены, объединенная область выступает как одна ячейка, идентифицированная верхним левым углом объединенной области. Целый интервал ячеек будет показан как объединенный при использовании функции isMerged(). Отдельная ячейка также будет показана как объединенная. Отдельная ячейка или первоначальный интервал ячеек для объединения может быть использована для установки свойства «объединенный» равным false. Я не знаю автоматического способа быстро найти ячейки, которые объединены.

Лист. 185: Объединённые ячейки

```
Sub MergeTest
  Dim oCell 'Временно хранит ячейку
  Dim oRange 'Первоначальный интервал ячеек
  Dim oSheet 'Четвертый лист

  oSheet = ThisComponent.getSheets().getByIndex(0)
  oRange = oSheet.getCellRangeByName("B2:D7")
  oRange.merge(True)
  Print "Объединен интервал = " & oRange.isMerged()  ' True

  REM Теперь получаем ячейку, которая входит в объединение merged
  REM и я могу сделать это!
  oCell = oSheet.getCellByPosition(2, 3) 'C4
  Print "Cell C4 merged = " & oCell.isMerged()  ' False
  oCell = oSheet.getCellByPosition(1, 1) 'B2
  Print "Cell B2 merged = " & oCell.isMerged()  ' True
  oCell.merge(False)
End Sub
```

IV.35. Написать свою функцию электронной таблицы Calc

Это только краткое введение в написание своих функций электронной таблицы Calc на языке Basic. Я пишу многое наизусть без проверки, поэтому без колебаний сообщайте мне об ошибках.

IV.35.1. Функции Calc, определенные пользователем

Обязательно проверьте сайт документации по OOo для более подробной информации по этой теме; я написал раздел по макросам для Calc. Также мои благодарности Робу Грею (Rob Gray [robberbaron@optusnet.com.au]) за материал для этого раздела.

Если Вы определяете функцию на языке Basic, Вы можете вызвать её из электронной таблицы Calc. Например, я создал документ Calc и добавил макрос `NumberTwo` в модуле Module1 в библиотеке Standard этого документа Calc. Я добавил вызов этой функции `=NumberTwo()` в вызове, и она вывела значение 2.

Лист. 186: Very Очень простая функция, которая возвращает значение 2.0.

```
Function NumberTwo() As Double
  NumberTwo() = 2.0
End Function
```

Чтобы функции, определенные пользователем (User Defined Functions = UDF), были доступны в Calc, библиотека должна быть загружена. Только библиотека Standard текущего документа и My Macros загружаются и становятся доступными при загрузке документа. Функции, определенные пользователем UDF, могут храниться в любой библиотеке, но такая библиотека должна быть загружена, иначе результатом будет ошибка #NAME? Вы можете загрузить макрос, используя менеджер макросов Сервис — Макросы — Управление макросами — (**Tools > Macros > Organize Macros >**) — OpenOffice.org Basic.

Дистрибутивы, использующие изменения фирмы Novell, распознают макросы VBA и оставляют корректными вызовы функций, определенных пользователем (UDF), в электронной таблице. Например, =Test1(34) сохраняется вместо преобразования формулы в =(test1;34); хотя изменение кода программы обычно требуется, оно так много проще.

Rob Gray использует вызов макроса BWStart для событий Document Open или Start Application в Лист. 187. Макрос BWStart загружает библиотеку Tools и две своих собственных библитеки макросов при загрузке документа.

Лист. 187: Загрузка трех библиотек макросов

```
Sub BWStart
  Dim oLibs As Object
  oLibs = GlobalScope.BasicLibraries
  'загрузим полезные или запрошенные библиотеки
  LibName="Tools"    'Из коллекции OO Macros & Dialogs
  If oLibs.HasByName (LibName) and (Not oLibs.isLibraryLoaded(LibName)) Then
    oLibs.LoadLibrary(LibName)
  End If
  LibName="VBA_Compat"  'из My Macros & Dialogs
```

```
  If oLibs.HasByName (LibName) and (Not oLibs.isLibraryLoaded(LibName)) Then
    oLibs.LoadLibrary(LibName)
  End If

  LibName="RecordedMacros"  'из текущего документа
  If oLibs.HasByName (LibName) and (Not oLibs.isLibraryLoaded(LibName)) Then
    oLibs.LoadLibrary(LibName)
  End If
End sub
```

IV.35.2. Оценка и проверка аргументов

Аргументы определенной пользователем функции электронной таблицы Calc определены для Ваших нужд. И Вы ответственны как за использование корректных аргументов, так и за проверку их в Вашем коде программы. Например, я могу определить функцию, которая принимает единственное числовое значение, но тогда я буду иметь проблему, если в качестве аргумента будет передан массив значений. Проверьте аргументы для определения, что нужно делать.

Лист. 188: Проверка аргумента, оформленная как функция Calc.

```
'Не забывайте, что строки и столбцы нумеруются с нуля — прим.перев.
Function EvalArgs(Optional x) As String
  Dim i As Integer
  Dim s As String
  If IsMissing(x) Then
    s = "Нет аргумента"
  ElseIf NOT IsArray(x) Then
    s = "Один аргумент типа " & TypeName(x)
  Else
    s = "Массив с границами x(" & _
         LBound(x, 1) & " по " & UBound(x, 1) & ", " & _
         LBound(x, 2) & " по " & UBound(x, 2) & ")"
  End If
  EvalArgs() = s
End Function
```

Небольшой эксперимент демонстрирует, что вызов =EVALARGS(A2:B4; A4:C7) возвращает строку «Array with bounds x(1 To 3, 1 To 2)». Другими словами, показывается первый аргумент как двумерный массив.

Единственный аргумент x соответствует интервалу ячеек A2:B4. Второй интервал ячеек полностью игнорируется. Добавьте второй, необязательный, аргумент для обработки второго интервала ячеек.

IV.35.3. Каков тип возвращаемого значения

(Прим. пер.: функция массива — возвращает двумерный массив, значения которого Calc размещает в прямоугольный интервал ячеек). Нет возможности узнать, какого типа значение будет возвращено. Обычно это не создаёт проблем. Для большинства типов Calc может правильно

обработать значение. Тем не менее, если функция вызвана как функция массива, то эта функция должна возвращать двумерный массив.

Лист. 189: Функция, которую можно вызывать в контексте массива.

```
Function ArrayCalc(a as Single)
 Dim x(0 To 4, 0 To 4) As Single
 Dim nC As Integer
 Dim n As Integer
 Dim m As Integer
 nC = 2
 For n = 0 To 4
   For m = 0 To 4
     x(n, m) = nC * a
     nC = nC + 1
   Next
 Next
 ArrayCalc = x()
End Function
```

Я поместил курсор в ячейку B8 и ввел: =ArrayCalc(3) , а затем нажал клавиши *Shift+Ctrl+Enter*, что означает для Calc ввести формулу массива. Если формула массива (array formula) не используется, то используется значение в верхнем левом углу интервала ячеек. Следующие значения появятся в ячейках B8:F12.

	B	C	D	E	F
8	6	9	12	15	18
9	21	24	27	30	33
10	36	39	42	45	48
11	51	54	57	60	63
12	66	69	72	75	78

Число ячеек, использованных для вывода данных, было определено размерностью возвращаемого массива. Размерность возвращаемого массива не определяется числом строк или столбцов, которые нужно заполнить. Не возможно оценить вызываемый контекст для того, чтобы узнать, сколько строк и столбцов нужно вернуть, по крайней мере, я не верю в такую возможность.

IV.35.4. Не изменяйте значений других ячеек электронной таблицы

Если функция вызвана из ячейки листа Sheet1, то вызванная функция не может менять любые другие ячейки sheet1. Такие изменения будут игнорироваться.

Глава V. Макросы текстового документа (Writer)

V.1. Что такое выделенный текст?

Выделенный текст (Selected text), по существу, является текстовым интервалом (text range) и ничем иным. После того, как выделение (selection) произведено, возможно получить текст [getString()] и установить значение текста [setString()]. Хотя строки ограничены размером 64K, размеры выделенного текста этим пределом не ограничены. Поэтому есть некоторые примеры, когда методы getString() и setString() дают результаты, которые я сам понять не могу. Поэтому, вероятно, лучше использовать перебор выделенного текста курсором и затем использования методов insertString() и insertControlCharacter() для объекта text, который поддерживает интерфейс Xtext. В документации специально подчеркивается, что следующие варианты пробела поддерживаются методом insertString(): blank (собственно пробел), tab (символ табуляции), cr (перевод каретки, который вставит конец параграфа «paragraph break») и lf (возврат каретки к началу строки, который вставит перевод строки «line break»).

Текст может быть выделен вручную, тогда курсор будет или в начале, или в конце выделения. Выделение имеет и точку начала, и точку конца, и Вы всегда знаете, которая из них начало (левый конец) и которая конец (правый конец выделения). Ниже этот метод и возникающие проблемы описаны подробно.

См. также:

`http://api.openoffice.org/docs/common/ref/com/sun/star/text/TextRange.html`

`http://api.openoffice.org/docs/common/ref/com/sun/star/text/XTextRange.html`

V.1.1. Находится ли курсор в текстовой таблице?

Часто ошибочно считают, что используются функции для выделенного текста с целью определить, находится ли курсов внутри текстовой таблицы. Контроллер (controller) текстового документа OOo Writer возвращает положение видимого курсора (view cursor), которое показывает, где находится курсор. Положение курсора имеет два интересных свойства: TextTable и Cell — и ещё текстовый курсор имеет свойства Textfield, TextFrame и TextSection. Используйте IsNull() для проверки этими свойствами, находится ли курсор в таблице. Убедитесь, экспериментируя с несколькими ячейками и выделенными областями.

Лист. 190: Находится ли курсор внутри текстовой таблицы?

```
Sub TableStuff
  Dim oVCurs    'Видимый курсор
  Dim oTable    'Текстовая таблица, содержащая текстовый курсор.
  Dim oCurCell  'Ячейка текстовой таблицы, содержащая текстовый курсор.

  oVCurs = ThisComponent.getCurrentController().getViewCursor()

  If IsEmpty(oVCurs.TextTable) Then
```

```
    Print "Курсор НЕ находится в текстовой таблице"
  Else
    oTable = oVCurs.TextTable
    oCurCell = oVCurs.Cell
    Print "Курсор находится в таблице " & oCurCell.CellName
  End If
End Sub
```

V.1.2. Можно ли проверить текущее выделение для текстовой таблицы (TextTable) или ячейки (Cell)?

Если я помещаю курсор внутрь текстовой таблицы и при этом не выделил целиком ячейку, то текущее выделение является объектом TextRanges. Выделенный текст из этой ячейки представлен объектом TextRange. Объект TextRange имеет свойство TextTable и свойство Cell. Эти два свойства не пустые, если интервал текста (text range) содержится в ячейке текстовой таблицы.

Обычно можно одновременно выделить несколько кусков текста. Когда выделены одна или несколько ячеек в текстовой таблице, тем не менее, можно получить только одно выделение, причем возвращенное текущее выделение равно TextTableCursor. Следующий макрос иллюстрирует эти возможности.

Лист. 191: Содержит ли текущее выделение текстовую таблицу?

```
Sub IsACellSelected
  Dim oSels         'Все части выделения
  Dim oSel          'отдельная часть выделения
  Dim i As Integer
  Dim sTextTableCursor$

  sTextTableCursor$ = "com.sun.star.text.TextTableCursor"

  oSels = ThisComponent.getCurrentController().getSelection()
  If oSels.supportsService("com.sun.star.text.TextRanges") Then
    For i=0 to oSels.getCount()-1
      oSel = oSels.getByIndex(i)
      If oSel.supportsService("com.sun.star.text.TextRange") Then
        If ( Not IsEmpty(oSel.TextTable) ) Then
          Print "Свойство "текстовая таблица" text table НЕ пустое"
        End If
        If ( Not IsEmpty(oSel.Cell) ) Then
          Print "Свойство Cell НЕ пустое"
        End If
      End If
    Next
  ElseIf oSels.supportsService(sTextTableCursor) Then
    REM По меньшей мере одна ячейка выделена целиком
    Print oSels.getRangeName()
  End If
End Sub
```

V.2. Что такое текстовые курсоры (Text Cursors)?

Обычный TextCursor является невидимым курсором, который независим от выводимого на экран курсора. Вы можете иметь больше одного текстового курсора одновременно, и Вы можете передвигать их, не влияя на видимый курсор. Видимый курсор также является текстовым курсором, который также поддерживает интерфейс XtextViewCursor. Существует только один видимый курсор, и он всегда виден на экране.

Объект TextCursor является объектом TextRange, который может быть перемещен внутри объекта Text. Стандартное передвижение включает варианты goLeft, goRight, goUp и goDown. Первый параметр является целым числом, показывающим, сколько символов или строк нужно передвинуть. Второй параметр булевский и показывает, должен ли при перемещении изменяться размер выделения (True) или нет. Значение True возвращается, если перемещение выполнено. Если курсор выделяет текст при передвижении влево и Вы хотите начать передвижение вправо, то Вы, вероятно, захотите использовать oCursor.goRight(0, False) для указанию курсору начать передвигаться вправо и не выделять текст. В результате не будет выделенного текста..

Объект TextCursor имеет и начало, и конец. Если начальная позиция равна конечной позиции, то текст не выделен и свойство IsCollapsed будет равно true.

Объект TextCursor поддерживает интерфейсы, которые позволяют передвижение и определение позиций в словах, предложениях и параграфах. Это может сберечь много времени.

> ***Внимание*** *Cursor.gotoStart() и Cursor.gotoEnd() передвигают курсор к началу и соответственно к концу документа, даже если курсор создан как интервал (over a range).*

См. также:

http://api.openoffice.org/docs/common/ref/com/sun/star/view/XViewCursor.html
http://api.openoffice.org/docs/common/ref/com/sun/star/text/TextCursor.html
http://api.openoffice.org/docs/common/ref/com/sun/star/text/XWordCursor.html
http://api.openoffice.org/docs/common/ref/com/sun/star/text/XSentenceCursor.html
http://api.openoffice.org/docs/common/ref/com/sun/star/text/XParagraphCursor.html
http://api.openoffice.org/docs/common/ref/com/sun/star/text/TextRange.html
http://api.openoffice.org/docs/common/ref/com/sun/star/text/XTextRange.html

V.2.1. Нельзя передвинуть курсор к «якорю» (anchor) TextTable.

Содержание текста привязано (заякорено) внутри текста. Якорь (anchor) доступен с использованием метода getAnchor(). К сожалению, текстовые курсоры не всегда совместимы с якорем. Следующий пример демонстрирует способы, которые не работают с якорем текстовой таблицы.

Лист. 192: Ошибка с якорями.

```
oAnchor = oTable.getAnchor()
oCurs.gotoRange(oAnchor.getStart(), False)
REM Error
oCurs = oAnchor.getText().createTextCursorByRange(oAnchor.getStart())
REM Error
oText.insertTextContent(oAnchor.getEnd(), oTable2, False)
REM Error
```

Меня недавно спрашивали, как удалить существующую таблицу и затем создать другую таблицу взамен. Можно использовать маленькую хитрость и передвинуть курсор к таблице с использованием текущего контроллера (controller).

```
ThisComponent.getCurrentController().select(oTable)
```

Следующий макрос демонстрирует, как создать таблицу и затем заменить её на другую таблицу.

Лист. 193: Создать текстовую таблицу взамен другой.

```
Sub Main
  Dim sName$
  Dim oTable
  Dim oAnchor
  Dim oCurs
  Dim oText

  oText = ThisComponent.getText()
  oTable = ThisComponent.createInstance("com.sun.star.text.TextTable")
  oTable.initialize(3, 3)
  'oTable.setName("wow")
  oCurs = ThisComponent.getCurrentController().getViewCursor()
  oText.insertTextContent(oCurs, oTable, False)
  oTable.setDataArray(Array(Array(1,2,3), Array(4,5,6), Array(7,8,9))
  sName = oTable.getName()

  Print "Создана таблица с именем " & sName
  oTable = ThisComponent.getTextTables().getByName(sName)
  oAnchor = oTable.getAnchor()

  REM Теперь передвинем видимый курсор view cursor к началу документа
  REM так, что можно продемонстрировать, как это работает.
  oCurs = ThisComponent.getCurrentController().getViewCursor()
  oCurs.gotoStart(False)
```

```
  REM Я бы очень хотел уметь передвигать курсор к якорю,
  REM но я не могу создать курсор на основе якоря, передвинуть его
  REM на якорь и т.п. Поэтому я использую хитрость и позволяю
  REM контроллеру передвинуть видимый курсор к таблице.
  REM К сожалению, нельзя передвинуть этот курсор к якорю...
  ThisComponent.getCurrentController().select(oTable)
  oTable.dispose()

  oTable = ThisComponent.createInstance("com.sun.star.text.TextTable")
  oTable.initialize(2, 2)
  oCurs = ThisComponent.getCurrentController().getViewCursor()
  oText.insertTextContent(oCurs, oTable, False)
  oTable.setDataArray(Array(Array(1, 2), Array(3, 4))
  sName = oTable.getName()
  Print "Создана таблица с именем " & sName
End Sub
```

Вы можете использовать особенность API, позволяющую передвинуть текстовый курсор в положение сразу после текстовой таблицы.

Лист. 194: Передвинуть текстовый курсор в положение ПЕРЕД текстовой таблицей.

```
  Dim oTable
  Dim oCurs

  oTable = ThisComponent.getTextTables.getByIndex(2)

  REM Передвигаем курсор на первую строку и первый столбец
  ThisComponent.getCurrentController().select(oTable)
  oCurs = ThisComponent.getCurrentController().getViewCursor()
  oCurs.goLeft(1, False)
```

V.2.2. Вставка чего-либо перед (или после) текстовой таблицы.

Если текстовая таблица является первым или последним элементом в документе, то не так-то легко вставить что-либо перед (или после) этой таблицы. Используя GUI можно передвинуть курсор к началу первой ячейки, а затем нажать enter для вставки новой строки в начало. Используя API нужно использовать insertTextContentBefore (или insertTextContentAfter).

Лист. 195: Вставить новый параграф перед текстовой таблицей.

```
Sub InsertParBeforeTable
  Dim oTable
  Dim oText
  Dim oPar
  oTable = ThisComponent.getTextTables().getByIndex(0)
  oText = ThisComponent.getText()
  oCurs = oText.createTextCursor()
  oPar = ThisComponent.createInstance("com.sun.star.text.Paragraph")
  oText.insertTextContentBefore ( oPar, oTable )
End Sub
```

Вы можете использовать этот способ для параграфа (paragraph), раздела текста (text section) или для текстовой таблицы.

V.2.3. Можно передвинуть курсор на закладку (Bookmark anchor).

Закладки (Bookmark anchors) совместимы с обычным текстовым курсором.

Лист. 196: Выделить закладку напрямую.

```
  Dim oAnchor  'Закладка Bookmark anchor
  Dim oCursor  'Курсор на левом конце закладки
  Dim oMarks

  oMarks = ThisComponent.getBookmarks()
  oAnchor = oMarks.getByName("MyMark").getAnchor()
  oCursor = ThisComponent.getCurrentController().getViewCursor()
  oCursor.gotoRange(oAnchor, False)
```

Можно также сравнить положение курсора для определения, находится он до или после закладки, но это не работает с закладкой (anchor) TextTable.

Лист. 197: Сравнить видимый курсор с закладкой.

```
Sub CompareViewCursorToAnchor()
  Dim oAnchor  'Закладка Bookmark anchor
  Dim oCursor  'Курсор на левом краю закладки
  Dim oMarks

  oMarks = ThisComponent.getBookmarks()
  oAnchor = oMarks.getByName("MyMark").getAnchor()
  oCursor = ThisComponent.getCurrentController().getViewCursor()

  If NOT EqualUNOObjects(oCursor.getText(), oAnchor.getText()) Then
    Print "Видимый курсор и закладка используют разные текстовые объекты"
    Exit Sub
  End If

  Dim oText, oEnd1, oEnd2
  oText = oCursor.getText()
  oEnd1 = oCursor.getEnd() : oEnd2 = oAnchor.getEnd()
  If oText.compareRegionStarts(oEnd1, oEnd2) >= 0 Then
    Print "Конец курсора Cursor END слева от конца закладки"
  Else
    Print "Конец курсора Cursor END справа от конца закладки"
  End If
End Sub
```

Нужны ещё эксперименты с различными типами закладок (anchor), которые из них правильно работают с курсорами.

V.2.4. Вставка текста в место закладки

Лист. 198: Вставить текст в место закладки.

```
oBookMark = oDoc.getBookmarks().getByName("<ЗдесьИмяВашейЗакладки>")
oBookMark.getAnchor.setString("Строка, которую Вы хотите вставить")
```

V.3. Работа с текстом (Andrew's Selected Text Framework)

Большинство проблем с использованием выделенного текста выглядят одинаково на абстрактном уровне.

```
If ничего не выделено then
  выполнить работу с документом в целом
else
  для каждой выделенной области
    выполнить работу с выделенной областью
```

Трудность в том, чтобы написать работающий макрос, который перебирает все выделенные области или переключается между двумя курсорами.

V.3.1. Выделен ли текст?

Документация утверждает, что если нет текущего контроллера (controller), то getCurrentSelection() возвращает Null вместо выделения. Я не все понимаю в этом, но я поймал их на слове и проверил сказанное.

Если счетчик числа выделенных областей равен нулю, то ничего не выделено. Я никогда не видел, чтобы счетчик числа выделенных областей был равен нулю, но я это проверил. Если никакой текст не выделен, то существует одна выделенная область нулевой длины. Я видел примеры, где нулевая длина выделения определялась так:

Лист. 199: Этот способ создаёт проблемы, если выделенный текст больше 64K .

```
If Len(oSel.getString()) = 0 Then ничего не выделено
```

Проблема с этим способом в том, что выделенный текст может содержать больше 64K символов, а строка не может содержать больше 64K символов. Я считаю этот способ ненадежным. Лучше будет создать текстовый курсор из выделенной области, а затем проверять, одинаковы ли начало и конец выделения

Лист. 200: Вот так я использвал «oDoc.Text» вместо «oSel.Text».

```
oCursor = oDoc.Text.CreateTextCursorByRange(oSel)
If oCursor.IsCollapsed() Then ничего не выделено
```

Вот функция, которая выполняет полную проверку. Код программы в Лист. 200 использует текстовый объект документа. Если выделение не входит

в текстовый объект первоначального документа (the document's primary text object), то будет ошибка. Лист. 201 получает текстовый объект из объекта выделения, что намного надежнее.

Лист. 201: Находится ли курсор внутри текстовой таблицы?

```
Function IsAnythingSelected(oDoc As Object) As Boolean
  Dim oSels  'Все элементы выделения
  Dim oSel         'Отдельный элемент выделения
  Dim oCursor      'Временный курсор

  IsAnythingSelected = False
  If IsNull(oDoc) Then Exit Function
  ' Текущее выделение в текущем контроллере.
  'Если нет текущего контроллера current controller, возвращается NULL.
  oSels = oDoc.getCurrentSelection()
  If IsNull(oSels) Then Exit Function

  REM Я никогда не видел количество элементов выделения равным нулю
  If oSels.getCount() = 0 Then Exit Function

  REM Если имеем множественное выделение, то обязательно
  REM что-то выделено
  If oSels.getCount() > 1 Then
    IsAnythingSelected = True
  Else
    REM Если имеем только один элемент выделения, то проверяем его,
    REM является ли он выделением нулевой длины collapsed. Другими словами,
    REM совпадают ли конец и начало выделения.
    REM Заметьте, что использован текстовый объект из объекта выделения
    REM потому что это надежнее, чем предполагать, что это тот же
    REM текстовый объект, что и текстовый объект документа.
    oSel = oSels.getByIndex(0)
    oCursor = oSel.getText().CreateTextCursorByRange(oSel)
    If Not oCursor.IsCollapsed() Then IsAnythingSelected = True
  End If
End Function
```

V.3.2. Как получить выделенную область (Selection)

Получить выделение не так просто, потому что возможно одновременное выделение нескольких не связанных областей. Некоторые выделения могут быть пустыми, а другие — не пустыми. Нижеприведенный код программы обрабатывает текстовые выделения во всех этих случаях. Он перебирает все выделенные области и печатает их.

Лист. 202: Можно обрабатывать одновременное выделение нескольких не связанных областей.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub MultipleTextSelectionExample
  Dim oSels As Object, oSel As Object
  Dim lSelCount As Long, lWhichSelection As Long
```

```
  ' Текущее выделение текущего контроллера.
  ' Если нет текущего контроллера, возвращается NULL.
  oSels = ThisComponent.getCurrentSelection()
  If Not IsNull(oSels) Then
    lSelCount = oSels.getCount()
    For lWhichSelection = 0 To lSelCount – 1
      oSel = oSels.getByIndex(lWhichSelection)
      MsgBox oSel.getString()
    Next
  End If
End Sub
```

См. также:

http://api.openoffice.org/docs/common/ref/com/sun/star/text/XTextRange.html

V.3.3. Выделенный текст: Какой конец — начало, какой — конец

Выделения, по существу, являются текстовыми областями, которые имеют начало и конец. Хотя выделения имеют и начало, и конец, но какая сторона выделения является началом, а какая — концом, определяется методом выделения (selection method). Текстовый объект предоставляет методы для ставнения позиций начала и конца текстовых областей. Метод «короткое сравнение начала выделенных областей compareRegionStarts (XTextRange R1, XTextRange R2)» возвращает 1, если R1 начинается раньше чем R2 , возвращает 0 если R1 начинается в той же позиции, что и R2, и возвращает -1 если R1 начинается позже, чем R2. Метод «короткое сравнение концов выделенных областей compareRegionEnds (XTextRange R1, XTextRange R2)» возвращает 1, если R1 заканчивается раньше, чем R2 , 0, если R1 заканчивается в той же позиции, что и R2 и -1, если R1 заканчивается после R2. Я использую следующие два метода для того, чтобы найти минимальную левую позицию и максимальную правую позицию выделенного текста.

Лист. 203: Определить, начало или конец находятся раньше (if the start or end comes first).

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
'oSel = текстовое выделение или интервал курсора
Function GetLeftMostCursor(oSel As Object) As Object
  Dim oRange   'Самый левый интервал  range.
  Dim oCursor  'Курсор на самом левом интервале.

  If oSel.getText().compareRegionStarts(oSel.getEnd(), oSel) >= 0 Then
    oRange = oSel.getEnd()
  Else
    oRange = oSel.getStart()
  End If
  oCursor = oSel.getText().CreateTextCursorByRange(oRange)
  oCursor.goRight(0, False)
  GetLeftMostCursor = oCursor
```

```
End Function

'Author: Andrew Pitonyak; email: andrew@pitonyak.org
'oSel = текстовое выделение или интервал курсора
Function GetRightMostCursor(oSel As Object) As Object
  Dim oRange   'Самый правый интервал.
  Dim oCursor  'Курсор на самом правом интервале.

  If oSel.getText().compareRegionStarts(oSel.getEnd(), oSel) >= 0 Then
    oRange = oSel.getStart()
  Else
    oRange = oSel.getEnd()
  End If
  oCursor = oSel.getText().CreateTextCursorByRange(oRange)
  oCursor.goLeft(0, False)
  GetRightMostCursor = oCursor
End Function
```

Совет *Я изменил эти разделы, чтобы получить текстовый объект из объектов выделения, а не использовать текстовый объект документа.*

См. также:

http://api.openoffice.org/docs/common/ref/com/sun/star/text/XTextCursor.html

http://api.openoffice.org/docs/common/ref/com/sun/star/text/XSimpleText.html

http://api.openoffice.org/docs/common/ref/com/sun/star/text/XTextRangeCompare.html

V.3.4. Работа макроса с выделенным текстом (Selected Text Framework Macro)

Мне потребовалось много времени, чтобы понять, как перебирать выделенный текст с использованием курсоров, поэтому я написал много макросов для того, что я считаю неверным путем. Сейчас я использую высокий уровень фреймов (high level framework) для выполнения этих задач. Идея в том, что если ничего не выделено, то макрос спрашивает, должен ли макрос выполняться для всего документа. Если ответ «Да», то создается курсор в начале и в конце документа, а затем вызывается «рабочий» макрос. Если выделенный текст есть, то перебирается каждое выделение, создается курсор в начале и конце выделения, а затем вызывается «рабочий» макрос для каждого такого выделения.

V.3.4.A. Непригодный фрейм (Rejected Framework)

Я решительно отвергаю фрейм (framework), приведенный ниже, потому что это слишком длинно и громоздко повторять каждый раз, что я хотел перебрать текст. Он, тем не менее, логичный. Вы можете предпочесть его и работать с ним.

Лист. 204: Громоздкий, но работающий вариант работы с выделенным текстом (selected text framework).

```
Sub IterateOverSelectedTextFramework
  Dim oSels As Object, oSel As Object, oText As Object
  Dim lSelCount As Long, lWhichSelection As Long
  Dim oLCurs As Object, oRCurs As Object

  oText = ThisComponent.Text
  If Not IsAnythingSelected(ThisComponent) Then
    Dim i%
    i% = MsgBox("No text selected!" + Chr(13) + _
      "Вызвать обработку для ВСЕГО документа?", _
      1 OR 32 OR 256, "Warning")
    If i% <> 1 Then Exit Sub
    oLCurs = oText.createTextCursor()
    oLCurs.gotoStart(False)
    oRCurs = oText.createTextCursor()
    oRCurs.gotoEnd(False)
    CallYourWorkerMacroHere(oLCurs, oRCurs, oText)
  Else
    oSels = ThisComponent.getCurrentSelection()
    lSelCount = oSels.getCount()
    For lWhichSelection = 0 To lSelCount – 1
      oSel = oSels.getByIndex(lWhichSelection)
      'Если я хочу проверить, что никакой текст НЕ выделен, я могу
      'сделать следующее:
      'oLCurs = oText.CreateTextCursorByRange(oSel)
      'If oLCurs.isCollapsed() Then ...
      oLCurs = GetLeftMostCursor(oSel, oText)
      oRCurs = GetRightMostCursor(oSel, oText)
       CallYourWorkerMacroHere(oLCurs, oRCurs, oText)
    Next
  End If
End Sub
```

V.3.4.B. Приемлемый способ (Framework)

Я предпочел создать нижеследующий фрейм (framework). Он возвращает двумерный массив позиций начала и конца курсоров, каждый из которых нужно перебирать. Это позволяет написать минимальный код программы, который следует использовать для перебора выделений текста или для целого документа.

Лист. 205: Создать курсоры, охватывающие выбранные текстовые области.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
'sPrompt : Как спросить, нужно ли просматривать весь текст
'oCurs() : Содержит возвращаемый массив курсоров
'Возвращает true, если нужно выполнять итерации, и false, если не нужно
Function CreateSelectedTextIterator(oDoc As Object, sPrompt As String,
oCurs()) As Boolean
  Dim lSelCount As Long       'Число элементов выделения
  Dim lWhichSelection As Long 'Текущий элемент выделения
```

```
  Dim oSels  'Все элементы выделения
  Dim oSel          'Отдельный элемент выделения
  Dim oLCurs      'Курсор на левом конце текущего элемента выделения
  Dim oRCurs      'Курсор на правом краю текущего элемента выделения

  CreateSelectedTextIterator = True
  If Not IsAnythingSelected(ThisComponent) Then
    Dim i%
    i% = MsgBox("Нет Выделенного текста!" + Chr(13) + sPrompt, _
      1 OR 32 OR 256, "Warning")
    If i% = 1 Then
      oLCurs = oDoc.getText().createTextCursor()
      oLCurs.gotoStart(False)
      oRCurs = oDoc.getText().createTextCursor()
      oRCurs.gotoEnd(False)
      oCurs = DimArray(0, 1)
      oCurs(0, 0) = oLCurs
      oCurs(0, 1) = oRCurs
    Else
      oCurs = DimArray()
      CreateSelectedTextIterator = False
    End If
  Else
    oSels = ThisComponent.getCurrentSelection()
    lSelCount = oSels.getCount()
    oCurs = DimArray(lSelCount - 1, 1)
    For lWhichSelection = 0 To lSelCount - 1
      oSel = oSels.getByIndex(lWhichSelection)
      REM Если нужно проверить, что никакой текст НЕ выделен, то можно
      REM сделать следующее:
      REM oLCurs = oSel.getText().CreateTextCursorByRange(oSel)
      REM If oLCurs.isCollapsed() Then ...
      oLCurs = GetLeftMostCursor(oSel)
      oRCurs = GetRightMostCursor(oSel)
      oCurs(lWhichSelection, 0) = oLCurs
      oCurs(lWhichSelection, 1) = oRCurs
    Next
  End If
End Function
```

V.3.4.C. Основной рабочий макрос

Это пример того, что называется «рабочая» процедура.

Лист. 206: Вызов процедуры PrintEachCharacterWorker для каждого выделения.

```
Sub PrintExample
  Dim oCurs(), i%
  If Not CreateSelectedTextIterator(ThisComponent, _
    "Печатать символы для всего документа?", oCurs()) Then Exit Sub
  For i% = LBound(oCurs()) To UBound(oCurs())
    PrintEachCharacterWorker(oCurs(i%, 0), oCurs(i%, 1))
  Next i%
End Sub
```

```
Sub PrintEachCharacterWorker(oLCurs As Object, oRCurs As Object)
  Dim oText
  oText = oLCurs.getText()
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Sub
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub
  oLCurs.goRight(0, False)
  Do While oLCurs.goRight(1, True) and oText.compareRegionEnds(oLCurs,
oRCurs) >= 0
    Print "Символ Character = '" & oLCurs.getString() & "'"
    REM Это приведет к тому, что текущий выделенный текст
    REM больше не будет выделеннымno longer selected
    oLCurs.goRight(0, False)
  Loop
End Sub
```

V.3.5. Подсчет числа предложений

Это написано наспех, поэтому можете использовать на Ваш страх и риск. Я нашел несколько ошибок в этом «курсоре предложений» при написании моей книги, см. подробнее там на странице 286.

Лист. 207: Подсчет числа предложений с использованием «курсора предложений» (sentence cursor).

```
'Этот макрос, скорее всего, даст ошибку, если в документе есть таблицы,
'так как курсор предложений не сможет войти внутрь таблицы, но я это не
'проверял.
Sub CountSentences
  Dim oCursor          'Текстовый курсор
  Dim oSentenceCursor  'Текстовый курсор.
  Dim oText
  Dim i

  oText = ThisComponent.Text
  oCursor = oText.CreateTextCursor()
  oSentenceCursor = oText.CreateTextCursor()

  'Передвигаем курсор к началу документа
  oCursor.GoToStart(False)
  Do While oCursor.gotoNextParagraph(True)
    'Теперь целый параграф выделен
    oSentenceCursor.gotoRange(oCursor.getStart(), False)
    Do While oSentenceCursor.gotoNextSentence(True) AND_
            oText.compareRegionEnds(oSentenceCursor, oCursor) >= 0
      oSentenceCursor.goRight(0, False)
      i = i + 1
    Loop
    oCursor.goRight(0, False)
  Loop
  MsgBox i, 0, "Число предложений"
End Sub
```

V.3.6. Удаление пустых промежутков текста и пустых строк, более подробный пример

Этот набор макросов заменяет все куски текста, состоящие только из пробелов , на один пробел. Макрос можно легко модифицировать для замены разных типов пробелов. Разные типы пробелов упорядочены по важности, так что если несколько обычных пробелов, за которыми следует новый параграф, то можно оставить этот новый параграф, а пробелы удалить. Это приведет также к удалению начальных и завершающих пробелов в строке.

V.3.6.A. Определение пробела (White Space)

Для решения этой проблемы первой задачей было определить, какие символы считать пробелами (white space). Можно просто изменить нижеследующее определение пробела (white space), чтобы игнорировать определенные символы.

Лист. 208: Пробелы

```
'Обычно это выполняется циклом по массиву, что должно быть быстрее,
'но я не знаю, как при этом использовать статическую инициализацию static
initializers
Function IsWhiteSpace(iChar As Integer) As Boolean
  Select Case iChar
  Case 9, 10, 13, 32, 160
    IsWhiteSpace = True
  Case Else
    IsWhiteSpace = False
  End Select
End Function
```

V.3.6.B. Последовательности символов для удаления

Далее нужно определить, что удалять и что оставлять. Я предпочел следующую процедуру.

Лист. 209: Удаление пробелов

```
'-1 означает удаление предыдущего символа
' 0 означает игнорирование данного символа
' 1 означает удаление данного символа
' Значимость от наибольшей к наименьшей: 0, 13, 10, 9, 160, 32
Function RankChar(iPrevChar, iCurChar) As Integer
  If Not IsWhiteSpace(iCurChar) Then      'Текущий символ не пробел,
игнорируем его
    RankChar = 0
  ElseIf iPrevChar = 0 Then               'Начало строки, а текущий символ -
пробел
    RankChar = 1                          'поэтому удаляем текущий символ.
  ElseIf Not IsWhiteSpace(iPrevChar) Then 'Текущий символ – пробел, а
предыдущий – нет
    RankChar = 0                          'игнорируем текущий символ
```

```
  ElseIf iPrevChar = 13 Then                'Предыдущий символ – пробел
наибольшей важности (новый параграф)
    RankChar = 1                            'удаляем текущий символ
  ElseIf iCurChar = 13 Then                 'Текущий символ – пробел
наибольшей важности
    RankChar = -1                           'удаляем предыдущий символ
  ElseIf iPrevChar = 10 Then                'предыдущий символ – новая строка
    RankChar = 1                            'удаляем текущий символ
  ElseIf iCurChar = 10 Then                 'текущий символ – новая строка
    RankChar = -1                           'удаляем предыдущий символ
  ElseIf iPrevChar = 9 Then                 'предыдущий символ - табуляция
    RankChar = 1                            'удаляем текущий символ
  ElseIf iCurChar = 9 Then                  'текущий символ - табуляция
    RankChar = -1                           'удаляем предыдущий символ
  ElseIf iPrevChar = 160 Then               'предыдущий символ – "твердый
пробел"
    RankChar = 1                            'удаляем текущий символ
  ElseIf iCurChar = 160 Then                'текущий символ – твердый пробел
    RankChar = -1                           'удаляем предыдущий символ
  ElseIf iPrevChar = 32 Then                'предыдущий символ - пробел
    RankChar = 1                            'удаляем текущий символ
  ElseIf iCurChar = 32 Then                 'текущий символ - пробел
    RankChar = -1                           'удаляем предыдущий символ
  Else                                      'Сюда мы не должны попасть
    RankChar = 0                            'просто игнорируем
  End If
End Function
```

V.3.6.C. Стандартный макрос перебора выделенного текста

Это стандартный формат для решения, нужно ли выполнить работу для всего документа или только для части документа.

Лист. 210: Удаление последовательностей пробелов (white space).

```
'Удаляем все последовательности пробелов!
'Если текст выделен, то только из выделенной области.
Sub RemoveEmptySpace
  Dim oCurs(), i%
  If Not CreateSelectedTextIterator(ThisComponent, _
    "Все пробелы удалить из ВСЕГО докумета?", oCurs()) Then Exit Sub
  For i% = LBOUND(oCurs()) To UBOUND(oCurs())
     RemoveEmptySpaceWorker (oCurs(i%, 0), oCurs(i%, 1))
  Next i%
End Sub
```

V.3.6.D. «Рабочий» макрос

В этом макросе выполняется основная работа.

Лист. 211: Перебрать все курсоры и удалить пробелы (white space).

```
Sub RemoveEmptySpaceWorker(oLCurs As Object, oRCurs As Object)
  Dim sParText As String, i As Integer
  Dim oText

  oText = oLCurs.getText()
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Sub
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub

  Dim iLastChar As Integer, iThisChar As Integer, iRank As Integer
  iLastChar = 0
  iThisChar = 0
  oLCurs.goRight(0, False)
  Do While oLCurs.goRight(1, True)
    iThisChar = Asc(oLCurs.getString())
    i = oText.compareRegionEnds(oLCurs, oRCurs)
    'Если находимся на последнем символе!
    'то всегда удаляем пробел
    If i = 0 Then
      If IsWhiteSpace(iThisChar) Then oLCurs.setString("")
      Exit Do
    End If
    'Если прошли за конец, то выходим из цикла
    If i < 0 Then Exit Do
    iRank = RankChar(iLastChar, iThisChar)
    If iRank = 1 Then
      'Собираемся удалить этот символ.
      'Не меняем iLastChar, потому что он не менялся!
      'Print "Удаление текущего символа, предыдущий " + iLastChar + " и
текущий " + iThisChar
      oLCurs.setString("")
    ElseIf iRank = -1 Then
      'Отменяем выделение текущего символа и выделяем предыдущий
      '(слева).
      oLCurs.goLeft(2, True)
      'Print "Удаление предыдущего символа, предыдущий " + iLastChar + " и
текущий " + iThisChar
      oLCurs.setString("")
      oLCurs.goRight(1, False)
      iLastChar = iThisChar
    Else
      oLCurs.goRight(0, False)
      iLastChar = iThisChar
    End If
  Loop
End Sub
```

V.3.7. Удаление пустых параграфов, другой пример

Лучше будет установить Автоформат (AutoFormat) для удаления пустых параграфов, а затем применить его к документу в запросе. Выберите в меню Сервис — Автозамена (Tools=>AutoCorrect/AutoFormat...) и затем выберите вкладку Параметры. Одной из строк будет «Удалять пустые абзацы» (Remove Blank Paragraphs). Убедитесь, что это включено. Теперь можно выполнить автоформат документа, и все пустые параграфы исчезнут.

Для удаления пустых параграфов только из выделенной части документа придётся использовать макрос. Если текст выделен, то пустые параграфы удаляются из выделенного текста. Если ничего не выделено, то удаление выполняется для всего документа. Первый макрос перебирает весь выделенный текст. Если ничего не выделено, макрос создаёт курсор от начала до конца документа и затем работает с документом в целом. Первым делом нужно найти способ перебрать текст по параграфам (paragraphs). Макрос удаления пустого места более надежен, потому что он не извлекает строки для обработки..

Лист. 212: Удаление пустых параграфов (paragraphs).

```
Sub RemoveEmptyParsWorker(oLCurs As Object, oRCurs As Object)
  Dim sParText As String, i As Integer
  Dim oText

  oText = oLCurs
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Sub
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub
  oLCurs.goRight(0, False)
  Do While oLCurs.gotoNextParagraph(TRUE) and
oText.compareRegionEnds(oLCurs, oRCurs) > 0
    'Да, я знаю, что ограничен здесь длиной 64K
    'Если один из параграфов длиннее чем  64K
    'тогда это не сработает!
    sParText = oLCurs.getString()
    i = Len(sParText)
    'Не получился короткий логичный цикл. Увы!
    Do While i > 0
      If (Mid(sParText,i,1) = Chr(10)) OR (Mid(sParText,i,1) = Chr(13)) Then
        i = i – 1
      Else
        i = -1
      End If
    Loop
    If i = 0 Then
      oLCurs.setString("")
    Else
      oLCurs.goLeft(0,FALSE)
    End If
  loop
End Sub
```

V.3.8. Выделенный текст, вопросы скорости выполнения, подсчет слов

Каждый, кто изучал алгоритмы, скажет, что хороший алгоритм предпочтительнее быстрого компьютера. Когда я писал свой первый макрос обработки пустых строк и пробелов, я использовал извлечение строк. Это может привести к потере информации или к ошибкам, если длина строк больше 64K. Позднее я написал макрос, использующий курсоры, и получил жалобы на медленную работу. Возникает вопрос, есть ли способ получше?

V.3.8.A. Поиск в выделенном тексте для подсчета слов

Andrew Brown, автор сайта `http://www.darwinwars.com` (содержит полезную информацию о макросах) спросил о выполнении поиска внутри выделенного текста. См. раздел по поиску в выделенном тексте.

Я обнаружил, что это было использовано для подсчёта количества слов в документе и что это очень медленно, слишком медленно.

V.3.8.B. Использование строк (Strings) для подсчёта числа слов

Существующий код программы подсчитывал число пробелов в выделенном тексте и использовал его для определения числа слов. Позднее я написал свой вариант, который был чуть более универсальным, немного быстрее, и давал верный результат.

Лист. 213: Подсчитать слова надежным, но медленным способом.

```
Function ADPWordCountStrings(vDoc) As String
  REM Добавьте, если нужно, еще символы, которые Вы считаете разделителями
слов!
  Dim sSeps$
  sSeps = Chr$(9) & Chr$(13) & Chr$(10) & " ,;."
  Dim bSeps(256) As Boolean, i As Long
  For i = LBound(bSeps()) To UBound(bSeps())
    bSeps(i) = False
  Next
  For i = 1 To Len(sSeps)
    bSeps(Asc(Mid(sSeps, i, 1))) = True
  Next

  Dim nSelChars As Long, nSelwords As Long, nSel%, nNonEmptySel%, j As Long,
s$
  Dim vSelections, vSel, oText, oCursor
  'Текущее выделение в текущем контроллере
  'Если текущего контроллера нет, возвращается NULL.
  vSelections = vDoc.getCurrentSelection()
  If IsNull(vSelections) Then
    nSel = 0
  Else
    nSel = vSelections.getCount()
  End If
  nNonEmptySel = 0
  Dim lTemp As Long, bBetweenWords As Boolean, bIsSep As Boolean
```

```
  On Local Error Goto BadOutOfRange
  Do While nSel > 0
    nSel = nSel — 1
    s = vSelections.GetByIndex(nSel).getString()
    REM Проверяем, пустое ли выделение
    lTemp = Len(s)
    If lTemp > 0 Then
      nSelChars = nSelChars + lTemp
      nNonEmptySel = nNonEmptySel + 1
      REM Начинается ли выделение на слове?
      If bSeps(Asc(Mid(s, 1, 1))) Then
        bBetweenWords = True
      Else
        bBetweenWords = False
        nSelWords = nSelWords + 1
      End If
      For j = 2 To lTemp
        bIsSep = bSeps(Asc(Mid(s, j, 1)))
        If bBetweenWords <> bIsSep Then
          If bBetweenWords Then
            REM Считаем, что получили еще одно слово, только если
            REM были между словами и перестали быть между словами!
            bBetweenWords = False
            nSelWords = nSelWords + 1
          Else
            bBetweenWords = True
          End If
        End If
      Next
    End If
  Loop
  On Local Error Goto 0

  Dim nAllChars As Long, nAllWords As Long, nAllPars As Long
  ' доступ к статистике (о документе)
  nAllChars = vDoc.CharacterCount
  nAllWords = vDoc.WordCount
  nAllPars = vDoc.ParagraphCount

  Dim sRes$
  sRes = "В документе находятся:" & chr(13) & nAllWords & "  слов. "  &_
    chr(13) & "(" & nAllChars & " символов." & chr(13) & nAllPars &_
    " параграфов (абзацев).)" & chr(13) & chr(13)
  If nNonEmptySel > 0 Then
    sRes = sRes  & "В выделенном тексте находятся:" & chr(13) & nSelWords
&_
      "  слов" & chr(13) & "(" & nSelChars  & " символов)" &_
      chr(13) & "В " & str(nNonEmptySel) & " элементах выделения"
    If nNonEmptySel > 1 Then sRes = sRes & " (selections)"
    sRes=sRes & "." & chr(13) & chr(13) & "Невыделенная часть документа
содержит:" &_
      chr(13)& str(nAllWords-nSelWords) & " слов."
  End If
  'MsgBox(sRes,64,"ADP Word Count")
  ADPWordCountStrings = sRes
```

```
  Exit Function

BadOutOfRange:
  bIsSep = False
  Resume Next
End Function
```

Каждый выделенный интервал извлекается как строка (string). Это приводит к ошибке, если выделенный текст больше 64K. Код ASCII каждого символа проверяется, считается ли он разделителем слов. Это делается через просмотр массива. Такой способ был эффективен, но приводил к ошибке, если встречался специальный символ, чей код ASCII был больше, чем размерность массива, поэтому используется процедура обработки ошибок. Специальная обработка выполняется, чтобы получить корректные значения для различных вариантов выделения. Уходит около 2.7 секунд для подсчета 8000 слов.

V.3.8.C. Использование курсора (Character Cursor) для подсчета числа слов

Пытаясь избежать ограничения на размер 64K, я написал вариант, который использует курсоры для просмотра текста по одному символу за раз. Этот вариант требует 47 секунд для подсчета тех же 8000 слов. Он использует тот же алгоритм, что и макрос с извлечением срок (string), но потери от использования курсора для поиска каждого символа не позволяют его использовать.

Лист. 214: Подсчёт слов с использованием курсора (character cursor).

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Function ADPWordCountCharCursor(vDoc) As String
  Dim oCurs(), i%, lNumWords As Long
  lNumWords = 0
  If Not CreateSelectedTextIterator(vDoc, _
    "Подсчитать слова во всем документе?", oCurs()) Then Exit Function
  For i% = LBound(oCurs()) To UBound(oCurs())
    lNumWords = lNumWords + WordCountCharCursor(oCurs(i%, 0), oCurs(i%, 1),
vDoc.Text)
  Next
  ADPWordCountCharCursor = "Всего слов = " & lNumWords
End Function

Function WordCountCharCursor(oLCurs, oRCurs, oText)
  Dim lNumWords As Long
  lNumWords = 0
  WordCountCharCursor = lNumWords
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Function
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Function

  Dim sSeps$
  sSeps = Chr$(9) & Chr$(13) & Chr$(10) & " ,;."
  Dim bSeps(256) As Boolean, i As Long
```

```
  For i = LBound(bSeps()) To UBound(bSeps())
    bSeps(i) = False
  Next
  For i = 1 To Len(sSeps)
    bSeps(Asc(Mid(sSeps, i, 1))) = True
  Next

  On Local Error Goto BadOutOfRange
  Dim bBetweenWords As Boolean, bIsSep As Boolean
  oLCurs.goRight(0, False)
  oLCurs.goRight(1, True)
  REM Начинается ли элемент выделения со слова?
  If bSeps(Asc(oLCurs.getString())) Then
    bBetweenWords = True
  Else
    bBetweenWords = False
    lNumWords = lNumWords + 1
  End If
  oLCurs.goRight(0, False)

  Do While oLCurs.goRight(1, True) and oText.compareRegionEnds(oLCurs,
oRCurs) >= 0
    bIsSep = bSeps(Asc(oLCurs.getString()))
    If bBetweenWords <> bIsSep Then
      If bBetweenWords Then
        REM Добавляем к счетчику еще одно слово, только если были
        REM между словами и теперь не находимся между словами!
        bBetweenWords = False
        lNumWords = lNumWords + 1
      Else
        bBetweenWords = True
      End If
    End If
    oLCurs.goRight(0, False)
  Loop
  WordCountCharCursor = lNumWords
  Exit Function

BadOutOfRange:
  bIsSep = False
  Resume Next
End Function
```

V.3.8.D. Использование курсора слов (Word Cursor) для подсчета числа слов

Это до сих пор наиболее быстрый способ. Он использует курсор слов (word cursor) и позволяет OOo определить, где слова начинаются и где заканчиваются. Он проверяет те же 8000 слов за 1.7 секунд. Этот макрос передвигается от слова к слову, подсчитывая, сколько разделителей слов он нашел. Поэтому результат может быть ошибаться (отличаться) на одно слово. Я нашел ошибки в курсорах слов (word cursors) во время написания моей книги.

Лист. 215: Подсчитать слова, используя курсор слов (word cursor).

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Function ADPWordCountWordCursor(vDoc) As String
  Dim oCurs(), i%, lNumWords As Long
  lNumWords = 0
  If Not CreateSelectedTextIterator(vDoc, _
    "Подсчитать слова во всем документе?", oCurs()) Then Exit Function
  For i% = LBound(oCurs()) To UBound(oCurs())
    lNumWords = lNumWords + WordCountWordCursor(oCurs(i%, 0), oCurs(i%, 1),
vDoc.Text)
  Next
  ADPWordCountWordCursor = "Всего слов = " & lNumWords
End Function

Function WordCountWordCursor(oLCurs, oRCurs, oText)
  Dim lNumWords As Long
  lNumWords = 0
  WordCountWordCursor = lNumWords
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Function
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Function
  oLCurs.goRight(0, False)
  Do While oLCurs.gotoNextWord(False) and oText.compareRegionEnds(oLCurs,
oRCurs) >= 0
    lNumWords = lNumWords + 1
  Loop
  WordCountWordCursor = lNumWords
End Function
```

V.3.8.E. Итоговые размышления о подсчете числа слов и скорости выполнения

Если решение проблемы слишком медленное, то может существовать другой способ. В OOo курсоры могут передвигаться на символы, слова, параграфы (paragraphs). Курсор, который Вы выбираете, значительно меняет скорость работы.

Если нужно подсчитать число слов в выделенном тексте, я рекомендую посмотреть на сайт Andrew Brown, `http://www.darwinwars.com`, потому что он активно занимается подсчетом числа слов. Действительно, он представил макрос, который я считаю работающим правильно, см. следующий раздел.

V.3.9. Подсчет слов: нужно использовать вот этот макрос!

Следующий макрос прислан мне Andrew Brown, как уже говорилось. Действительно стоит посетить его сайт, там много хороших вещей, а если Вы пропустили его, то есть написанная им книга. Она не о программировании макросов, но я просто наслаждался многими её разделами. Убедитесь!

Лист. 216: Надежный подсчет слов.

```
Sub acbwc
' v2.0.1
' 5 sept 2003
```

```
' does footnotes и selections of all sizes
' still slow with large selections, but I blame Hamburg :-)
' v 2.0.1 slightly faster with improved cursorcount routine
' not unendurably slow with lots of footnotes, using cursors.

' acb, June 2003
' rewritten version of the
' dvwc macro by me и Daniel Vogelheim

' september 2003 changed the selection count to use a word cursor for large
selections
' following hints from Andrew Pitonyak.
' this is not perfect, either, largely because move-by-word is erratic.
' it will slightly exaggerate the number of words in a selection, counting
extra
' for paragraph ends и some punctuation.
' but it is still much quicker than the old method.

Dim xDoc, xSel, nSelcount
Dim nAllChars
Dim nAllWords
Dim nAllPars
Dim thisrange, sRes
Dim nSelChars, nSelwords, nSel
Dim atext, bigtext
Dim fnotes, thisnote, nfnotes, fnotecount
Dim oddthing,startcursor,stopcursor
  xDoc = thiscomponent
  xSel = xDoc.getCurrentSelection()
  nSelCount = xSel.getCount()
  bigText=xDoc.getText()

  ' согласно обычным требованиям (считаются сноски)...
  fnotes=xdoc.getFootNotes()
  If fnotes.hasElements() Then
    fnotecount=0
    For nfnotes=0 To fnotes.getCount()-1
    thisnote=fnotes.getbyIndex(nfnotes)
    startcursor=thisnote.getStart()
    stopcursor=thisnote.getEnd()
    Do While thisnote.getText().compareRegionStarts(startcursor,stopcursor)
and _
           startcursor.gotoNextWord(FALSE)
      fnotecount=fnotecount+1
    Loop
'         msgbox(startcursor.getString())
'         fnotecount=fnotecount+stringcount(thisnote.getstring())
'         fnotecount=fnotecount+CursorCount(thisnote,bigtext)
    Next nfnotes
  End If

  ' следующий оператор "If" обходит проблему, когда Вы выделили текст,
  ' затем сделали его пустым (нулевой длины) и снова подсчитываете,
  ' то это пустое выделение еще раз подсчитывается, что является ошибкой
  If nSelCount=1 and xSel.getByIndex(0).getString()="" Then
```

```
    nSelCount=0
  End If

  ' доступ к статистике (о документе)
  nAllChars = xDoc.CharacterCount
  nAllWords = xDoc.WordCount
  nAllPars = xDoc.ParagraphCount

  ' инициализируем счетчики
  nSelChars = 0
  nSelWords = 0
  ' Начало части макроса
  ' Перебираем элементы множественного выделения
  For nSel = 0 To nSelCount — 1
    thisrange=xSel.GetByIndex(nSel)
    atext=thisrange.getString()
    If len(atext)< 220 Then
      nselwords=nSelWords+stringcount(atext)
    Else
      nselwords=nSelWords+Cursorcount(thisrange)
    End If
    nSelChars=nSelChars+len(atext)
  Next nSel

  ' Код диалога переписан для удобочитаемости
    If fnotes.hasElements() Then
      sRes="В документе находится (считая сноски): "  + nAllWords + " слов.
"  + chr(13)
      sRes= sRes + "Количество слов без сносок: " + str(nAllWords-
fnotecount) +_
            " слов. "  + chr(13)+"(Всего: " +nAllChars +" символов в "
    Else
      sRes= "В документе находится: "  + nAllWords +"  слов. "  +
chr(13)+"(" + _
            nAllChars +" Символов в "
    End If
    sRes=sRes + nAllPars +" параграфах (абазцах).)"+ chr(13)+ chr(13)
    If nselCount>0 Then
      sRes=sRes  + "В выделенном тексте находится: " + nSelWords  + "  слов"
+ chr(13) +_
                  "(" + nSelChars  + " символов"
      If nSelcount=1 Then
        sRes=sRes  + " в " + str(nselCount) + " элементах выделения.)"
      Else
        REM Не знаю почему, но это уточнение необходимо
        sRes=sRes +  " В " + str(nselCount-1) +" элементах выделения.)"
      End If
      sRes=sRes+chr(13)+chr(13)+"В невыделенной части документа находится:"
+ chr(13)+_
        str(nAllWords-nSelWords) + " слов." +chr(13) +chr(13)
    End If
    If fnotes.hasElements() Then
      sRes=sRes+"Имеется "+ str(fnotecount) + " слов в "+ fnotes.getCount()
+_
               " сносках." +chr(13) +chr(13)
```

```
    End If
    msgbox(sRes,64,"acb Word Count")
End Sub

function Cursorcount(aRange)
' acb September 2003
' quick count for use with large selections
' based on Andrew Pitonyak's WordCountWordCursor() function
' but made cruder, in line with my general tendency.
Dim lnumwords as long
Dim atext
Dim startcursor, stopcursor as object
  atext=arange.getText()
  lnumwords=0
  If not atext.compareRegionStarts(aRange.getStart(),aRange.getEnd()) Then
    startcursor=atext.createTextCursorByRange(aRange.getStart())
    stopcursor=atext.createTextCursorByRange(aRange.getEnd())
  Else
    startcursor=atext.createTextCursorByRange(aRange.getEnd())
    stopcursor=atext.createTextCursorByRange(aRange.getStart())
  End If
  Do while aText.compareRegionEnds(startCursor, stopcursor) >= 0 and _
          startCursor.gotoNextWord(False)
     lnumwords=lnumwords+1
  Loop
  CursorCount=lnumwords-1
end function

Function stringcount(astring)
' acb June 2003
' slower, more accurate word count
' for use with smaller strings
' sharpened up by David Hammerton (http://crazney.net/) in September
2003
' to allow those who put two spaces after punctuation to escape their just
deserts
Dim nspaces,i,testchar,nextchar
  nspaces=0
  For i= 1 To len(astring)-1
    testchar=mid(astring,i,1)
    select Case testchar
    Case " ",chr(9),chr(13)
      nextchar = mid(astring,i+1,1)
      select Case nextchar
         Case " ",chr(9),chr(13),chr(10)
           nspaces=nspaces
         Case Else
           nspaces=nspaces+1
      end select
    end select
  Next i
  stringcount=nspaces+1
End Function
```

V.4. Замена выделенного пробела с использованием строк (Strings). Вставка специальных символов

Обычно нужно удалить лишние пробелы при чтении выделенного текста и затем записать новый текст обратно. Первая причина в том, что строки (strings) ограничены размером 64K, а вторая причина — так можно потерять информацию о форматировании. Я оставил эти примеры макросов, потому что они решали поставленную задачу до того, как я нашел способ сделать то же самое с использованием курсоров, и потому, что эти макросы демонстрируют технику вставки специальных символов. Они также показывают способы вставки управляющих символов (новых параграфов, символов конца строки и т.д.) в текст.

Лист. 217: Изменение текста, ненадежный способ.

```
Sub SelectedNewLinesToSpaces
  Dim lSelCount&, oSels As Object
  Dim iWhichSelection As Integer, lIndex As Long
  Dim s$, bSomethingChanged As Boolean

  oSels = ThisComponent.getCurrentSelection()
  lSelCount = oSels.getCount()
  For iWhichSelection = 0 To lSelCount - 1
    bSomethingChanged = False
    REM Что, если строка больше чем  64K? Увы! Oops
    s = oSels.getByIndex(iWhichSelection).getString()
    lIndex = 1
    Do While lIndex < Len(s)
      Select Case Asc(Mid(s, lIndex, 1))
      Case 13
        'Нашли маркер нового параграфа.
        'Следующим символом будет  10!
        If lIndex < Len(s) and Asc(Mid(s, lIndex+1, 1)) = 10 Then
          Mid(s, lIndex, 2, " ")
        Else
          Mid(s, lIndex, 1, " ")
        End If
        lIndex = lIndex + 1
        bSomethingChanged = True
      Case 10
        'Новая строка, был ли предыдущим символом конец параграфа 13
        'Удалите всю эту часть макроса case 10, если хотите игнорировать
случай с отдельным символом новой строки 10!
        If lIndex > 1 and Asc(Mid(s, lIndex-1, 1)) <> 13 Then
          'Это действительно новая строка без нового параграфа
          Mid(s, lIndex, 1, " ")
          lIndex = lIndex + 1
          bSomethingChanged = True
        Else
          'Да, это все-таки был новый параграф!
          lIndex = lIndex + 1
```

```
        End If
      Case Else
        'Ничего не делать, если не будем проверять что-нибудь еще
        lIndex = lIndex + 1
      End Select
    Loop
    If bSomethingChanged Then
      oSels.getByIndex(iWhichSelection).setString(s)
    End If
  Next
End Sub
```

Меня также просили преобразовать новые параграфы (paragraphs) в новые строки. Использование курсоров явно лучшая идея, но я не знаю, как сделать это. Я думаю, что следующий пример все ещё полезен, поэтому я оставил его. Сначала удаляется выделенный текст, а затем начинается вставка его обратно.

Лист. 218: Использование текстовых курсоров для замены пробелов..

```
Sub SelectedNewParagraphsToNewLines
  Dim lSelCount&, oSels As Object, oSelection As Object
  Dim iWhichSelection As Integer, lIndex As Long
  Dim oText As Object, oCursor As Object
  Dim s$, lLastCR As Long, lLastNL As Long

  oSels = ThisComponent.getCurrentSelection()
  lSelCount = oSels.getCount()
  oText=ThisComponent.Text

  For iWhichSelection = 0 To lSelCount - 1
    oSelection = oSels.getByIndex(iWhichSelection)
    oCursor=oText.createTextCursorByRange(oSelection)
    s = oSelection.getString()
    'Удаляем выделенный текст!
    oCursor.setString("")
    lIndex = 1
    Do While lIndex <= Len(s)
      Select Case Asc(Mid(s, lIndex, 1)
      Case 13
        oText.insertControlCharacter(oCursor, _
    com.sun.star.text.ControlCharacter.LINE_BREAK, False)
        'Хотел бы я иметь более короткий цикл!
        'Пропускаем следующий символ LF, если он есть.
        'Думаю, он будет всегда, но я это не проверял
        If (lIndex < Len(s)) Then
          If Asc(Mid(s, lIndex+1, 1)) = 10 Then lIndex = lIndex + 1
        End If
      Case 10
        oText.insertControlCharacter(oCursor, _
          com.sun.star.text.ControlCharacter.LINE_BREAK, False)
      Case Else
        oCursor.setString(Mid(s, lIndex, 1))
        oCursor.GoRight(1, False)
```

```
      End Select
      lIndex = lIndex + 1
    Loop
  Next
End Sub
```

V.4.1. Сравнение макросов, основанных на курсорах, и макросов, основанных на извлечении строк (String)

Вот некоторые макросы, которые я написал с использованием методов курсоров и вытекающих из этого способов, я написал их до того, как создал мои фреймы (framework)!

Лист. 219: Я написал это очень давно!

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
'Целью этого макроса является использование метода Text<-->Table
'для удаления ведущих и замыкающих пробелов.
'Макрос также вставляет символы нового параграфа взамен символов новой
строки!
Sub CRToNLMain
  Dim oCurs(), i%, sPrompt$
  sPrompt$ = "Преобразовать символы нового параграфа в символы новой строки
для ВСЕГО документа?"
  If Not CreateSelectedTextIterator(ThisComponent, sPrompt$, oCurs()) Then
Exit Sub
  For i% = LBOUND(oCurs()) To UBOUND(oCurs())
    CRToNLWorker(oCurs(i%, 0), oCurs(i%, 1), ThisComponent.Text)
  Next i%
End Sub
Sub CRToNLWorker(oLCurs As Object, oRCurs As Object, oText As Object)
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Sub
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub
  oLCurs.goRight(0, False)
  Do While oLCurs.gotoNextParagraph(False) and
oText.compareRegionEnds(oLCurs, oRCurs) >= 0
    oLCurs.goLeft(1, True)
    oLCurs.setString("")
    oLCurs.goRight(0, False)
    oText.insertControlCharacter(oLCurs,_
      com.sun.star.text.ControlCharacter.LINE_BREAK, True)
  Loop
End Sub

'Настоящей целью этого макроса является облегчить использование метода
'Text<-->Table, который требует удалить символы пробелов.
'Макрос также вставляет символы нового параграфа взамен символов новой
строки!
Sub SpaceToTabsInWordsMain
  Dim oCurs(), i%, sPrompt$
  sPrompt$ = "Преобразовать пробелы в символы табуляции для ВСЕГО
документа?"
  If Not CreateSelectedTextIterator(ThisComponent, sPrompt$, oCurs()) Then
Exit Sub
```

```
  For i% = LBOUND(oCurs()) To UBOUND(oCurs())
    SpaceToTabsInWordsWorker(oCurs(i%, 0), oCurs(i%, 1), ThisComponent.Text)
  Next i%
End Sub
Sub SpaceToTabsInWordsWorker(oLCurs As Object, oRCurs As Object, oText As
Object)
  Dim iCurrentState As Integer, iChar As Integer, bChanged As Boolean
  Const StartLineState = 0
  Const InWordState = 1
  Const BetweenWordState = 2

  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Sub
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub
  oLCurs.goRight(0, False)
  iCurrentState = StartLineState
  bChanged = False
  Do While oLCurs.goRight(1, True) and oText.compareRegionEnds(oLCurs,
oRCurs) >= 0
    iChar = Asc(oLCurs.getString())
    If iCurrentState = StartLineState Then
      If IsWhiteSpace(iChar) Then
        oLCurs.setString("")
      Else
        iCurrentState = InWordState
      End If
    ElseIf iCurrentState = InWordState Then
      bChanged = True
      Select Case iChar
      Case 9
        REM Уже символ табуляции, игнорируем
        iCurrentState = BetweenWordState
      Case 32, 160
        REM Преобразуем пробел в символ табуляции
        oLCurs.setString(Chr(9))
        oLCurs.goRight(1, False)
        iCurrentState = BetweenWordState
      Case 10
        REM Удаляем символ новой строки и вставляем символ нового параграфа
        oText.insertControlCharacter(oLCurs,_
          com.sun.star.text.ControlCharacter.PARAGRAPH_BREAK, True)
        oLCurs.goRight(1, False)
        iCurrentState = StartLineState
      Case 13
        iCurrentState = StartLineState
      Case Else
        oLCurs.gotoEndOfWord(True)
      End Select
    ElseIf iCurrentState = BetweenWordState Then
      Select Case iChar
      Case 9, 32, 160
        REM Уже добавляли символ табуляции, еще добавлять будет уже лишним!
        oLCurs.setString("")
      Case 10
        REM Удаляем символ новой строки и вставляем символ параграфа
        REM и проверяем, что удалили ведущий символ табуляции,
```

```
        REM который мы уже добавили левее!
        oText.insertControlCharacter(oLCurs,_
          com.sun.star.text.ControlCharacter.PARAGRAPH_BREAK, True)
        oLCurs.goLeft(0, False)
        REM и удаляем предыдущий символ табуляции
        oLCurs.goLeft(1, True)
        oLCurs.setString("")
        oLCurs.goRight(1, False)
        iCurrentState = StartLineState
      Case 13
        REM Сначала возвращаемся к символу  CR, затем выделяем символ
табуляции и удаляем его
        oLCurs.goLeft(0, False)
        oLCurs.goLeft(1, False)
        oLCurs.goLeft(1, True)
        oLCurs.setString("")
        REM Наконец, возвращаемся к символу  CR, который будем игнорировать
        oLCurs.goRight(1, True)
        iCurrentState = StartLineState
      Case Else
        iCurrentState = InWordState
        oLCurs.gotoEndOfWord(False)
      End Select
    End If
    oLCurs.goRight(0, False)
  Loop
  If bChanged Then
    REM Попасть сюда можно, если продвинулись вправо на один лишний символ
    oLCurs.goLeft(1, False)
    oLCurs.goLeft(1, True)
    If Asc(oLCurs.getString()) = 9 Then oLCurs.setString("")
  End If
End Sub

Sub TabsToSpacesMain
  Dim oCurs(), i%, sPrompt$
  sPrompt$ = "Преобразовать символы табуляции в пробелы для ВСЕГО
документа?"
  If Not CreateSelectedTextIterator(ThisComponent, sPrompt$, oCurs()) Then
Exit Sub
  For i% = LBOUND(oCurs()) To UBOUND(oCurs())
    TabsToSpacesWorker(oCurs(i%, 0), oCurs(i%, 1), ThisComponent.Text)
  Next i%
End Sub
Sub TabsToSpacesWorker(oLCurs As Object, oRCurs As Object, oText As Object)
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oText) Then Exit Sub
  If oText.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub
  oLCurs.goRight(0, False)
  Do While oLCurs.goRight(1, True) and oText.compareRegionEnds(oLCurs,
oRCurs) >= 0
    If Asc(oLCurs.getString()) = 9 Then
      oLCurs.setString("    ")  'Change a tab into 4 spaces
    End If
    oLCurs.goRight(0, False)
  Loop
```

```
End Sub

'sPrompt : текст вопроса, если приходится обрабатывать весь текст
'oCurs() : массив возвращаемых курсоров
'Возвращает true, если нужно просматривать, и false, если не нужно
Function CreateSelectedTextIterator(oDoc As Object, sPrompt As String,
oCurs()) As Boolean
  Dim oSels As Object, oSel As Object, oText As Object
  Dim lSelCount As Long, lWhichSelection As Long
  Dim oLCurs As Object, oRCurs As Object

  CreateSelectedTextIterator = True
  oText = oDoc.Text
  If Not IsAnythingSelected(ThisComponent) Then
    Dim i%
    i% = MsgBox("Текст не выбран!" + Chr(13) + sPrompt, _
      1 OR 32 OR 256, "Warning")
    If i% = 1 Then
      oLCurs = oText.createTextCursor()
      oLCurs.gotoStart(False)
      oRCurs = oText.createTextCursor()
      oRCurs.gotoEnd(False)
      oCurs = DimArray(0, 1)
      oCurs(0, 0) = oLCurs
      oCurs(0, 1) = oRCurs
    Else
      oCurs = DimArray()
      CreateSelectedTextIterator = False
    End If
  Else
    oSels = ThisComponent.getCurrentSelection()
    lSelCount = oSels.getCount()
    oCurs = DimArray(lSelCount - 1, 1)
    For lWhichSelection = 0 To lSelCount - 1
      oSel = oSels.getByIndex(lWhichSelection)
      'Чтобы проверить, что никакой текст не выделен, можно сделать
      'следующее:
      'oLCurs = oText.CreateTextCursorByRange(oSel)
      'If oLCurs.isCollapsed() Then ...
      oLCurs = GetLeftMostCursor(oSel, oText)
      oRCurs = GetRightMostCursor(oSel, oText)
      oCurs(lWhichSelection, 0) = oLCurs
      oCurs(lWhichSelection, 1) = oRCurs
    Next
  End If
End Function

'oDoc - объект Ooo Writer
Function IsAnythingSelected(oDoc As Object) As Boolean
  Dim oSels As Object, oSel As Object, oText As Object, oCursor As Object
  IsAnythingSelected = False
  If IsNull(oDoc) Then Exit Function
  ' Текущее выделение в текущем контроллере.
  'Если текущего контроллера нет, возвращается NULL.
  oSels = oDoc.getCurrentSelection()
```

```
  If IsNull(oSels) Then Exit Function
  If oSels.getCount() = 0 Then Exit Function
  If oSels.getCount() > 1 Then
    IsAnythingSelected = True
  Else
    oSel = oSels.getByIndex(0)
    oCursor = oDoc.Text.CreateTextCursorByRange(oSel)
    If Not oCursor.IsCollapsed() Then IsAnythingSelected = True
  End If
End Function

'oSelection = выделенный текст или интервал курсора
'oText = текстовый объект
Function GetLeftMostCursor(oSel As Object, oText As Object) As Object
  Dim oRange As Object, oCursor As Object

  If oText.compareRegionStarts(oSel.getEnd(), oSel) >= 0 Then
    oRange = oSel.getEnd()
  Else
    oRange = oSel.getStart()
  End If
  oCursor = oText.CreateTextCursorByRange(oRange)
  oCursor.goRight(0, False)
  GetLeftMostCursor = oCursor
End Function

'oSelection = выделенный текст или интервал курсора
'oText = текстовый объект
Function GetRightMostCursor(oSel As Object, oText As Object) As Object
  Dim oRange As Object, oCursor As Object

  If oText.compareRegionStarts(oSel.getEnd(), oSel) >= 0 Then
    oRange = oSel.getStart()
  Else
    oRange = oSel.getEnd()
  End If
  oCursor = oText.CreateTextCursorByRange(oRange)
  oCursor.goLeft(0, False)
  GetRightMostCursor = oCursor
End Function

'oSelection = выделенный текст или интервал курсора
'oText = текстовый объект
Function IsWhiteSpace(iChar As Integer) As Boolean
  Select Case iChar
  Case 9, 10, 13, 32, 160
    IsWhiteSpace = True
  Case Else
    IsWhiteSpace = False
  End Select
End Function

'Здесь начинается СТАРЫЙ макрос!
Sub ConvertSelectedNewParagraphToNewLine
  Dim lSelCount&, oSels As Object, oSelection As Object
```

```
  Dim iWhichSelection As Integer, lIndex As Long
  Dim oText As Object, oCursor As Object
  Dim s$, lLastCR As Long, lLastNL As Long

  'Здесь возможно множественное выделение!
  oSels = ThisComponent.getCurrentSelection()
  lSelCount = oSels.getCount()
  oText=ThisComponent.Text

  For iWhichSelection = 0 To lSelCount — 1
    oSelection = oSels.getByIndex(iWhichSelection)
    oCursor=oText.createTextCursorByRange(oSelection)
    s = oSelection.getString()
    oCursor.setString("")
    lLastCR = -1
    lLastNL = -1
    lIndex = 1
    Do While lIndex <= Len(s)
      Select Case Asc(Mid(s, lIndex, 1)
      Case 13
        oText.insertControlCharacter(oCursor,_
          com.sun.star.text.ControlCharacter.LINE_BREAK, False)
        'Хотел бы я иметь короткий цикл для этого!
        'Пропускаем следующий символ LF, если он есть.
        'Я думаю, он есть всегда, но я это не проверял
        If (lIndex < Len(s)) Then
          If Asc(Mid(s, lIndex+1, 1)) = 10 Then lIndex = lIndex + 1
        End If
      Case 10
        oText.insertControlCharacter(oCursor,_
          com.sun.star.text.ControlCharacter.LINE_BREAK, False)
      Case Else
        oCursor.setString(Mid(s, lIndex, 1))
        oCursor.GoRight(1, False)
      End Select
      lIndex = lIndex + 1
    Loop
  Next
End Sub

'Я решил написать это как машину с конечными состояниями.
'Машины с конечными состояниями - прекрасная вещь :-)
Sub ConvertSelectedSpaceToTabsBetweenWords
  Dim lSelCount&, oSels As Object, oSelection As Object
  Dim iWhichSelection As Integer, lIndex As Long
  Dim oText As Object, oCursor As Object
  Dim s$, lLastCR As Long, lLastNL As Long
  REM Какие состояния поддерживаются
  Dim iCurrentState As Integer
  Const StartLineState = 0
  Const InWordState = 1
  Const BetweenWordState = 2
  REM Точки перехода
  Dim iWhatFound As Integer
  Const FoundWhiteSpace = 0
```

```
Const FoundNewLine = 1
Const FoundOther = 2
Const ActionIgnoreChr = 0
Const ActionDeleteChr = 1
Const ActionInsertTab = 2
REM Определяем переходы между состояниями
Dim iNextState(0 To 2, 0 To 2, 0 To 1) As Integer
iNextState(StartLineState, FoundWhiteSpace, 0)  = StartLineState
iNextState(StartLineState, FoundNewLine, 0)     = StartLineState
iNextState(StartLineState, FoundOther, 0)       = InWordState

iNextState(InWordState, FoundWhiteSpace, 0)    = BetweenWordState
iNextState(InWordState, FoundNewLine, 0)       = StartLineState
iNextState(InWordState, FoundOther, 0)         = InWordState

iNextState(BetweenWordState, FoundWhiteSpace, 0)= BetweenWordState
iNextState(BetweenWordState, FoundNewLine, 0)   = StartLineState
iNextState(BetweenWordState, FoundOther, 0)     = InWordState

REM Определяем действия для состояний
iNextState(StartLineState, FoundWhiteSpace, 1)  = ActionDeleteChr
iNextState(StartLineState, FoundNewLine, 1)     = ActionIgnoreChr
iNextState(StartLineState, FoundOther, 1)       = ActionIgnoreChr

iNextState(InWordState, FoundWhiteSpace, 1)    = ActionDeleteChr
iNextState(InWordState, FoundNewLine, 1)       = ActionIgnoreChr
iNextState(InWordState, FoundOther, 1)         = ActionIgnoreChr

iNextState(BetweenWordState, FoundWhiteSpace, 1) = ActionDeleteChr
iNextState(BetweenWordState, FoundNewLine, 1)    = ActionIgnoreChr
iNextState(BetweenWordState, FoundOther, 1)      = ActionInsertTab

'Здесь может быть множественное выделение!
oSels = ThisComponent.getCurrentSelection()
lSelCount = oSels.getCount()
oText=ThisComponent.Text

For iWhichSelection = 0 To lSelCount – 1
  oSelection = oSels.getByIndex(iWhichSelection)
  oCursor=oText.createTextCursorByRange(oSelection)
  s = oSelection.getString()
  oCursor.setString("")
  lLastCR = -1
  lLastNL = -1
  lIndex = 1
  iCurrentState = StartLineState
  Do While lIndex <= Len(s)
    Select Case Asc(Mid(s, lIndex, 1))
    Case 9, 32, 160
      iWhatFound = FoundWhiteSpace
    Case 10
      iWhatFound = FoundNewLine
      lLastNL = lIndex
    Case 13
      iWhatFound = FoundNewLine
```

```
        lLastCR = lIndex
      Case Else
        iWhatFound = FoundOther
      End Select
      Select Case iNextState(iCurrentState, iWhatFound, 1)
      Case ActionDeleteChr
        'Путем выбора "не вставлять" символ удаляется!
      Case ActionIgnoreChr
        'На самом деле это означает вставить символ обратно!
        If lLastCR = lIndex Then
          'Вставка управляющего символа похожа на передвижение курсора
          'вокруг
          oText.insertControlCharacter(oCursor,_
            com.sun.star.text.ControlCharacter.PARAGRAPH_BREAK, False)
'          oText.insertControlCharacter(oCursor,_
'            com.sun.star.text.ControlCharacter.APPEND_PARAGRAPH, False)
          'oCursor.goRight(1, False)
          'Print "Вставлен символ возврата каретки CR"
        ElseIf lLastNL = lIndex Then
          If lLastCR + 1 <> lIndex Then
            oText.insertControlCharacter(oCursor,_
              com.sun.star.text.ControlCharacter.PARAGRAPH_BREAK, False)
              'com.sun.star.text.ControlCharacter.LINE_BREAK, False)
            'oCursor.goRight(1, False)
            'Print "Вставлен символ новой строки NL"
          End If
          'Игнорируем
        Else
          oCursor.setString(Mid(s, lIndex, 1))
          oCursor.GoRight(1, False)
          'Print "Что-то вставлено"
        End If
      Case ActionInsertTab
        oCursor.setString(Chr$(9) + Mid(s, lIndex, 1))
        oCursor.GoRight(2, False)
        'Print "Вставлен символ табуляции tab"
      End Select
      lIndex = lIndex + 1
      'MsgBox "index = " + lIndex + Chr(13) + s
      iCurrentState = iNextState(iCurrentState, iWhatFound, 0)
    Loop
  Next
End Sub

Sub ConvertAllTabsToSpace
  DIM oCursor As Object, oText As Object

  Dim nSpace%, nTab%, nPar%, nRet%, nTot%
  Dim justStarting As Boolean

  oText=ThisComponent.Text          'Получаем текстовый объект документа
  oCursor=oText.createTextCursor()  'Создаем курсор в тексте
  oCursor.gotoStart(FALSE)          'Переходим к началу, но НЕ выделяем
текст при этом
```

```
  Do While oCursor.GoRight(1, True) 'Передвигаемся на один символ вправо и
выделяем его
    If Asc(oCursor.getString()) = 9 Then
      oCursor.setString("    ")     'Заменяем символ табуляции на 4 пробела
    End If
    oCursor.goRight(0,FALSE)        'Делаем текст не выделенным!
  Loop
End Sub

Sub ConvertSelectedTabsToSpaces
  Dim lSelCount&, oSels As Object
  Dim iWhichSelection As Integer, lIndex As Long
  Dim s$, bSomethingChanged As Boolean

  'Здесь может быть множественное выделение!
  'Вероятно, будет на один элемент выделения больше, так как положение
  'текущего курсора будет считаться одним элементом выделения
  'текста, как Вы знаете!
  oSels = ThisComponent.getCurrentSelection()
  lSelCount = oSels.getCount()
  'Print "Всего элементов выделения = " + lSelCount
  For iWhichSelection = 0 To lSelCount - 1
    bSomethingChanged = False
    s = oSels.getByIndex(iWhichSelection).getString()
    'Print "Элемент выделения " + iWhichSelection + " имеет длину " + Len(s)
    lIndex = 1
    Do While lIndex < Len(s)
      'Print "Код ASCII символа в позиции " + lIndex + " = " + Asc(Mid(s,
lIndex, 1))
      If Asc(Mid(s, lIndex, 1)) = 9 Then
        s = ReplaceInString(s, lIndex, 1, "    ")
        bSomethingChanged = True
        lIndex = lIndex + 3
      End If
      lIndex = lIndex + 1
      'Print ":" + lIndex + "(" + s + ")"
    Loop
    If bSomethingChanged Then
      oSels.getByIndex(iWhichSelection).setString(s)
    End If
  Next
End Sub

Function ReplaceInString(s$, index&, num&, replaces$) As String
  If index <= 1 Then
    'Помещаем это в начало строки
    If num < 1 Then
      ReplaceInString = replaces + s
    ElseIf num > Len(s) Then
      ReplaceInString = replaces
    Else
      ReplaceInString = replaces + Right(s, Len(s) - num)
    End If
  ElseIf index + num > Len(s) Then
    ReplaceInString = Left(s,index - 1) + replaces
```

```
  Else
    ReplaceInString  = Left(s,index – 1) + replaces + Right(s, Len(s) –
index – num + 1)
  End If
End Function
```

V.5. Установка атрибутов текста

Когда макрос выполняется, он воздействует на параграф (paragraph), где находится курсор. Шрифт и размер устанавливаются. Атрибут CharPosture управляет курсивом, CharWeight управляет жирным шрифтом и CharUnderline управляет подчёркиванием. Допустимые значения найдены здесь:

http://api.openoffice.org/docs/common/ref/com/sun/star/style/CharacterProperties.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/FontWeight.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/FontSlant.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/FontUnderline.html

Лист. 220: Показать, как устанавливать атрибуты текста.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub SetTextAttributes
  Dim document As Object
  Dim Cursor
  Dim oText As Object
  Dim mySelection As Object
  Dim Font As String

  document=ThisComponent
  oText = document.Text
  Cursor = document.currentcontroller.getViewCursor()
  mySelection = oText.createTextCursorByRange(Cursor.getStart())
  mySelection.gotoStartOfParagraph(false)
  mySelection.gotoEndOfParagraph(true)
  mySelection.CharFontName="Courier New"
  mySelection.CharHeight="10"
  'Пора установить курсив (Italic) или не курсив с помощью
  'NONE, OBLIQUE, ITALIC, DONTKNOW, REVERSE_OBLIQUE, REVERSE_ITALIC
  mySelection.CharPosture = com.sun.star.awt.FontSlant.ITALIC
  'Хотите ли Вы жирный шрифт (BOLD)?
  'DONTKNOW, THIN, ULTRALIGHT, LIGHT, SEMILIGHT,
  'NORMAL, SEMIBOLD, BOLD, ULTRABOLD, BLACK
  'Реально существую только константы  THIN = 50, NORMAL = 100
  ' BOLD = 150, и BLACK = 200.
  mySelection.CharWeight = com.sun.star.awt.FontWeight.BOLD
  'Для подчеркивания возможны варианты
  'NONE, SINGLE, DOUBLE, DOTTED, DONTKNOW, DASH, LONGDASH,
  'DASHDOT, DASHDOTDOT, SMALLWAVE, WAVE, DOUBLEWAVE, BOLD,
  'BOLDDOTTED, BOLDDASH, BOLDLONGDASH, BOLDDASHDOT,
  'BOLDDASHDOTDOT, BOLDWAVE
  mySelection.CharUnderline = com.sun.star.awt.FontUnderline.SINGLE
  'Я недостаточно экспериментировал с этим, но я
  'знаю, что это, похоже, работает для немецкой локали German.
```

```
  Dim aLanguage As New com.sun.star.lang.Locale
  aLanguage.Country = "de"
  aLanguage.Language = "de"
  mySelection.CharLocale = aLanguage
End Sub
```

V.6. Вставить текст

Лист. 221: Демонстрирует, как вставлять строки (strings) в текстовый объект.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub InsertSimpleText
  Dim oDoc As Object
  Dim oText As Object
  Dim oVCurs As Object
  Dim oTCurs As Object

  oDoc = ThisComponent
  oText = oDoc.Text
  oVCurs = oDoc.CurrentController.getViewCursor()
  oTCurs = oText.createTextCursorByRange(oVCurs.getStart())
  ' укажите здесь текст, который нужно вставить
  oText.insertString(oTCurs, "—", FALSE)
End Sub
```

V.6.1. Вставить новый параграф (paragraph)

Используйте insertString() для вставки обычного текста. Используйте insertControlcharacter() для вставки специальных симовлов, таких как «новый параграф» (new paragraph). Для объекта «интервал текста» этот пример использует текстовый курсор, определяет, куда текст вставляется. Итоговое булевское значение показывает, нужно ли расширять текстовый курсор, чтобы он включал в себя только что вставленный текст. Я использую значение False, чтобы не расширять курсор.

Лист. 222: Вставить символ конца страницы (line break).

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub InsertTextWithABreak
  Dim oDoc
  Dim oText
  Dim oVCurs
  Dim oCursor

  oText = ThisComponent.getText()
  oVCurs = ThisComponent.CurrentController.getViewCursor()
  oCursor = oText.createTextCursorByRange(oVCurs.getStart())
  ' укажите здесь текст, который нужно вставить
  oText.insertString(oCursor, _
     "— I am new text before the break -", FALSE)
  oText.insertControlCharacter(oCursor, _
     com.sun.star.text.ControlCharacter.LINE_BREAK, False)
```

```
  oText.insertString(oCursor, "— I am new text after the break -", FALSE)
End Sub
```

V.7. Поля

V.7.1. Вставить форматированное поле даты в документ OOo Writer

Следующий макрос вставляет текст «Today is <date> », где «date» — форматирована так: «DD. MMM YYYY». Это создаёт формат даты, если он не существовал. Подробнее о допустимых форматах см. справку OOo по теме «форматы дат;коды числовых форматов» (number formats; formats).

Лист. 223: Insert Вставить форматированное поле даты в документ OOo Writer.

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
'uses:  FindCreateNumberFormatStyle
Sub InsertDateField
  Dim oDoc
  Dim oText
  Dim oVCurs
  Dim oTCurs
  Dim oDateTime
  Dim s$

  oDoc = ThisComponent
  If oDoc.SupportsService("com.sun.star.text.TextDocument") Then
    oText = oDoc.Text
    oVCurs = oDoc.CurrentController.getViewCursor()
    oTCurs = oText.createTextCursorByRange(oVCurs.getStart())
    oText.insertString(oTCurs, "Today is ", FALSE)
    ' Создаем тип DateTime
    s = "com.sun.star.text.TextField.DateTime"
    ODateTime = oDoc.createInstance(s)
    oDateTime.IsFixed = TRUE
    oDateTime.NumberFormat = FindCreateNumberFormatStyle(_
      "DD. MMMM YYYY", oDoc)

    oText.insertTextContent(oTCurs,oDateTime,FALSE)
    oText.insertString(oTCurs," ",FALSE)  Else
    MsgBox "К сожалению, этот макрос работает только с текстовым документом"
  End If
End Sub
```

V.7.2. Вставка примечания (аннотация)

Лист. 224: Добавить примечание в месте нахождения курсора..

```
Sub AddNoteAtCursor
  Dim vDoc, vViewCursor, oCurs, vTextField
  Dim s$

  'Lets lie и say that this was added ten days ago!
  Dim aDate As New com.sun.star.util.Date
```

```
  With aDate
    .Day   = Day(Now - 10)
    .Month = Month(Now - 10)
    .Year  = Year(Now - 10)
  End With

  vDoc = ThisComponent
  vViewCursor = vDoc.getCurrentController().getViewCursor()
  oCurs=vDoc.getText().createTextCursorByRange(vViewCursor.getStart())
  s = "com.sun.star.text.TextField.Annotation"
  vTextField = vDoc.createInstance(s)
  With vTextField
    .Author  = "AP"
    .Content = "It sure is fun to insert notes into my document"
    'заменяем дату и ее значения по умолчанию на сегодняшнюю дату!
    .Date    = aDate
  End With
  vDoc.Text.insertTextContent(oCurs, vTextField, False)
End Sub
```

V.8. Вставка новой страницы

При попытке вставить новую страницу в документ я остановился на следующей ссылке:

`http://api.openoffice.org/docs/common/ref/com/sun/star/style/ParagraphProperties.html`

где обсуждаются два свойства. Свойство *PageNumberOffset* описывается: «если свойство разрыва страницы установлено на параграф (paragraph), то это свойство содержит новое значение номера страницы.» Свойство *PageDescName* описывается: «Если это свойство установлено, то оно создаёт разрыв страницы перед параграфом, к которому это свойство относится, и присваивает свое значение в качестве имени стиля новой страницы.» Я понял, что если я установлю свойство PageDescName, то я смогу создать новую страницу и установить номер страницы. Однако осталось не сказанным, что значение свойства PageDescName станет именем стиля новой страницы после разрыва страницы. Если не указать существующий стиль страницы, то возникнет ошибка!

Лист. 225: Вставка разрыва страницы.

```
Sub ExampleNewPage
  Dim oSels As Object, oSel As Object, oText As Object
  Dim lSelCount As Long, lWhichSelection As Long
  Dim oLCurs As Object, oRCurs As Object

  oText = ThisComponent.Text
  oSels = ThisComponent.getCurrentSelection()
  lSelCount = oSels.getCount()
  For lWhichSelection = 0 To lSelCount - 1
    oSel = oSels.getByIndex(lWhichSelection)
```

```
    oLCurs = oText.CreateTextCursorByRange(oSel)
    oLCurs.gotoStartOfParagraph(false)
    oLCurs.gotoEndOfParagraph(true)
    REM  Оставляем предыдущий стиль страницы!
    oLCurs.PageDescName = oLCurs.PageStyleName
    oLCurs.PageNumberOffset = 7
  Next
End Sub
```

V.8.1. Удаление разрывов страниц

Я не во всем разобрался, но я выполнил несколько экспериментов. Установка свойства PageDescName вызывает разрыв страницы. При этом может также быть установлено свойство BreakType. Следующий макрос находит и удаляет разрывы страниц. Я выполнил только минимальное тестирование. Я не занимался установкой новых параметров страниц.

Лист. 226: Найти и удалить разрывы страниц.

```
Sub FindPageBreaks
  REM Author: Andrew Pitonyak
  Dim iCnt As Long
  Dim oCursor as Variant
  Dim oText As Variant
  Dim s As String

  oText = ThisComponent.Text
  oCursor = oText.CreateTextCursor()
  oCursor.GoToStart(False)
  Do
    If NOT oCursor.gotoEndOfParagraph(True) Then Exit Do
    iCnt = iCnt + 1
    If NOT IsEmpty(oCursor.PageDescName) Then
      s = s & "параграф " & iCnt & " имеет новый стиль страницы " & _
               oCursor.PageDescName & CHR$(10)
      oCursor.PageDescName = ""
    End If
    If oCursor.BreakType <> com.sun.star.style.BreakType.NONE Then
      s = s & "параграф " & iCnt & " содержит разрыв страницы" & CHR$(10)
      oCursor.BreakType = com.sun.star.style.BreakType.NONE
    End If
  Loop Until NOT oCursor.gotoNextParagraph(False)
  MsgBox s
End Sub
```

V.9. Установить стиль страницы в документе

Стиль страницы устанавливается при модификации имени Page Description. Это очень просто начать новую страницу.

Лист. 227: Установить стиль страницы для всего документа.

```
Sub SetDocumentPageStyle
  Dim oCursor As Object
  oCursor = ThisComponent.Text.createTextCursor()
  oCursor.gotoStart(False)
  oCursor.gotoEnd(True)
  Print "Current style = " & oCursor.PageStyleName
  oCursor.PageDescName = "Wow"
End Sub
```

V.10. Включение и выключение верхних и нижних заголовков

Каждый верхний и нижний заголовок ассоциирован со стилем страницы. Это означает, что включать и выключать верхние и нижние заголовки нужно в стиле страницы. Следующий макрос включает или выключает верхний заголовок страницы по месту нахождения курсора. Таким образом будет включен или выключен верхний заголовок для каждой страницы, использующей этот стиль страницы.

Лист. 228: Включить или выключить верхний заголовок страницы

```
Sub HeaderOnAtCursor(oDoc, bHeaderState As boolean)
  Dim oVC
  Dim sName$
  Dim oStyle

  REM  Получить стиль страницы, используемый по месту нахождения видимого
курсора
  sName = oDoc.getCurrentController().getViewCursor().PageStyleName
  oStyle = oDoc.StyleFamilies.getByName("PageStyles").getByName(sName)
  REM Use FooterIsOn to toggle the footer state.
  If oStyle.HeaderIsOn <> bHeaderState Then
    oStyle.HeaderIsOn = bHeaderState
  End If
End Sub
```

Стиль страницы получен из списка стилей страниц. Атрибут HeaderIsOn («верхний заголовок включен») переключается. Для переключения нижнего заголвка нужно использовать свойство FooterIsOn.

V.11. Вставить OLE-объект

Юмор в том, что для версии OpenOffice 1.1 следующий макрос вставит объект OLE в документ OOo Writer. Здесь CLSID может быть внешним OLE-объектом.

Лист. 229: Вставить объект OLE.

```
SName = "com.sun.star.text.TextEmbeddedObject"
obj   = ThisComponent.createInstance(sName)
```

```
obj.CLSID = "47BBB4CB-CE4C-4E80-A591-42D9AE74950F"
obj.attach(ThisComponent.currentController().Selection.getByIndex(0))
```

Если выбирать встроенный в Writer объект, то можно получить доступ к API так:

```
oModel = ThisComponent.currentController().Selection.Model
```

Это предоставляет тот же интерфейс доступа к объекту, как если бы объект был создан путем загрузки документа с помощью loadComponentFromURL

V.12. Установка стиля параграфа (Paragraph)

Многие стили могут быть заданы напрямую для выделенного текста, включая стиль параграфа.

Лист. 230: Установить стиль параграфа для выделенного параграфа равным «Heading 2».

```
Sub SetParagraphStyle
  Dim oSels As Object, oSel As Object, oText As Object
  Dim lSelCount As Long, lWhichSelection As Long
  Dim oLCurs As Object, oRCurs As Object

  oText = ThisComponent.Text
  oSels = ThisComponent.getCurrentSelection()
  lSelCount = oSels.getCount()
  For lWhichSelection = 0 To lSelCount – 1
    oSel = oSels.getByIndex(lWhichSelection)
    oSel.ParaStyleName = "Heading 2"
  Next
End Sub
```

Следующий пример устанавливает один и тот же стиль для всех параграфов.

Лист. 231: Установить одинаковый стиль для всех параграфов.

```
'Author: Marc Messeant
'email: marc.liste@free.fr
Sub AppliquerStyle()
  Dim oText, oVCurs, oTCurs
  oText = ThisComponent.Text
  oVCurs = ThisComponent.CurrentController.getViewCursor()
  oTCurs = oText.createTextCursorByRange(oVCurs.getStart())

  While oText.compareRegionStarts(oTCurs.getStart(),oVCurs.getEnd())=1
    oTCurs.paraStyleName = "YourStyle"
    oTCurs.gotoNextParagraph(false)
  Wend
End Sub
```

V.13. Создать свой собственный стиль

Я не проверял этот код макроса, но верю, что он работает.

Лист. 232: Create a paragraph style.

```
vFamilies = oDoc.StyleFamilies
vStyle = oDoc .createInstance("com.sun.star.style.ParagraphStyle")
vParaStyles = vFamilies.getByName("ParagraphStyles")
vParaStyles.insertByName("MyStyle",vStyle)
```

V.14. Поиск и замена

Компоненты, где возможен поиск, поддерживают возможность создать описание поиска. Такие компоненты могут также находить первый, следующий, а все экземпляры заданного текста. См. подробнее:
`http://api.openoffice.org/docs/common/ref/com/sun/star/util/XSearchable.html`

Простой пример показывает, как нужно выполнять поиск (Лист. 233).

Лист. 233: Выполнить простой поиск на основе слов..

```
Sub SimpleSearchExample
  Dim vDescriptor, vFound
  ' создаем новый дескриптор для документа, в котором ищем.
  vDescriptor = ThisComponent.createSearchDescriptor()
  ' Задаем текст, который ищем, и др.
  '
  With vDescriptor
    .SearchString = "hello"
    ' Следующие параметры по умолчанию были бы равны false
    .SearchWords = true
    .SearchCaseSensitive = False
  End With
  ' Находим первое
  vFound = ThisComponent.findFirst(vDescriptor)
  Do While Not IsNull(vFound)
    Print vFound.getString()
    vFound.CharWeight = com.sun.star.awt.FontWeight.BOLD
    vFound = ThisComponent.findNext( vFound.End, vDescriptor)
  Loop
End Sub
```

Объект, возвращаемый методами findFirst и findNext , аналогичен курсору, поэтому многие вещи, которые можно сделать с курсором, такие как установление атрибутов, можно делать и с этим объектом.

V.14.1. Замена текста

Замена текста очень похожа на поиск текста, за исключением того, что замена должна поддерживать:

http://api.openoffice.org/docs/common/ref/com/sun/star/util/XReplaceable.html

Единственный полезный метод, который замена предоставляет и не поддерживает поиск — то, что документ не дает возможности заменить все экземпляры текста для поиска на что-то другое. Идея состоит в том, что если искать по одному экземпляру за один раз, то можно вручную заменить найденный текст на текст для замены. Следующий макрос ищет в тексте и заменяет каждый экземпляр «a@» на символ Unicode с кодом 257.

Лист. 234: Заменить несколько символов

```
'Author:  Birgit Kellner
'email: birgit.kellner@univie.ac.at
Sub AtToUnicode
  'Andy says that sometime in the future these may have to be Variant types
to work with Array()
  Dim numbered(5) As String, accented(5) As String
  Dim n as long
  Dim oDoc as object, oReplace as object
  numbered() = Array("A@", "a@", "I@", "i@", "U@", "u@", "Z@", "z@",_
    "O@", "o@", "H@", "h@", "D@", "d@", "L@", "l@", "M@", "m@",_
    "G@", "g@", "N@", "n@", "R@", "r@",_
    "Y@", "y@", "S@", "s@", "T@", "t@", "C@", "c@", "j@", "J@")
  accented() = Array(Chr$(256), Chr$(257), Chr(298), Chr$(299),_
    Chr$(362),  Chr$(363),  Chr$(377),  Chr$(378),  Chr$(332),_
    Chr$(333),  Chr$(7716), Chr$(7717), Chr$(7692), Chr$(7693),_
    Chr$(7734), Chr$(7735), Chr$(7746), Chr$(7747), Chr$(7748),_
    Chr$(7749), Chr$(7750), Chr$(7751), Chr$(7770), Chr$(7771),_
    Chr$(7772), Chr$(7773), Chr$(7778), Chr$(7779), Chr$(7788),_
    Chr$(7789), Chr$(346),  Chr$(347),  Chr$(241),  Chr$(209))
  oReplace = ThisComponent.createReplaceDescriptor()
  oReplace.SearchCaseSensitive = True
  For n = LBound(numbered()) To UBound(accented())
    oReplace.SearchString = numbered(n)
    oReplace.ReplaceString = accented(n)
    ThisComponent.ReplaceAll(oReplace)
  Next n
End Sub
```

V.14.2. Поиск в выделенном тексте

Хитрость поиска только в выделенном тексте в том, чтобы понять, что курсор может использоваться в процедуре findNext. Можно затем проверить конец найденного текста, не зашел ли поиск за пределы выделенного текста. Это также позволит начать поиск с любого положения курсора. Метод findFirst не обязателен, если уже есть объект курсора, который описывает позицию, с которой начинается поиск процедурой findNext. Следующий макрос

использует рамки моего выделенного текста и содержит некоторые улучшения, предложенные Bernard Marcelly. См. также:
http://api.openoffice.org/docs/common/ref/com/sun/star/text/XTextRangeCompare.html

Лист. 235: Поиск в выделенном тексте

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub SearchSelectedText
  Dim oCurs(), i%
  If Not CreateSelectedTextIterator(ThisComponent, _
    "Search text in the entire document?", oCurs()) Then Exit Sub
  For i% = LBound(oCurs()) To UBound(oCurs())
    SearchSelectedWorker(oCurs(i%, 0), oCurs(i%, 1), ThisComponent)
  Next i%
End Sub

Sub SearchSelectedWorker(oLCurs, oRCurs, oDoc)
  If IsNull(oLCurs) Or IsNull(oRCurs) Or IsNull(oDoc) Then Exit Sub
  If oDoc.Text.compareRegionEnds(oLCurs, oRCurs) <= 0 Then Exit Sub
  oLCurs.goRight(0, False)
  Dim vDescriptor, vFound
  vDescriptor = oDoc.createSearchDescriptor()
  With vDescriptor
    .SearchString = "Paragraph"
    .SearchCaseSensitive = False
  End With
  ' Нет причин выполнять findFirst.
  vFound =  oDoc.findNext(oLCurs, vDescriptor)
  REM Would you kill for short-circuit evaluation?
  Do While Not IsNull(vFound)
    REM If Not vFound.hasElements() Then Exit Do
    'See if we searched past the end
    'Not really safe because this assumes that vFound и oRCurs
    'are in the same text object (warning).
    If -1 = oDoc.Text.compareRegionEnds(vFound, oRCurs) Then Exit Do
    Print vFound.getString()
    vFound = ThisComponent.findNext( vFound.End, vDescriptor)
  Loop
End Sub
```

V.14.3. Сложный поиск и замена

Лист. 236: Удалить (текст) между двумя разделителями с помощью поиска и замены.

```
'Deleting text between two delimiters is actually very easy
Sub deleteTextBetweenDlimiters
  Dim vOpenSearch, vCloseSearch  'Open и Close descriptors
  Dim vOpenFound, vCloseFound    'Open и Close find objects
  Dim oDoc

  oDoc = ThisComponent
  ' Создать дескрипторы для документа, в котором возможен поиск.
  vOpenSearch = oDoc.createSearchDescriptor()
```

```
  vCloseSearch = oDoc.createSearchDescriptor()

  ' Задать текст, который нужно искать, и др.
  vOpenSearch.SearchString = "["
  vCloseSearch.SearchString = "]"

  ' Найти первый открывающий разделитель
  vOpenFound = oDoc.findFirst(vOpenSearch)
  Do While Not IsNull(vOpenFound)

    'Поиск закрывающего разделителья, начиная с найденного открывающего
    vCloseFound = oDoc.findNext( vOpenFound.End, vCloseSearch)
    If IsNull(vCloseFound) Then
      Print "Найден открывающий разделитель без закрывающего!"
      Exit Do
    Else
      ' Очистить откыающий разделитель, если это ещё не сделано,
      ' затем завершить обработку текста внутри разделителей
      vOpenFound.setString("")
      ' Выделить текст внутри разделителей
      vOpenFound.gotoRange(vCloseFound, True)
      Print "Found " & vOpenFound.getString()
      ' очистить текст между разделителями
      vOpenFound.setString("")
      ' Очистить закрывающий разделитель
      vCloseFound.setString("")
      ' Действительно ли нужно удалить ВСЕ пробелы?
      ' Если да, то это делается здесь!
      If vCloseFound.goRight(1, True) Then
        If vCloseFound.getString() = " " Then
          vCloseFound.setString("")
        End If
      End If
      vOpenFound = oDoc.findNext( vOpenFound.End, vOpenSearch)
    End If
  Loop
End Sub
```

V.14.4. Поиск и замена с атрибутами и регулярными выражениями (Regular Expressions)

Этот макрос заключает все **ЖИРНЫЕ** элементы в фигурные скобки «{{ }}» и меняет атрибут **Bold** (жирный) на Normal. Регулярное выражение (regular expression) использовано при указании текста для поиска.

Лист. 237: Замена форматирования с использованием регулярного выражения.

```
Sub ReplaceFormatting
  'original code : Alex Savitsky
  'modified by : Laurent Godard
  'The purpose of this macro is to surround all
  'BOLD elements with {{ }}
  'and change the Bold attribute to NORMAL
  'This uses regular expressions
```

```
  'The styles have to be searched too

  Dim oDoc As Object
  Dim oReplace As Object
  Dim SrchAttributes(0) As New com.sun.star.beans.PropertyValue
  Dim ReplAttributes(0) As New com.sun.star.beans.PropertyValue

  oDoc = ThisComponent
  oReplace = oDoc.createReplaceDescriptor

  'Регулярное выражение — соответствует любому тексту
  oReplace.SearchString = ".*"
  'Заметьте: знак &  означает вставку обратно найденного текста
  oReplace.ReplaceString = "{{ & }}"
  oReplace.SearchRegularExpression=True  'Использовать регулярные выражения
  oReplace.searchStyles=True             'Искать по стилям
  oReplace.searchAll=True                'Обрабатывать весь документ

  REM  Это атрибут, который нужно найти
  SrchAttributes(0).Name = "CharWeight"
  SrchAttributes(0).Value =com.sun.star.awt.FontWeight.BOLD

  REM Это значение атрибута, которым нужно заменить атрибут найденного
текста
  ReplAttributes(0).Name = "CharWeight"
  ReplAttributes(0).Value =com.sun.star.awt.FontWeight.NORMAL

  REM Установить атрибуты в дескрипторе замены
  oReplace.SetSearchAttributes(SrchAttributes())
  oReplace.SetReplaceAttributes(ReplAttributes())

  REM Теперь выполнить!
  oDoc.replaceAll(oReplace)
End Sub
```

V.14.5. Искать только первую текстовую таблицу

Первым поиском может быть поиск с помощью findNext (не обязательно findFirst), но нужно указать позицию, с которой начинать поиск. Эта позиция старта может быть почти любым текстовым интервалом.

Лист. 238: Поиск в первой текстовой таблице.

```
Sub SearchInFirstTable
  Dim oDescriptor, oFound
  Dim oTable
  Dim oCell
  Dim oDoc

  oDoc = ThisComponent
  REM Получить ячейку A1 в первой текстовой таблице
  oTable = oDoc.getTextTables().getByIndex(0)
  oCell = oTable.getCellByName("A1")

  REM Создать дескриптор поиска
```

```
  oDescriptor = oDoc.createSearchDescriptor()
  With oDescriptor
    .SearchString = "one"
    .SearchWords = False
    .SearchCaseSensitive = False
  End With

  REM Начать поиск с начала текстового объекта в ячейке A1
  REM первой текстовой таблицы.
  REM oFound = ThisComponent.findFirst(oDescriptor)
  oFound = oDoc.findNext( oCell.getText().getStart(), oDescriptor)
  Do While Not IsNull(oFound)
    REM Если найденный текст не входит в текстовую таблицу, то закончить
    If IsNull(oFound.TextTable) Then
      Exit Sub
    End If
    REM  Если найденный текст не входит в ту же текстовую таблицу —
закончить.
    REM  Это не полная проверка, так как текстовая таблица может содержать
    REM  другую текстовую таблицу или другой объект, например, фрейм,
    REM  но этот макрос подходит для простых случаев.
    If NOT EqualUnoObjects(oTable, oFound.TextTable) Then
      Exit Sub
    End If
    oFound.CharWeight = com.sun.star.awt.FontWeight.BOLD
    oFound = oDoc.findNext( oFound.End, oDescriptor)
  Loop
End Sub
```

V.15. Изменение строчных букв на прописные и наоборот (Case) в словах

OOo определяет прописные/строчные буквы в слове на основе свойства объекта character. Теоретически это означает, что можно выделить весь документ и затем установить для него все букуы строчными или все прописными. На практике, тем не менее, выделенная часть документа может не иметь свойства «character case» (прописные/строчные). Как компромисс между скоростью и возможными проблемами, я выбрал использование курсора слов (word cursor) для перебора всего текста, устанавливая для каждого слова индивидуально свойство прописные/строчные (case). Я написал макрос для обработки целых слов как возможное решение. Если Вас это не устраивает, измените макрос. Я использовал мою технику работы с выделенным текстом, поэтому Вам потребуется свой макрос для нужной Вам обработки.

Если Вы обнаруживете, что текст не поддерживает установку прописных/строчных букв (character case), то можно избежать ошибок добавлением оператора «On Local Error Resume Next» перед SetWordCase().

Внимание *Установка прописных/строчных (case) не меняет сам символ, а только способ его вывода. Если установить атрибут строчных (маленьких) букв, то Вы не сможете вручную ввести прописную букву.*

Внимание *В версии OOo 1.0.3.1 есть ошибка для значения case, равного title; «heLLo» выводится как «HeLLo».*

См. также:

`http://api.openoffice.org/commmon/ref/com/sun/star/style/CaseMap.html`

Лист. 239: Изменение букв

```
'Author: Andrew Pitonyak; email: andrew@pitonyak.org
Sub ADPSetWordCase()
  Dim oCurs(), i%, sMapType$, iMapType%
  iMapType = -1
  Do While iMapType < 0
    sMapType = InputBox("Какой вариант прописных/строчных букв нужно
присвоить словам?" & chr(10) &_
      "None (восстановить исходный размер), UPPER (все большие), lower (все
малые), Title (первая большая и остальные малые) или small caps (маленькие
заглавные)?", "Изменить размер букв", "Title")
    sMapType = UCase(Trim(sMapType))
    If sMapType = "" Then sMapType = "EXIT"
    Select Case sMapType
    Case "EXIT"
      Exit Sub
    Case "NONE"
      iMapType = com.sun.star.style.CaseMap.NONE
    Case "UPPER"
      iMapType = com.sun.star.style.CaseMap.UPPERCASE
    Case "LOWER"
      iMapType = com.sun.star.style.CaseMap.LOWERCASE
    Case "TITLE"
      iMapType = com.sun.star.style.CaseMap.TITLE
    Case "SMALL CAPS"
      iMapType = com.sun.star.style.CaseMap.SMALLCAPS
    Case Else
      Print "Извините, " & sMapType & " не является правильным вариантом"
    End Select
  Loop
  If Not CreateSelectedTextIterator(ThisComponent, _
    "Изменить весь документ?", oCurs()) Then Exit Sub
  For i% = LBound(oCurs()) To UBound(oCurs())
```

```
    SetWordCase(oCurs(i%, 0), oCurs(i%, 1), ThisComponent.Text, iMapType%)
  Next
End Sub

Function SetWordCase(vLCursor, vRCursor, oText, iMapType%)
  If IsNull(vLCursor) OR IsNull(vRCursor) OR IsNull(oText) Then Exit
Function
  If oText.compareRegionEnds(vLCursor, vRCursor) <= 0 Then Exit Function
  vLCursor.goRight(0, False)
  Do While vLCursor.gotoNextWord(True)
    If oText.compareRegionStarts(vLCursor, vRCursor) > 0 Then
      vLCursor.charCasemap = iMapType%
      vLCursor.goRight(0, False)
    Else
      Exit Function
    End If
  Loop
  REM Если за ПОСЛЕДНИМ  словом документа нет символов пунктуации и конца
строки,
  REM  то не возможно перейти к следующему слову. Проверим этот случай!
  If oText.compareRegionStarts(vLCursor, vRCursor) > 0 and _
          vLCursor.gotoEndOfWord(True) Then
    vLCursor.charCasemap = iMapType%
  End If
End Function
```

V.16. Перебор параграфов (поведение текстового курсора)

Во внутреннем представлении абзац представлен как «что-то<cr><lf>». Использование GUI для ручного выделения параграфа выделяет текст без завершающих символов перевода строки (carriage return) и без возврата строки (line feed).

Лист. 240: Передвинуть курсор на ноль позиций для очистки выделения.

```
oVCurs = ThisComponent.getCurrentController().getViewCursor()
MsgBox "(" & oVCurs.getString() & ")"
oVCurs.goRight(0, False)
```

Передвижение курсора на ноль позиций направо очищает выделение и оставляет курсор непосредственно перед символами <cr><lf>; и поэтому не может использоваться для перехода к следующему параграфу. Тем не менее, передвигаться между параграфами можно.

Лист. 241: Выделить следующий параграф целиком.

```
  Dim oVCurs
  Dim oTCurs
  oVCurs = ThisComponent.getCurrentController().getViewCursor()
  oTCurs = oVCurs.getText().createTextCursorByRange(oVCurs)
  oTCurs.gotoNextParagraph(False) 'перейти к началу следующего параграфа.
  oTCurs.gotoEndOfParagraph(True) 'выделить параграф целиком.
  MsgBox "(" & oTCurs.getString() & ")"
```

> ***Совет*** *Метод gotoEndOfParagraph не включает в выделение разделяющие символы <cr><lf>*

Следующий макрос демонстрирует способ перебрать каждый параграф и напечатать имя стиля параграфа. Если выделено больше одного параграфа, то нельзя получить стиль параграфа. Что удобно, этот способ полностью пропускает вставленные текстовые таблицы.

Лист. 242: Напечатать все стили параграфов.

```
Sub PrintAllStyles
  Dim s As String
  Dim oCurs as Variant
  Dim sCurStyle As String

  oCurs = ThisComponent.Text.CreateTextCursor()
  oCurs.GoToStart(False)
  Do
    If NOT oCurs.gotoEndOfParagraph(True) Then Exit Do
    sCurStyle = oCurs.ParaStyleName
    s = s & """" & sCurStyle & """" & CHR$(10)
  Loop Until NOT oCurs.gotoNextParagraph(False)
  MsgBox s, 0, "Styles in Document"
End Sub
```

V.16.1. Форматирование параграфов с макросами (пример)

Я написал макрос, проверяющий документ и устанавливающий для каждой части с кодами макросов заданный стиль параграфа.

Таблица 8. Стили параграфа, использованные для форматирования кода макросов.

Описание	***Первоначальный стиль***	***Новый стиль***
Макрос из одной строки	_code_one_line	_OooComputerCodeLastLine
Первая строка	_code_first_line	_OooComputerCode
Последняя строка	_code_last_line	_OooComputerCodeLastLine
Строки посередине	_code	_OooComputerCode

Я хотел просмотреть весь документ, определить части с кодами макросов (основываясь на стиле параграфа) и присвоить им нужный нужный новый стиль параграфа.

Лист. 243: Форматировать параграфы с макросами.

```
Sub user_CleanUpCodeSections
  worker_CleanUpCodeSections("_code_first_line", "_code", _
                             "_code_last_line", "_code_one_line")
End Sub

Sub worker_CleanUpCodeSections(firstStyle$, midStyle$, _
                               lastStyle$, onlyStyle$)
  Dim vCurCurs as Variant     'Текущщий курсор
  Dim vPrevCurs as Variant    'Предыдущий курсор, на один авбзац раньше
  Dim sPrevStyle As String    'Предыдущий стиль
  Dim sCurStyle As String     'Текущий стиль

  REM  Позиция текущего курсора в начале второго параграфа
  vCurCurs  = ThisComponent.Text.CreateTextCursor()
  vCurCurs.GoToStart(False)
  If NOT vCurCurs.gotoNextParagraph(False) Then Exit Sub

  REM  Позиция предыдущего курсора для выделения первого параграфа
  vPrevCurs = ThisComponent.Text.CreateTextCursor()
  vPrevCurs.GoToStart(False)
  If NOT vPrevCurs.gotoEndOfParagraph(True) Then Exit Sub
  sPrevStyle = vPrevCurs.ParaStyleName

  Do
    If NOT vCurCurs.gotoEndOfParagraph(True) Then Exit Do
    sCurStyle = vCurCurs.ParaStyleName

    REM Здесь выполняется основная работа
    If sCurStyle = firstStyle$ Then
      REM  Текущий стиль равен первому стилю
      REM  Проверяем, равен ли предыдущий стиль одному из следующих!
      Select Case sPrevStyle
        Case onlyStyle$, lastStyle$
          sCurStyle = midStyle$
          vCurCurs.ParaStyleName  = sCurStyle
          vPrevCurs.ParaStyleName = firstStyle$

        Case firstStyle$, midStyle$
          sCurStyle = midStyle$
          vCurCurs.ParaStyleName = sCurStyle
      End Select

    ElseIf sCurStyle = midStyle$ Then
      REM  Текущий стиль равен стилю середины макроса
      REM  Проверка, равен ли предыдущий стиль одному из следующих!
      Select Case sPrevStyle
        Case firstStyle$, midStyle$
          REM do nothing!

        Case onlyStyle$
          REM   Стиль конца макроса был единственным стилем, но он
предшествует стилю середины макроса!
          vPrevCurs.ParaStyleName = firstStyle$
```

```
        Case lastStyle$
          vPrevCurs.ParaStyleName = midStyle$

        Case Else
          sCurStyle = firstStyle$
          vCurCurs.ParaStyleName = sCurStyle
      End Select

    ElseIf sCurStyle = lastStyle$ Then
      Select Case sPrevStyle
        Case firstStyle$, midStyle$
          REM  Ничего не делать!

        Case onlyStyle$
          REM  Стиль конца макроса был  единственным стилем, но он
предшествует стилю середины макроса!
          vPrevCurs.ParaStyleName = firstStyle$

        Case lastStyle$
          vPrevCurs.ParaStyleName = midStyle$

        Case Else
          sCurStyle = firstStyle$
          vCurCurs.ParaStyleName = sCurStyle
      End Select

    ElseIf sCurStyle = onlyStyle$ Then
      Select Case sPrevStyle
        Case firstStyle$, midStyle$
          sCurStyle = midStyle$
          vCurCurs.ParaStyleName = sCurStyle

        Case lastStyle$
          sCurStyle = lastStyle$
          vCurCurs.ParaStyleName = sCurStyle
          vPrevCurs.ParaStyleName = midStyle$

        Case onlyStyle$
          sCurStyle = lastStyle$
          vCurCurs.ParaStyleName = sCurStyle
          vPrevCurs.ParaStyleName = firstStyle$
      End Select

    Else
      Select Case sPrevStyle
        Case firstStyle$
          vPrevCurs.ParaStyleName = onlyStyle$

        Case midStyle$
          vPrevCurs.ParaStyleName = lastStyle$
      End Select
    End If

    REM  Работа выполнена, передвигаем курсор предыдущего параграфа
```

```
    vPrevCurs.gotoNextParagraph(False)
    vPrevCurs.gotoEndOfParagraph(True)
    sPrevStyle = vPrevCurs.ParaStyleName
  Loop Until NOT vCurCurs.gotoNextParagraph(False)
End Sub
```

Макрос Лист. 243 легко переделать для форматирования и макрос станет много меньше. У меня нехватило на это времени.

V.16.2. Находится ли курсор в последнем параграфе

Передвижение курсора может быть использовано для определения, находится ли курсор в последнем параграфе. Следующий макрос предполагает, что ответом будет «нет», если курсор не находится в первичном объекте документа.

Лист. 244: Находится ли курсор в последнем параграфе?

```
Function IsCursorInlastPar(oCursor, oDoc)
  Dim oTC

  IsCursorInlastPar = False
  If EqualUNOObjects(oCursor.getText(), oDoc.getText()) Then
    oTC = oCursor.getText().createTextCursorByRange(oCursor)
    IsCursorInlastPar = NOT oTC.gotoNextParagraph(false)
  End If
End Function
```

V.16.3. Что означает перебрать содержимое текста?

Перебор (enumerating) объектов означает, что Вы просмотрели каждый объект. Текстовый объект может перебрать его содержимое (content). Текстовый объект перебирает параграфы и текстовые таблицы. Другими словами, текстовый объект может по отдельности извлечь каждый параграф и каждую текстовую таблицы, которые в нем находятся (см. Лист. 245).

Лист. 245: Перебрать уровень параграфов содержимого текста.

```
Sub EnumerateParagraphs
  REM Author: Andrew Pitonyak
  Dim oParEnum     'Счетчик (Enumerator) используемый для перебора
параграфов
  Dim oPar         'Перебираемый параграф

  REM  Перебрать параграфы
  REM  Таблицы перебираются вместе с параграфами
  oParEnum = ThisComponent.getText().createEnumeration()
  Do While oParEnum.hasMoreElements()
    oPar = oParEnum.nextElement()

    REM  Это пропускает таблицы. Добавьте оператор else, чтобы
    REM  обработать таблицы
    If oPar.supportsService("com.sun.star.text.Paragraph") Then
      MsgBox oPar.getString(), 0, "Найден параграф"
```

```
    ElseIf oPar.supportsService("com.sun.star.text.TextTable") Then
      Print "Найдена текстовая таблица TextTable"
    Else
      Print "Что найдено?"
    End If
  Loop
End Sub
```

Каждый текстовый параграф может перебрать его содержимое. Перебор может быть использован для поиска полей, закладок (bookmarks) и всех видов другого текстового содержимого. Тем не менее, это может перебрать только содержимое, привязанное (anchored) к параграфу. Графика, которая привязывается к странице, не включается в такой перебор. «Считаем, что имеем простой текстовый документ, который содержит только *этот* параграф». («Consider a **simple** text document that contains only *this* paragraph.»)

Лист. 246: Перебрать уровень параграфов текстового содержимого.

```
Sub EnumerateContent
  REM Author: Andrew Pitonyak
  Dim oParEnum     'Счетчик, используемый для перебора параграфов
  Dim oPar         'Перебираемый параграф
  Dim oSectionEnum 'Счетчик, используемый для перебора текстовых разделов
(text sections)
  Dim oSection     'Перебираемый текстоваый раздел
  Dim s As String  'Содержит  нумерацию (enumeration)
  Dim i As Integer 'Считать параграфы

  REM  Перебрать параграфы
  REM  Таблицы перебираются одновременно с параграфами
  oParEnum = ThisComponent.getText().createEnumeration()
  Do While oParEnum.hasMoreElements()
    oPar = oParEnum.nextElement()

    REM  Это пропускает таблицы. Добавьте оператор else, чтобы
    REM  обработать таблицы.
    If oPar.supportsService("com.sun.star.text.Paragraph") Then

      i = i + 1 : s = ""
      REM  Теперь перебираем текстовые разделы и посмотрим графические
      REM  объекты, которые привязаны (anchored) к символу или оформлены как
символл.
      oSectionEnum = oPar.createEnumeration()
      Do While oSectionEnum.hasMoreElements()
        oSection = oSectionEnum.nextElement()

        If oSection.TextPortionType = "Text" Then
          REM  Это прото текстовый объект!
          s = s & oSection.TextPortionType & " : "
          s = s & oSection.getString() & CHR$(10)
        Else
          s = s & oSection.TextPortionType & " : " & CHR$(10)
```

```
        End If
      Loop
      MsgBox s, 0, "Paragraph " & i
    End If
  Loop
End Sub
```

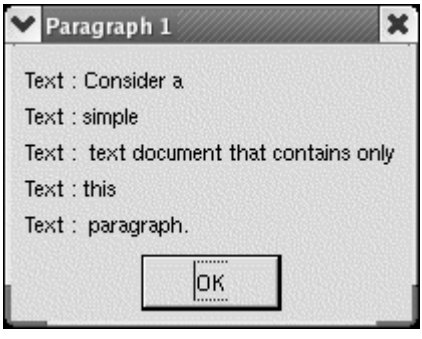


Рисунок 3: Перебранный параграф.

Более сложный пример показан в Лист. 247 следующего раздела, который перебирает графическое содержимое. В моей книге это рассмотрено подробнее!

V.16.4. Перебор текста и поиск текстового содержимого (content)

Основной причиной для перебора текстового содержимого является экспорт документа. Недавно меня спрашивали, как распознать графические объекты, встроенные в текст. Макрос FindGraphics находит графические объекты, которые связаны с параграфом, связанные с символом, а также встроенные в качестве символа. Он не находит изображения, привязанные к странице.

Процедура FindGraphics находит TextGraphicObjects (графические объекты текста) и GraphicObjectShapes (графические объекты формы). Графические объекты текста TextGraphicObject созданы для вставки в текстовый объект и используются через меню Вставка — Изображение для документа OOo Writer. Я могу выполнить двойной клик на графическом объекте текста TextGraphicObject и получить множество свойств, имеющихся для этого объекта; но это не так для графических объектов формы GraphicObjectShape.

Лист. 247: Найти графические объекты, встроенные в текст.

```
Sub FindGraphics
  REM Author: Andrew Pitonyak
```

```
  Dim oParEnum     'Счетчик, используемый для перебора параграфов
  Dim oPar         'Перебираемый параграф
  Dim oSectEnum    'Счетчик, используемый для перебора текстовых разделов
  Dim oSect        'Перебиремый текстовый раздел (section)
  Dim oCEnum 'Перебирает содержимое, такое как графические объекты
  Dim oContent     'Перебираемое содержимое
  Dim msg1$, msg2$, msg3$, msg4$
  Dim textGraphService$, graphicService$, textCService$

  textGraphService$ = "com.sun.star.text.TextGraphicObject"
  graphicService$   = "com.sun.star.drawing.GraphicObjectShape"
  textCService$     = "com.sun.star.text.TextContent"

  msg1$ = "Найден графический объект текста TextGraphicObject привязанный
к параграфу: URL = "
  msg2$ = "Найден графический объект формы GraphicObjectShape anchored to a
Paragraph: URL = "
  msg3$ = "Found a TextGraphicObject anchored " & _
          "to or as a character:  URL = "
  msg4$ = "Found a GraphicObjectShape anchored " & _
          "to or as a character:  URL = "

  REM  Перебрать параграфы
  REM  Таблицы перебираются одновременно с параграфами
  oParEnum = ThisComponent.getText().createEnumeration()
  Do While oParEnum.hasMoreElements()
    oPar = oParEnum.nextElement()

    REM  Это пропускает таблицы. Добавьте оператор else, чтобы
    REM  обработать таблицы
    If oPar.supportsService("com.sun.star.text.Paragraph") Then

      REM  Если нужно увидеть типы, которые допустимы для
      REM  перебора в качестве содержимого, привязанного к параграфу,
      REM  посмотрите имена допустимых сервисов.
      REM MsgBox Join(oPar.getAvailableServiceNames(), CHR$(10)

      REM  Обычно я использую пустую строку для перебора ВСЕГО содержимого
      REM  Но это вызывает здесь ошибку выполнения макроса. Если хоть
      REM  какие-то графические объекты присутствуют, то они перебираются
      REM как TextContent.
      oCEnum = oPar.createContentEnumeration(textCService$)
      Do While oCEnum.hasMoreElements()
        oContent = oCEnum.nextElement()
        If oContent.supportsService(textGraphService$) Then
          Print msg1$ & oContent.GraphicURL
        ElseIf oContent.supportsService(graphicService$) Then
          Print msg2$ & oContent.GraphicURL
        '  EmbedLinkedGraphic(oContent)
        'Else
        '  Inspect(oContent)
        End If
      Loop

      REM  Теперь переберем текстовые разделы (sections) и поищем
```

```
      REM  графические объекты, которые привязаны к символу, или вставлены
      REM  как символы
      oSectEnum = oPar.createEnumeration()
      Do While oSectEnum.hasMoreElements()
        oSect = oSectEnum.nextElement()

        If oSect.TextPortionType = "Text" Then
          REM  Это просто текстовый объект!
          'MsgBox oSect.TextPortionType & " : " & _
          '       CHR$(10) &  oSect.getString()
        ElseIf oSect.TextPortionType = "Frame" Then

          REM  Используйте пустую строку для перебора ВСЕГО содержимого
          oCEnum = oSect.createContentEnumeration(textGraphService$)
          Do While oCEnum.hasMoreElements()
            oContent = oCEnum.nextElement()
            If oContent.supportsService(textGraphService$) Then
              Print msg3$ & oContent.GraphicURL
            ElseIf oContent.supportsService(graphicService$) Then
              Print msg4$ & oContent.GraphicURL
            '  EmbedLinkedGraphic(oContent)
            'Else
            '  Inspect(oContent)
            End If
          Loop
        End If
      Loop
    End If
  Loop
End Sub
```

V.16.5. Но я только хочу найти графические объекты

Я проверил этот макрос на документе OOo Writer и он пересмотрел все графические объекты. Это явно наиболее быстрый способ найти графические объекты.

Лист. 248: Find all items on a draw page Найти все элементы страницы чертежа, графики (draw page).

```
  Dim i As Integer
  Dim oGraph
  For i=0 To ThisComponent.Drawpage.getCount()-1
    oGraph = ThisComponent.Drawpage.getByIndex(i)
  Next i
```

V.16.6. Найти текстовое поле, содержащееся в текущем параграфе?

CPH (Mr. Hennessy) хотел текстовое поле в начале текстового параграфа

1. Можно перебирать текстовые поля, проверяя, привязаны ли они к текущему параграфу.
2. Текстовый курсор может перебирать содержимое, но текстовые поля не включаются в перебор.

3. Можно передвигать курсор на один симовол за один раз через весь текущий параграф в поисках полей (см. Лист. 249). Похожий способ использован для определения, содержится ли текущий курсор в текстовой таблице или ячейке (см. Лист. 190 и Лист. 191).

Лист. 249: Использовать курсор для поиска текстового поля.

```
  oTCurs.gotoStartOfParagraph(False)
  Do While oTCurs.goRight(1, False) and NOT oTCurs.isEndOfParagraph()
    If NOT IsEmpty(oTCurs.TextField) Then
      Print "Найдено поле путем перемещения курсора по тексту."
    End If
  Loop
```

Перебор текстового содержимого из текстового курсора будет перебирать графические объекты, но это НЕ перебирает текстовые поля.

Лист. 250: Перебрать текстовое содержимое.

```
sTContentService = "com.sun.star.text.TextContent"
oEnum = oTCurs.createContentEnumeration(sTContentService)
Do While oEnum.hasMoreElements()
  oSect = oEnum.nextElement()
  Print "Enumerating TextContent: " & oSect.ImplementationName
Loop
```

Текущий параграф может быть перебран из текстового курсора.

Лист. 251: Начать перебор из текстового курсора.

```
  oEnum = oTCurs.createEnumeration()
  Do While oEnum.hasMoreElements()
    v = oEnum.nextElement()
    oSecEnum = v.createEnumeration()
    Do While oSecEnum.hasMoreElements()
      oSubSection = oSecEnum.nextElement()
      If oSubSection.TextPortionType = "TextField" Then
        REM Заметьте, что доступ осуществляется к Textfield ,
        REM Можно также получить доступ к
        REM TextFrame, TextSection, или  TextTable.
        'Текстовое поле здесь
      ElseIf oSubSection.TextPortionType = "Frame" Then
        'Графические объекты здесь!
      End If
    Loop
  Loop
```

Собирая макросы вместе в единую программу, получаем следующее:

Лист. 252: Найти графические объекты и текстовые поля в текущем параграфе.

```
Sub GetTextFieldFromParagraph
  Dim oEnum   'Cursor enumerator.
  Dim oSect   'Current Section.
```

```
Dim s$       'Generic string variable.
Dim oVCurs   'Holds the view cursor.
Dim oTCurs   'Created text cursor.
Dim oText    'Text object that contains the view cursor
Dim  sTContentService$

sTContentService = "com.sun.star.text.TextContent"
REM Only the view cursor knows where a line ends.
oVCurs = ThisComponent.CurrentController.getViewCursor()

REM Use the text object that contains the view cursor.
oText = oVCurs.Text

REM Require a text cursor so that you know where the paragraph ends.
REM Too bad the view cursor is not a paragraph cursor.
oTCurs = oText.createTextCursorByRange(oVCurs)
oTCurs.gotoStartOfParagraph(False)
oTCurs.gotoEndOfParagraph(True)

REM This does NOT work to enumerate text fields,
REM but it enumerates graphics.
oEnum = oTCurs.createContentEnumeration(sTContentService)
Do While oEnum.hasMoreElements()
  oSect = oEnum.nextElement()
  Print "Enumerating TextContent: " & oSect.ImplementationName
Loop

REM focus the cursor over the paragraph again.
oTCurs.gotoStartOfParagraph(False)
oTCurs.gotoEndOfParagraph(True)
REM и this provides the paragraph!
oEnum = oTCurs.createEnumeration()
Dim v
Do While oEnum.hasMoreElements()
  v = oEnum.nextElement()
  Dim oSubSection
  Dim oSecEnum
  oSecEnum = v.createEnumeration()
  s = "Enumerating section type: " & v.ImplementationName
  Do While oSecEnum.hasMoreElements()
    oSubSection = oSecEnum.nextElement()
    s = s & CHR$(10) & oSubSection.TextPortionType
    If oSubSection.TextPortionType = "TextField" Then
      REM Notice how a Textfield is accessed, you can also access a
      REM TextFrame, TextSection, or a TextTable.
      s = s & " <== here is a text field "
      s = s &  oSubSection.TextField.ImplementationName
    ElseIf oSubSection.TextPortionType = "Frame" Then
      s = s & " <== here is a Frame "
    End If
  Loop
  MsgBox s, 0, "Пересчитать единственный параграф"
Loop

REM Move the cursor one character at a time looking
```

```
  REM for a text field.
  oTCurs.gotoStartOfParagraph(False)
  Do While oTCurs.goRight(1, False) and NOT oTCurs.isEndOfParagraph()
    If NOT IsEmpty(oTCurs.TextField) Then
      Print "Найти поле передвижением курсора через текст."
    End If
  Loop
End Sub
```

V.17. Где находится курсор дисплея (Display Cursor)?

Не время для длинных описаний, но вот сообщение от Giuseppe Castagno [castagno@tecsa-srl.it], который предложил следующую идею.

Здесь много интересных вещей, но, думаю, они не корректны. Во-первых, получить позицию (get position) выглядит так, что она считается относительно первой позиции в начале страницы, которая может содержать текст. Если задан верхний заголовок, позиция считается относительно него, а если верхнего заголовка нет, то позиция считается от верха рамки текста. Это получается что-то вроде того, что верхнее поле считается от верха страницы до первой позиции, в которой может содержаться текст..

Ваше измерение нижней позиции очень хорошо, потому что показывает расстояние от верхней границы нижнего заголовка до курсора. Я приятно поражен, не нашел возражений. Но, с другой стороны, что если увеличится размер нижнего заголовка? Вы, думается, не принимали этого в расчет.

Вероятно, можно сделать что-то вроде следующего:

(Высота страницы) минус (верхнее поле) минус (позиция курсора)

И нет необходимости двигать курсор туда или сюда.

Лист. 253: Где курсор?

```
Sub PrintCursorLocation
  Dim xDoc
  Dim xViewCursor
  Dim s As String

  xDoc = ThisComponent
  xViewCursor = xDoc.CurrentController.getViewCursor()
  s = xViewCursor.PageStyleName

  Dim xFamilyNames As Variant, xStyleNames As Variant
  Dim xFamilies
  Dim xStyle, xStyles

  xFfamilies = xDoc.StyleFamilies
  xStyles = xFamilies.getByName("PageStyles")
  xStyle  = xStyles.getByName(xViewCursor.PageStyleName)
'  RunSimpleObjectBrowser(xViewCursor)
```

```
  Dim lHeight As Long
  Dim lWidth  As Long
  lHeight = xStyle.Height
  lWidth = xStyle.Width

  s = "Page size is " & CHR$(10) &_
      "     " & CStr(lWidth / 100.0) & " mm By " &_
      "     " & CStr(lHeight / 100.0) & " mm" & CHR$(10) &_
      "     " & CStr(lWidth / 2540.0) & " inches By " &_
      "     " & CStr(lHeight / 2540.0) & " inches" & CHR$(10) &_
      "     " & CStr(lWidth *72.0 / 2540.0) & " picas By " &_
      "     " & CStr(lHeight *72.0 / 2540.0) & " picas" & CHR$(10)

  Dim dCharHeight As Double
  Dim iCurPage As Integer

  Dim dXCursor As Double
  Dim dYCursor As Double
  Dim dXRight  As Double
  Dim dYBottom As Double
  Dim dBottomMargin As Double
  Dim dLeftMargin As Double


  dCharHeight = xViewCursor.CharHeight / 72.0
  iCurPage =  xViewCursor.getPage()

  Dim v
  v = xViewCursor.getPosition()
  dYCursor = (v.Y + xStyle.TopMargin)/2540.0 + dCharHeight / 2
  dXCursor = (v.X + xStyle.LeftMargin)/2540.0
  dXRight  = (lWidth - v.X - xStyle.LeftMargin)/2540.0
  dYBottom = (lHeight - v.Y - xStyle.TopMargin)/2540.0 - dCharHeight / 2
'  Print "Left margin = " & xStyle.LeftMargin/2540.0


  dBottomMargin = xStyle.BottomMargin / 2540.0
  dLeftMargin = xStyle.LeftMargin / 2540.0
  s = s & "Cursor is " & Format(dXCursor, "0.##")& " inches from left "   &
CHR$(10)
  s = s & "Cursor is " & Format(dXRight, "0.##") & " inches from right "  &
CHR$(10)
  s = s & "Cursor is " & Format(dYCursor, "0.##")& " inches from top "    &
CHR$(10)
  s = s & "Cursor is " & Format(dYBottom, "0.##")& " inches from bottom " &
CHR$(10)
  s = s & "Left margin = " & dLeftMargin & " inches" & CHR$(10)
  s = s & "Bottom margin = " & dBottomMargin & " inches" & CHR$(10)
  s = s & "Char height = " & Format(dCharHeight, "0.####") & " inches" &
CHR$(10)

'  RunSimpleObjectBrowser(xStyle)

'  Dim dFinalX As Double
'  Dim dFinalY As Double
```

```
'  dFinalX = dXCursor + dLeftMargin
'  dFinalY = (v.Y + xStyle.TopMargin)/2540 + dCharHeight / 2
'  s = s & "Cursor in page is at (" & Format(dFinalX, "0.####") & ", " &_
'          Format(dFinalY, "0.####") & ") in inches" & CHR$(10)

  REM  Теперь проверяем нижний заголовок!
'  If xStyle.FooterIsOn Then
'    v = IIF(iCurPage MOD 2 = 0, xStyle.FooterTextLeft,
xStyle.FooterTextRight)
'    If IsNull(v) Then v = xStyle.FooterText
'    If Not IsNull(v) Then
'      REM Save the postion
'      Dim xOldCursor
'      xOldCursor = xViewCursor.getStart()
'      xViewCursor.gotoRange(v.getStart(), false)
'      Print "footer position = " & CStr(xViewCursor.getPosition().Y/2540.0)
'      dFinalY = xViewCursor.getPosition().Y/2540.0 — dYCursor +
dBottomMargin
'      xViewCursor.gotoRange(xOldCursor, false)
'    End If
'  Else
'    Print "Нижнего заголовка нет"
'  End If
'  s = s & "Курсор на странице находится в позиции (" & Format(dFinalX,
"0.####") & ", " &_
'          Format(dFinalY, "0.####") & ")  в дюймах" & CHR$(10)
  MsgBox s, 0, "Информация о странице"
End Sub
```

V.17.1. Какой курсор нужно использовать для удаления строки или параграфа

Видимый курсор «знает» о строках и страницах, обычные курсоры знают о словах, предложениях и параграфах.

Лист. 254: Удаление строки.

```
Sub DeleteCurrentLine  ' Удаляем текущую строку
  Dim oVCurs
  oVCurs = ThisComponent.getCurrentController().getViewCursor()
  oVCurs.gotoStartOfLine(False)
  oVCurs.gotoEndOfLine(True)
  oVCurs.setString("")
End Sub
```

Чтобы удалить текущий параграф, используйте запрос курсора параграфов (paragraph cursor), который не является видимым курсором. Видимый курсор нужен, если требуется найти текущую позицию курсора.

Лист. 255: Удаление параграфа.

```
Sub DeleteParagraph
  Dim oCurs
  Dim oText
```

```
  Dim oVCurs
  oVCurs = ThisComponent.getCurrentController().getViewCursor()

  REM  Получить текстовый объект из курсора — более надежно, чем
предполагать,
  REM  что объект текстового документа верхнего уровня содержит видимый
курсор
  REM
  oText = oVCurs.getText()
  oCurs = oText.createTextCursorByRange(oVCurs)
  oCurs.gotoStartOfParagraph(False)
  If oCurs.gotoNextParagraph(True) Then
    oCurs.setString("")
  Else
    REM  Теперь мы находимся уже  в (AT) последнем параграфе
    If oCurs.gotoPreviousParagraph(False) Then
      oCurs.gotoEndOfParagraph(False)
      oCurs.gotoNextParagraph(True)
      oCurs.gotoEndOfParagraph(True)
      oCurs.setString("")
    Else
      Rem  Существует ровно один параграф
      REM  Удаляем его
      oCurs.gotoStartOfParagraph(False)
      oCurs.gotoEndOfParagraph(True)
      oCurs.setString("")
    End If
  End If
End Sub
```

V.17.2. Удалить текущую страницу

Чтобы удалить страницу целиком, требуется видимый курсор (view cursor), потому что только видимый курсор «знает», где начинается и заканчивается страница. См. следующий упрощенный пример:

Лист. 256: Удаление страницы.

```
Sub removeCurrentPage()
  REM Author: Andrew Pitonyak
  Dim oVCurs
  Dim oCurs

  oVCurs = ThisComponent.getCurrentController().getViewCursor()
  If oVCurs.jumpToStartOfPage() Then
    oCurs = ThisComponent.getText().CreateTextCursorByRange(oVCurs)
    If (oVCurs.jumpToEndOfPage()) Then
      oCurs.gotoRange(oVCurs, True)
      oCurs.setString("")
    Else
      Print "Нельзя перескочить к концу страницы"
    End If
  Else
    Print "Нельзя перескочить к началу страницы"
```

```
  End If
End Sub
```

Я частично проверил этот макрос. Если страница начинается с параграфа и заканчивается параграфом, макрос может оставить лишний пустой параграф в тексте. Очень похоже, что метод jumpToEndOfPage(), вероятно, не включает символ конца параграфа.

V.18. Вставка индекса или оглавления (table of contents)

Оглавление (TOC = table of contents) — это просто другой тип индекса. Я вставил достаточно комментариев, чтобы ответить на большинство Ваших вопросов о том, как он работает.

Лист. 257: Вставка оглавления.

```
Sub InsertATOC   ' Вставить TOC (оглавление)
  REM Author: Andrew Pitonyak
  Dim oCurs                  'Используется для вставки текстового
содержимого
  Dim oIndexes               'Все существующие индексы
  Dim oIndex                 'TOC  , если существует, или новый, если
не было
  Dim i As Integer           'Найти существующий TOC
  Dim bIndexFound As Boolean 'Флаг (признак), что TOC  уже найден
  Dim s$

  REM  Во-первых, находим существующий TOC , если он есть. Если удалось,
  REM  то он будет просто обновлен
  oIndexes = ThisComponent.getDocumentIndexes()
  bIndexFound = False
  For i = 0 To oIndexes.getCount() - 1
    oIndex = oIndexes.getByIndex(i)
    If oIndex.supportsService("com.sun.star.text.ContentIndex") Then
      bIndexFound = True
      Exit For
    End If
  Next

  If Not bIndexFound Then
    Print "Не найден существующий индекс (content index)"
    REM  Возможно, нужно создать и вставить новый индекс!
    REM  Заметьте, что он ДОЛЖЕН быть создан в том документе, который
    REM  будет содержать этот индекс.
    S = "com.sun.star.text.ContentIndex"
    oIndex = ThisComponent.createInstance(s)

    REM  На моем компьютере есть значения по умолчанию
    REM  Как Вы хотите создавать индекс?
    REM CreateFromChapter = False
    REM CreateFromLevelParagraphStyles = False
    REM CreateFromMarks = True
    REM CreateFromOutline = False
```

```
    oIndex.CreateFromOutline = True

    REM Вы можете установить ещё много видов других свойств, таких как
    REM  заголовок (Title) или уровень (Level)

    oCurs = ThisComponent.getText().createTextCursor()
    oCurs.gotoStart(False)
    ThisComponent.getText().insertTextContent(oCurs, oIndex, False)
  End If

  REM  Даже только что вставленный индекс не обновляется до этого места!
  oIndex.update()
End Sub
```

V.19. Вставка адреса URL в документ OOo Writer

Хотя документ OOo Calc сохраняет адреса URL в текстовом поле URL, как говорится в другом месте этой статьи,а документ OOo Writer идентифицирует содержащиеся в нем адреса URL на основе свойств символа. Ссылка становится ссылкой, когда установлено свойство HyperLinkURL

Лист. 258: URL

```
Sub InsertURLAtTextCursor
  Dim oText    'Текстовый объект для текущего объекта
  Dim oVCursor 'Текущий видимый курсор

  oVCursor = ThisComponent.getCurrentController().getViewCursor()
  oText = oVCursor.getText()
  oText.insertString(oVCursor, "andrew@pitonyak.org", True)
  oVCursor.HyperLinkTarget = "mailto:andrew@pitonyakorg"

  oVCursor.HyperLinkURL = "andrew@pitonyak.org"
'  oVCursor.HyperLinkName = "andrew@pitonyak.org"
'  oVCursor.UnvisitedCharStyleName = "Internet Link"
'  oVCursor.VisitedCharStyleName = "Visited Internet Link"
End Sub
```

V.20. Сортировка текста

Текстовый курсор можно использовать для сортировки данных в документе OOo Writer.

Лист. 259: Сортировка.

```
Sub SortTextInWrite
  Dim oText    'Текстовый объект для текущего объекта
  Dim oVCursor 'Текущий видимый курсор
  Dim oCursor  'Текстовый курсор
  Dim oSort
  Dim i%
  Dim s$
```

```
  REM Предположим, что нужно сортировать выделенный (selected) текст!
  REM  К несчастью, видимый курсор НЕ может создать дескриптор сортировки
  REM  Создаем новый текстовый курсор, который может создать дескриптор
сортировки
  oVCursor = ThisComponent.getCurrentController().getViewCursor()
  oText = oVCursor.getText()
  oCursor = oText.createTextCursorByRange(oVCursor)
  oSort = oCursor.createSortDescriptor()

'  On Error Resume Next
'  For i = LBound(oSort) To UBound(oSort)
'    s = s & "(" & oSort(i).Name & ", "
'    s = s & oSort(i).Value
'    s = s & ")" & CHR$(10)
'  Next
'  MsgBox s, 0, "Условия сортировки"
  oCursor.sort(oSort)
End Sub
```

Используйте SortDescriptor2, применять SortDescriptor не рекомендуется. См. подробнее:

http://api.openoffice.org/docs/common/ref/com/sun/star/text/TextSortDescriptor2.html

http://api.openoffice.org/docs/common/ref/com/sun/star/table/TableSortDescriptor2.html

Значения по умолчанию для дескриптора сортировки (sort descriptor), похоже, следующие: (IsSortInTable, False), (Delimiter, 32), (IsSortColumns, True), (MaxSortFieldsCount, 3), (SortFields,). Для установки порядка сортировки и языка (локали, Locale) нужно изменить параметр SortFields.

V.21. Нумерация структур (Outline)

Меня несколько раз спрашивали, как изменить правило, какие стили используются для нумерации структур (outline numbering). Сначала для нумерации структур нужно иметь стиль параграфа. Макрос в Лист. 260 создаёт новый стиль параграфа с указанием высоты символа. Родительским стилем указывается «Heading». Можете изменить стиль параграфа для Ваших нужд.

Лист. 260: Создать новый пользовательский стиль параграфа.

```
Function CreateParaStyle(oDoc, NewParaStyle, nCharHeight%)
  Dim pFamilies, pStyle, pParaStyles

  pFamilies = oDoc.StyleFamilies
  pParaStyles = pFamilies.getByName("ParagraphStyles")

  If  Not pParaStyles.hasByName(NewParaStyle) then
    pStyle = oDoc.createInstance("com.sun.star.style.ParagraphStyle")
    pStyle.setParentStyle("Heading")
    pStyle.CharHeight = nCharHeight
    pParaStyles.insertByName(NewParaStyle, pStyle)
```

```
  End if
End Function
```

После создания нового пользовательского стиля параграфа или по меньшей мере выбора стилей параграфа для использования можно установить нумерацию. Макрос в Лист. 261 принимает массив имен стилей параграфа и устанавливает нумерацию структур для использования этих стилей.

Информация о нумерации структур хранится в объекте правил нумерации разделов. Получить каждое правило можно используя getByIndex(), который возвращает массив свойств. Просмотрите имя каждого свойства, чтобы найти то свойство, которое нужно изменить. Каждое свойство копируется ИЗ этого массива. Если эти свойства хранились как сервис UNO, а не структура UNO, то будет скопирована ссылка из этого массива, а не копия (я объясняю копирование по значению, а не копирование по ссылке в моей книге). После обнаружения и изменения значений нужных свойств это правило копируется обратно в объект правил.

Лист. 261: Установить нумерацию структур.

```
Sub SetNumbering(sNames())
  Dim i%, j%
  Dim oRules
  Dim oRule()
  Dim oProp

  oRules = ThisComponent.getChapterNumberingRules()
  For i = 0 To UBound(sNames())
    If i >= oRules.getCount() Then Exit Sub
    oRule() = oRules.getByIndex(i)
    REM  Я не устанавливаю следующее:
    REM Adjust, StartWith, LeftMargin,
    REM SymbolTextDistance, FirstLineOffset
    For j = LBound(oRule()) To Ubound(oRule())
      REM oProp  — это только копия свойства.
      REM  Нужно присвоить значение свойства обратно в массив.
      oProp = oRule(j)
      Select Case oProp.Name
      Case "HeadingStyleName"
        oProp.Value = sNames(i)
      Case "NumberingType"
        oProp.Value = com.sun.star.style.NumberingType.ARABIC
      Case "ParentNumbering"
        oProp.Value = i + 1
      Case "Prefix"
        oProp.Value = ""
      Case "Suffix"
        oProp.Value = " "
      'Case "CharStyleName"
      '  oProp.Value =
      End Select
```

```
      oRule(j) = oProp
    Next
    oRules.replaceByIndex(i, oRule())
  Next
End Sub
```

Если массив, переданный макросу в Лист. 261 , содержит три элемента, то только первые три элемента в структуре модифицируются. Используйте этот макрос как отправную точку. Лист. 262 показывает, как использовать вместе Лист. 260 и Лист. 261.

Лист. 262: Установить и использовать нумерацию структур

```
Sub SetOutlineNumbering
  CreateParaStyle(ThisComponent, "_New_Heading_1", 16)
  CreateParaStyle(ThisComponent, "_New_Heading_2", 14)
  setNumbering(Array("_New_Heading_1", "_New_Heading_2"))
End Sub
```

V.22. Вставить оглавление (table of contents =TOC) или другой индекс.

Следующий макрос вставляет оглавление(table of contents TOC) в документ. Если TOC уже существует, тогда вызывается метод обновления (update) существующего индекса. Используйте метод dispose для удаления существующего индекса из документа.

Лист. 263: Вставка оглавления

```
Sub InsertATOC
  REM  Автор: Andrew Pitonyak
  Dim oCurs                  'Используется для вставки текстового
содержимого.
  Dim oIndexes               'Все существующие индексы
  Dim oIndex                 'Оглавление TOC , если существует, лии новое
оглавление, если не было
  Dim i As Integer           'Найти существующее оглавление TOC
  Dim bIndexFound As Boolean 'Флаг (признак) для указания, что TOC уже
найдено
  Dim s$

  REM  Сначала находим TOC , если оно существует. В этом случае,
  REM  оно будет просто обновлено (updated).
  oIndexes = ThisComponent.getDocumentIndexes()
  bIndexFound = False
  For i = 0 To oIndexes.getCount() - 1
    oIndex = oIndexes.getByIndex(i)
    If oIndex.supportsService("com.sun.star.text.ContentIndex") Then
      bIndexFound = True
      Exit For
    End If
  Next
```

```
  If Not bIndexFound Then
    Print "Не найден существующий индекс"
    REM  Возможно, нужно создать и вставить новое оглавление!
    REM  Заметьте, что оглавление ДОЛЖНО создаваться в документе, который
    REM  содержит это оглавление.
    S = "com.sun.star.text.ContentIndex"
    oIndex = ThisComponent.createInstance(s)

    REM  На моем компьютере заданы значения по умолчанию
    REM  Как Вы хотите создавать этот индекс?
    REM CreateFromChapter = False
    REM CreateFromLevelParagraphStyles = False
    REM CreateFromMarks = True
    REM CreateFromOutline = False
    oIndex.CreateFromOutline = True

    REM  Вы можете установить все разновидности других вещей, таких как
    REM Title (заголовок) или Level (уровень)

    oCurs = ThisComponent.getText().createTextCursor()
    oCurs.gotoStart(False)
    ThisComponent.getText().insertTextContent(oCurs, oIndex, False)
  End If

  REM  Даже вновь созданный индекс ещё не обновлен до ЭТОГО оператора!
  oIndex.update()
End Sub
```

V.23. Текстовые секции (sections)

Текстовая секция TextSection является интервалом целых параграфов, содержащим текстовый объект (contained a text object). Содержимое может быть содержимым ссылки на другой документ, ссылкой на тот же документ, или результатом операции DDE. Экземпляры TextSection могут ссылаться на и на них можно ссылаться из других текстов. Содержимое текстовой секции (text section) нумеруется через весь обычный текст и может быть перебрано текстовым курсором. Текст может быть пересмотрен, даже если он помечен как невидимый, что можно проверить с помощью свойства oSect.IsVisible.

Лист. 264: Вывести тест первой текстовой секции

```
'Автор: Andrew Pitonyak
Sub GetTextFromTextSection
  Dim oSect
  Dim oCurs

  If ThisComponent.getTextSections().getCount() = 0 Then
    Print "Нет текстовых секций"
    Exit Sub
  End If
```

```
  REM  Можно получить текстовые секции по имени или по индексу (номеру).
  'oSect = ThisComponent.getTextSections().getByName("Special")
  oSect = ThisComponent.getTextSections().getByIndex(0)

  REM  Создаем курсор и передвигаем этот курсор по текстовой секции.
  oCurs = ThisComponent.getText().createTextCursor()
  oCurs.gotoRange(oSect.getAnchor(), False)

  REM  В этом примере предполагается, что есть текст после
  REM данной текстовой секции. В противном случае, получится зацикливание
  REM  Тогда лучше проверять, что
  REM gotoNextParagraph не возвращает значение False.
  Do While NOT IsEmpty(oCurs.TextSection)
    oCurs.gotoNextParagraph(False)
  Loop

  oCurs.gotoPreviousParagraph(False)
  oCurs.gotoRange(oSect.getAnchor(), True)
  MsgBox oCurs.getString()
End Sub
```

V.23.1. Вставить текстовую секцию, установка столбцов и значений ширины

Вставить текстовую секцию с двумя столбцами. Затем установить расстояние ½ между столбцами. Подробнее объяснить не смог.

Лист. 265: Вывести текст первой текстовой секции

```
Sub AddTextSection
  Dim oSect
  Dim sName$
  Dim oVC
  Dim oText
  Dim oCols
  Dim s$
  sName = "ADPSection"
  If ThisComponent.getTextSections().hasByName(sName) Then
    Print "Текстовая секция " & sName & " уже существует"
    oSect = ThisComponent.getTextSections().getByName(sName)
  Else
    REM  Создаем текстовую секцию в месте нахождения курсора
    oVC = ThisComponent.getCurrentController().getViewCursor()
    oText = oVC.getText()
    REM  Вставляем новый параграф
    oText.insertControlCharacter(oVC, _
        com.sun.star.text.ControlCharacter.LINE_BREAK, False)
    REM  Вставляем новый параграф и выделяем его
    oText.insertControlCharacter(oVC, _
        com.sun.star.text.ControlCharacter.LINE_BREAK, True)
    s = "com.sun.star.text.TextSection"
    oSect = ThisComponent.createInstance(s)
    oSect.setName(sName)
    REM  Теперь создаем столбцы...
     s = "com.sun.star.text.TextColumns"
```

```
    oCols = ThisComponent.createInstance(s)
    oCols.setColumnCount(2)
    oSect.TextColumns = oCols
    oText.insertTextContent(oVC, oSect, True)
    oText.insertString(oVC, "Это — новый текст. " & _
        "Я предполагаю, что я мог бы пересчитать и повторить себя как " & _
        "пример того, как текст может повторяться снова и снова и снова. " &
_
        "Я предполагаю, что я мог бы пересчитать и повторить себя как " & _
        "пример того, как текст может повторяться снова и снова и снова. " &
_
        "Я предполагаю, что я мог бы пересчитать и повторить себя как " & _
        "пример того, как текст может повторяться снова и снова и снова. " &
_
        "Я предполагаю, что я мог бы пересчитать и повторить себя как " & _
        "пример того, как текст может повторяться снова и снова и снова. " &
_
        "Я предполагаю, что я мог бы пересчитать и повторить себя как " & _
        "пример того, как текст может повторяться снова и снова и снова. " &
_
        "Я предполагаю, что я мог бы пересчитать и повторить себя как " & _
        "пример того, как текст может повторяться снова и снова и снова. " &
_
        "И наконец, я останавливаюсь.", True)
    Print "Созданная текстовая секция"
  End If
  oCols = oSect.TextColumns
  Dim OC()
  OC() = oCols.getColumns()
  REM  Устанавливаем размер правого поля первого столбца равным 1/4 дюйма.
  OC(0).RightMargin = 635
  REM  Устанавливаем размер левого поля второго столбца равным  1/4  дюйма.
  OC(1).LeftMargin = 635
  oCols.setColumns(OC())
  oSect.TextColumns = oCols
End Sub
```

V.24. Сноски на странице (Footnotes) и сноски в конце текста (Endnotes)

Есть два типа сносок на странице (footnotes): автоматические и указанные пользователем. Автоматические сновки на странице автоматически устанавливают номер сноски по порядку, этот номер не совсем поле (field). Если указана метка (label), то сноска на странице не включается в автоматическую нумерацию.

Лист. 266: Очень легко пронумеровать сноски на странице.

```
oNotes = ThisComponent.getFootnotes()
 For i = 0 To oNotes.getCount() — 1
   oNote = oNotes.getByIndex(i)
 Next
```

Используйте getString() и setString() для получения и установки текста сноски.

Используйте getLabel() и setLabel() для получения метки, заданной пользователем. Если метка — пустая строка, то это автоматическая сноска. Преобразуйте эти два типа сносок путем установки значения метки, ненулевой или нулевой длины соответственно.

Используйте getEndNotes() для документа, чтобы получить сноски к концу документа вместо сносок внизу страницы.

Не игнорируйте возможности установки значений (getFootnoteSettings). Так, можно указать, как сноски нумеруются, например, начинать нумерацию сначала для каждой главы.

Глава VI. Текстовые таблицы

Текстовые таблицы трудны для понимания, возникает множество вопросов по текстовым таблицам. Поэтому я начал новый раздел специально о текстовых таблицах. Со временем я расширю этот раздел и помещу в него все примеры текстовых таблиц.

VI.1. Поиск текстовых таблиц

Наиболее лёгкий способ найти текстовые таблицы — получить объект TextTables и выбирать их напрямую. Объект TextTable возвращает все текстовые таблицы документа.

Лист. 267: Получить текстовые таблицы из объета TextTables.

```
Sub GetTextTablesDirectly
  Dim s As String
  Dim i As Long
  Dim oTables
  Dim oTable

  oTables = ThisComponent.getTextTables()
  Print "Этот документ содержит " & oTables.getCount() & " таблиц"

  REM  Можно также проверить условие getCount() = 0.
  If NOT oTables.hasElements() Then Exit Sub

  For i = 0 To oTables.getCount() - 1
    oTable = oTables.getByIndex(i)
    s = s & oTable.getName() & CHR$(10)
  Next
  MsgBox s, 0, "Использование нумерации (перебора)"

  REM  Даже быстрее
  s = Join(oTables.getElementNames(), CHR$(10))
  MsgBox s, 0, "Using getElementNames()"

  REM  Получаем имя первой таблицы.
  s = oTables.getByIndex(0).getName()

  REM  Можно проверить, существует ли таблица с заданным именем
  REM и затем получить таблицу по её имени.
  If oTables.hasByName(s) Then
    oTable = oTables.getByName(s)
  End If
End Sub
```

VI.1.1. Где находится текстовая таблица

Используйте метод `getAnchor()` (получить привязку, «якорь») текстовой таблицы для определения, где расположена эта текстовая таблица. «Якорь» — это текстовый интервал, также текстовым интервалом является текстовый курсор. Текстовый курсор можно передвинуть на заданный текстовый интервал. К сожалению, «якорь», возвращаемый текстовой таблицей,

работает не так, как многие другие «якоря», поэтому курсор нельзя передвинуть на «якорь» текстовой таблицы.

Каждый видимый документ имеет контроллер (controller), и этот контроллер может выделять разные вещи. Выделение таблицы заставляет видимый курсор переместиться на первую ячейку этой текстовой таблицы. Можно затем использовать этот видимый курсор для передвижения на один символ влево, что поместит курсор прямо перед этой текстовой таблицей.

Лист. 268: Передвинуть видимый курсор перед текстовой таблицей.

```
'Передвинуть курсор на первую строку и первый столбец
ThisComponent.getCurrentController().select(oTable)
oCurs = ThisComponent.getCurrentController().getViewCursor()
oCurs.goLeft(1, False)
```

Окончательное расположение видимого курсора зависит от того, что находится перед текстовой таблицей. Если сразу перед текстовой таблицей находится параграф, то видимый курсор будет непосредственно перед этой текстовой таблицей. К сожалению, это не всегда так. Например, если таблица находится в самом начале документа, или если первой в документе стоит ячейка таблицы.

Стандартная хитрость для определения, находится ли видимый курсор в объекте конкретного типа, состоит в исследовании свойств видимого курсора. Как уже говорилось, «якорь» возвращаемый текстовой таблицей, не работает так, как большинство объектов типа интервалов, поэтому, хотя это и работает для видимого курсора, это не работает для «якоря» текстовой таблицы.

Лист. 269: Находится ли видимый курсор в текстовой таблице?

```
Dim oVC
oVC = ThisComponent.getCurrentController().getViewCursor()
If NOT IsEmpty(oVC.Cell) Then Print "В ячейке Cell"
If NOT IsEmpty(oVC.TextField) Then Print "В поле Field"
If NOT IsEmpty(oVC.TextFrame) Then Print "Во фрейме Frame"
If NOT IsEmpty(oVC.TextSection) Then Print "В секции Section"
If NOT IsEmpty(oVC.TextTable) Then Print "В таблице Table"
```

«Якорь» таблицы может возвращать текстовый объект, в котором он размещен. Создайте текстовый курсор из текстового объекта и затем исследуйте этот курсор для определения, находится ли внутри другого текстового объекта тот текстовый объект, где расположен «якорь» текстовой таблицы.

Лист. 270: Находится ли видимый курсор в текстовой таблице, содержащейся в другой текстовой таблице?

```
Dim oTable
Dim oVC
Dim oText
Dim oCurs

oVC = ThisComponent.getCurrentController().getViewCursor()
If IsEmpty(oVC.TextTable) Then
  Print "Видимый курсор не внутри текстовой таблицы"
  Exit Sub
End If

oTable = oVC.TextTable
oText = oTable.getAnchor().getText()
oCurs = oText.createTextCursor()
If NOT IsEmpty(oCurs.Cell) Then Print "Таблица в ячейке Cell"
If NOT IsEmpty(oCurs.TextField) Then Print "Таблица в поле Field"
If NOT IsEmpty(oCurs.TextFrame) Then Print "Таблица во фрейме Frame"
If NOT IsEmpty(oCurs.TextSection) Then Print "Таблица в секции Section"
If NOT IsEmpty(oCurs.TextTable) Then Print "Таблица в таблице Table"
```

Все эти примеры — крайние случаи, иллюстрирующие статьи, которые должны быть адресованы. Обычно Вы знаете свой документ и насколько он сложен, поэтому такой уровень детализации не требуется.

VI.1.2. Перебор (Enumerating) всех текстовых таблиц.

Текстовый документ содержит один первичный текстовый объект (primary text object), который содержит большинство текстового содержимого в документе. Примерами текстового содержимого ялвяются параграфы, текстовые таблицы, текстовые поля, графические объекты. Каждый текстовый объекты поддерживает метод для пересмотра текстового содержимого.

Лист. 271: Пересмотреть параграфы в текстовом объекте.

```
Sub EnumerateParagraphs
  Dim oParEnum
  Dim oPar
  Dim i As Long

  oParEnum = ThisComponent.getText().createEnumeration()
  Do While oParEnum.hasMoreElements()
    i = i + 1
    oPar = oParEnum.nextElement()
  Loop
  Print "Имеется " & i & " параграфов"
End Sub
```

Параграфы и текстовые таблицы перебираются на верхнем уровне. Другими словами, текстовые таблицы действуют очень похоже на параграфы.

Поэтому имеет смысл то, что одиночный параграф или одиночная строка не могут содержать больше чем одну текстовую таблицу.

Лист. 272: Перебрать параграфы и текстовые таблицы в текстовом объекте.

```
Sub EnumerateTextConent
  Dim oParEnum
  Dim oPar
  Dim nPar As Long
  Dim nTable As Long

  oParEnum = ThisComponent.getText().createEnumeration()
  Do While oParEnum.hasMoreElements()
    oPar = oParEnum.nextElement()
    If oPar.supportsService("com.sun.star.text.Paragraph") then
      nPar = nPar + 1
    ElseIf oPar.supportsService("com.sun.star.text.TextTable") Then
      nTable = nTable + 1
    End if
  Loop
  Msgbox "Имеется " & nPar & " параграфов" & CHR$(10) & _
         "Имеется " & nTable & " таблиц"
End Sub
```

Есть много способов вывести больше чем одну текстовую таблицу на одной и той же строке. Текстовая таблица может быть вставлена во фрейм, который может быть привязан (anchored) в качестве символа. Тексовая таблица может быть вставлена в текстовую секцию с несколькими столбцами, а также текстовая таблица может быть вставлена в ячейку другой текстовой таблицы. Макрос в Лист. 272 не найдет текстовую таблицу, которая находится в нестандартной позиции, такой как вложенной в верхний или нижний заголовок страницы, внутри другой текстовой таблицы, во фрейме или в секции.

VI.2. Перебор ячеек в текстовой таблице.

Обычный текст в документе содержится в текстовом объекте документа, который можно получить методом `getText()`. Каждая ячейка в текстовой таблице также содержит текстовый объект, который может быть перебран. Метод перебора ячеек зависит от текстовой таблицы. Простая таблица не содержит объединённых или разделённых ячеек (Таблица 9).

Таблица 9. Простая таблица с выведенными именами ячеек.

A1	B1	C1	D1
A2	B2	C2	D2
A3	B3	C3	D3

A1	B1	C1	D1
A4	B4	C4	D4

Используйте `getRows()` для получения строк из текстовой таблицы. Возвращаемый объект строк таблицы полезен для определения, сколько строк имеется, выборки, вставки и удаления строк (см. Таблица 10).

Таблица 10. Основные методы доступа, поддерживаемые объектом строк таблицы

Метод	***Описание***
getByIndex	Выбирает указанную строку
getCount	Число строк в таблице
hasElements	Определяет, есть ли строки в данной таблице
insertByIndex	Добавляет строки в таблицу
removeByIndex	Удаляет строки из таблицы

Хотя легко перебрать отдельные строки, строка бесполезна для получения ячеек, которые в ней содержатся. Отдельный объект строки таблицы в основном используется для следующего:

- Установить высоту строки, используя атрибут Height.
- Установить IsAutoHeight в значение true, чтобы высота строки устанавливалась автоматически.
- Установить IsSplitAllowed равным false, чтобы строка не могла быть разделена на несколько ячеек.
- Изменить TableColumnSeparators для изменения ширины столбцов.

Используйте `getColumns()` для получения столбцов из этой текстовой таблицы. Объект столбца таблицы поддерживаает те же методы, что и объект строки таблицы (см. Таблица 10). Тем не менее, нельзя получить указанный столбец из объекта столбца строки; существуе метод `getByIndex()`, но он возвращает null. Хотя я ожидал, что можно установить ширину столбца путем выборки указанного столбца и установки свойства width, но оказалось необходимым изменить разделители столбцов таблицы, которые доступны через объект строки таблицы.

VI.2.1. Простые текстовые таблицы

Перебрать ячейки простой текстовой таблицы очень легко. Макрос в Лист. 273 перебирает ячейки в текстовой таблице, содержащей видимый курсор.

Каждая ячейка извлекается с помощью метода `getCellByPosition()`, который доступен только для простой текстовой таблицы. Если текстовая таблица содержит объединенные ячейки или ячейки, разделенные на несколько ячеек (split), то нужно использовать другой метод

Лист. 273: Перебрать ячейки простой текстовой таблицы.

```
Dim s As String
Dim oTable
Dim oVC
Dim oCell
Dim nCol As Long
Dim nRow As Long

oVC = ThisComponent.getCurrentController().getViewCursor()
If IsEmpty(oVC.TextTable) Then
  Print "Видимый курсор не в текстовой таблице"
  Exit Sub
End If

oTable = oVC.TextTable
For nRow = 0 To oTable.getRows().getCount() - 1
  For nCol = 0 To oTable.getColumns().getCount() - 1
    oCell = oTable.getCellByPosition(nCol, nRow)
    s = s & oCell.CellName & ":" & oCell.getString() & CHR$(10)
  Next
Next
MsgBox s
```

VI.2.2. Форматирование простой текстовой таблицы

Я использую метод форматирования текстовой таблицы, аналогичный предложенному на сайте OOoAuthors (см. Таблица 10). Я помещаю текстовый курсор в текстовую таблицу и затем выполняю макрос из Лист. 274. Во-первых, этот макрос запрещает вертикальные линии между строками и разрешает все остальное. Далее, все ячейки перебираются способом, показанным в Лист. 273. Цвет фона каждой ячейки устанавливается в зависимости от позиции ячейки в таблице. Наконец, для каждой ячейки таблицы извлекается текстовый объект и перебираются параграфы, причем устанавливается стиль параграфа.

Лист. 274: Форматировать текстовую таблицу, как указано на сайте OOoAuthors.

```
Sub Main
  FormatTable()
End Sub

Sub FormatTable(Optional oUseTable)
  Dim oTable
  Dim oCell
  Dim nRow As Long
  Dim nCol As Long
```

```
  If IsMissing(oUseTable) Then
    oTable = ThisComponent.CurrentController.getViewCursor().TextTable
  Else
    oTable = oUseTable
  End If
  If IsNull(oTable) OR IsEmpty(oTable) Then
    Print "Форматирование таблицы: Не указана таблица"
    Exit Sub
  End If

  Dim v
  Dim x
  v = oTable.TableBorder
  x = v.TopLine        : x.OuterLineWidth = 2 : v.TopLine        = x
  x = v.LeftLine       : x.OuterLineWidth = 2 : v.LeftLine       = x
  x = v.RightLine      : x.OuterLineWidth = 2 : v.RightLine      = x
  x = v.TopLine        : x.OuterLineWidth = 2 : v.TopLine        = x
  x = v.VerticalLine   : x.OuterLineWidth = 2 : v.VerticalLine   = x
  x = v.HorizontalLine : x.OuterLineWidth = 0 : v.HorizontalLine = x
  x = v.BottomLine     : x.OuterLineWidth = 2 : v.BottomLine     = x
  oTable.TableBorder = v

  For nRow = 0 To oTable.getRows().getCount() - 1
    For nCol = 0 To oTable.getColumns().getCount() - 1
      oCell = oTable.getCellByPosition(nCol, nRow)
      If nRow = 0 Then
        oCell.BackColor = 128
        SetParStyle(oCell.getText(), "OOoTableHeader")
      Else
        SetParStyle(oCell.getText(), "OOoTableText")
        If nRow MOD 2 = 1 Then
          oCell.BackColor = -1
        Else
          REM color is (230, 230, 230)
          oCell.BackColor = 15132390
        End If
      End If
    Next
  Next
End Sub

Sub SetParStyle(oText, sParStyle As String)
  Dim oEnum
  Dim oPar
  oEnum = oText.createEnumeration()
  Do While oEnum.hasMoreElements()
    oPar = oEnum.nextElement()
    If opar.supportsService("com.sun.star.text.Paragraph") Then
      'oPar.ParaConditionalStyleName = sParStyle
      oPar.ParaStyleName = sParStyle
    End If
  Loop
End Sub
```

Макрос в Лист. 274 предполагает, что таблица простая. Если ячейка содержит текстовые таблицы, фреймы или секции, то они просто игнорируются. Когда будете писать свой макрос, Вам нужно будет решить, насколько гибким и универсальным будет Ваш макрос. Мои предположения сделаны для моих целей. Я знаю, что я буду использовать этот макрос только для простых текстовых таблиц, поэтому мой макрос сравнительно простой и короткий.

Если у Вас есть стиль автоформата текстовых таблиц, то отформатировать таблицу ещё проще. Объект текстовой таблицы поддерживает метод `autoFormat(имя)`, который принимает имя формата (стиля) в качестве аргумента. One might argue that the macro in

VI.2.3. Что такое сложная текстовая таблица

Сложная текстовая таблица — это таблица, не являющаяся простой. Точнее, сложная текстовая таблица содержит ячейки, которые разделены (между страницами) или объединены. Чтобы продемонстрировать сложную текстовую таблицу, начнем с Таблица 9 и выполним следующие операции для получения Таблица 11.

1) Правый клик (нажатие правой клавиши мыши) на ячейке A2 и выбираем Ячейка — разбить — по горизонтали (Cell > Split > Horizontal). Ячейка A2 становится двумя ячейками, остальные ячейки не переименовываются. В этот момент API показывает, что имеются четыре строки и ноль столбцов.

2) Правый клик на ячейке B2 и выбираем Ячейка — Разбить — по вертикали (Cell > Split > Vertical). Ячейка B2 становится двумя разными ячейками, B2 и C2. Ячейки справа от B2 также переименовываются.

Таблица 11. Сложная таблица после разделения ячеек

<table>
<tr><td>A1</td><td colspan="2">B1</td><td>C1</td><td>D1</td></tr>
<tr><td>A2=>A2.1.1</td><td rowspan="2">B2=>B2</td><td rowspan="2">B2=>C2</td><td rowspan="2">C2=>D2</td><td rowspan="2">D2=>E2</td></tr>
<tr><td>A2=>A2.1.2</td></tr>
<tr><td>A3</td><td colspan="2">B3</td><td>C3</td><td>D3</td></tr>
<tr><td>A4</td><td colspan="2">B4</td><td>C4</td><td>D4</td></tr>
</table>

3) Выделить ячейки A3 и B3, правый клик и выбрать Ячейка — Объединить (Cell > Merge). Заметьте, что эти две ячейки A3 и B3 теперь называются A3. Ячейки C3 и D3 также переименовываются.

Таблица 12. Сложная таблица после объединения ячеек в одной строке.

A1	B1		C1	D1
A2=>A2.1.1	B2=>B2	B2=>C2	C2=>D2	D2=>E2
A2=>A2.1.2				
A3, B3=>A3			C3=>B3	D3=>C3
A4	B4		C4	D4

4) Выделить ячейки C3=>B3 и C4, правый клик и выбрать Ячейка — Объединить (Cell > Merge).

Таблица 13. Сложная таблица после объединения ячеек в одном столбце

A1	B1		C1	D1
A2=>A2.1.1	B2=>B2	B2=>C2	C2=>D2	D2=>E2
A2=>A2.1.2				
A3, B3=>A3=>A3.1.1			(C3=>B3),C4=>B3	D3=>C3=>C3.1.1
A4=>A3.1.2	B4=>A3.2.2			D4=>C3.1.2

Это последнее объединение вызвало все возможные виды хаоса. Заметьте, что ячейки A4 и B4 теперь именуются A3.1.2 и A3.2.2. Похожая вещь случилась с ячейками C3 и D4 из Таблица 12. После шага 4 API показывает, что имеются три строки и ноль столбцов.

> ***Совет*** *Очевидно, текстовая таблица не может иметь ноль столбцов, так что это может быть удобным способом быстро определить, является ли текстовая таблица простой. С другой стороны, это, вероятно, не документированное поведение и в будущем это может не срабоать.*

VI.2.4. Перебор ячеек в любой текстовой таблице

В простой текстовой таблице можно использовать метод `getCellByPosition(col, row)` для извлечения ячейки, но это невозможно для сложной текстовой таблицы. Каждая текстовая таблица поддерживает метод `getCellByName()`, который возвращает ячейку на основе её имени. Используйте метод `getCellNames()` для получения массива имен ячеек, содержащихся в таблице.

Лист. 275: Перебрать ячейки в любой текстовой таблице.

```
Dim sNames()
Dim i As Long
Dim s As String
Dim oTable
Dim oVC
Dim oCell

oVC = ThisComponent.getCurrentController().getViewCursor()
If IsEmpty(oVC.TextTable) Then
  Print "Видимый курсор не в текстовой таблице"
  Exit Sub
End If

oTable = oVC.TextTable
sNames() = oTable.getCellNames()
For i = LBound(sNames()) To UBound(sNames())
  oCell = oTable.getCellByName(sNames(i))
  s = s & sNames(i) & " = " & oCell.getString() & CHR$(10)
Next
MsgBox s
```

Хотя порядок, в котором извлекаются ячейки, не прост, но этот метод работает для всех типов текстовых таблиц. Можно легко получить список имен ячеек следующим образом:

```
MsgBox Join(oTable.getCellNames(), CHR$(10))
```

VI.3. Извлечение данных из простой текстовой таблицы

Если нужно получить данные из текстовой таблицы, используйте методы `getDataArray()` и `getData()`, которые возвращают все данные таблицы в виде массива. Получение данных быстрее, чем перебор ячеек и затем извлечение данных из этих ячеек. Каждый метод извлечения данных имеет соответствующий метод установки значений данных, который может установить все данные за один раз.

Данные возвращаются в виде массива или массивов, а не в двумерном массиве. Следующий макрос выводит все числовые данные в первой строке таблицы.

Лист. 276: Вывод числовых данных

```
Dim oData() : oData() = oTable.getDataArray()
MsgBox Join(oData(0), CHR$(10))
```

Предположим, что текстовая таблица содержит две строки и три столбца, тогда следующий код установит значения данных в этой таблице.

```
Otable.setData(Array(Array(0, 1, 2), Array(3, 4, 5))
```

Используйте `getData()` и `setData()` для установки числовых данных. Предполагается, что все элементы (entries) — числа с двойной точностью Текстовые данные преобразуются в нули.

> ***Совет*** *Ячейка считается содержащей числовые данные в том и только том случае, если ячейка отформатирована как число.*

Используйте `getDataArray()` и `setDataArray()` для установки значений данных, которые содержат строки. В версии 2.01 строковые данные не извлекаются и не устанавливаются, но этот способ уже обещан (an issue has been filed).

VI.4. Курсоры таблицы и интервалы ячеек

Текстовые таблицы поддерживают метод `createCursorByCellName()`. Курсоры таблиц могут быть передвинуты и установлены аналогично их родственникам (counter-parts) — текстовым курсорам. Можно также использовать курсор таблицы для разделения и объединения ячеек. Видимый курсор действует как и курсор таблицы, если он находится внутри текстовой таблицы, и поэтому может использоваться для копирования текстовой таблицы или интервала ячеек в новую текстовую таблицу (см. мою книгу «OpenOffice.org Macros Explained» страницы 309 — 311).

VI.5. Интервалы ячеек (Cell ranges)

Большинство возможностей текстовой таблицы также поддерживается интервалом ячеек. Другими словами, если можно что-то сделать с объектом таблицы, то обычно это можно сделать и с объектом интервала ячеек. Например, можно извлечь и установить функции данных (data functions) на интервале ячеек. Можно отсортировать интервал ячеек и можно извлечь отдельную ячейку из интервала ячеек.

Используйте методы `getCellRangeByPosition()` и `getCellRangeByName()` для извлечения интервала ячеек из текстовой таблицы. Курсор таблицы может также выделить интревал ячеек, но курсор таблицы не является интервалом ячеек. Курсор таблицы поддерживает метод `getRangeName() (получить имя интервала)`, который может поэтому использоваться для извлечения этого интервала ячеек.

VI.5.1. Использование интервала ячеек для очистки ячеек

В документе OOo Calc интервал ячеек поддерживает метод очистки ячейки; интервал ячеек из текстовой таблицы не поддерживает. Можно обойти это следующим способом:

Лист. 277: Очистка данных текстовой таблицы.

```
Sub ClearCells()
  Dim oTable
  Dim oRange
  Dim oData()
  Dim oRow()
  Dim i%, j%

  REM  Получить ПЕРВУЮ текстовую таблицу
  oTable = ThisComponent.getTextTables().getByIndex(0)
  oRange = oTable.getCellRangeByName("B1:D4")
  oData() = oRange.getDataArray()
  For i = LBound(oData()) To UBound(oData())
    oRow() = oData(i)
    For j = LBound(oRow()) To UBound(oRow())
      oRow(j) = ""
    Next
  Next
  oRange.setDataArray(oData())
End Sub
```

Я не пытался сделать этот код более эффективным.

VI.6. Данные диаграммы (Chart data)

Не беспокойтесь об этой теме... Я бы не упоминал об этом, если бы не хотел что-то сделать с диаграммами.

ARRAY getColumnDescriptions (void)

ARRAY getRowDescriptions (void)

VOID setColumnDescriptions (ARRAY)

VOID setRowDescriptions (ARRAY)

VI.7. Ширина столбцов

Разделитель столбца (column separator) указывает, где заканчивается столбец в процентах от ширины таблицы. Позиция конца столбца 5000 означает ширину 50% от ширины таблицы. Макрос в Лист. 278 устанавливает, что первый столбец заканчивается на 50% ширины текущей таблицы, а второй столбец на 70% от общей ширины таблицы.

Таблица поддерживает это свойство, только если все строки имеют одинаковую структуру. Другими словами, нельзя установить ширину столбца для всей таблицы, если таблица сложная. Замечу ещё, что если данный разделитель столбца имеет значение флага IsVisible («видимый») равный false, то он невидим. Невидимые разделители нельзя переместить, и они не могут быть перекрыты видимыми разделителями.

Лист. 278: Установить ширину столбца для первых двух столбцов.

```
Sub SetTwoColsWidths
  Dim oTblColSeps  'Массив разделителей столбцов таблицы.
  Dim oTable       'Первая текстовая таблица в документе.

  'Печатаем
  oTable = ThisComponent.getTextTables().getByIndex(0)
  oTblColSeps = oTable.TableColumnSeparators

  Rem  Меняем позицию этих двух разделителей.
  oTblColSeps(0).Position = 5000
  oTblColSeps(1).Position = 7000

  REM  Нужно вернуть этот массив обратно в таблицу
  oTable.TableColumnSeparators = oTblColSeps
End Sub
```

VI.8. Установка оптимальной ширины столбца

В документе OOo Calc можно установить свойство столбца OptimalWidth равным True. Это не так просто для текстовой таблицы с использованием API.

GUI поддерживает метод, который может установить ширину столбца на основе положения текстового курсора, или выделенной части текстовой таблицы. Этот метод доступен через диспетчер (dispatch). В моей книге объяснено, как выделять области в текстовой таблице, поэтому я не буду повторять это обсуждение здесь. Макрос в Лист. 279 выделяет текстовую таблицу целиком и затем устанавливает ширину столбца для всей текстовой таблицы.

Лист. 279: Установить для всей таблицы оптимальную ширину столбцов.

```
Sub SetTableOptimumWidth
  Dim oDispHelper 'Dispatch helper
  Dim oFrame      'Текущий фрейм окна (window frame)
  Dim oTable      'Первая таблица в документе
  Dim oVCursor    'Видмый курсор
  Dim s$

  oTable = ThisComponent.getTextTables().getByIndex(0)
  ThisComponent.getCurrentController().select(oTable)
  oVCursor = ThisComponent.getCurrentController().getViewCursor()
  oVCursor.gotoEnd(True)
  oVCursor.gotoEnd(True)
  oFrame = ThisComponent.CurrentController.Frame
  oDispHelper = createUnoService("com.sun.star.frame.DispatchHelper")
  s$ = ".uno:SetOptimalColumnWidth"
  oDispHelper.executeDispatch(oFrame, s, "", 0, Array())
End Sub
```

VI.9. Насколько широка текстовая таблица?

На первый взгляд, очень просто определить ширину текстовой таблицы; используем свойство Width. К сожалению, этого недостаточно. Чтобы определить реальную ширину текстовой таблицы,нужно исследовать свойства, показанные в Таблица 14.

Таблица 14: Свойства текстовой таблицы, связанные с шириной текстовой таблицы.

Свойство	*Описание*
LeftMargin	Левое поле таблицы
RightMargin	Правое поле таблицы
HoriOrient	Содержит горизонтальную ориентацию из группы констант com.sun.star.text.HoriOrientation
RelativeWidth	Определяет ширину таблицы относительно её окружения.
IsWidthRelative	Определяет, действует ли относительная ширина.
Width	Иногда содержит абсолютную ширину таблицы

В моей книге (стр. 307) я упоминаю HoriOrient как управляющее значение для всех остальных свойств. Если свойство HoriOrient содержит значение по умолчанию FULL, то остальные свойства, по существу, бесполезны, включая свойство Width. В этом случае нужно определять ширину столбца, который содержит текстовую таблицу. У меня недостаточно времени объяснить это подробнее.

VI.10. Курсор в текстовой таблице

Можно проверить, находится ли курсор внутри текстовой таблицы, используя макрос в Лист. 191, который также демонстрирует, как определить случай, когда выделено больше одной ячейки. Макрос в Лист. 280 печатает информацию о текстовом курсоре внутри текстовой таблицы.

Лист. 280: Проверка текстовой таблицы, в которой находится курсор.

```
Sub InspectCurrentTable
  Dim oTable  'Таблица, в которой находистя курсор
  Dim oCell   'Ячейка, в которой находистя курсор
  Dim oVCurs  'Текущий видимый курсор
  Dim s$      'Содержит поясняющий текст
  Dim nCol%   'Столбец, который содержит курсор
  Dim nRow%   'Строка, которая содержит курсор
```

```
  oVCurs = ThisComponent.getCurrentController().getViewCursor()
  If IsEmpty(oVCurs.TextTable) Then
    s = s & "Курсор НЕ в текстовой таблице"
  Else
    oTable = oVCurs.TextTable
    oCell  = oVCurs.Cell
    REM  Предположим, что число столбцов меньше 26
    nCol = Asc(oCell.Cellname) - 65
    nRow = CInt(Right(oCell.Cellname, Len(oCell.Cellname) - 1)) - 1
    s = s & "Курсор в текстовой таблице " & _
            oTable.getName() & CHR$(10) & _
            "Текущая ячейка - это " & oCell.Cellname & CHR$(10) & _
            "Эта ячейка расположена: (" & nCol & ", " & nRow & ")" & _
            CHR$(10) "В таблице " & _
            oTable.getColumns().getCount() & " столбцов и " & _
            oTable.getRows().getCount() & " строк" & CHR$(10)
    REM ***  Можете здесь дописывать  макрос!
  End If
  MsgBox s
End Sub
```

Чтобы выделить строку целиком, сначала передвиньте курсор на первую ячейку в строке:

Лист. 281: Передвинуть курсор на первую ячейку таблицы.

```
    oCell1 = oTable.getCellByPosition(0, nRow)
    ThisComponent.getCurrentController().select(oCell1)
```

Затем выделите ячейку целиком. Если ячейка содержит текст, то оператор oVCurs.gotoEnd(True) передвинет курсор к концу этого текста в текущей ячейке. Если курсор уже находится в конце ячейки, например, если ячейка не содержит текста совсем, то куосор передвинется на конец всей таблицы. Метод gotoEndOfLine() всегда передвигает курсор на конец текущей строки, что удобно, если ячейка содержит только одну строку текста. После выделения ячейки целиком можно передвинуть видимый курсор направо для выделения остальных ячеек. Макрос в Лист. 282 написан для вставки в макрос в Лист. 280.

Лист. 282: Выделить строку целиком в простой текстовой таблице.

```
Dim oCell1
Dim oText
oCell1 = oTable.getCellByPosition(0, nRow)
ThisComponent.getCurrentController().select(oCell1)
oText = oCell1.getText()
Dim oStart : oStart = oText.getStart()
If oText.compareRegionStarts(oStart(), oText.getEnd()) <> 0 Then
  oVCurs.gotoEnd(True)
End If
oVCurs.goRight(oTable.getColumns().getCount()-1, True)
```

VI.10.1. Установить курсор после текстовой таблицы

Сначала предположим, что видимый курсор находится в текстовой таблице. Используя хитрость, которуя я описал в своей книге, я могу легко передвинуть видимый курсор на позицию после текущей текстовой таблицы.

Лист. 283: Передвинуть видимый курсор после текущей текстовой таблицы.

```
Sub CursorAfterCurrentTable()
  Dim oCursor

  REM  Получить видимый курсор
  oCursor = ThisComponent.getCurrentController().getViewCursor()

  REM  Проверить, что курсор внутри текстовой таблицы
  If IsNull(oCursor.TextTable) OR IsEmpty(oCursor.TextTable) Then
    Print "Не выделена текстовая таблица"
    Exit Sub
  End If

  REM  Теперь пердвигаемся к последней ячейке таблицы (см. подробнее
пояснения в моей книге)
  REM  Это работает, ТОЛЬКО если текущая ячейка не пуста.
  'oCursor.gotoEnd(False)
  'oCursor.gotoEnd(False)
  'oCursor.goDown(1, False)
  REM  Это решение предложил JohnV на OooForum.
  REM  Могут быть проблемы, если две таблицы идут подряд
  REM  или если одна таблица содержит другую. Можно легко обойти эти
  REM  проблемы, если потребуется
  While Not IsEmpty(oCursor.TextTable)
    oCursor.goDown(1,False)
  Wend
End Sub
```

Передвижение курсора на позицию после заранее заданной таблицы также очень легко. Следующий пример преполагает, что по крайней мере одна текстовая таблица имеется:

Лист. 284: Передвинуть видимый курсор после заданной таблицы.

```
Sub CursorAfterFirstTable()
  Dim oTable
  REM  Получить ПЕРВУЮ текстовую таблицу
  oTable = ThisComponent.getTextTables().getByIndex(0)
  REM  Передвигаем курсор на первую ячейку таблицы
  ThisComponent.GetCurrentController().Select(oTable)
  REM Передвигаемся ПОСЛЕ текущей таблицы
  CursorAfterCurrentTable()
End Sub
```

VI.11. Создание текстовой таблицы

Следующий макрос демонстрирует обработку текстовых таблиц и содержащихся в них ячеек

Лист. 285: Создать текстовую таблицу и вставить в нее ячейки

```
'Author: Hermann Kienlein
'Author: Christian Junker
Sub easyUse( )
 Dim odoc, otext, ocursor, mytable, tablecursor
 odoc = thisComponent
 otext = odoc.getText()
 mytable = CreateTable(odoc)

 'Создаем нормальный текстовый курсор TextCursor
 ocursor = otext.CreateTextCursor()
 ocursor.gotoStart(false)

 'теперь определен интервал, равный позиции в таблице, давайте вставим его
(now that we defined the range = position of the table, let's insert it)
 otext.insertTextContent(ocursor, myTable, false )
 tablecursor = myTable.createCursorByCellName("A1")

 InsertNextItem("first cell", tablecursor, mytable) 'Вставляем новый
элемент:
 InsertNextItem("second cell", tablecursor, mytable) 'и ещё один:
End Sub

Sub InsertNextItem(what, oCursor, oTable)
  Dim oCell As Object
  'задаем имя интервала ячеек, которые выделены данным курсором
  sName = oCursor.getRangeName()
  '  имя ячейку будет что-то вроде D3
  oCelle = oTable.getCellByName(sName)
  oCelle.String = what
  oCursor.goRight(1,FALSE)
End Sub

Function CreateTable(document) As Object
  oTextTable = document.createInstance("com.sun.star.text.TextTable")
  oTextTable.initialize(5, 8)
  oTextTable.HoriOrient = 0 'com.sun.star.text.HoriOrientation::NONE
  oTextTable.LeftMargin = 2000
  oTextTable.RightMargin = 1500
  CreateTable = oTextTable
End Function

Sub deleteTables()
  'Иногда удаление таблиц в GUI кажется видом глупости,
  'но все же эта процедура удалит все таблицы
  Dim enum, textobject
  enum = thisComponent.Text.createEnumeration
```

```
  While enum.hasMoreElements()
    txtcontent = enum.nextElement()
    If txtcontent.supportsService("com.sun.star.text.TextTable") Then
      thisComponent.Text.removetextcontent(txtcontent)
    End If
  Wend
End Sub
```

VI.12. Таблица без рамок границы (borders)

Lalaimia Samia [samia.lalaimia@infotel.com] спрашивала меня, как вставить таблицу в документ, который не содержит границ (borders). Нельзя обрабатывать границы таблицы до тех пор, пока оне не будет вставлена. Следующая проблема в том, что каждое из свойств является структурой, поэтому невозможно напрямую изменять структуру, потому что фактически Вы работаете с её копией. Ответом будет скопировать структуру во временную переменную, модифицировать эту переменную, а затем скопировать эту временную переменную обратно. Это скучновато, но просто.

Лист. 286: Вставить текстовую таблицу без рамок границы.

```
Sub InsertATableWithNoBorders
  Dim oTable   'Новая таблица для вставки
  Dim oEnd

  REM  Пусть документ создаст текстовую таблицу
  oTable = ThisComponent.createInstance( "com.sun.star.text.TextTable" )
  oTable.initialize(4, 1) 'четыре строки, один столбец

  REM  Теперь вставляем текстовую таблицу в конец документа
  Oend = ThisComponent.Text.getEnd()
  ThisComponent.Text.insertTextContent(oEnd, oTable, False)

  Dim x 'представляет каждый экземпляр BorderLine
  Dim v 'представляет объект рамок границ таблицы TableBorder в целом
  v = oTable.TableBorder
  x = v.TopLine        : x.OuterLineWidth = 0 : v.TopLine = x
  x = v.LeftLine       : x.OuterLineWidth = 0 : v.LeftLine = x
  x = v.RightLine      : x.OuterLineWidth = 0 : v.RightLine = x
  x = v.TopLine        : x.OuterLineWidth = 0 : v.TopLine = x
  x = v.VerticalLine   : x.OuterLineWidth = 0 : v.VerticalLine = x
  x = v.HorizontalLine : x.OuterLineWidth = 0 : v.HorizontalLine = x
  x = v.BottomLine     : x.OuterLineWidth = 0 : v.BottomLine = x

  oTable.TableBorder = v
  Dim a()
  a() = Array(Array("Files"), Array("One"), Array("Two"), Array("Three"))
  oTable.setDataArray(a())
End Sub
```

Глава VII. Форматирование макроса

Я люблю форматировать мои макросы с использованием цвета, так что они похожи по оформлению на те, что показываются в средстве отладки макросов IDE. Данный раздел содержит утилиты и методы, которые я для этого использую.

VII.1. Утилиты для строк и массивов

Этот макрос перебирает строки, определяя в них разные типы символов. Наиболее быстрый способ для этого — использование просмотра массив. Я предполагаю, что коды символов ASCII находятся в интервале от 0 до 256, что не всегда надёжно, но я обрабатываю именно так в данном макросе.

Лист. 287: Начало модуля для строковых утилит форматирования макросов.

```
Option Explicit
Private bCheckWhite(0 To 256) As Boolean
Private bCheckSpecial(0 To 256) As Boolean
private bWordSep(0 To 256) As Boolean
```

Каждое значение установлено равным False, и затем значения для специальных символов явным образом устанавливаются равными True.

Лист. 288: Инициализировать массивы специальных символов.

```
'Инициализировать переменные, которые содержат специальные симоволы
Sub FMT_InitSpecialCharArrays()
  Dim i As Long

  For i = LBound(bCheckWhite()) To UBound(bCheckWhite())
    bCheckWhite(i)   = False
    bCheckSpecial(i) = False
    bWordSep(i)      = False
  Next

  bCheckWhite(9)   = True
  bCheckWhite(10)  = True
  bCheckWhite(13)  = True
  bCheckWhite(32)  = True
  bCheckWhite(160) = True
  bCheckSpecial(Asc("+")) = True
  bCheckSpecial(Asc("-")) = True
  bCheckSpecial(Asc("&")) = True
  bCheckSpecial(Asc("*")) = True
  bCheckSpecial(Asc("/")) = True
  bCheckSpecial(Asc(":")) = True
  bCheckSpecial(Asc("=")) = True
  bCheckSpecial(Asc("<")) = True
  bCheckSpecial(Asc(">")) = True
  bCheckSpecial(Asc("(")) = True
  bCheckSpecial(Asc(")")) = True
  bCheckSpecial(Asc(",")) = True

  For i = LBound(bCheckWhite()) To UBound(bCheckWhite())
```

```
    bWordSep(i) = bCheckWhite(i) OR bCheckSpecial(i)
  Next
  bWordSep(ASC(".")) = True
End Sub
```

Специальные функции используются для проверки статуса специального символа. Большинство этих функций принимают аргументом целое число, которое соответствует ASCII коду проверяемого симовла.

Таблица 15: Функции, используемые для выявления специальных символов.

Функция	***Описание***
IsWhiteSpace(Integer)	True, если символ является пробелом или знаком табуляции
FMT_IsSpecialChar(Integer)	True , если символ является специальным, таким как *,;&.
FMT_IsWordSep(Integer)	True, если специальный символ или разновидность пробела (white space).
FMT_IsDigit(Integer)	True, если десятичная цифра [0 — 9].
FMT_StrIsDigit(String)	True, если первый символ строки является цифрой

Функции в Таблица 15 вставлены в Лист. 289. Большинство этих функций используют просмотр массива для скорости. Обработка ошибок используется в случае, когда код ASCII слишком велико (например, UNICODE). Цифра проверяется напрямую сравнением с кодами ASCII символов '0' и '9'. Целью этих функций было сделать выполнение более быстрым.

Лист. 289: Функции для проверки на специальные симоволы

```
'Просмотр массива — более быстрый способ
'Здесь предполагается, что получен символ ASCII, что я не смог
'сделать в версии OOo 2.0. Смог я это сделать в версии 2.01.
Function FMT_IsWhiteSpace(iChar As Integer) As Boolean
  On Error Resume Next
  FMT_IsWhiteSpace() = False
  FMT_IsWhiteSpace() = bCheckWhite(iChar)
'  Select Case iChar
'  Case 9, 10, 13, 32, 160
'    iIsWhiteSpace = True
'  Case Else
'    iIsWhiteSpace = False
'  End Select
End Function

'Возвращает значение true, если символ является специальным символом
Function FMT_IsSpecialChar(iChar As Integer) As Boolean
  On Error Resume Next
```

```
  FMT_IsSpecialChar() = False
  FMT_IsSpecialChar() = bCheckSpecial(iChar)
End Function

'Возвращает значение true, если символ является разделителем слов
Function FMT_IsWordSep(iChar As Integer) As Boolean
  On Error Resume Next
  FMT_IsWordSep() = False
  FMT_IsWordSep() = bWordSep(iChar)
End Function

'Является ли символ одной из цифр 0, 1, 2, ..., или 9?
Function FMT_IsDigit(iChar As Integer) As Boolean
  FMT_IsDigit() = (48 <= iChar and iChar <= 57)
End Function

'Является ли первый символ строки одной из цифр 0, 1, 2, ..., или 9?
Function FMT_StrIsDigit(s$) As Boolean
  FMT_StrIsDigit = FMT_IsDigit(ASC(s))
End Function
```

VII.1.1. Специальные символы и числа в строках

Специальные фунции используются для быстрого просмотра текста для того, чтобы найти «интересный текст». При поиске кавычки символ, который ищется, берется из строки в местоположении iPos (второй аргумент). Аргумент iPos увеличивается, пока не будет достигнут конец строки.

Таблица 16: Функции, используемые для поиска следующего подходящего специального символа.

Функция	***Описание***
FMT_FindNextNonSpace	Используется для быстрого пропуска всех видов пробелов, начиная с позиции iPos.
FMT_FindEndQuote	Поиск следующего символа кавычки.
FMT_FindEndQuoteEscape	Поиск следующего символа кавычки, игнорируя любой символ, перед которым стоит обратный слэш (\).
FMT_FindEndQuoteDouble	Поиск следующего символа кавычки, игнорируя любой символ, после которого идёт символ кавычки. Например, текст «""» найдёт все символы кавычки (Search for the next quote character, ignoring any character followed by a quote character. For example, the text «""» finds all quote characters.)
FMT_FindNumberEnd	Выделяет число (Identify a number.)

Лист. 290: Функции для поиска следующего подходящего специального символа.

```
'Увеличиваем iPos до тех пор, пока он не укажет на первый символ, не
'являющийся разновидностью пробела или пока не встанет после конца строки
'текста
Sub FMT_FindNextNonSpace(sLine$, iPos%, iLen%)
  If iPos <= iLen Then
    REM  Располагаем курсор ПОСЛЕ разновидности пробела (white space)
    Do While FMT_IsWhiteSpace(Asc(Mid(sLine, iPos, 1)))
      iPos = iPos + 1
      If iPos > iLen Then Exit Do
    Loop
  End If
End Sub

'Увеличиваем iPos, пока он не установится после закрывающей кавычки
'В строке не должно быть символов кавычки
Sub FMT_FindEndQuote(s$, iPos%, iLen%)
  Dim sQuote$  : sQuote = Mid(s, iPos, 1)

  iPos = iPos + 1
  If iPos <= iLen Then
    Do While Mid(s, iPos, 1) <> sQuote
      iPos = iPos + 1
      If iPos > iLen Then Exit Do
    Loop
    Rem iPos might point two past the string...
    iPos = iPos + 1
  End If
End Sub

'Увеличиваем iPos до тех пор, пока он не встанет после закрывающей кавычки
'Кавычка со стоящим перед ней обратным слэшем \ игнорируется (escapes it).
'Закрывающей кавычкой считается символ, равный символу в позиции iPos
'при входе в процедуру
Sub FMT_FindEndQuoteEscape(s$, iPos%, iLen%)
  Dim sQuote$  : sQuote = Mid(s, iPos, 1)
  Dim sCur$

  iPos = iPos + 1
  If iPos <= iLen Then
    Do
      sCur = Mid(s, iPos, 1)
      iPos = iPos + 1
      If sCur = "\" Then iPos = iPos + 1
      If iPos > iLen Then Exit Do
    Loop Until sCur = sQuote
  End If
End Sub

'Увеличиваем iPos до тех пор, пока он не встанет после закрывающей кавычки
'Две кавычки подряд игнорируются
Sub FMT_FindEndQuoteDouble(s$, iPos%, iLen%)
  Dim sQuote$  : sQuote = Mid(s, iPos, 1)
```

```
  iPos = iPos + 1
  Do While iPos <= iLen
    If Mid(s, iPos, 1) <> sQuote OR iPos = iLen Then
      iPos = iPos + 1
    Else
      REM  iPos  указывает на кавычку, при этом конец
      REM  строки ещё не достигнут
      iPos = iPos + 1
      If Mid(s, iPos, 1) <> sQuote Then Exit Do
      REM  Были две кавычки подряд, игнорируем их
      iPos = iPos + 1
    End If
  Loop
  REM  Результат не должен быть больше позиции за концом строки
  If iPos > iLen Then iPos = iLen + 1
End Sub

'Эта процедура вызывается в том и только том случае, когда oCurs указывает
'на позицию с числом.
'
'Определить, указывает ли курсор на число. Если да, то устанавливаем
'iEnd  на символ, следующий за концом этого числа, и возвращаем True
'Если нет, то просто возращаем False.
'Числом считается следующее:
'— число может начинастья с цифры или десятичной точки.
'— число (перед показателем степени) может иметь  десятичную точку, но
'только одну.
'-e  или f принимаются как признак перед началом показателя степени .
'— показатель степени может начинаться (после e,f) со знака + или -.
'— показатель степени также может содержать десятичную точку.
Function FMT_FindNumberEnd(sLine$, iPos%, iLen%, iEnd%) As Boolean
  Dim sChar$
  Dim bDecimal As Boolean

  iEnd = iPos
  bDecimal = False
  sChar = ""
  'пропускаем ведущие цифры
  Do While FMT_IsDigit(ASC(Mid(sLine, iEnd, 1)))
    iEnd = iEnd + 1
    If iEnd > iLen Then Exit do
  Loop

  'Теперь проверяем на десятичную точку
  If iEnd <= iLen Then
    If Mid(sLine, iEnd, 1) = "." Then
      iEnd = iEnd + 1
      bDecimal = True
      REM  пропускаем цифры после десятичной точки
      Do While iEnd <= iLen
        If NOT FMT_IsDigit(ASC(Mid(sLine, iEnd, 1))) Then Exit Do
        iEnd = iEnd + 1
      Loop
    End If
  End If
```

```
  'если в числе была только  ".", то  iEnd = iPos + 1
  If (bDecimal and iEnd = iPos + 1) OR (iEnd = iPos) Then
    FMT_FindNumberEnd() = False
    Exit Function
  End If

  'This is a number, now look for scientific notation.
  FMT_FindNumberEnd() = True
  If iEnd <= iLen Then
    sChar = Mid(sLine, iEnd, 1)
    If sChar <> "f" and sChar <> "e" Then Exit Function
    iEnd = iEnd + 1
  End If

  'это экспоненциальная (scientific) запись, поэтому проверяем на + или -.
  If iEnd <= iLen Then sChar = Mid(sLine, iEnd, 1)
  If sChar = "+" OR sChar = "-" Then iEnd = iEnd + 1

  'пропускаем ведущие цифры показателя степени
  Do While iEnd <= iLen
    If NOT FMT_IsDigit(ASC(Mid(sLine, iEnd, 1))) Then Exit Do
    iEnd = iEnd + 1
  Loop

  'Теперь проверяем на десятичную точку
  If iEnd <= iLen Then
    If Mid(sLine, iEnd, 1) = "." Then
      iEnd = iEnd + 1
      'пропускаем завершающие цифры (после десятичной точки в показателе
      'степени)
      'они могут быть только нулями или отсутствовать совсем
      Do While iEnd <= iLen
        If NOT FMT_IsDigit(ASC(Mid(sLine, iEnd, 1))) Then Exit Do
        iEnd = iEnd + 1
      Loop
    End If
  End If
End Function
```

Я использую следующий макрос для проверки моих специальных функций. Я использую его же для оценки времени выполнения этих функций для целей оптимизации программы.

Лист. 291: Макрос тестирования специальных функций

```
Sub TestSpecialChars
  Dim ss$
  Dim n%
  Dim s$    : s    = "how    are you?"
  Dim iPos% : iPos = 4
  Dim iLen% : iLen = Len(s)
  Dim x
  Dim i       As Long
  Dim nIts    As Long
```

```
  Dim nMinIts As Long  : nMinIts = 1000
  Dim nNow    As Long
  Dim nNum    As Long  : nNum    = 2000
  Dim nFirst  As Long  : nFirst  = GetSystemTicks()
  Dim nLast   As Long  : nLast   = nFirst + nNum
  Dim iEnd%
  Dim b As Boolean

  ss = "1.3f+3.2x"
  iPos = 1
  b = FMT_FindNumberEnd(ss$, iPos%, Len(ss), iEnd%)
  Print ss & " ==> " & b & " end = " & iEnd
  'Exit Sub

  FMT_InitSpecialCharArrays()

  Do
    For i = 1 To nMinIts
      FMT_FindNextNonSpace(s, iPos%, iLen%)
    Next
    nIts = nIts + 1
    nNow = GetSystemTicks()
  Loop Until nNow >= nLast
  MsgBox "Завершено после " & CStr(nIts * nMinIts) & _
         " итераций " & CHR$(10) & _
         CStr(CDbl(nIts * nMinIts) * 1000 / CDbl(nNow - nFirst) & _
         " итераций в секунду"
End Sub
```

VII.1.2. Массивы строк

Этот макрос создаёт списки строк, в которых нужно выполнить поиск. Первая процедура используется для сортировки массива. Метод пузырьковой сортировки (bubble sort) достаточно быстр и достаточно прост. Данный поиск, как бы то ни было, использует двоичный поиск. Двоичный поиск (в том виде, как он используется ниже) будет быстрее для данной задачи, но для маленьких массивов линейный поиск, похоже, будет быстрее.

Лист. 292: Функция массивов

```
'Сортируем массив sItems() в возрастающем порядке
'Этот алгоритм прост и не эффективен.
'В худщем случае время выполнения пропорционально квадрату n.  Если Вы
'посчитаете точнее, то получите время выполнения(n^2-4n+1)/2, неважно, что
'это означает
Sub FMT_SortStringArrayAscending(sItems())
  Dim i As Integer        'Внешний индекса
  Dim j As Integer        'Внутренний индекс
  Dim s As String         'Временная переменная для обмена значений
  Dim bChanged As Boolean 'Становится True если хоть что-то изменилось
  For i = LBound(sItems()) To UBound(sItems()) - 1
    bChanged = False
    For j = UBound(sItems()) To i+1 Step -1
      If sItems(j) < sItems(j-1) Then
```

```
        s = sItems(j) : sItems(j) = sItems(j-1) : sItems(j-1) = s
        bChanged = True
      End If
    Next
    If Not bChanged Then Exit For
  Next
End Sub

'определяем, содержит ли массив заданную строку
'хотя двоичный поиск быстрее для больших массивов,
'но для маленьких массивов линейный поиск может быть быстрее
Function FMT_ArrayHasString(s$, sItems()) As Boolean
  Dim i   As Integer
  Dim iUB As Integer
  Dim iLB As Integer
  FMT_ArrayHasString() = False

  iUB = UBound(sItems())
  iLB = LBound(sItems())
  Do
    i = (iUB + iLB) \ 2
    If sItems(i) = s Then
      FMT_ArrayHasString() = True
      iLB = iUB + 1
      'Exit Do
    ElseIf sItems(i) > s Then
      iUB = i – 1
    Else
      iLB = i + 1
    End If
  Loop While iUB >= iLB
End Function
```

VII.2. Утилиты для поиска разделов/секций с кодами макросов

Когда я вставляю макрос в документ, каждая строка является отдельным параграфом. Конкретные стили параграфа идентифицируют параграф как «строку кода программы». После помещения курсора в текст, содержащий макрос, мне необходимо найти первую и последнюю строку этого макроса.

Первый нижеприведенный макрос принимает как параметры текстовый курсор и список имён, используемых для идентификации параграфа в качестве кода макроса. Этот курсор передвигается назад по параграфам до тех пор, пока будет найден параграф, не являющийся кодом программы. Если нужно, этот курсор затем передвигается на один параграф вперед, чтобы попасть на параграф с кодом программы. Последний параграф, содержащий код программы, находится аналогичным способом.

Лист. 293: Найти область вокруг курсора, содержащую код программы.

```
'Передвигаем курсор на первый параграф, использующий заданные стили кода
'программы
Sub FMT_CursorToFirstCodeParagraph(oCurs, sStyles())

  oCurs.gotoStartOfParagraph(False)
'  Эта проверка должна быть выполнена до вызова данной процедуры:
'  If NOT FMT_ArrayHasString(oCurs.ParaStyleName, sStyles()) Then
'    Print "Текстовый курсор должен находиться внутри кода программы"
'    Exit sub
'  End If

  REM  находим первый параграф с кодом программы
  Do While FMT_ArrayHasString(oCurs.ParaStyleName, sStyles())
    If NOT oCurs.gotoPreviousParagraph(False) Then Exit Do
  Loop

  If NOT FMT_ArrayHasString(oCurs.ParaStyleName, sStyles()) Then
    oCurs.gotoNextParagraph(False)
  End If
  oCurs.gotoStartOfParagraph(False)
End Sub

'передвигаем курсор на последний параграф со стилем кода программы
Sub FMT_CursorToLastCodeParagraph(oCurs, sStyles())

  REM  найти последний параграф, который имеет стиль кода программы
  Do While FMT_ArrayHasString(oCurs.ParaStyleName, sStyles())
    If NOT oCurs.gotoNextParagraph(False) Then Exit Do
  Loop

  If NOT FMT_ArrayHasString(oCurs.ParaStyleName, sStyles()) Then
    oCurs.gotoPreviousParagraph(False)
  End If
  oCurs.gotoEndOfParagraph(False)
End Sub

'Для заданного курсора выбрать область вокруг этого курсора, содержащую
'параграфы заданных стилей
Sub FMT_FindCodeAroundCursor(oCurs, sStyles())
  Dim oEndCurs

  REM  найти последний параграф имеющий стиль кода программы
  oEndCurs = oCurs.getText().createTextCursorByRange(oCurs)
  FMT_CursorToLastCodeParagraph(oEndCurs, sStyles())
  FMT_CursorToFirstCodeParagraph(oCurs, sStyles())

  REM  выделить все целиком и затем отформатировать
  oCurs.gotoRange(oEndCurs, True)
End Sub
```

Последний макрос в Лист. 293 используется для выделения целого текста макроса. Находим первый и последний параграфы, а затем курсор выделяет

весь интервал текста целиком. Разумеется, предполагаем, что курсор начинается в параграфе с кодом программы.

VII.3. Основной модуль макроса

Цвета, использованные для каждой части кода программы, задаются с помощью стилей символов. Предполагается, что эти стили имеются в документе и они указаны в начале модуля.

Лист. 294: Начало основного модуля макроса форматирования

```
OPTION Explicit
Private Const sCharStyleComment    = "_OOoComputerComment"
Private Const sCharStyleLiteral    = "_OOoComputerLiteral"
Private Const sCharStyleKeyWord    = "_OOoComputerKeyWord"
Private Const sCharStyleIdentifier = "_OOoComputerIdent"
Private Const sCharStyleBase       = "_OooComputerBase"
```

Ключевые слова и стили параграфов хранятся в отсортированных массивах. Эти массивы созданы с использованием процедур. Поэтому легко модифицировать код макроса, чтобы он распознавал новые характерные элементы.

Лист. 295: Инициализируем массивы ключевых слов и стилей параграфов

```
'Следующие слова являются ключевыми словами, распознаваемые отладчиком
'Basic IDE.
'Этот список отсортирован в алфавитном порядке. Список загружается из
'файла: basic/source/comp/tokens.cxx.
'Многословные ключевые элементы, такие как "end enum" , нежелатилеьны,
'так как данный макрос будет распознавать только одиночные слова.  Оба
'слова "end enum" уже есть в списке, поэтому даже в самом худшем случае
'это просто замедлит поиск, потому что там есть дополнительные слова
Sub FMT_InitTokensBasic(sTokens())
  sTokens() = Array("access", _
    "alias", "and", "any", "append", "as", _
    "base", "binary", "boolean", "byref", "byval", _
    "call", "case", "cdecl", "classmodule", "close", _
    "compare", "compatible", "const", "currency", _
    "date", "declare", "defbool", "defcur", "defdate", _
    "defdbl", "deferr", "defint", "deflng", "defobj", _
    "defsng", "defstr", "defvar", "dim", "do", "double", _
    "each", "else", "elseif", "end", _
    "end enum", "end function", "end if", "end property", _
    "end select", "end sub", "end type", _
    "endif", "enum", "eqv", "erase", "error", _
    "exit", "explicit", _
    "for", "function", _
    "get", "global", "gosub", "goto", _
    "if", "imp", "implements", "in", "input", _
    "integer", "is", _
    "let", "lib", "line", "line input", "local", _
```

```
    "lock", "long", _
    "loop", "lprint", "lset", _
    "mod", _
    "name", "new", "next", "not", _
    "object", "on", "open", "option", _
    "optional", "or", "output", _
    "paramarray", "preserve", "print", _
    "private", "property", "public", _
    "random", "read", "redim", "rem", "resume", _
    "return", "rset", _
    "select", "set", "shared", "single", "static", _
    "step", "stop", "string", "sub", "system", _
    "text", "then", "to", "type", "typeof", _
    "until", _
    "variant", _
    "wend", "while", "with", "write", _
    "xor")
    FMT_SortStringArrayAscending(sTokens())
End Sub

' Тесты программ форматируются с использованием определенных стилей
' параграфов. Соответствующие стили параграфов перечислены ниже
Sub FMT_InitParStyles(sStyles())
  sStyles() = Array( "_OOoComputerCode", _
   "_OOoComputerCodeLastLine", _
   "_code", _
   "_code_first_line", _
   "_code_last_line", _
   "_code_one_line")
   FMT_SortStringArrayAscending(sStyles())
End Sub
```

VII.3.1. Первичные точки входа

Чаще всего используемый (мной) макрос добавляет выделение цветом для текста программы, содержащего видимый курсор. Этот макрос очень прост

1. Инициализируем массивы
2. Проверяем, что видимый курсор находится в тексте программы
3. Находим первый и последний параграф (см. Лист. 293).
4. Используем основной макрос (см. Лист. 299).

Основная работа выполняется в главном рабочем макросе (main worker macro)

Лист. 296: Подчеркнуть текущий макрос.

```
'Выделить цветом текст программы на языке Basic, окружающий видимый курсор
Sub FMT_ColorCodeCurrentBasic()
  Dim oVCurs
  Dim oCurs
  Dim sStyles()     ' Стили параграфа для текста программы
  Dim sTokens()     ' Ключевые слова/комбинации (tokens).
```

```
  FMT_InitSpecialCharArrays()
  FMT_InitParStyles(sStyles())
  FMT_InitTokensBasic(sTokens())

  REM  Получить видимый курсор в качестве начальной позиции
  oVCurs = ThisComponent.getCurrentController().getViewCursor()
  oCurs  = oVCurs.getText().createTextCursorByRange(oVCurs)
  If NOT FMT_ArrayHasString(oCurs.ParaStyleName, sStyles()) Then
    Print "Текстовый курсор должен быть в сегменте с кодом программы"
    Exit sub
  End If
  FMT_FindCodeAroundCursor(oCurs, sStyles())
  FMT_ColorCodeOneRangeStringBasic(oCurs, sTokens())
End Sub
```

Найдите и отформатируйте все макросы в текущем документе. Сначала сохраните этот документ на случай серьезной ошибки.

Лист. 297: Выделить цветом все макросы в документе

```
'Выделить все тексты программ в данном документе
Sub HighlightDoc()
  Dim oCurs
  Dim oStartCurs
  Dim oEndCurs
  Dim bFoundCompStyle As Boolean
  Dim sStyles()
  Dim sTokens()

  FMT_InitSpecialCharArrays()
  FMT_InitParStyles(sStyles())
  FMT_InitTokensBasic(sTokens())
  bFoundCompStyle = False

  oCurs = ThisComponent.getText().createTextCursor()
  oStartCurs = ThisComponent.getText().createTextCursor()
  oEndCurs = ThisComponent.getText().createTextCursor()
  oEndCurs.gotoStart(False)
  Do
    If FMT_ArrayHasString(oEndCurs.ParaStyleName, sStyles()) Then
      If NOT bFoundCompStyle Then
        bFoundCompStyle = True
        oCurs.gotoRange(oEndCurs, False)
        oCurs.gotoEndOfParagraph(True)
      Else
        oCurs.gotoNextParagraph(True)
        oCurs.gotoEndOfParagraph(True)
      End If
    Else
      If bFoundCompStyle Then
        bFoundCompStyle = False
        FMT_ColorCodeOneRangeStringBasic(oCurs, sTokens())
      End If
    End If
  Loop While oEndCurs.gotoNextParagraph(False)
```

```
  If bFoundCompStyle Then
    bFoundCompStyle = False
    FMT_ColorCodeOneRangeStringBasic(oCurs, sTokens())
  End If
End Sub
```

Выделите цветом текст программы только тех параграфов, которые выделены. Главный рабочий макрос передвигается по одному параграфу за раз, так что последний параграф не может быть обработан, если он не выделен целиком.

Лист. 298: Выделить

```
'форматировать только выделенный текст
Sub HighlightSel()
  Dim oSels
  Dim oSel
  Dim i%
  Dim sTokens()

  FMT_InitSpecialCharArrays()
  FMT_InitTokensBasic(sTokens())
  oSels = ThisComponent.getCurrentController().getSelection()
  For i = 0 To oSels.getCount() - 1
    oSel = oSels.getByIndex(i)
    FMT_ColorCodeOneRangeStringBasic(oSel, sTokens())
  Next
End Sub
```

VII.3.2. Основной рабочий макрос

В нем выполняется основной объем работы. Весь выделенный текст считается кодом программы, поэтому стили параграфов уже не проверяются. Текстовые курсоры, используемые для применения форматирования, продвигаются по выделенному тексту по одному параграфу за раз.

Лист. 299: Выделение цветом

```
'Пометить цветом код программы на Basic и интервале текста oSel
'Используйте ключевые слова из массива sTokens()
Sub FMT_ColorCodeOneRangeStringBasic(oSel, sTokens())
  Dim oCurs    'Проходит по параграфам в выделенном тексте
  Dim oTCurs   'Проходит по символам в параграфе
  Dim oText    'Текстовый объект, содержащий выделение
  Dim iPos%
  Dim iLen%
  Dim i%       'Временная переменная — целое число
  Dim sChar$   'Текущий символ
  Dim sLine$   'Текущая строка, все символы переведены в малые буквы (in
lower case).

  REM  Позиция oTCurs в начале выделенного текста
  oText = oSel.getText()
  oTCurs = oText.createTextCursorByRange(oSel.getStart())
```

```
oTCurs.goRight(0, False)

REM oCurs содержит первый параграф
oCurs = oText.createTextCursorByRange(oSel.getStart())
oCurs.gotoEndOfParagraph(True)

Do While oText.compareRegionEnds(oCurs, oSel) >= 0
  REM  Теперь обрабатываем одну строку текста!
  REM oCurs выделил целый параграф
  REM oTCurs  находится в начале этого параграфа
  sLine = LCase(oCurs.getString())
  iLen = Len(sLine)
  iPos = 1
  Do While iPos <= iLen
    REM  Пропускаем ведущие символы всех типов пробела
    FMT_FindNextNonSpace(sLine, iPos%, iLen%)
    If iPos > iLen Then Exit Do

    sChar = Mid(sLine, iPos, 1)
    If sChar = "'" Then
      Rem  Находим комменталий, помечаем остаток строки
      REM  Передвигаем курсор символа от начала параграфа
      REM  на символ одиночной кавычки '
      REM  Выделяем остаток параграфа
      oTCurs.goRight(iPos-1, False)
      oTCurs.gotoEndOfParagraph(True)
      oTCurs.CharStyleName = sCharStyleComment
      iPos = iLen + 1
    ElseIf sChar = """" Then
      REM  Передвигаемся на первую двойную кавычку @
      oTCurs.goRight(iPos-1, False)

      REM  запоминаем положение первой двойной кавычки
      REM  и затем находим конец текста, ограниченного кавычками
      i = iPos
      FMT_FindEndQuoteDouble(sLine$, iPos%, iLen%)

      REM  передвигаем курсор на завершающую двойную кавычку "
      REM  Устанавливаем стиль симовлов для данной строки
      REM  передвигаем курсор обратно к началу параграфа
      oTCurs.goRight(iPos — i, True)
      oTCurs.CharStyleName = sCharStyleLiteral
      oTCurs.gotoRange(oCurs.start, False)
    ElseIf FMT_FindNumberEnd(sLine, iPos, iLen, i) Then
      REM  Передвигаем к началу числа
      oTCurs.goRight(iPos-1, False)
      oTCurs.goRight(i — iPos, True)
      oTCurs.CharStyleName = sCharStyleLiteral
      oTCurs.gotoRange(oCurs.start, False)
      iPos = i
    ElseIf sChar = "." OR FMT_IsSpecialChar(ASC(sChar)) Then
      i = iPos
      oTCurs.goRight(iPos — 1, False)

      Do
```

```
          iPos = iPos + 1
          If iPos > iLen Then Exit Do
        Loop Until NOT FMT_IsSpecialChar(ASC(Mid(sLine, iPos, 1)))
        oTCurs.goRight(iPos — i, True)
        oTCurs.CharStyleName = sCharStyleKeyWord
        oTCurs.gotoRange(oCurs.start, False)
      ElseIf sChar = "_" and iPos = iLen Then
        REM  Идентификатор может начинаться с символа"_" (я так думаю).
        REM  Похоже, что завершающие пробелы будут в тексте документа,
        REM   но сейчас мы их игнорируем!
        oTCurs.goRight(iPos-1, False)
        oTCurs.goRight(1, True)
        oTCurs.CharStyleName = sCharStyleKeyWord
        oTCurs.gotoRange(oCurs.start, False)
        iPos = iPos + 1
      Else
        REM  Не специальный символ, поэтому это переменная или логическое
        REM  выражение. Передвигаемся к первому символу
        i = iPos
        oTCurs.goRight(iPos-1, False)
        Do
          iPos = iPos + 1
          If iPos > iLen Then Exit Do
        Loop Until FMT_IsWordSep(Asc(Mid(sLine, iPos, 1)))

        oTCurs.goRight(iPos — i, True)
        sChar = LCase(oTCurs.getString())

        REM  Может быть проблема с именем переменной, похожим на
        REM "rem.doit.var". Отладчик Basic IDE пропускает такое имя,
        REM  поэтому я не буду волноваться об этом
        If sChar = "rem" Then
          oTCurs.gotoEndOfParagraph(True)
          oTCurs.CharStyleName = sCharStyleComment
          iPos = iLen + 1
        ElseIf FMT_ArrayHasString(sChar, sTokens()) Then
          oTCurs.CharStyleName = sCharStyleKeyWord
        Else
          oTCurs.CharStyleName = sCharStyleIdentifier
        End If

        oTCurs.gotoRange(oCurs.start, False)
      End If
    Loop
    If Not oCurs.gotoNextParagraph(False) Then Exit Do
    oTCurs.gotoRange(oCurs, False)
    If NOT oCurs.gotoEndOfParagraph(True) Then Exit Do
  Loop
End Sub
```

Удачного форматирования!

Глава VIII. Формы

> ***Внимание*** *Здесь очень мало информации о формах. Скачайте*
> `http://www.pitonyak.org/database/AndrewBase.odt`
> *со страницы о базах данных моего сайта*

Frank Schönheit [`fs@openoffice.org`] писал следующее: «Для обновления awt-control внесите изменения в модель (**model**) и элементы управления (control), которые входят в эту модель, автоматически обновятся так, чтобы соответствовать этой модели. Изменение элементов управления напрямую может привести к некоторому несоответствию (inconsistency). Например, для списка (list box) не используйте интерфейс XlistBox, предоставляемый этим элементом управления для выбора позиции списка (XListBox::selectItem), а лучше используйте свойство com.sun.star.awt.UnoControlListBoxModel::SelectedItems этой модели».

VIII.1. Введение

Форма или документ формы (form document) является набором форм и (или) элементов управления (controls), таких как кнопка или выпадающий список. формы могут использоваться для доступа к данным и другим сложным вещам. См. также:

`http://api.openoffice.org/docs/common/ref/com/sun/star/form/XFormsSupplier.html`

`http://api.openoffice.org/docs/common/ref/com/sun/star/form/module-ix.html`

Чтобы получить форму, нужно сначала получить чертеж страницы (draw page). Каким способом это делать — зависит от типа документа. В качестве теста я вставил кнопку в электронную таблицу. Я назвал кнопку «TestButton» и форму «TestForm». Затем я создал следующий макрос, который изменяет текст кнопки (button label) при нажатии на неё.

Лист. 300: Изменить текст кнопки в документе OOo Calc.

```
Sub TestButtonClick
  Dim vButton, vForm
  Dim oForms

  oForms = ThisComponent.CurrentController.ActiveSheet.DrawPage.Forms
  vForm=oForms.getByName("TestForm")
  vButton = vForm.getByName("TestButton")
  vButton.Label = "Wow"
  Print vButton.getServiceName()
End Sub
```

Из текстового документа OOo Writer можно было бы получить форму следующим образом:

```
oForm=THISCOMPONENT.DrawPage.Forms.getByName("FormName")
```

VIII.2. Диалоги

Обычно примеры диалога начинаются так:

Лист. 301: Диалог.

```
Dim oDlg As Object
Sub StartDialog
  oDlg = CreateUnoDialog(DialogLibraries.Standard.Dialog1 )
  oDlg.execute()
End Sub
```

Заметьте, что есть переменная с именем oDlg, когорая определена вне метода, который её создаёт. Это необходимо, потому что к диалогу обращаются другие процедуры, которые называются обработчиками событий (event handlers). Если такие процедуры будут обращаться к данному диалогу, то они должны иметь возможность доступа к переменной, которая ссылается на диалог.

Диалоги, как и макро библиотеки, имеют свою иерархию. В данном примере я ввел текст макроса, показанный выше. Затем я выбрал меню Сервис — Макросы и затем Управление макросами (Tools->Macros , «Organizer»). Вы увидите, что это организует/показывает вещи из текущего документа в секции, названной Standard. Когда Вы (на вкладке Диалоги) кликнете (нажмете левую клавишу мыши) на кнопке «Новый диалог» («New Dialog»), то имя диалога по умолчанию будет Диалог1 («Dialog1»). Так как этот текст программы выполняется из макроса, встроеенного в документ, то библиотека DialogLibraries ссылается на иерархию библиотек данного документа. Если Вы хотите обратиться к иерархии библиотек приложения из макроса данного документа, то нужно использовать GlobalScope.DialogLibraries.

Стандартный способ закрыть диалог включает установку обработчика событий, который закроет этот диалог. Я обычно добавляю кнопку «Закрыть» («close»), которая вызывает метод, похожий на следующий макрос

Лист. 302: Закрытие диалога.

```
Sub CloseDialog
  oDlg.endExecute()
End Sub
```

Простите за множество деталей, но я считаю их необходимыми для начинающих. Во время редактирования нового диалога я кликнул на кнопке «Кнопка» («control») на панели управления и выбрал таким образом вставку кнопки. Затем я вставил кнопку. Правый клик (нажатие правой клавиши мыши) на новую кнопку и выбрал «Свойства» (properties). На вкладке Общие (General tab) я установил имя этой кнопки (идентификатор) равным

«ExitButton» и метку (текст на кнопке) равным «Exit». Затем я кликнул на вкладку События («Events» tab) и затем в строке «Во время инициализации» («When Initiating») кликнул на кнопку «...» для открытия диалога, который позволяет назначить действия событиям. В разделе событий я выбрал строку «Во время инициализации». Нажав кнопку «Макрос» я в окне «Выбор макроса» выбрал и затем кликнул на кнопку ОК. Хотя это событие обрабатывает активацию и мышью, и клавиатурой, возможно назначить разные макросы для событий «нажатие клавиши» и «нажатие клавиши мыши». У меня не было причин различать эти способы инициализации..

Совет *Нужно иметь переменную, ссылающуюся на данный диалог, причем переменную, определенную вне процедуры диалога, так чтобы другие процедуры могли обращаться к диалогу.*

Совет *Из макроса документа DialogLibraries.Standard.Dialog1 ссылается на диалог в текущем документе, а GlobalScope.DialogLibraries.Standard... ссылается на глобальный диалог.*

VIII.2.1. Элементы управления

Все элементы управления (Controls) имеют общие черты. Некоторые из них описаны здесь:

`http://api.openoffice.org/docs/common/ref/com/sun/star/awt/UnoControl.html`
`http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XWindow.html`
`http://api.openoffice.org/docs/common/ref/com/sun/star/awt/module-ix.html`

Это позволяет управлять свойствами элементов управления, такими как видимость (visibility), включение (enabled), размер (size) и другие общиее свойства. Многие элементы управления имеют общие методы, такие как setLabel(string). Много разных типов событий поддерживается. По моему опыту, наиболее часто используемые события — уведомления об изменении статуса элемента управления.

Элемент управления может быть получен из диалога с использованием метода getControl(control_name). Это возможно также путем перебора всех элементов управления, если захотите.

VIII.2.2. Текст/метка элемента управления (Control Label)

Текст элемента управления (label) действует как обычный текст в поле диалога. Это обычно используется для пометки элемента управления. Можно получить и установить текст (label), используя getText() и setText(string). Можно также указать расположение этого текста в отведенном для него пространстве, выбрав left (слева), centered (по центру), or right (справа). Часто выбирают расположение справа, так что текст будет ближе к элементу управления, который он поясняет. См.:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XFixedText.html

VIII.2.3. Управляющая кнопка (Button)

Обычно кнопка (button) используется только для вызова процедуры, когда кнопка нажата. Можно также вызвать setLabel(string) для изменения текста кнопки (button label). См. также:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XButton.html

VIII.2.4. Поле текста (Text Box)

Текстовое поле (text box) используется для содержания обычного текста. Можно ограничить максимальную длину текста и управлять максимальным числом строк текста. Можно написать Ваши собственные управляющие элементы форматирования, если захотите. Чаще всего используются методы getText() и setText(string). Если Вам нужна полоса прокрутки (scroll bar), то следует установить её при задании свойств создаваемого диалога

Имеются специальные виды окошек ввода для дат, времени, чисел, ввода по образцу, форматированных данных, денежных сумм. Если Вы вставляете такое окошко, убедитесь, что уделили достаточно внимания его свойствам, чтобы получить нужное поведение. Например, можно отключить строгую проверку форматирования и задать интервал допустимых числовых значений. Поле для ввода по образцу содержит маску ввода (input mask) и маску символов (character mask). Маска ввода определяет, какие данные пользователя можно вводить. Маска символов определяет состояние поля с маской во время загрузки формы. Поле форматированных данных допускает любое форматирование, поддерживаемое в OOo. Если мне понадобится поле для ввода номера социального страхования, я буду использовать этот тип поля ввода. См. также:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XTextComponent.html
http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XTimeField.html
http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XDateField.html
http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XCurrencyField.html
http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XNumericField.html
http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XPatternField.html

VIII.2.5. Поле списка (List Box)

Поле списка (list box) предоставляет список значений, из которого можно выбрать нужное значение. Можно разрешить выбор нескольких значений (multiple selections). Чтобы добавить элементы в поле списка, я обычно использую что-то похожее на addItems(Array(«one», «two», «three»), 0). Можно также удалить элементы из поля списка.

Для варианта выбора одного элемента списка можно использвовать getSelectedItemPos() для определения, какой элемент выбран. Число -1 возвращается, если не выбрано ничего. Если что-то выбрано, то первому элементу списка соответствует возвращаемое значение 0. Для варианта выделения нескольких элементов списка используйте getSelectedItemsPos(), который возвращает последовательность коротких целых чисел (shorts). См.:
`http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XListBox.html`

VIII.2.6. Комбинированное поле: ввод с присоединенным списком (Combo Box)

Комбинированное поле ввода с присоединенным списком (combo box) представляет собой поле ввода с с присоединенным к нему полем списка. Часто его называют полем с выпадающим списком. В приводимом ниже примере я создаю комбинированное поле в два этапа. Сначала я задаю список значений для поля, а затем задаю поле ввода для свойств.

Лист. 303: Значения и поле.

```
aControl.addItems(Array("properties", "methods", "services"), 0)
aControl.setText("properties")
```

Затем я перехожу к событиям и выбираю для события«Item status changed» (изменен статус элемента) вызов процедуры NewDebugType. Она будет выводить все методы, свойства и сервисы, поддерживаемые этим диалогом. Я сделал это для демонстрации события обратного вызова (call back event), что фактически выдает полезную информацию для отладки. См.:
`http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XComboBox.html`

VIII.2.7. Поле флажка (Check Box)

Хотя обычно поле флажка (check box) обычно используется только для вывода состоятия «Да» или «Нет», имеется вид поля флажка, называемый также полем тройного состояния (tristate). В OOo поле флажка может иметь состояние 0 (не помечен) или 1 (помечен). Можно использовать для получения текущего состояния поля методgetState(). Если вызвать метод enableTriState(true), то становится разрешенным состояние 2. Это состояние показывается как наличие метки флажка, но затем поле становится тусклым (Dim). Установить такое состояние можно с использованием метода setState(int). См.:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XCheckBox.html

VIII.2.8. Кнопка «радио» (включена/выключена) (Option/Radio Button)

Цель создания радио-кнопки radio button обычно в выборе одного элемента из группы элементов. Поэтому типичное использование её в диалоге — вставить рамку группы (group box) и затем поместить внутрь этой рамки радио-кнопок. Чтобы определить, какая из радио-кнопок выбрана, можно вызывать метод getState() для каждой из радио-кнопок, пока не найдете выбранную. Можно было бы использовать радио-кнопки вместо выпадающего списка в моем примере для выбора, что выводить в поле списка отладки. См.:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XRadioButton.html

VIII.2.9. Индикатор выполнения (Progress Bar)

В моем примере я использую индикатор выполнения (progress bar), чтобы показать ход процесса заполнения поля списка отладки (debug list box). Вероятно, это не самый лучший пример, потому что процесс идет слишком быстро, но это же только пример.

Можно установить интервал для индикатора выполнения с помощью метода setRange(min, max). Это делает более легким наблюдение за ходом выполнения. Так как я обрабатываю строку, то я установил min равным 0 и max равным длине строки. Затем я вызываю метод setValue(int) со значением текущей позиции в строке, чтобы показать, как далеко продвинулся процесс. См.:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XProgressBar.html

VIII.3. Получение элементов управления (Controls)

Если Вы хотите перебрать элементы управления в форме документа, то будет работать следующее

Лист. 304: Перебрать элементы управления в форме

```
Sub EnumerateControlsInForm
  Dim oForm, oControl, iNumControls%, i%
  'По умолчанию это — место, где находятся элементы управления
  oForm = ThisComponent.Drawpage.Forms.getByName("Standard")
  oControl = oForm.getByName("MyPushButton")
  MsgBox "Использовано getByName для получения элементов управления с именем
" & oControl.Name
  iNumControls = oForm.Count()
  For i = 0 To iNumControls -1
    MsgBox "Control " & i & " is named " & oControl.Name
  Next
End Sub
```

Для элементов управления в диалоге используйте следующий код макроса:

Лист. 305: Перебрать элементы управления диалога.

```
  x = oDlg.getControls()
  For ii=LBound(x) To UBound(x)
    Print x(ii).getImplementationName()
  Next
```

Обычно элементы управления извлекают по имени, а не полным перебором.

VIII.3.1. Управление размерами и местоположением элементов управления по именам

Paolo Mantovani [mantovani.paolo@tin.it] показал, что элемент управления формы находится на подложке формы (drawing shape) (com.sun.star.drawing.XControlShape), поэтому нужно получить эту подложку формы, являющуюся фоном для элемента управления, чтобы управлять его размером и положением. Некоторые полезные макросы см. Сервис — Макросы — Управление макросами — Макросы OpenOffice.org — FormWizard — Tools дают возможность работать с элементами управления форм. Попробуйте следующий функции:

```
GetControlModel()
GetControlShape()
GetControlView()
```

Я написал следующий макрос, который хотя и получает элемент управления, но потом его не использует.

Лист. 306: Получить элемент управления и его подложку формы (shape) по имени

```
Sub GetControlAndShape
  Dim oForm, oControl

  'это место, где по умолчанию  находятся элементы управления
  oForm = ThisComponent.Drawpage.Forms.getByName("Standard")
  oControl = oForm.getByName("CheckBox")
  Dim vShape
  Dim vPosition, vSize
  Dim s$
  vShape = GetControlShape(ThisComponent, "CheckBox")
  vPosition = vShape.getPosition()
  vSize = vShape.getSize()
  s = s & "Position = (" & vPosition.X & ", " & vPosition.Y & ")"
  s = s & CHR$(10)
  s = s & "Height = " & vSize.Height & " Width = " & vSize.Width
  MsgBox s
End Sub
```

Следующий макрос изменяет размер элемента управления.

Лист. 307: Изменить размер элемента управления

```
Sub ModifyControlSize
  Dim oShape
  Dim oSize

  REM  Библиотека "tools" содержит GetControlShape
  GlobalScope.BasicLibraries.LoadLibrary("Tools")

  REM  Имя главной формы по умолчанию (Default main form name).
  oShape = GetControlShape(ThisComponent, "MainPushButton")
  oSize = oShape.getSize()

  REM  Теперь делаем элемент управления больше!
  oSize.Height = 1.10 * oSize.Height
  oSize.Width  = 1.10 * oSize.Width
  oShape.setSize(oSize)
  Print "Элемент управления теперь на  10%  больше"
End Sub
```

VIII.3.2. Какой элемент управления вызвал обработку и где он расположен?

Рассмотрим текстовую таблицу, содержащую кнопки (buttons). Все эти кнопки вызывают один и тот же макрос. Какая из ячеек содержит кнопку (вызвавшую макрос в этот раз)?

1. Добавим аргумент для перехватчика. Этот аргумент содержит свойство «Source» (источник), который ссылается на данную кнопку.
2. Подложки формы элементов управления имеют свойство «Control», которое ссылается на модель элемента управления (control's model). Перебираем эти подложки формы и сравниваем их модель (shape's model) с моделью элемента управления (control's model)

Следующий макрос предполагает, что все возвращаемые подложки формы (shapes) являются подложками формы элемента управления (control shapes), а Вам, наверно, захочется проверить, правда ли это.

Лист. 308: Найти подложку формы элемента управления (control's shape), если Вы НЕ знаете имя.

```
Sub ButtonCall(x)
  Dim oButton           ' Кнопка, которая использована для вызова
перехватчика
  Dim oModel            ' Модель этой кнопки.
  Dim oShape            ' Подложка формы кнопки
  Dim i As Long         ' Переменная индекса
  Dim bFound As Boolean ' становится равной True, когда найдена
соответствующая подложка формы

  REM  Сначала получим кнопку, которая вызвала процедуру
  REM  Сохраним модель этой кнопки
  oButton = x.Source
```

```
  oModel = oButton.getModel()

  REM  перебрать все элементы управления
  i = ThisComponent.getDrawPage().getCount()
  bFound = False
  Do While (i > 0 and NOT bFound)
    i = i – 1
    oShape = ThisComponent.getDrawPage().getByIndex(i)
    bFound = EqualUNOObjects(oShape.Control, oModel)
  Loop
  If bFound Then
    Print "Кнопка в ячейке " & oShape.getAnchor().Cell.CellName
  End If
End Sub
```

VIII.4. Выбор файла с использованием диалога (File Dialog)

Следующий пример выводит стандартный диалог выбора файла

Лист. 309: Использование стандартного диалога выбора файла.

```
Sub ExampleGetAFileName
  Dim filterNames(1) As String
  filterNames(0) = "*.txt"
  filterNames(1) = "*.sxw"
  Print GetAFileName(filterNames())
End Sub

Function GetAFileName(Filternames()) As String
  Dim oFileDialog as Object
  Dim iAccept as Integer
  Dim sPath as String
  Dim InitPath as String
  Dim RefControlName as String
  Dim oUcb as object
  'Dim ListAny(0)
  'Заметьте: следующие сервисы нужно вызвать с указанном ниже порядке,
  '  иначе сервис FileDialog не будет удален/завершен
  oFileDialog = CreateUnoService("com.sun.star.ui.dialogs.FilePicker")
  oUcb = createUnoService("com.sun.star.ucb.SimpleFileAccess")
  'ListAny(0) = _
  ' com.sun.star.ui.dialogs.TemplateDescription.FILEOPEN_SIMPLE
  'oFileDialog.initialize(ListAny())
  AddFiltersToDialog(FilterNames(), oFileDialog)

  'Здесь установите начальный путь!
  'InitPath = ConvertToUrl(oRefModel.Text)

  If InitPath = "" Then
    InitPath = GetPathSettings("Work")
  End If
  If oUcb.Exists(InitPath) Then
    oFileDialog.SetDisplayDirectory(InitPath)
  End If
```

```
  iAccept = oFileDialog.Execute()
  If iAccept = 1 Then
    sPath = oFileDialog.Files(0)
    GetAFileName = sPath
    'If oUcb.Exists(sPath) Then
    '  oRefModel.Text = ConvertFromUrl(sPath)
    'End If
  End If
  oFileDialog.Dispose()
End Function
```

VIII.5. Центрировать диалог на экране

Благодарности Berend Cornelius [Berend.Cornelius@sun.com] за предоставление следующего макроса. Главная хитрость в использовании текущего контроллера (controller) для окончательного извлечения окна данного компонента и из него извлечение позиции и размера этого окна.

Лист. 310: Центрировать диалог на экране.

```
Sub CenterDialogOnScreen
  Dim CurPosSize As New com.sun.star.awt.Rectangle
  Dim oFrame
  oFrame =  ThisComponent.getCurrentController().Frame
  FramePosSize = oFrame.getComponentWindow.PosSize
  xWindowPeer = oDialog.getPeer()
  CurPosSize = oDialog.getPosSize()
  WindowHeight = FramePosSize.Height
  WindowWidth = FramePosSize.Width
  DialogWidth = CurPosSize.Width
  DialogHeight = CurPosSize.Height
  iXPos = ((WindowWidth/2) - (DialogWidth/2))
  iYPos = ((WindowHeight/2) - (DialogHeight/2))
  oDialog.setPosSize(iXPos, iYPos, DialogWidth, _
                     DialogHeight, com.sun.star.awt.PosSize.POS)
  oDialog.execute()
End Sub
```

VIII.6. Установить перехватчик события (event listener) для элемента управления

Следующий небольшой кусок кода программы устанавливает для события в элементе управления вызов процедуры Test из стандартной библиотеки документа в модуле с именем Module1. Благодарности Oliver Brinzing [OliverBrinzing@t-online.de] за этот код программы.

Лист. 311: Установить перехватчик события для элемента управления.

```
Sub SetEvent
  Dim oDoc as Object
  Dim oView as Object
```

```
  Dim oDrawPage as Object
  Dim oForm as Object
  Dim oEvents(0) As New com.sun.star.script.ScriptEventDescriptor

  oDoc = StarDesktop.getCurrentComponent
  oView = oDoc.CurrentController
  oDrawPage = oView.getActiveSheet.DrawPage

  '  получить первую форму
  oForm = oDrawPage.getForms.getByIndex(0)

  oEvents(0).ListenerType = "XActionListener"
  oEvents(0).EventMethod  = "actionPerformed"
  oEvents(0).AddListenerParam = ""
  oEvents(0).ScriptType = "StarBasic"

  oEvents(0).ScriptCode = "document:Standard.Module1.Test"
  oForm.registerScriptEvent(0, oEvents(0))
End Sub

Sub Test(oEvt)
  Print oEvt.Source.Model.Name
End Sub
```

Я предполагаю, что краткое описание метода registerScriptEvent в порядке. См.

`http://api.openoffice.org/docs/common/ref/com/sun/star/script/XEventAttacherManager.html`

в качестве хорошей отправной точки для изучения.

`registerScriptEvent(index, ScriptEventDescriptor)`

Переменная oEvents является массивом описаний событий, см.

`http://api.openoffice.org/docs/common/ref/com/sun/star/script/ScriptEventDescriptor.html`,

где описано событие, которое будет присоединено (attached) (EventMethod), и что это событие вызовет (ScriptCode). Вышеприведенный пример мог бы также легко использовать простую переменную вместо массива, потому что этот код использует только один элемент массива. Первый аргумент у метода registerScriptEvent является индексом объекта, который будет «использовать» описание события (event descriptor). Страницы помощи предполагают, что Вы знаете, какие объекты индексированы, и формулируете так: «Если любой объект будет присоединен под этим индексом, то это событие присоединяется автоматически». Что пока не сказано — что форма выступает как объект-контейнер для компонентов формы. Эти компоненты формы доступны и по имени, и по индексу.

VIII.7. Управление диалогом я не создал.

Некоторые команды диспетчера (dispatch) загружают диалог, которым Вам нужно управлять. Интересное решение предложено автором «ms777» на oooforum:

http://www.oooforum.org/forum/viewtopic.phtml?t=22845

1. Создать TopWindowListener.
2. Выполнить диспетчер (dispatch).
3. Этот диспетчер откроет новое главное окно (top window).
4. Вызывается перехватчик событий этого окна
5. Внутри перехватчика событий этого главного окна используется интерфейс доступности (accessibility interface) для выполнения нужных операций (interactions) пользователя.
6. Внутри перехватчика событий этого главного окна нажимается кнопка ОК диалога диспетчера для прекращения выполнения диспетчера.
7. Сам перехватчик событий TopWindowListener удаляется.

Есть пара предостережений.

- Не пытайтесь использовать точку прерывания (breakpoint) и не выполняйте других действий после создания перехватчика событий главного окна. Этот перехватчик попытается обработать все вводимое..
- Вам нужно понять расположение диалога, которым Вы хотите управлять. Это не простая задача, особенно потому, что Вы не можете изучать и перебирать объекты.

VIII.7.1. Вставка формулы

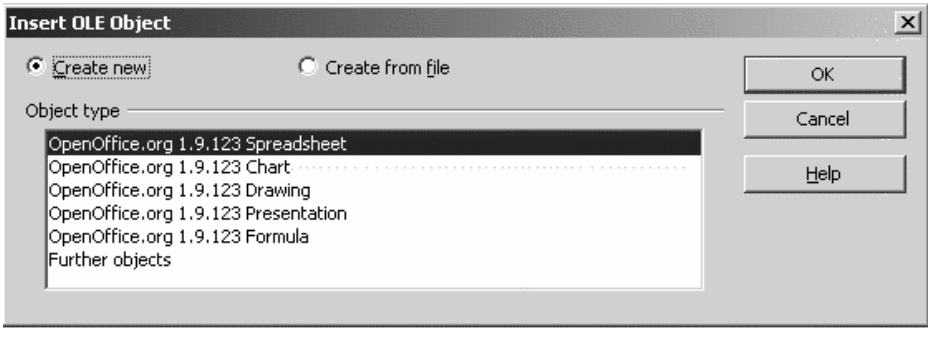


Рисунок 4: Вставка диалога объекта OLE

Используйте диалог Вставить OLE объект для вставки формулы или другого объекта (см. Рисунок 4). Работа с данным диалогом требует, чтобы Вы знали, какие объекты вставлены, тогда Вы можете получить к ним доступ. Вам также нужно знать, как взаимодействовать с каждым объектом.

Таблица 17: Доступное содержимое в диалоге Вставить объект OLE.

Child	*Имя*	*Тип*
0	Create new (создать новый)	RADIO_BUTTON
1	Create from file (создать из файла)	RADIO_BUTTON
2		PANEL
3	Object type (тип объекта)	SEPARATOR
4	OK	PUSH_BUTTON
5	Cancel (прервать)	PUSH_BUTTON
6	Help (помощь)	PUSH_BUTTON

Макрос из Лист. 312 демонстрирует, как использовать диалог Вставить объект OLE на Рисунке Рисунок 1. Этот макрос вставляет равенство и оставляет курсор в режиме вставки равенства (поэтому он фактически не устанавливает равенство).

Лист. 312: Вставить формулу в OOo Writer.

```
Sub InsertFormulaIntoWriter()
  Dim oFrame    ' Фрейм из текущего окна
  Dim oToolkit ' Для контейнета окна —  com.sun.star.awt.Toolkit
  Dim oDisp     '  Помощник диспетчера (Dispatch helper).
  Dim oList     ' XTopWindowListener который обрабатывает интерактивные
действия пользователя (interactions).
  Dim s$

  REM  Получить com.sun.star.awt.Toolkit
  oFrame   = ThisComponent.getCurrentController().getFrame()
  oToolkit = oFrame.getContainerWindow().getToolkit()
  s$ = "com.sun.star.awt.XTopWindowListener"
  oList    = createUnoListener("TopWFormula_", s$)
  oDisp    = createUnoService("com.sun.star.frame.DispatchHelper")

  REM  Вставляем объект OLE !
  oToolkit.addTopWindowListener(oList)
  oDisp.executeDispatch(oFrame, ".uno:InsertObject", "", 0, Array())
  oToolkit.removeTopWindowListener(oList)
End Sub

Sub TopWFormula_windowOpened(e As Object)
  Dim oAC
  Dim oACRadioButtonNew
  Dim oACList
  Dim oACButtonOK

  REM  Получить доступное окно, которое является целым диалогом
  oAC = e.source.AccessibleContext

  REM  Получить кнопки
  oACRadioButtonNew = oAC.getAccessibleChild(0).AccessibleContext
  DIM oAC2
  oAC2 = oAC.getAccessibleChild(2)
  oACList           = oAC2.AccessibleContext.getAccessibleChild(0)
  oACButtonOK       = oAC.getAccessibleChild(4).AccessibleContext

  REM  Выбрать создане нового "Create New"
  oACRadioButtonNew.doAccessibleAction(0)

  REM  доступ к пятому элементу в списке (как в 0, 1, 2, 3, 4...)
  oACList.selectAccessibleChild(4)

  REM  доступным действием для кнопки является использовать (use) её
  oACButtonOK.doAccessibleAction(0)
End Sub

Sub TopWFormula_windowClosing(e As Object)
End Sub

Sub TopWFormula_windowClosed(e As Object)
End Sub
```

```
Sub TopWFormula_windowMinimized(e As Object)
End Sub

Sub TopWFormula_windowNormalized(e As Object)
End Sub

Sub TopWFormula_windowActivated(e As Object)
End Sub

Sub TopWFormula_windowDeactivated(e As Object)
End Sub
```

VIII.7.2. Раскрытие доступного содержимого (автор Andrew)

Вы не можете прямо исследовать объекты, поэтому трудно определить доступное содержимое, так что я написал макрос в Лист. 313. Этот макрос просматривает все дочернее содержимое и затем печатает имя и тип. Я не сильно заботился об обработке ошибок, поэтому будьте предупреждены.

Лист. 313: Просмотреть и исследовать доступное содержимое

```
Function InspectAccessibleContent(sLead$, oAC) As String
  REM  Автор Andrew Pitonyak
  Dim x
  Dim s$
  Dim i%
  Dim sL$
  Dim s1$, s2$
  Dim oACUse
  Dim oACChild

  s1 = "com.sun.star.accessibility.XAccessibleContext"
  s2 = "com.sun.star.accessibility.XAccessible"
  If HasUnoInterfaces(oAC, s1) Then
    oACUse = oAC
    sL = sLead
  ElseIf HasUnoInterfaces(oAC, s2) then
    oACUse = oAC.AccessibleContext
    sL = sLead & ".AC."
  Else
    Exit Function
  End If

  x = Array( "UNKNOWN", "ALERT", "COLUMN_HEADER", "CANVAS", _
    "CHECK_BOX", "CHECK_MENU_ITEM", "COLOR_CHOOSER", "COMBO_BOX", _
    "DATE_EDITOR", "DESKTOP_ICON", "DESKTOP_PANE", _
    "DIRECTORY_PANE",  "DIALOG", "DOCUMENT", _
    "EMBEDDED_OBJECT", "END_NOTE", "FILE_CHOOSER", "FILLER", _
    "FONT_CHOOSER", "FOOTER", "FOOTNOTE", "FRAME", "GLASS_PANE", _
    "GRAPHIC", "GROUP_BOX", "HEADER", "HEADING", "HYPER_LINK", _
    "ICON", "INTERNAL_FRAME", "LABEL", "LAYERED_PANE", "LIST", _
    "LIST_ITEM", "MENU", "MENU_BAR", "MENU_ITEM", "OPTION_PANE", _
    "PAGE_TAB", "PAGE_TAB_LIST", "PANEL", "PARAGRAPH", _
    "PASSWORD_TEXT", "POPUP_MENU", "PUSH_BUTTON", "PROGRESS_BAR", _
```

```
    "RADIO_BUTTON", "RADIO_MENU_ITEM", "ROW_HEADER", "ROOT_PANE", _
    "SCROLL_BAR", "SCROLL_PANE", "SHAPE", "SEPARATOR", "SLIDER", _
    "SPIN_BOX", "SPLIT_PANE", "STATUS_BAR", "TABLE", "TABLE_CELL", _
    "TEXT", "TEXT_FRAME", "TOGGLE_BUTTON", "TOOL_BAR", "TOOL_TIP", _
    "TREE", "VIEW_PORT", "WINDOW")

  For i=0 to oACUse.AccessibleChildCount-1
    oACChild = oACUse.getAccessibleChild(i)
    If HasUnoInterfaces(oACChild, s1) Then
      s = s & sL & i & " " & oACChild.AccessibleName  & " " & _
          x(oACChild.AccessibleRole) & CHR$(10)
    ElseIf HasUnoInterfaces(oACChild, s2) Then
      oACChild = oACChild.AccessibleContext
      s = s & sL & "AC." & i & " " & oACChild.AccessibleName  & _
          " " & x(oACChild.AccessibleRole) & CHR$(10)
    Else
        Exit Function '    неожиданная ситуация
    End If
    Dim nTemp As Long
     nTemp =  com.sun.star.accessibility.AccessibleRole.PANEL
    If oACChild.AccessibleRole = nTemp Then
      s = s & InspectIt(sL & " " & x(oACChild.AccessibleRole) & _
                                ".", oACChild)
    End If
  Next
  InspectIt = s
End Function
```

Диалог Опции очень сложен, что делает его исключительным случаем для изучения. Код программы в Лист. 314 не завершен. Этот код получен модификацией кода из Лист. 312; требуемые изменения должны быть очевидны. Этот макрос открывает диалог Опции и оставляет его открытым. Выводится информация о том, какая вкладка выбрана для показа.

Лист. 314: Как исследовать доступное содержимое

```
Option Explicit
Private sss$

Sub ManipulateOptions()
  Dim oFrame   ' Фрейм из текущего окна
  Dim oToolkit ' Контейнер для окна (Container window's)
com.sun.star.awt.Toolkit
  Dim oDisp    ' Помощник диспетчера (Dispatch helper).
  Dim oList    ' XTopWindowListener который обрабатывает интерактивные
действия пользователя (interactions).

  REM  Получить com.sun.star.awt.Toolkit
  oFrame   = ThisComponent.getCurrentController().getFrame()
  oToolkit = oFrame.getContainerWindow().getToolkit()
  oList    = createUnoListener("TopWFormula_", _
                       "com.sun.star.awt.XTopWindowListener")
  oDisp    = createUnoService("com.sun.star.frame.DispatchHelper")
```

```
  oToolkit.addTopWindowListener(oList)
  oDisp.executeDispatch(oFrame, ".uno:OptionsTreeDialog", "", _
              0, Array())
  oToolkit.removeTopWindowListener(oList)
  MsgBox sss
End Sub

Sub TopWFormula_windowOpened(e As Object)
  sss = InspectIt("", e.source.AccessibleContext)
End Sub
```

Ввиду сложности некоторых диалогов, автор ms777 написал некоторые поисковые процедуры для нахождения элементов управления на основе имени, типа или описания. Во время поиска происходит приостановка (delay), если нужное содержимое не найдено. Когда производятся манипуляции с диалогом, возможно, доступное содержимое ещё не обновлено для вывода на экран (поэтому оно может ещё не существовать).

Лист. 315: Поиск заданного содержимого (child).

```
Function SearchOneSecForChild(oAC As Object, sName$, sDescription$, lRole As
Long) As Object
  Dim k%
  Dim oRes

  For k=1 To 50
    oRes = SearchForChild(oAC, sName, sDescription, lRole)
    If NOT IsNull(oRes) Then
      SearchOneSecForChild = oRes
      Exit Function
    End If
    Wait(20)
  Next
End Function

Function SearchForChild(oAC As Object, sName$, sDescription$, lRole As Long)
As Object
  Dim oAC1
  Dim oACChild
  Dim k%
  Dim bFound As Boolean

  If HasUnoInterfaces(oAC, "com.sun.star.accessibility.XAccessibleContext")
Then
    oAC1 = oAC
  ElseIf HasUnoInterfaces(oAC, "com.sun.star.accessibility.XAccessible")
Then
    oAC1 = oAC.AccessibleContext
  Else
    Exit Function ' неожиданая ситуация, может быть, вызванная некоторым
промежуточным состоянием
  End If

  For k=0 To oAC.AccessibleChildCount-1
```

```
    oACChild = oAC.getAccessibleChild(k)
    If NOT HasUnoInterfaces(oACChild,
"com.sun.star.accessibility.XAccessibleContext") Then
      If HasUnoInterfaces(oACChild,
"com.sun.star.accessibility.XAccessible") Then
        oACChild = oACChild.AccessibleContext
      Else
        Exit Function ' неожиданая ситуация, может быть, вызванная некоторым
промежуточным состоянием
      End If
    End If
    bFound = true

    If sName<>""        Then bFound = bFound and (oACChild.AccessibleName =
sName)
    If sDescription<>"" Then bFound = bFound and _
                                  (oACChild.AccessibleDescription =
sDescription)
    If lRole > -1       Then bFound = bFound and (oACChild.AccessibleRole =
lRole)

    If bFound Then
      SearchForChild = oACChild
      Exit Function
    End If
  Next
End Function
```

VIII.7.3. Выполнение манипуляций с диалогом Опции (Options)

Диалог Опции (Tools | Options) очень сложный, даже ещё сложнее из-за того, что этот диалог может быть в любом состоянии. Следующий макрос разрушает (collapses) все дерево структур для того, чтобы привести этот диалог в известное состояние.

Лист. 316: Разрушить (Collapse) все деревья в диалоге Опции (Options).

```
Sub CollapseAllTrees(oAC)
  REM Author ms777
  REM Modified by Andrew Pitonyak
  Dim oACChild
  Dim oACChild1
  Dim k%
  Dim k1%

  'установить определенное состояние: разрушить все элементы во всех
деревьях
  For k=0 To oAC.getAccessibleChildCount-1
    oACChild = oAC.getAccessibleChild(k)
    If HasUnoInterfaces(oACChild,
"com.sun.star.accessibility.XAccessibleContext") Then
      If oACChild.AccessibleRole =
com.sun.star.accessibility.AccessibleRole.TREE Then
        For k1=0 To oACChild.getAccessibleChildCount-1
```

```
          oACChild1 = oACChild.getAccessibleChild(k1)
          If HasUnoInterfaces(oACChild1,
"com.sun.star.accessibility.XAccessibleAction") Then
            If oACChild1.AccessibleStateSet.contains(_
                                com.sun.star.accessibility.AccessibleStateTy
pe.EXPANDED) Then
              oACChild1.doAccessibleAction(0)
            End If
          End If
        Next k1
      End If
    End If
  Next k
End Sub
```

Наконец, после открытия диалога Опции, Вам нужно выполнять с ним манипуляции

Лист. 317: Установить пользовательский заголовок (Title) в диалоге Опции (Options).

```
Sub TopWFormula_windowOpened(e As Object)
  Dim oAC
  Dim oACTree
  Dim oACOO
  Dim oACButtonOK
  Dim oACUserDataPanel
  Dim oACTitle
  Dim bResult As boolean

  REM Получаем доступное окно, которое является диалогом в целом.
  oAC = e.source.AccessibleContext
  CollapseAllTrees(oAC)
  'e.source.setVisible(false)

  'Открыть элемент дерева "OpenOffice.org"
  oACTree = SearchOneSecForChild(oAC, "", "", _
       com.sun.star.accessibility.AccessibleRole.TREE)
  oACOO   = SearchOneSecForChild(oACTree, "OpenOffice.org", "", _
       com.sun.star.accessibility.AccessibleRole.LABEL)
  oACOO.doAccessibleAction(0)

  'Выбрать первый элемент в OpenOffice.org (то есть элемент UserData)
  oACOO.selectAccessibleChild(0)

  'Теперь устанавливаем UserData
  oACUserDataPanel = SearchOneSecForChild(oAC, "User Data", "", _
            com.sun.star.accessibility.AccessibleRole.PANEL)
  oACTitle = SearchOneSecForChild(oACUserDataPanel, "Title/Position", _
            "Наберите Ваш заголовок для этого поля.", _
            com.sun.star.accessibility.AccessibleRole.TEXT)
  If NOT IsNULL(oACTitle) and NOT IsEmpty(oACTitle) Then
    bResult = oACTitle.setText("Meister aller Klassen")
  EndIf
```

```
  REM нажать кнопку OK
  oACButtonOK = SearchOneSecForChild(oAC, "OK", "", _
                com.sun.star.accessibility.AccessibleRole.PUSH_BUTTON)
  oACButtonOK.doAccessibleAction(0)
End Sub
```

VIII.7.4. Листинг поддерживаемых принтеров

Это — один из моих любимых примеров, представленный автором ms777. Получить список всех поддерижваемых принтеров птем вывода диалога Принтеры, извлечение списка притеров из этого диалога, а затем закрываем диалог. Макрос в Лист. 318 требует наличия ещё макроса, приведенного в Лист. 315.

Лист. 318: Получить список всех поддерживаемых принтеров.

```
Option Explicit
Public GetPrinterArray735 As Any

Sub PrintAllPrinters
  REM Author ms777.
  Dim arPrinter
  Dim s$
  Dim k%

  arPrinter = GetAllPrinters()
  s = ""
  For k = 0 To UBound(arPrinter)
    s = s + k + ": " + arPrinter(k) +Chr(10)
  Next k
  MsgBox s
End Sub

Function GetAllPrinters() As Any
  'On Error Resume Next
  Dim oFrame   ' Ферйм из текущего окна.
  Dim oToolkit ' Контейнер для этого окна com.sun.star.awt.Toolkit  этого
окна
  Dim oDisp    ' Помощник диспетчера (Dispatch).
  Dim oList    ' XTopWindowListener который обрабатывает интерактивные
действия (interactions).

  REM получаем com.sun.star.awt.Toolkit
  oFrame   = ThisComponent.getCurrentController().getFrame()
  oToolkit = oFrame.getContainerWindow().getToolkit()
  oList    = createUnoListener("TopWPrint_",
"com.sun.star.awt.XTopWindowListener")
  oDisp    = createUnoService("com.sun.star.frame.DispatchHelper")

  oToolkit.addTopWindowListener(oList)
  REM Вставим объект OLE !
  oDisp.executeDispatch(oFrame, ".uno:Print", "", 0, Array())
  oToolkit.removeTopWindowListener(oList)
  GetAllPrinters = GetPrinterArray735
```

```
End Function

Sub TopWPrint_windowOpened(e As Object)
  Dim oAC
  Dim oACComboBox
  Dim oACList
  Dim oACCancel
  Dim k%

  oAC = e.source.AccessibleContext
  e.source.setVisible(false)

  oACComboBox = SearchOneSecForChild(oAC,         "", "", _
                    com.sun.star.accessibility.AccessibleRole.COMBO_BOX)
  oACList     = SearchOneSecForChild(oACComboBox, "", "", _
                    com.sun.star.accessibility.AccessibleRole.LIST)
  oACCancel   = SearchOneSecForChild(oAC, "Cancel", "", _
                    com.sun.star.accessibility.AccessibleRole.PUSH_BUTTON)

  Dim sResult(oACList.AccessibleChildCount-1) As String

  For k = 0 to oACList.AccessibleChildCount-1
    sResult(k) = oACList.getAccessibleChild(k).AccessibleName
  Next k
  oACCancel.doAccessibleAction(0)
  GetPrinterArray735 = sResult()
End Sub

Sub TopWPrint_windowClosing(e As Object)
End Sub

Sub TopWPrint_windowClosed(e As Object)
End Sub

Sub TopWPrint_windowMinimized(e As Object)
End Sub

Sub TopWPrint_windowNormalized(e As Object)
End Sub

Sub TopWPrint_windowActivated(e As Object)
End Sub

Sub TopWPrint_windowDeactivated(e As Object)
End Sub
```

VIII.7.5. Поиск открытого окна

Следующий макрос представлен автором ms777 как часть его процедур исследования объектов. По-моему, этот макрос достаточно важен. Идея состоит в том, что Вы открываете диалог, замечаете заголовок (title), а затем вызываете этот макрос длдя получения ссылки на нужное окно. Этот макрос использован в

Лист. 319: Найти открытый диалог по его заголовку (title) .

```
'GetWindowOpen
'Перебираем открытые диалоги и находим тот, который начинается с sTitle.
Function GetWindowOpen(sTitle as String) As Object
  Dim oToolkit
  Dim lCount As Long
  Dim k As Long
  Dim oWin

  oToolkit = Stardesktop.ActiveFrame.ContainerWindow.Toolkit
  lCount = oToolkit.TopWindowCount

  For k=0 To lCount -1
    oWin = oToolkit.getTopWindow(k)
    If HasUnoInterfaces(oWin, "com.sun.star.awt.XDialog") Then
      If left(oWin.Title, len(sTitle)) = sTitle Then
        GetWindowOpen = oWin
        Exit Function
      EndIf
    EndIf
  Next k
End Function
```

VIII.7.6. Исследование доступного содержимого (автор ms777)

Я написал макрос в Лист. 313 для исследования содержимого (content). Ms777 затем представил мне свой макрос. Мне больше наравится его способ. Я не знал, как найти открытое окно, показанным в Лист. 319 способом. Хитрость при использовании нижеследующего макроса в том, чтобы сначала открыть диалог, а затем выполнить макрос с нужным именем. У меня были планы создать диалог, позволяющий выбрать нужное окно, но пока они не реализованы. Я сделал минимальные модификации по декларации всех переменных и небольших улучшений вывода результатов.

Лист. 320: Исследовать диалог с заданным именем

```
'Сначала открываем диалог, который Вы хотите исследовать. Затем выполняем
'следующий макрос:
Sub InspectOpenWindow
  Dim oWin
  Dim oAC

  'Эта функция идентифицирует главное окно, чей заголовок начинается
с "Options"
  ' Я использовал менд  Tools | Options для открытия этого окна.
  oWin = GetWindowOpen("Options")
  oAC = oWin.AccessibleContext

  ' Это создаёт иерархический список дерева доступности дерева
  call AnalyzeCreateSxc(oAC)
End Sub
```

```
'- GetWindowOpen
'Перебираем все открытые диалоги и находим тот, который начинается с
'sTitle.
Function GetWindowOpen(sTitle as String) As Object
  Dim oToolkit
  Dim lCount As Long
  Dim k As Long
  Dim oWin

  oToolkit = Stardesktop.ActiveFrame.ContainerWindow.Toolkit
  lCount = oToolkit.TopWindowCount

  For k=0 To lCount -1
    oWin = oToolkit.getTopWindow(k)
    If HasUnoInterfaces(oWin, "com.sun.star.awt.XDialog") Then
      If left(oWin.Title, len(sTitle)) = sTitle Then
        GetWindowOpen = oWin
        Exit Function
      EndIf
    EndIf
  Next k
End Function

'-AnalyzeCreateSxc
Sub AnalyzeCreateSxc(oAC As Object)
  Dim kRowSheetOut As Long
  Dim kColSheetOut as Long
  Dim oDoc as Object
  Dim oSheetOut

  oDoc = StarDesktop.LoadComponentFromUrl("private:factory/scalc","_default",0,Array())

  oSheetOut = oDoc.sheets.getByIndex(0)
  kRowSheetOut = 0
  kColSheetOut = 0
  call Analyze(oAC, oSheetOut, kRowSheetOut, kColSheetOut, "")
End sub

Sub Analyze(oAC as Object, oSheetOut as Object, kRowSheetOut as Long,
kColSheetOut as Long, sParentTrace as String)
  Dim sName As String
  Dim k As Long
  Dim kMax As Integer
  Dim lRole As Long
  Dim sProps() As String
  Dim sObjName As String
  Dim k1 As Long
  Dim k2 As Long
  Dim kColSheetOut As Long
  Dim oAC1
  Dim oCell
  Dim sTraceHelper As String

  If HasUnoInterfaces(oAC, "com.sun.star.accessibility.XAccessibleContext") Then
    oSheetOut.getCellByPosition(kColSheetOut, kRowSheetOut).String = _
```

```
        AccessibleObjectDescriptionString(oAC) + " (" + sParentTrace + ")"
    kRowSheetOut = kRowSheetOut + 1

    kMax = -1
    on error resume next
    kMax = oAC.getAccessibleChildCount()-1
    on error goto 0

    ' Показываем максимум 30 дочерних объектов (childs)
    If kMax>=0 Then
      If kMax>30 Then
        k1 = 15
        k2 = kMax-15
      Else
        k1=-1
        k2=0
      EndIf
      For k=0 To k1
        kColSheetOut = kColSheetOut + 1
        sTraceHelper = IIF(Len(sParentTrace) = 0, "", ", ")
        Call Analyze(oAC.getAccessibleChild(k), oSheetOut, kRowSheetOut, _
                     kColSheetOut, sParentTrace + sTraceHelper + k)
        kColSheetOut = kColSheetOut - 1
      Next k
      For k=k2 To kMax
        kColSheetOut = kColSheetOut + 1
        sTraceHelper = IIF(Len(sParentTrace) = 0, "", ", ")
        Call Analyze(oAC.getAccessibleChild(k), oSheetOut, kRowSheetOut, _
                     kColSheetOut, sParentTrace + sTraceHelper + k)
        kColSheetOut = kColSheetOut - 1
      Next k
    EndIf
  EndIf

  If HasUnoInterfaces(oAC, "com.sun.star.accessibility.XAccessible") Then
    oAC1 = oAC.AccessibleContext
    If EqualUnoObjects(oAC1, oAC) Then
      Exit Sub
    EndIf
    kColSheetOut = kColSheetOut + 1
    Call Analyze(oAC1, oSheetOut, kRowSheetOut, kColSheetOut, sParentTrace +
", AC"  )
    kColSheetOut = kColSheetOut - 1
  EndIf
End sub

Function AccessibleObjectDescriptionString(oAC As Object) As String
  Dim s As String
  Dim sText As String
  Dim sProps()
  Dim k As Long
  Dim lRole As Long
  Dim lActionCount As Long

  s = ""
```

```
  'AccessibleName
  On Error Resume Next
  sText = oAC.getAccessibleName()
  On Error Goto 0
  If sText ="" Then
    sText = "--"
  EndIf
  s = s + sText

  'AccessibleObject name
  sProps = Split(oAC.Dbg_Properties,"""")
  s = s + " " + sProps(1)

  'AccessibleRole
  lRole=-1
  On Error Resume Next
  lRole = oAC.getAccessibleRole()
  On Error Goto 0
  If lRole <>-1 Then
    s = s + " Role: " + Choose(lRole+1, "UNKNOWN", "ALERT", "COLUMN_HEADER",
"CANVAS", "CHECK_BOX", "CHECK_MENU_ITEM", "COLOR_CHOOSER", "COMBO_BOX",
"DATE_EDITOR", "DESKTOP_ICON", "DESKTOP_PANE", "DIRECTORY_PANE", "DIALOG",
"DOCUMENT", "EMBEDDED_OBJECT", "END_NOTE", "FILE_CHOOSER", "FILLER",
"FONT_CHOOSER", "FOOTER", "FOOTNOTE", "FRAME", "GLASS_PANE", "GRAPHIC",
"GROUP_BOX", "HEADER", "HEADING", "HYPER_LINK", "ICON", "INTERNAL_FRAME",
"LABEL", "LAYERED_PANE", "LIST", "LIST_ITEM", "MENU", "MENU_BAR",
"MENU_ITEM", "OPTION_PANE", "PAGE_TAB", "PAGE_TAB_LIST", "PANEL",
"PARAGRAPH", "PASSWORD_TEXT", "POPUP_MENU", "PUSH_BUTTON", "PROGRESS_BAR",
"RADIO_BUTTON", "RADIO_MENU_ITEM", "ROW_HEADER", "ROOT_PANE", "SCROLL_BAR",
"SCROLL_PANE", "SHAPE", "SEPARATOR", "SLIDER", "SPIN_BOX", "SPLIT_PANE",
"STATUS_BAR", "TABLE", "TABLE_CELL", "TEXT", "TEXT_FRAME", "TOGGLE_BUTTON",
"TOOL_BAR", "TOOL_TIP", "TREE", "VIEW_PORT", "WINDOW")
  EndIf

  'AccessibleDescription
  On Error Resume Next
  s = s + " " + oAC.getAccessibleDescription()
  On Error Goto 0

  If HasUnoInterfaces(oAC, "com.sun.star.accessibility.XAccessibleAction")
Then
    s = s + " Actions:"
    lActionCount = oAC.getAccessibleActionCount
    For k=0 To lActionCount-1
      s = s + " " + k + ": " + oAC.getAccessibleDescription(k)
    Next k
  EndIf

  AccessibleObjectDescriptionString = s
End Function
```

Глава IX. Пример по инвестициям (Investment)

IX.1. Внутренняя ставка возврата средств (Internal Rate of Return = IRR)

Возьмём сумму *P* и инвестируем её в банк с простой ставкой процента *r*, тогда через год заработаем дополнительную сумму rP процентов. Другими словами, в конце первого года будем иметь сумму *P(1+r).* Если оставить деньги в банке на *n* лет, то будущая сумма будет равна:

$$FV = P(1+r)^n \tag{1}$$

Если оставить деньги в банке на *m* дней, где *m* меньше, чем один год, то набежавшие проценты обычно распределяются пропорционально:

$$FV = P\left(1 + r\frac{m}{365}\right) \tag{2}$$

Предположим, что Вы периодически помещаете деньги на счет определенного типа. Позднее Вы будете знать, сколько денег будет на счете.

Таблица 18. Термины для вычисления процентов.

Термин	*Описание*
P_i	Сумма депозита с номером i^{th} (P остается для обозначения основного долга). Например, P_0 это сумма первог депозита или платежа.
t_i	Дата (t остается для обозначения времени) депозита с номером i^{th}.
FV	Окончательная (будущая) сумма
r	Годовая ставка процентов (сколько можно заработать каждый год).
C_i	Сумма, которую я добавил в качестве депозита с номером i^{th} .

Добавление всех депозитов составляет инвестированную сумму.

$$C_n = \sum_{i=0}^{n} P_i \tag{3}$$

Использование только простых процентов

Максимально упрощенным примером в формуле 2, расширенным на случай расчета нескольких депозитов в течение всего времени (даже если депозит НИКОГДА не используется таким способом).

$$FV = \sum_{i=0}^{n} P_i \left(1 + r(t - t_i)/365\right) \tag{4}$$

Теперь развернем эти уравнения

$$FV = \sum_{i=0}^{n} P_i + \sum_{i=0}^{n} P_i r(t - t_i)/365 = \sum_{i=0}^{n} P_i + \frac{r}{365} \sum_{i=0}^{n} P_i (t - t_i) \tag{5}$$

Для этого конкретного примера мы знаем все, кроме банковского процента, и теперь его можно найти из уравнения. Во-первых, вычтем из обеих частей положенные на депозит суммы

$$FV - \sum_{i=0}^{n} P_i = \frac{r}{365} \sum_{i=0}^{n} P_i (t - t_i) \tag{6}$$

Упростив уравнение получим:

$$365 \frac{FV - \sum_{i=0}^{n} P_i}{\sum_{i=0}^{n} P_i (t - t_i)} = r \tag{7}$$

Лист. 321, используемый как функция для OOo Calc, вычисляет ставку процента согласно формуле (7).

Лист. 321: Вычисляем годовую ставку процента.

```
Function YearlyRateOfReturn(FinalValue As Double, _
         CurrentDate As Date, DepositDates(), Deposits()) As Double
  Dim iRow As Integer
  Dim iCol As Integer
  Dim dSum As Double
  Dim d    As Double
  Dim dTotalDeposit As Double
  Dim dRate As Double
  Dim nMaxRows As Integer
  Dim nLB As Integer

  If UBound(DepositDates()) <> UBound(Deposits()) Then
    MsgBox "Количество дат не равно количеству депозитов"
    YearlyRateOfReturn = 0.0
    Exit Function
  End If

  dSum = 0
  dTotalDeposit = 0
  nLB = LBound(Deposits(), 1)
```

```
  For iRow = nLB To UBound(Deposits(), 1)
    d = 0
    For iCol = nLB To UBound(Deposits(), 2)
      d = d + Deposits(iRow, iCol)
    Next
    dTotalDeposit = dTotalDeposit + d
    dSum = dSum + (CurrentDate – DepositDates(iRow, nLB)) * d
  Next

  If dSum = 0 Then
    dRate = 0.0
  Else
    dRate = 365 * (FinalValue – dTotalDeposit) / dSum
  End If
  'Print "Final rate = " & dRate
  YearlyRateOfReturn = dRate
End Function
```

IX.1.1. Сложные проценты

Предположим, что проценты добавляются к основной сумме *k* раз в год и на них начисляются проценты *n* раз по ставке *r*.

$$FV = P\left(1+\frac{r}{k}\right)^{n} \tag{8}$$

Если я сделал *n* платежей, тогда формула будет намного сложнее, в основном, потому что я не произвожу платежи равномерно по времени. Самое лучшеее, что можно сказать — формула будет похожей на (9).

$$FV = \sum_{i=0}^{n} P_i\left(1+\frac{r}{k}\right)^{(T-t_i)} \tag{9}$$

Все переменные (9) известны, кроме ставки *r*. Это уравнение с многочленом относительно неизвестной *r* и потому не имеет легкого решения.

OOo Calc поддерживает функцию IRR для вычисления внутренней ставки процента для выплат вкладчику равномерно по заданным интервалам времени. Есть две проблемы с этой функцией IRR: 1) платежи должны быть равномерными по времени 2) задана общая сумма получаемых процентов.

Наилучший путь решения задачи, видимо, следующий:

1. Преобразовать данные в форму, пригодную для применения функции IRR.
2. Использовать макрос из Лист. 321 для создания начального приближения к решению, если это требуется.

Первый способ кажется мне очень трудным. Может быть, когда-нибудь я это сделаю?

Глава X. Обработчики (Handlers) и перехватчики (Listeners)

Прим. пер.: хотя сначала обработчики и перехватчики упоминаются как синонимы, есть различие между ними: перехватчик события обычно выполняет действия и передает событие следующим перехватчикам, а обработчик события может после выполнения вернуть (хотя может и не вернуть) признак того, что событие обработано полностью и передавать его оставшимся перехватчикам не нужно. Обработчик (handler) — особый случай перехватчика (listener).

Обработчки (handlers), для целей данной главы, это любой код программы, который использует «обратный вызов» (call back) или что-то связанное с обработкой событий определенного сорта.

X.1. xKeyHandler пример

Leston Buell [bulbul@ucla.edu] написал ключевой обработчик, который следит за событиями нажатия клавиш и преобразует конкретные комбинации клавиш в символы языка Эсперанто Esperanto.

Глобальные переменные используются для хранения ссылки на обработчик нажатий клавиш. Это важно, потому что глобальная переменная сохраняет свое значение между обращениями к макросу. Ссылка на обработчик необходима, поэтому удалять её можно только позднее.

Лист. 322: Глобальные переменные для переводчика Esperanto

```
'Автор:Leston Buell [bulbul@ucla.edu]
Global oComposerDocView
Global oComposerKeyHandler
Global oComposerInputString
```

Когда Вы набираете текст на клавиатуре, символы сохраняются в буфере. Когда две клавиши нажаты, символ будет переведен в один симовол Esperanto.

Лист. 323: Преобразование строки из двух символов в символ Esperanto.

```
Function GetTranslation( oString ) as String
  Select Case oString
    Case "^C", "Ch"
      GetTranslation = "Ĉ"
    Case "^c", "ch"
      GetTranslation = "ĉ"
    Case "^G", "Gh"
      GetTranslation = "Ĝ"
    Case "^g", "gh"
      GetTranslation = "ĝ"
    Case "^H", "Hh"
      GetTranslation = "Ĥ"
    Case "^h", "hh"
```

```
      GetTranslation = "ĥ"
    Case "^J", "Jh"
      GetTranslation = "Ĵ"
    Case "^j", "jh"
      GetTranslation = "ĵ"
    Case "^S", "Sh"
      GetTranslation = "Ŝ"
    Case "^s", "sh"
      GetTranslation = "ŝ"
    Case "uU", "Uh"
      GetTranslation = "Ŭ"
    Case "uu", "uh"
      GetTranslation = "ŭ"
  End Select
End Function
```

После того, как два нажатия клавиши преобразованы в символ Esperanto, этот символ вставляется в документ. Меня удивило, что oDocView и oKeyHandler передаются в качестве аргументов, потому что они доступны с помощью глобальных переменных.

Лист. 324: Вставить символ Esperanto в документ

```
Function InsertString( oString, oDocView, oKeyHandler )
  Dim oVCurs
  Dim oText
  Dim oCursor

  'oVCurs = ThisComponent.getCurrentController().getViewCursor()
  'oText = ThisComponent.getText()
  oVCurs = oDocView.getViewCursor()
  oText = oVCurs.getText()
  oCursor = oText.createTextCursorByRange( oVCurs.getStart() )
  'Вставка текста следует за нажатием клавиш (двойным событием!),
  'Удаляем обработчик (handler) перед вставкой, затем в конце добавляем его
снова
  oDocView.removeKeyHandler( oKeyHandler )
  oText.insertString( oCursor.getStart(), oString, true )
  oDocView.addKeyHandler( oKeyHandler )
End Function
```

Вызовем комбинированный макрос для создания и регистрации обработчика нажатий клавиш для текущего контроллера (controller) документа. Ссылка на обработчик нажатий клавиш сохраняется в глобальной переменной, так что он может быть удален позднее. Ссыока на текущий контроллер документа также сохраняется и поэтому также может быть удалена позже. После вызова макроса Лист. 325, каждое Ваше нажатие клавиши будет передано этому перехватчику (listener).

Лист. 325: Зарегистрировать преобразователь

```
Sub Compose
  oComposerDocView = ThisComponent.getCurrentController
```

```
  oComposerKeyHandler = createUnoListener( "Composer_", _
    "com.sun.star.awt.XKeyHandler" )
  oComposerDocView.addKeyHandler( oComposerKeyHandler )
  oComposerInputString = ""
End Sub
```

Удаление перехватчика нажатий клавиш (listener) — это легко.

Лист. 326: Удалить преобразователь

```
Sub ExitCompose
  oComposerDocView.removeKeyHandler( oComposerKeyHandler )
  oComposerInputString = ""
End Sub
```

Методы обработчика («handler»), который вызывается автоматически, имеют префикс «Composer_», как показано в Лист. 325. Этот перехватчик (listener), который создается, определяет методы, которые должны быть созданы. Первый метод делает немногое: возвращает false , как признак того, что данное событие не может быть обработано.

Лист. 327: Метод keyReleased .

```
Function Composer_keyReleased( oEvt ) as Boolean
  Composer_keyReleased = False
End Function
```

Когда клавиши нажимаются, они сохраняются в oComposerInputString. Событие содержит клавишу, которая только что нажата. Если oComposerInputString уже содержит один символ, то передаются два символа, и Лист. 323 используется для преобразования этой строки в подходящий символ. Преобразованный симовл вставляетс в документ с использованием макроса Лист. 324.

Лист. 328: Основной рабочий метод.

```
Function Composer_keyPressed( oEvt ) as Boolean
  If len( oComposerInputString ) = 1 Then
    oComposerInputString = oComposerInputString & oEvt.KeyChar
    Dim translation
    translation = GetTranslation( oComposerInputString )
    InsertString( translation, oComposerDocView, oComposerKeyHandler )
    oComposerInputString = ""
    ExitCompose
  Else
    oComposerInputString = oComposerInputString & oEvt.KeyChar
  End If
  Composer_KeyPressed = True
End Function
```

X.2. Перехватчик (Listener), описанный автором Paolo Mantovani

Текст в следующем разделе написан автором Paolo Mantovani. Я (Andrew Pitonyak) сделал очень немного изменений. Благодарю за потраченное время Paolo. Это одно из лучших описаний, которые я видел по этой теме. Полученная мной статья содержала следующее заявления, которое я повторяю!

© 2003 Paolo Mantovani

Документ распространяется по лицензии Public Documentation License Version 1.0

Копия этой лицензии доступна по ссылке
http://www.openoffice.org/licenses/pdl.pdf

X.2.1. Функция CreateUnoListener (создать перехватчик)

Среда времени выполнения OOo Basic предоставляет функцию, называемую CreateUnoListener, которая требует двух строковых аргументов: префиксх и полное имя (включая вышестоящие объекты) интерфейса перехватчика.

```
oListener = CreateUnoListener( sPrefix , sInterfaceName )
```

Эта функция очень хорошо описана в справкее OOo Basic.

```
sListenerName = «com.sun.star.lang.XEventListener"
oListener = CreateUnoListener("prefix_», sListenerName)
MsgBox oListener.Dbg_supportedInterfaces
MsgBox oListener.Dbg_methods
```

Интерфейс com.sun.star.lang.XEventListener является базовым интерфейсом для всех перехватчиков (listeners); поэтому он является наипростейшим перехватчиком. Интерфейс XEventListener работает только как базовый интерфейс для других перехватчиков, поэтому не следует использовать его напрямую; но для этого примера он отлично подходит.

X.2.2. Хорошо, но что этот макрос делает?

Макрос OOo Basic может вызывать методы и свойства API. Обычно макрос делает множество обращений к API. С другой стороны, само API обычно не могут вызывать процедуры OOo Basic. Рассмотрим следующий пример:

Лист. 329: Простой перехватчик событий (listener).

```
Sub Example_Listener
  sListenerName = "com.sun.star.lang.XEventListener"
  oListener = CreateUnoListener("prefix_", sListenerName)
```

```
  Dim oArg As New com.sun.star.lang.EventObject
  oListener.disposing( oArgument )
End Sub

Sub prefix_disposing( vArgument )
  MsgBox "Hi all!!"
End Sub
```

Когда процедура Example_Listener вызывает «oListener.disposing()», то фактически вызывается «prefix_disposing». Другими словами, функция CreateUnoListener создаёт сервис, который может вызывать Ваши процедуры OOo Basic.

Вы должны создать процедуры и функции с именами, совпадающими с именами метода перехватчика (listener), с добавлением префикса, указанного при вызове CreateUnoListener. Например, вызов CreateUnoListener передает первый аргумент как «prefix_» и процедура «prefix_disposing» начинается с «prefix_».

Документация для интерфейса com.sun.star.lang.XEventListener говорит, что аргумент должен быть структурой типа com.sun.star.lang.EventObject.

X.2.3. Как я знаю, какие методы создавать?

UNO требует от перехватчика, чтобы он вызывал Ваши макросы (requires a listener to call your macros). Когда Вы ходите использовать перехватчик (listener) для перехвата событий, Вам требуется объект UNO, способный разговаривать с Вашим перехватчиком. Этот объект UNO object, который создаёт вызов Вашего перехватчика, называется передатчиком (broadcaster). Объекты передатчика UNO поддерживают методы для добавления и удаления соответствующих перехватчиков.

Чтобы создать объект перехватчика, передайте полное имя (fully qualified name) интерфейса перехватчика в функцию CreateUnoListener. Извлеките методы, поддерживаемые перехватчиком, путем доступа к свойству Dbg_methods (или прочтите документацию IDL по интерфейсу перехватчика). Наконец, вставьте основную процедуру для каждого метода, даже для метода вывода (disposing method).

Многие сервисы UNO предоставляют методы для регистрации и удаления регистрации перехватчиков. Например, сервис com.sun.star.OfficeDocumentView поддерживает интерфейс com.sun.star.view.XSelectionSupplier. Это — передатчик (broadcaster). Этот интерфейс поддерживает следующие методы:

`addSelectionChangeListener`	Регистрирует перехватчик события, который вызывается, когда изменяется выделение (selection)
`removeSelectionChangeListener`	Удаляет регистрацию перехватчика

	события, который был зарегистрирован с помощью addSelectionChangeListener.

Оба метода принимают в качестве аргумента SelectionChangeListener (который является сервисом UNO, который поддерживает интерфейс com.sun.star.view.XSelectionChangeListener)

Объект передатчика (broadcaster) добавляет один или более аргументов в вызове. Первый аргумент — это структура UNO, а следующие аргументы зависят от определения интерфейса. Прочтите документацию IDL по интерфейсу перехватчика, который Вы используете, и посмотрите детальное описание его методов. Часто передаваемая структура является экземпляром com.sun.star.lang.EventObject. Как бы то ни было, все структуры событий должны быть расширением структуры com.sun.star.lang.EventObject, поэтому они имеют как минимум этот исходный пункт.

X.2.4. Пример 1: com.sun.star.view.XSelectionChangeListener

Следующий макрос является полной реализацией перехватчика события «изменение выделения» (selection change). Этот перехватчик может использоваться со всеми документами OpenOffice.org.

Лист. 330: Перехватчик изменения выделения.

```
Option Explicit

Global oListener As Object
Global oDocView As Object

' выполните этот макрос для начала перехвата событий
Sub Example_SelectionChangeListener
  Dim sName$
  oDocView = ThisComponent.getCurrentController

  ' создайте перехватчик для перехвата события "изменение выделения"
   sName = "com.sun.star.view.XSelectionChangeListener"
  oListener = CreateUnoListener( "MyApp_", sName )

  ' зарегистрировать этот перехватчик в контроллере документа
  oDocView.addSelectionChangeListener(oListener)
End Sub

' Выполните этот макрос для прекращения перехвата событий
Sub Remove_Listener
  ' удаляет перехватчик
  oDocView.removeSelectionChangeListener(oListener)
End Sub

' Все  перехватчики должны поддерживать это событие
Sub MyApp_disposing(oEvent)
  msgbox "Вывод перехватчика (disposing the listener)"
```

```
End Sub

Sub MyApp_selectionChanged(oEvent)
  Dim oCurrentSelection As Object
  ' Исходное свойствоо структуры события
  ' получает ссылку на текущее выделение
   oCurrentSelection =  oEvent.source
   MsgBox oCurrentSelection.dbg_properties
End Sub
```

Заметьте, что все методы перехватчика должны быть включены в Вашу программу на basic, потому что если вызвавший сервис не найдет подходящие процедуры, то произойдет ошибка выполнения.

Связанные ссылки по API:

http://api.openoffice.org/docs/common/ref/com/sun/star/view/OfficeDocumentView.html

http://api.openoffice.org/docs/common/ref/com/sun/star/view/XSelectionSupplier.html

http://api.openoffice.org/docs/common/ref/com/sun/star/view/XSelectionChangeListener.html

http://api.openoffice.org/docs/common/ref/com/sun/star/lang/EventObject.html

X.2.5. Пример 2: com.sun.star.view.XPrintJobListener

Объект, который может быть напечатан (будем называть его объектом документа), моежт поддерживать интерфейс com.sun.star.view.XPrintJobBroadcaster . Этот интерфейс позволяет регистрировать (и удалять регистрацию) com.sun.star.view.XPrintJobListener для перехвата событий во время печати. Когда Вы перехватываете события печати, Вы получаете структуру com.sun.star.view.PrintJobEvent . Эта структура имеет общее свойство «source» (источник); источником для этого события является Print Job (процесс печати), то есть сервис, который описывает текущий процесс печати и должен поддерживать интерфейс com.sun.star.view.XPrintJob.

Лист. 331: Перехватчик для процесса печати.

```
Option Explicit

Global oPrintJobListener As Object

' выполните этот макрос для начала перехвата событий
Sub Register_PrintJobListener

  oPrintJobListener = _
  CreateUnoListener("MyApp_", "com.sun.star.view.XPrintJobListener")

  ' эта функция определена в библиотеке  "Tools"
  'writedbginfo oPrintJobListener

  ThisComponent.addPrintJobListener(oPrintJobListener)
```

```
End Sub

' выполните этот макрос для прекращения перехвата событий
Sub Unregister_PrintJobListener

  ThisComponent.removePrintJobListener(oPrintJobListener)
End Sub

' все перехватчики должны поддерживать это событие
Sub MyApp_disposing(oEvent)
  ' здесь ничего не делается
End sub

' это событие вызывается несколько раз
' во время процесса печати
Sub MyApp_printJobEvent(oEvent)

  ' источником события printJob является PrintJob,
  ' это — сервис, который поддерживает интерфейс com.sun.star.view.XPrintJob
  '
  ' Этот сервис описывает текущий процесс печати
  MsgBox oEvent.source.Dbg_methods

  Select Case oEvent.State

    Case com.sun.star.view.PrintableState.JOB_STARTED
      Msgbox " печать (предоставление документа) началась"

    Case com.sun.star.view.PrintableState.JOB_COMPLETED
       sMsg = "печать (предоставление документа) "
       sMsg = sMsg & " закончилась, началась буферизация"
      Msgbox sMsg

    Case com.sun.star.view.PrintableState.JOB_SPOOLED
      sMsg = " буферизация закончилась успешно."
      sMsg = sMsg & " Это — единственное состояние, которое "
      sMsg = sMsg & "может считаться  'успехом'"
      sMsg = sMsg & " для процесса печати."
      Msgbox sMsg

    Case com.sun.star.view.PrintableState.JOB_ABORTED
      sMsg = "печать прервана (например, пользователем) "
      sMsg = sMsg & " во время печати или буферизации."
      Msgbox sMsg

    Case com.sun.star.view.PrintableState.JOB_FAILED
      sMsg = "печать привела к ошибке."
      Msgbox sMsg

    Case com.sun.star.view.PrintableState.JOB_SPOOLING_FAILED
      sMsg = "документ может быть напечатано, но не буферизован."
      Msgbox sMsg
  End Select
End sub
```

Связанные ссылки по API:

http://api.openoffice.org/docs/common/ref/com/sun/star/document/OfficeDocument.html
http://api.openoffice.org/docs/common/ref/com/sun/star/view/XPrintJobBroadcaster.html
http://api.openoffice.org/docs/common/ref/com/sun/star/view/XPrintJobListener.html
http://api.openoffice.org/docs/common/ref/com/sun/star/view/PrintJobEvent.html
http://api.openoffice.org/docs/common/ref/com/sun/star/view/XPrintJob.html

X.2.6. Пример 3: com.sun.star.awt.XKeyHandler

Обработчики (handlers) являются специальным типом обработчика. Как и все перехватчики (listeners), они могут перехватывать событие, но к тому же обработчик (handler) действует как потребитель события, другими словами, обработчик (handler) может «съесть» событие. В отличие от перехватчиков (listeners), методы в обработчиках (handlers) должны выдавать результат (булевый): True сообщает передатчику (broadcaster), что событие полностью обработано, так что передатчик не будет посылать это событие оставшимся обработчикам (handlers).

Обработчик com.sun.star.awt.XKeyHandler позволяет перехватывать события клавиш в документе. Данный пример демонстрирует обработчик событий клавиш, который действует как потребитель для событий нажатия некоторых клавиш (клавиш «t», «a», «b», «u») :

Лист. 332: Обработчик нажатий клавиш.

```
Option Explicit
Global oDocView
Global oKeyHandler

Sub RegisterKeyHandler
  oDocView = ThisComponent.getCurrentController
  oKeyHandler = _
  createUnoListener("MyApp_", "com.sun.star.awt.XKeyHandler")

'   writedbginfo oKeyHandler

  oDocView.addKeyHandler(oKeyHandler)
End Sub

Sub UnregisterKeyHandler
  oDocView.removeKeyHandler(oKeyHandler)
End Sub

Sub MyApp_disposing(oEvt)
' здесь ничего не делается
End Sub

Function MyApp_KeyPressed(oEvt) As Boolean
  select case oEvt.KeyChar
    case "t", "a", "b", "u"
       MyApp_KeyPressed = True
```

```
       msgbox "клавиша """ & oEvt.KeyChar & """ не разрешена!"
    case else
       MyApp_KeyPressed = False
  end select
End Function

Function MyApp_KeyReleased(oEvt) As Boolean
  MyApp_KeyReleased = False
End Function
```

Связанные ссылки по API:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XUserInputInterception.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XExtendedToolkit.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XKeyHandler.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/KeyEvent.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/Key.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/KeyFunction.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/KeyModifier.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/InputEvent.html

X.2.6.A. Небольшие дополнения от автора Andrew

Возник вопрос, как можно перехватить нажатия F1 или Alt+z. KeyChar для функциональных клавиш имеет ASCII значение, равное нулю. Проверяйте KeyCode для специальных символов. Хотя не не видел где-либо упоминаний об этом, Вам следует также проверить свойство MODIFIERS для уверенности, что клавиши control, alt и shift НЕ нажаты. В следующем примере я сравниваю напрямую модификатор клавиш MOD2, хотя это всего лишь флаг. Я только хочу ловить Alt+z и не ловить ни Ctrl+Alt+z, ни другой вариант с z.

Лист. 333: Ловить специальные символы в обработчике нажатий клавиш.

```
If oEvt.KeyCode = com.sun.star.awt.Key.F1 and oEvt.MODIFIERS = 0 Then
  MsgBox "Ха-ха, я НЕ позволю Вам использовать сегодня клавишуF1 !"
  MyApp_KeyPressed = True
  Exit Function
End If
If oEvt.KeyChar = "z" and _
          oEvt.MODIFIERS = com.sun.star.awt.KeyModifier.MOD2 Then
  MsgBox "Ха-ха, не НЕ позволю Вам использовать сегодня комбинациюAlt+z !"
  MyApp_KeyPressed = True
  Exit Function
End If
```

К сожалению, этот код программы может оказаться не способным находить комбинацию клавиш Alt+z. Если нажата CapsLock, то shift+z выдаст «z» вместо «Z», и модификатор будет иметь оба флага MOD2 и MOD1. Следующий пример ловит Alt+z, даже если нажата CapsLock.

Лист. 334: Ловить специальные символы в обработчике нажаний клавиш

```
  If oEvt.KeyChar = "z" and _
   ((oEvt.MODIFIERS and com.sun.star.awt.KeyModifier.MOD2) <> 0) and _
   ((oEvt.MODIFIERS  and com.sun.star.awt.KeyModifier.MOD1)  = 0) Then
    MsgBox "Ха-ха, я НЕ позволю Вам использовать сегодня комбинацию клавиш
Alt+z  !"
    MyApp_KeyPressed = True
    Exit Function
  End If
```

X.2.6.B. Замечания о клавишах модификации (Ctrl и Alt)

Обработчик событий нажатий клавиш показывает, нажаты ли в это время клавиши Ctrl и (или) Alt, он не различает левые и правые клавиши. Кроме того, если нажата только одна клавиша — клавиша Alt, то это вызывает обработчик нажатий клавиш, но это не так для нажатия клавиши Ctrl. key. Вот что написал Philipp Lohmann из фирмы Sun (ответ отредактирован):

Само по себе нажатие клавиши модификации не вызывает событие нажатия клавиш в VCL, но оно создаёт специальное событие «KeyModChange» (изменился подификатор). Оно не связано с AWT и поэтому не доступно для обработчика (customer) AWT. Более того, изменение клавиши-модификатора не каждый раз передается диспетчеру (dispatched), каждый раз передается только отжатие клавиши. Это событие KeyModChange было разработано для различения левой и правой клавиш shift — нажатия и отжатия, что позволяет переключить направление ввода текста, это и было причиной такого поведения.

В Windows клавиша Alt также работает как вызов меню — нажатие этой клавиши в одиночку делает активным меню. Эта функциональность эмулируется и другими операционными системами. Побочный эффект состоит в том, что клавиша ALT создаёт событие нажатия клавиш.

X.2.7. Пример 4: com.sun.star.awt.XMouseClickHandler

Этот обработчик позволяет перехватить нажатия клавиш мыши в документе.

Лист. 335: Полный обработчик нажатия клавиш мыши.

```
Option Explicit

Global oDocView As Object
Global oMouseClickHandler As Object

Sub RegisterMouseClickHandler
  oDocView = ThisComponent.currentController
  oMouseClickHandler = _
  createUnoListener("MyApp_", "com.sun.star.awt.XMouseClickHandler")
'  writedbginfo oMouseClickHandler
  oDocView.addMouseClickHandler(oMouseClickHandler)
```

```
End Sub

Sub UnregisterMouseClickHandler
  on error resume next
  oDocView.removeMouseClickHandler(oMouseClickHandler)
  on error goto 0
End Sub

Sub MyApp_disposing(oEvt)
End Sub

Function MyApp_mousePressed(oEvt) As Boolean
  MyApp_mousePressed = False
End Function

Function MyApp_mouseReleased(oEvt) As Boolean
Dim sMsg As String
  With oEvt
    sMsg = sMsg & "Modifiers = " & .Modifiers & Chr(10)
    sMsg = sMsg & "Buttons = " & .Buttons & Chr(10)
    sMsg = sMsg & "X = " & .X & Chr(10)
    sMsg = sMsg & "Y = " & .Y & Chr(10)
    sMsg = sMsg & "ClickCount = " &  .ClickCount & Chr(10)
    sMsg = sMsg & "PopupTrigger = " & .PopupTrigger '& Chr(10)
    'sMsg = sMsg & .Source.dbg_Methods
  End With

  ThisComponent.text.string = sMsg

  MyApp_mouseReleased = False
End Function
```

Связанные ссылки API:

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XUserInputInterception.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/XMouseClickHandler.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/MouseEvent.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/MouseButton.html

http://api.openoffice.org/docs/common/ref/com/sun/star/awt/InputEvent.html

X.2.8. Пример 5: Ручная привязка к событиям

Обычно программируя в OOo Basic, Вы не нуждаетесь в перехватчиках, потому что Вы можете вручную связать событие и макрос. Например, из диалога меню Сервис — Настройка — вкладка События (menu «Tools»=>»Configure..» — Tab «Events») можно привязаться к событиям приложения или событиям документа. Более того, многие объекты, которые Вы можете вставить в документ, предлагает диалог свойств на этой вкладке События. Наконец, диалоги и панели управления OOo Basic также имеют эту возможность.

Полезно заметить, что при ручной привязке используется тот же механизм, что и в перехватчиках, поэтому Вы можете добавить ещё параметр

— событие — в Ваш макрос, чтобы получить дополнительную информацию об этом событии.

Чтобы выполнить следующий макрос, откройте новый документ OOo Writer, добавьте управляющий элемент (кнопку) Text Edit и вручную привяжите данный макрос с событием нажатия этой кнопки. Заметьте, что имя макроса и имя события могут быть любыми.

Лист. 336: Вручную добивать обработчик события

```
Option Explicit

' Этот макрос вручную присоединен к событию нажатия клавиши
'  управляющего элемента в документе
Sub MyTextEdit_KeyPressed(oEvt)
Dim sMsg As String

  With oEvt
    sMsg = sMsg & "Modifiers = " & .Modifiers & Chr(10)
    sMsg = sMsg & "KeyCode = " & .KeyCode & Chr(10)
    sMsg = sMsg & "KeyChar = " & .KeyChar & Chr(10)
    sMsg = sMsg & "KeyFunc = " & .KeyFunc & Chr(10)
    sMsg = sMsg & .Source.Dbg_supportedInterfaces
  End With

  msgbox sMsg
End Sub
```

X.3. Что произошло с моим перехватчиком ActiveSheet?

Jim Thompson представил этот фрагмент программы для создания перехватчика события (Event Listener) изменения свойства «ActiveSheet» в текущем контроллере (controller). Этот перехватчик узнает, когда другой лист (sheet) выделен в том же самом документе, и выполняет обработку в зависимости от листа. Активация и деактивация режима предварительного просмотра страницы (меню Файл — Предварительный просмотр) (File | Page Preview), тем не менее, отключает перехватчик, поэтому теперь переход на другой лист больше не определяется. Следующий код программы действует как перехватчик и не демонстрирует полного решения:

Лист. 337: Перехватчик

```
'Автор: Jim Thompson
'Email: jimthompson5802@aol.com

Global oActiveSheetListener as Object
Global CurrentWorksheetName as String
Global oListeningController as Object

Sub Workbook_Open()
  Rem  Процедура Workbook_Open привязанк к событию "Document Open"
```

```
  Rem  Активировать различные перехватчики для событий во время обработки
  Rem  Включить перехватчик активации рабочего листа
  Call WorksheetActivationListenerOn
End Sub

Sub WorksheetActivationListenerOn
  CurrentWorksheetName = ""
  oListeningController = ThisComponent.CurrentController
  oActiveSheetListener = _
     createUnoListener("ACTIVESHEET_", _
                       "com.sun.star.beans.XPropertyChangeListener")
  oListeningController.addPropertyChangeListener("ActiveSheet", _
                       oActiveSheetListener)
End Sub

Sub WorksheetActivationListenerOff
  oListeningController.removePropertyChangeListener("ActiveSheet", _
                       oActiveSheetListener)
End Sub

Sub ACTIVESHEET_propertyChange(oEvent)
  REM  вызов подходящей процедуры деактивации рабочего листа
  Select Case CurrentWorksheetName
    Case "Example5"
      Call Example5Code.Worksheet_Deactivate
  End select

  ' msgbox "лист изменился: старый лист OldSheet =" & _
  ' CurrentWorksheetName & ", новый лист NewSheet=" & _
  ' oEvent.Source.ActiveSheet.Name
  REM  вызвать подходящую процедуру активации рабочего листа
  Select case oEvent.Source.ActiveSheet.Name
   Case "Example1"
     Call Example1Code.Worksheet_Activate
   Case "Example5"
     Call Example5Code.Worksheet_Activate
  End Select
  CurrentWorksheetName = oEvent.Source.ActiveSheet.Name
End Sub

Sub ACTIVESHEET_disposing(oEvent)
  msgbox "Вывод активного листа Disposing ACTIVESHEET"
End Sub
```

Согласно автору Mathias Bauer, объект контроллера документа (document's Controller) изменился, если изменился его вид (view). Решением будет — не регистрировать FrameActionListener с фреймом, который содержит этот контроллер (Controller). Каждый раз, когда новый компонент (в данном случае — контроллер Controller) присоединен (attached) к этому фрейму, этот фрейм посылает уведомление.

Глава XI. Язык (Language)

Я знаю, что по некоторым вопросам этот раздел неполон, он основан на очень старой версии OOo (не уверен, что язык сильно изменился), и он содержит несколько ошибок. Моя книга намного аккуратнее, полнее и современнее, покупайте её!

XI.1. Комментарии

Всегда полезно подробно комментировать свой код программы. То, что понятно сегодня, не будет понятно завтра. Символ одиночной кавычки и слово REM оба означают начало комментария. Весь текст после этого будет игнорироваться.

Лист. 338: Комментарии

```
REM  Это комментарий
REM и это тоже комментарий
' и ещё один комментарий
' это можно повторять ещё и ещё
Dim i As Integer  REM  i  используется как переменная индекса в циклах
Print i           REM  Этот оператор будет печатать значение i
```

XI.2. Переменные

XI.2.1. Имена

Имена переменных ограничены длиной 255 символов, они должны начинаться со стандартного символа алфавита и могут содержать цифры. Символ подчеркивания и символы пробелы — также допустимые символы. Не различаются большие и малые буквы. Имена переменных с пробелами должны быть заключены в квадратные кавычки «[ ]». Это расширено в более новых версиях OOo.

XI.2.2. Объявление/декларация переменных (Declaration)

Считается хорошей привычкой декларировать переменные перед их использованием. Оператор «Option Explicit» заставляет делать именно так. Этот оператор должен быть в первой строке кода программы. Если не использовать «Option Explicit», то, возможно, опечатки в именах переменных будут преследовать Вас как источник ошибок.

Чтобы объявить переменную, используйте оператор Dim. Синтакс Dim следующий:

```
[ReDim]Dim Name1 [(начало To конец)] [As тип][, Name2 [(начало To конец)]
[As тип][,...]]
```

Это позволяет объявлять несколько переменных за один раз. *Имя* — это любое имя переменной или массива. Значения *начало* и *конец* могут быть в интервале от -32768 до 32767. Это определяет индексы элементов (включительно), так что и *Name1(начало)* и *Name1(конец)* являются

допустимыми. Если используется оператор ReDim , то значения *начало* и *конец* могут быть числовыми выражениями. Возможные значения для *тип* включают: Boolean, Currency, Date, Double, Integer, Long, Object, Single, String, и Variant.

Variant является типом по умолчанию, если тип не указан, кроме случая команд DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, или DefVar. Эти команды позволяют сразу задать тип данных, основываясь на первой букве имени переменной.

String переменные/объекты ограничены размером 64K символов.

Variant переменные/объекты могут содержать все типы значений. Тип определяется при присвоении значения переменной.

Object переменные — после имени должен идти набор значений (subsequent set).

Следующий пример демонстрирует проблемы, которые могут возникнуть, если не объявлять переменные. Не объявленная переменная «truc» будет по умолчанию иметь тип Variant. Выполните этот макрос и посмотрите, какие типы использует не объявленная переменная:

Лист. 339: Некорректное объявление переменных

```
Sub TestNonDeclare
  Print "1 :  ", TypeName(truc), truc
  truc= "ab217"
  Print "2 :  ", TypeName(truc), truc
  truc= true
  Print "3 :  ", TypeName(truc), truc
  truc= 5=5 ' Должна быть булевой (Boolean)
  Print "4 :  ", TypeName(truc), truc
  truc= 123.456
  Print "5 :  ", TypeName(truc), truc
  truc=123
  Print "6 :  ", TypeName(truc), truc
  truc= 1217568942 ' должна быть длинным целым (Long)
  Print "7 :  ", TypeName(truc), truc
  truc= 123123123123123.1234 ' должна быть денежной суммой (Currency)
  Print "8 :  ", TypeName(truc), truc
End Sub
```

Это сильный аргумент за то, чтобы явно объявлять все переменные.

> ***Внимание*** *Каждый тип переменной должен быть объявлен, иначе переменная будет типа Variant. «Dim a, b As Integer» эквивалентно «Dim a As Variant, b As Integer».*

Лист. 340: Объявление переменных.

```
Sub MultipleDeclaration
  Dim a, b As Integer
  Dim c As Long, d As String
  Dim e As Variant, f As Float
  Print TypeName(a)  REM  пустая, по умолчанию будет Variant
  Print TypeName(b)  REM  целое Integer, объявлена как Integer
  Print TypeName(c)  REM  длинное целое Long, объявлена как Long
  Print TypeName(d)  REM  строка String, объявлена как String
  Print TypeName(e)  REM пустая, объявлена как Variant
  Print TypeName(f)  REM  объект Object, потому что Float является
неизвестным типом
  Print TypeName(g)  REM Object, потому что НЕ инициализировна
End Sub
```

XI.2.3. Страшные глобальные переменные (Global) и статические (Statics)

Глобальные переменные (Global) обычно нежелательны, потому что их может изменить любая процедура в любой момент, и трудно узнать, какие переменные какими методами изменены, если глобальные переменные используются. Поэтому я начал заголовок раздела со слова «страшные» (evil) перед термином «глобальные переменные» во время преподавания в университете The Ohio State University. Я использовал это слово для напоминания моим студентам, что хотя есть время и место для использования глобальных переменных, нужно хорошо подумать перед их использованием.

Глобальная переменная должна быть объявлена вне процедуры. Вы можете затем использовать ключевые слова Public и Private для указания, являются ли данная переменная глобальной для всех модулей или только для данного модуля. Если ни Public, ни Private явно не указаны, предполагается Private. Синтаксис тот же самый, что и для операторов Dim и ReDim

Хотя переменные передаются по ссылке, если только они не запрошены иным способом, глобальные переменные передаются по значению. Это вызвало по меньшей мере одну ошибку в моем коде программы.

Каждый раз, когда процедура вызывается, те переменные, которые являются локальными для этой процедуры, создаются заново. Если объявить переменную как Static, то она сохранит свое значение. В следующем примере процедура Worker Sub подсчитывает, сколько раз она была вызвана. Помните, что переменные Numeric инициализируются со значением нуль, а Strings инициализируются со значением пустой строки.

Лист. 341: Подсчёт вызова процедуры.

```
Option Explicit
Public Author As String   REM  глобальная Global для ВСЕХ модулей
Private PrivateOne$       REM  глобальная Global только для ЭТОГО модуля
```

```
Dim PrivateTwo$           REM  глобальная Global только для ЭТОГО модуля

Sub PublicPrivateTest
  Author = "Andrew Pitonyak"
  PrivateOne = "Hello"
  Worker()
  Worker()
End Sub

Sub Worker()
  Static Counter As Long  REM  сохраняет свое значение между вызовами
процедуры
  Counter = Counter + 1  REM  подсчитывает каждый раз, когда вызывается
Worker
  Print "Counter = " + Counter
  Print "Author = " + Author
End Sub
```

XI.2.4. Типы

Абстрактно говоря, OpenOffice.org Basic поддерживает типы переменных числовые (numeric), строковые (string), булевские (boolean), и объект (object). Объекты в основном используются для ссылки на встроенные типы, такие как документы, таблицы и т.д. Со значением переменной типа объекта можно использовать соответствующие методы и свойства. Числовые типы инициализируются равными нулю, а строки инициализируются равными пустой строке «».

Если Вам нужно узнать тип переменной во время выполнения программы, функция TypeName возвращает строковое представление типа переменной, а функция VarType возвращает целое число , соответствующее типу переменной.

Ключевое слово = TypeName	***Тип переменной***	***VarType***	***Автоматическое присвоение типа по первому символуAuto Type***	***Операторы объявления данного типа переменной Defxxx***
Boolean	Булевское	11		DefBool
Currency	Денежная сумма с 4 цифрами после десятичной точки	6	@	
Date	Дата	7		DefDate
Double	Двойное с плавающей	5	#	DefDbl

Ключевое слово = TypeName	*Тип переменной*	*VarType*	*Автоматическое присвоение типа по первому символуAuto Type*	*Операторы объявления данного типа переменной Defxxx*
	точкой Double Floating Point			
Integer	Целое	2	%	DefInt
Long	Длинное целое Long	3	&	DefLng
Object	Объект Object	9		DefObj
Single	Одинарное с плавающей точкой Single Floating Point	4	!	
String	Строка	8	$	
Variant	Может содержать все типы значений	12		DefVar
Empty	Переменная не инициализирована	0		
Null	Нет данных допустимого типа	1		

Макрос ExampleTypes демонстрирует поведение типов переменных.

Лист. 342: Пример типов переменных.

```
Sub ExampleTypes
  Dim b As Boolean             REM  Булевская 11
  Dim c As Currency            REM  Денежная сумма 6
  Dim t As Date                REM  Дата 7
  Dim d As Double              REM  Двойная с плавающей точкой 5
  Dim i As Integer             REM  Целое 2
  Dim l As Long                REM  Длинное целое Long 3
  Dim o As Object              REM  Объект Object 9
  Dim f As Single              REM  Одинарная с плавающей точкой  4
  Dim s As String              REM  Строка 8
  Dim v As Variant             REM  Пустая Empty 0
  Dim n As Variant : n = NULL REM  Без допустимых значений Null 1
  Dim x As Variant : x = f     REM   стала Одинарная с плавающей точкой 4
```

```
  Dim oData()
  Dim sName

  oData = Array(b, "b", c, "c", t, "t", d, "d", _
        i, "i", l, "l", o, "o", f, "f", s, "s", _
        v, "v", n, "n", x, "x")
  For i = LBound(oData()) To UBound(oData()) Step 2
    sName = oData(i+1)
    s = s & "TypeName(" & sName & ")=" & TypeName(oData(i)) & CHR$(10)
  Next
  s = s & CHR$(10)
  For i = LBound(oData()) To UBound(oData()) Step 2
    sName = oData(i+1)
    s = s & " VarType(" & sName & ")=" & VarType(oData(i)) & CHR$(10)
  Next
  MsgBox s
End Sub
```

XI.2.4.A. Булевские переменные

Хотя булевские переменные используют значения «True» или «False,» но внутренне они представлены целыми числами «-1» и «0» соответственно. Если присвоить булевской переменной что-либо, что не равно в точности «0», то в переменной будет сохранено «True». Типичное использование булевских переменных следующее:

Лист. 343: Использование булевских переменных.

```
Dim b as Boolean
b = True
b = False
b = (5 = 3)' Устанавливается равным False
Print b    ' Печатается 0
b = (5 < 7)' устанавливается равным True
Print b    ' печатается -1
b = 7      ' устанавливается равным True потому что 7 не равно 0
```

XI.2.4.B. Целые переменные (Integer)

Целые переменные (Integer) являются 16-битовыми числами, принимают значения от -32768 до 32767. Принимая число с плавающей точкой, целое число округляется до ближайшего целого. Если при объявлении переменной после её имени поставить символ «%» то переменная считается целой (Integer).

Лист. 344: Целые переменные.

```
Sub AssignFloatToInteger
  Dim i1 As Integer, i2%
  Dim f2 As Double
  f2= 3.5
  i1= f2
  Print i1  REM 4
  f2= 3.49
```

```
  i1= f2
  Print i1  REM 3
End Sub
```

XI.2.4.C. Переменные длинного целого (Long)

Длинные целые переменные (Long) являются 32-битовыми числами и принимают значения от -2,147,483,648 до 2,147,483,647. Если число с плавающей точкой присваивается длинной целой переменной, то оно округляется до ближайшего целого. Если при объявлении переменной после её имени поставить символ «&», то она является длинной целой (Long).

```
Dim Age&
Dim Dogs As Long
```

XI.2.4.D. Переменные денежной суммы (Currency)

Переменные денежной суммы (Currency) являются 64-битовыми с фиксированной точкой и четырьмя цифрами после десятичной точки и 15 цифрами до десятичной точки. Они принимают значения от -922,337,203,658,477.5808 до +922,337,203, 658,477.5807. Если при объявлении переменной после её имени поставить символ «@», то она становится денежной суммой (Currency).

```
Dim Income@
Dim Cost As Currency
```

XI.2.4.E. Переменные одиночные с плавающей точкой (Single)

Переменные одиночные с плавающей точкой (Single) являются 32-битовыми числами, наибольшая абсолютная величина их 3.402823 x 10E38. Наименьшая ненулевая абсолютная величина равна 1.401298 x 10E-45. Если при объявлении переменной после её имени поставить символ «!», она становится одиночной с плавающей точкой (Single).

```
Dim Weight!
Dim Height As Single
```

XI.2.4.F. Переменные двойной точности с плавающей точкой (Double)

Переменные двоойные с плавающей точкой (Double) являются 64-битовыми числами. Наибольшая абсолютная величина их 1.79769313486232 x 10E308. Наименьшая ненулевая абсолютная величина 4.94065645841247 x 10E-324. Если при объявлении переменной после её имени поставить символ «#», то она является двойной с плавающей точкой (Double).

```
Dim Weight#
Dim Height As Double
```

XI.2.4.G. Строковые переменные (String)

Строковые переменные используют один байт с кодом ASCII каждого символа и ограничены длиной 64Кбайта. Если при объявлении переменной после её имени поставить символ «$», то она является строковой (String).

```
Dim FirstName$
Dim LastName As String
```

XI.2.5. Объект (Object), произвольный (Variant), пустой (Empty), и отсутствующий (Null) значения и переменные

Два специальных значения «пустой» (Empty) и отсутствующий (Null) интересны при обсуждении типов переменных объект (Object) и (Variant). Значение Empty показывает, что никакое значение не было присвоено данной переменной. Это можно проверить функцией IsEmpty(переменная). Значение Null показывает, что не присутствует никакого допустимого значения. Это можно проверить функцией IsNull(переменная).

Когда переменная типа Object объявляется первый раз, она содержит значение Null. Когда переменная типа Variant объявляется первый раз, её значение равно Empty.

Лист. 345: Объявление переменных

```
Sub ExampleObjVar
  Dim obj As Object, var As Variant
  Print IsNull(obj)   REM True
  Print IsEmpty(obj)  REM False
  obj = CreateUnoService("com.sun.star.beans.Introspection")
  Print IsNull(obj)   REM False
  'obj = Null         REM Not valid...
  'Print IsNull(obj)  REM True

  Print IsNull(var)   REM False
  Print IsEmpty(var)  REM True
  var = obj
  Print IsNull(var)   REM True
  Print IsEmpty(var)  REM False
  var = 1
  Print IsNull(var)   REM False
  Print IsEmpty(var)  REM False
  'var = Empty        REM Not valid!
  'Print IsEmpty(var) REM True
End Sub
```

XI.2.6. Использовать тип переменной Object или Variant

При написании кода программы, которая взаимодействует с объектами UNO, нужно решить, какой тип переменной использовать, хотя большинство примеров использует тип Object.

Всегда используйте тип Variant для объявления переменных для объектов UNO, а не тип Object. В OpenOffice.org Basic тип переменной Object приспособлен для объектов самого OpenOffice.org Basic , но не для объектов UNO OpenOffice.org Basic. Тип переменных Variant наилучший для объектов UNO во избежание проблем, которые могут привести к особому поведению OpenOffice.org Basic в отношении переменных типа Object:

```
Dim oService1 ' Ok
oService1 = CreateUnoService( «com.sun.star.anywhere.Something» )
Dim oService2 as Object ' NOT recommended
oService2 = CreateUnoService( «com.sun.star.anywhere.SomethingElse» )
```

Andreas Bregas добавил, что в большинстве случаев оба варианта работают. Руководство Разработчика предпочитает тип variant, потому что в некоторых редких ситуациях использование типа object ведет к ошибке из-за устаревшей семантики (смысла) типа object. Но если программа использует тип object и работает правильно с этим типом, то проблем быть не должно.

XI.2.7. Константы (Constants)

OpenOffice.org Basic уже знает значения «True», «False» и «PI» (чило «пи»). Вы можете определить свои собственные константы. Каждая константа может быть определена один и только один раз. Для констант не указывается тип, они просто вставляются как имеющие определенный тип.

```
Const Gravity = 9.81
```

XI.2.8. Массивы (Arrays)

Массив позволяет сохранить множество различных значений в одной переменной. По умолчанию, первый элемент массива имеет индекс (location) 0. Тем не менее, можно указать начальное и конечное значения индекса массива. Вот несколько примеров

Лист. 346: Примеры массивов

```
Dim a(5) As Integer          REM 6 элементов от 0  до 5  включительно
Dim b$(5 to 10) As String    REM 6  элементов от 5  до 10 включительно
Dim c(-5 to 5) As String     REM 11 элементов от -5 до 5  включительно
Dim d(5 To 10, 20 To 25) As Long
```

Если есть массив типа variant и нужно быстро его заполнить, используйте функцию Array. Эта функция возвращает массив типа Variant с включенными в него данными. Вот как можно создать список данных.

Лист. 347: Заполнение массива

```
Sub ExampleArray
  Dim a(), i%
  a = Array(0, 1, 2)
```

```
  a = Array("Zero", 1, 2.0, Now)
  REM String, Integer, Double, Date
  For i = LBound(a()) To UBound(a())
    Print TypeName(a(i))
  Next
End Sub
```

XI.2.8.A. Опция начала индексов массивов Option Base

Можно изменить нижнюю границу индексов массивов по умолчанию, сделав её 1 вместо 0. Это должно быть сделано до каких-либо исполняемых операторов программы

Синтаксис:

```
Option Base { 0 | 1 }
```

XI.2.8.B. Функция Lbound(ИмяМассива[,НомерРазмерности])

Возвращает нижнюю границу индексов массива. Необязательный второй параметр означает номер размерности многомерного массива, для которой требуется получить нижнюю границу, начинается с 1 (а не с 0) для первой размерности, далее 2 для второй и т.д. Пример для массивов из пункта 10.2.8

Лист. 348: Пример массивов

```
LBound(a())     REM 0
LBound(b$())    REM 5
LBound(c())     REM -5
LBound(d())     REM 5
LBound(d(), 1)  REM 5
LBound(d(), 2)  REM 20
```

XI.2.8.C. Функция UBound(ИмяМассива[,НомерРазмерности])

Возвращает верхнюю границу индексов массива. Необязательный второй параметр является номером размерности многомерного массива, для которой нужно получить верхнюю границу, причем нумерация размерностей начинается с 1 (а не с 0).

Лист. 349: Верхняя граница индексов массивов

```
UBound(a())     REM 5
UBound(b$())    REM 10
UBound(c())     REM 5
UBound(d())     REM 10
UBound(d(), 1)  REM 10
UBound(d(), 2)  REM 25
```

XI.2.8.D. Определен ли данный массив

Если массив фактически не содержит ни одного элемента, то нижняя граница индексов больше верхней границы индексов этого массива

XI.2.9. Изменение размерности DimArray

Функция DimArray используется для установки или изменения числа размерностей и числа индексов каждой размерности массива типа Variant. DimArray(2, 2, 4) означает то же, что и DIM a(2, 2, 4).

Лист. 350: Управление массивами

```
Sub ExampleDimArray
  Dim a(), i%
  a = Array(0, 1, 2)
  Print "" & LBound(a()) & " " & UBound(a())    REM 0 2
  a = DimArray()
  ' Empty array
  i = 4
  a = DimArray(3, i)
  Print "" & LBound(a(),1) & " " & UBound(a(),1)  REM 0, 3
  Print "" & LBound(a(),2) & " " & UBound(a(),2)  REM 0, 4
End Sub
```

XI.2.10. Изменение числа элементов массива ReDim

Оператор ReDim используется для изменения размера массива

Лист. 351: Изменение размерностей массивов

```
Dim e() As Integer      REM  Пока не указан размер массива
ReDim e(5) As Integer   REM  индексы от 0  до 5 допустимы
ReDim e(10) As Integer  REM  индексы от 0  до 10  допустимы
```

Ключевое слово Preserve может использоваться с оператором ReDim для сохранения содержимого массива, размерность которого изменяется.

Лист. 352: Сохранение содержимого массивов

```
Sub ReDimExample
  Dim a(5) As Integer
  Dim b()
  Dim c() As Integer
  a(0) = 0
  a(1) = 1
  a(2) = 2
  a(3) = 3
  a(4) = 4
  a(5) = 5
  REM  a имеет индексы от 0  до 5 , пока что a(i) = i
  PrintArray("a от начала", a())
  REM a  меняет размерность от 1 до 3, пока a(i) = i
  ReDim Preserve a(1 To 3) As Integer
  PrintArray("a после применения ReDim", a())
  REM Array()  возвращает тип variant
  REM  b  получает размерность от 0 до 9, пока b(i) = i+1
  b = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
  PrintArray("b при первичном присвоении значений ", b())
```

```
  REM  b  получает размерность от 1 до 3, пока b(i) = i+1
  ReDim Preserve b(1 To 3)
  PrintArray("b после применения ReDim", b())
  REM  следующий оператор был бы НЕ правильным
  REM  потому что этот массив уже имеет размерность
  REM  причем другую размерность
  REM a = Array(0, 1, 2, 3, 4, 5)

  REM  c  получает размерность от 0 до 5, пока a(i) = i
  REM Если бы "ReDim" уже была выполнена для массива c, то это НЕ сработало
бы
  c = Array(0, 1, 2, 3, 4, 5)
  PrintArray("c, теперь типа Integer после присвоения значений ", c())
  REM  Вы будете смеяться, но этот оператор разрешен, хотя  c не будет
содержать никаких данных после этого!
  ReDim Preserve c(1 To 3) As Integer
  PrintArray("c после применения ReDim", c())
End Sub

Sub PrintArray (lead$, a() As Variant)
  Dim i%, s$
  s$ = lead$ + Chr(13) + LBound(a()) + " to " + _
                         UBound(a()) + ":" + Chr(13)
  For i% = LBound(a()) To UBound(a())
    s$ = s$ + a(i%) + " "
  Next
  MsgBox s$
End Sub
```

Функция Array, упомянутая выше, создаёт только массив типа Variant. Чтобы инициализировать массив нужного типа, можно использовать следующий способ

Лист. 353: Инициализация массивов

```
Sub ExampleSetIntArray
  Dim iA() As Integer
  SetIntArray(iA, Array(9, 8, 7, 6))
  PrintArray("", iA)
End Sub
Sub SetIntArray(iArray() As Integer, v() As Variant)
  Dim i As Long
  ReDim iArray(LBound(v()) To UBound(v())) As Integer
  For i = LBound(v) To UBound(v)
    iArray(i) = v(i)
  Next
End Sub
```

XI.2.11. Изучение объектов

Чтобы определить тип переменной, можно использовать булевские функции IsArray, IsDate, IsEmpty, IsMissing, IsNull, IsNumeric, IsObject и IsUnoStruct. Функция IsArray возвращает значение true, если парамтр является массивом. Функция IsDate возвращает значение true, если

оказывается возможным конвертировать этот объект в дату (Date). Строка с правильно отформатированной датой поэтому даст значение true этой функции IsDate. Метод IsEmpty используется для проверки, был ли инициализирован объект типа Variant. Функция IsMissing показывает, отсутствует ли необязательный параметр (Optional). Метод IsNull проверяет, содержит ли переменная типа Variant значение Null, что означает, что эта переменная не содержит данных. Функция IsNumeric используется для проверки, содержит ли строка числовые значения. Функция IsUnoStruct принимает в качестве параметра имя структуры UNO и возвращает значение true, только если эта строка является реальным именем структуры.

XI.2.12. Операторы сравнения

Операторы сравнения обычно работают так, как от них ожидают, но они не выполняют быстрых вычислений. Работают следующие стандартные операторы сравнения.

Оператор	*Значение*
=	Равно Equal To
<	Меньше Less Than
>	Больше Greater Than
<=	Меньше или равно Less Than or Equal To
>=	Больше или равно Greater Than or Equal To
<>	Не равно Not Equal To
Is	Является ли тем же самым объектом Are these the same Object

Оператор and выполняет логическую операцию над булевскими типами и побитовую операцию над числовыми типами. Оператор OR выполняет логическую операцию над булевскими типами и побитовую операцию над числовыми типами. Оператор XOR выполняет логическую операцию над булевскими типами и побитовую операцию над числовыми типами. Помните, что это «Исключительное ИЛИ». Оператор NOT выполняет логическую операцию над булевскими типами и побитовую операцию над числовыми типами. Простой тест показывает, что стандартное правило старшинства операторов действует, а именно, что and старше (выполняется первым), чем операторы or.

Лист. 354: Старшинство операторов

```
Option Explicit
Sub ConditionTest
```

```
  Dim msg As String
  msg = "AND has "
  msg = msg & IIf(False OR True and False, "equal", "greater")
  msg = msg & " precedence than OR" & Chr(13) & "OR does "
  msg = msg + IIF(True XOR True OR True, "", "not ")
  msg = msg + "have greater precedence than XOR" + Chr(13)
  msg = msg & "XOR does "
  msg = msg + IIF(True OR True XOR True, "", "not ")
  msg = msg + "have greater precedence than OR"
  MsgBox msg
End Sub
```

XI.3. Функции (Functions) и процедуры (SubProcedures)

Функция — это процедура (Sub procedure), которая может возвращать результатом какое-либо значение. Это позволяет использовать её в выражениях. Начинаются процедуры и функции (Subs и Functions) так:

Синтаксис:

```
Function FuncName[(Var1 [As Type][, Var2 [As Type][,...]]]) [As Type]
Sub SubName[(Var1 [As Type][, Var2 [As Type][,...]]])
```

Функции объявляют (declare) тип возвращаемого значения, потому что они возвращают значение. Чтобы задать возвращаемое значение, используйте оператор в форме «ИмяФункции = ВозвращаемоеЗначение». Хотя можно выполнить подобное присвоение несколько раз, возвращаемым значением будет то, которое задано последним присвоением.

Чтобы немедленно выйти из процедуры, используйте соответствующий оператор Exit.

XI.3.1. Необязательные параметры

Параметр может быть объявлен как необязательный путем использования ключевого слова Optional. Метод IsMissing используется тогда для того, чтобы определить, был ли этот параметр получен.

Лист. 355: Объявление необязательного параметра

```
Sub testOptionalParameters()
  Print TestOpt()    REM MMM
  Print TestOpt(,)    REM MMM
  Print TestOpt(,,)    REM MMM
  Print TestOpt(1)    REM 1MM
  Print TestOpt(1,)  REM 1MM
  Print TestOpt(1,,)  REM 1MM
  Print TestOpt(1,2)  REM 12M
  Print TestOpt(1,2,)  REM 12M
  Print TestOpt(1,2,3)  REM 123
  Print TestOpt(1,,3)  REM 1M3
  Print TestOpt(,2,3)  REM M23
```

```
  Print TestOpt(,,3)  REM MM3
  Print TestOptI()    REM MMM
  Print TestOptI(,)    REM 488MM (Error)
  Print TestOptI(,,)  REM 488488M (Error)
  Print TestOptI(1)  REM 1MM
  Print TestOptI(1,)  REM 1MM
  Print TestOptI(1,,)  REM 1488M (Error)
  Print TestOptI(1,2)  REM 12M
  Print TestOptI(1,2,)  REM 12M
  Print TestOptI(1,2,3)REM 123
  Print TestOptI(1,,3)  REM 14883 (Error)
  Print TestOptI(,2,3)  REM 48823 (Error)
  Print TestOptI(,,3)  REM 4884883 (Error)
End Sub
Function TestOpt(Optional v1 As Variant, Optional v2 As Variant, Optional v3
As Variant) As String
  Dim s As String
  s = "" & IIF(IsMissing(v1), "M", Str(v1))
  s = s  & IIF(IsMissing(v2), "M", Str(v2))
  s = s  & IIF(IsMissing(v3), "M", Str(v3))
  TestOpt = s
End Function
Function TestOptI(Optional i1 As Integer, Optional i2 As Integer, Optional
i3 As Integer) As String
  Dim s As String
  s = "" & IIF(IsMissing(i1), "M", Str(i1))
  s = s  & IIF(IsMissing(i2), "M", Str(i2))
  s = s  & IIF(IsMissing(i3), "M", Str(i3))
  TestOptI = s
End Function
```

> ***Внимание*** *Что касается версии 1.0.3.1, IsMissing будет приводить к ошибке для необязательных (Optional) параметров , если и тип не Variant и отсутствующий необязательный параметр представлен двумя последовательными запятыми. Я начал это исследовать после разговора с Christian Anderson [ca@ofs.no]. Это статья номер 11678 в issuezilla.*

XI.3.2. Передача параметров по ссылке (By Reference) или по значению (Value)

Если переменная передана по значению (by value), то можно изменить параметр в вызванной процедуре, а исходная переменная не изменится. Если параметр передан по ссылке (reference), то при изменении параметра меняется и исходная переменная. По умолчанию параметр передается по ссылке. Чтобы передать его по значению, используйте ключевое слово ByVal перед объявлением параметра. Если параметр является константой,

например, «4», и Вы изменяете этот параметр в вызванной процедуре, то константа может измениться, а может и не измениться. Согласно автору Andreas Bregas [ab@openoffice.org], это ошибка OOo, поэтому я создал статью в issuezilla

http://www.openoffice.org/project/www/issues/show_bug.cgi?id=12272

Лист. 356: Передача параметров

```
Option Explicit
Sub LoopForever
  Dim l As Long
  l = 4
  LoopWorker(l)
  Print "Передан параметр  l по значению и он пока остался  " + l
  LoopForeverWorker(l)
  ' l теперь 1 поэтому будет напечатано 1.
  Print "Передан параметр l по ссылке и теперь он равен " + l
  ' Этот цикл будет выполняться бесконечно, потому что 4 является константой
и её НЕЛЬЗЯ
  ' изменить
  Print "Передача константы как параметра по ссылке, это будет забавно"
  Print LoopForeverWorker(4)
End Sub

Sub LoopWorker(ByVal n As Long)
  Do While n > 1
    Print n
    n = n — 1
  Loop
End Sub

Sub LoopForeverWorker(n As Long)
  Do While n > 1
    ' Это забавно, когда n является константой
    Print n
    n = n — 1
  Loop
End Sub
```

XI.3.3. Рекурсия (Recursion)

Ваши функции могут быть рекурсивными для версии OOo 1.1.1. Разные версии OOo под разными операционными системами поддерживают разные уровни рекурсии, поэтому будьте осторожны.

Лист. 357: Рекурсии

```
Option Explicit
Sub DoFact
  Print "Recursive = " + RecursiveFactorial(4)
  Print "Normal Factorial = " + Factorial(4)
End Sub
```

```
Function Factorial(n As Long) As Long
  Dim answer As Long
  Dim i As Long
  i = n
  answer = 1
  Do While i > 1
    answer = answer * i
    i = i – 1
  Loop
  Factorial = answer
End Function

' Это приведет к ошибке, потому что Вы не можете использовать рекурсию
Function RecursiveFactorial(n As Long) As Long
  If n > 1 Then
    RecursiveFactorial = n * RecursiveFactorial(n-1)
  Else
    RecursiveFactorial = 1
  End If
End Function
```

XI.4. Управление последовательностью выполнения программы (Flow Control)

XI.4.1. Операторы If Then Else

Конструкция If использована для выполнения блока текста программы на основании значения выражения. Хотя можно использовать GoTo или GoSub для выхода из блока If, нельзя перескочить внутрь блока If.

Синтаксис:

```
If condition=true Then
  Statementblock
[ElseIf condition=true Then]
  Statementblock
[Else]
  Statementblock
End If

If condition=true Then Statement
```

Пример:

Лист. 358: Операторы If Then Else

```
If x < 0 Then
  MsgBox "Число отрицательное"
ElseIf x > 0 Then
  MsgBox "Число положительное"
Else
  MsgBox "Число нулевое"
```

```
End If
```

XI.4.2. IIF

Конструкция IIF используется для возвращения выражения в зависимости от условия. Это похоже на синтаксис «?» в языке C.

Синтаксис:

```
IIf (Условие, ВыраженияДляTrue, ВыраженияДляFalse)
```

Это очень похоже на следующий код программы:

```
If (Condition) Then
  object = TrueExpression
Else
  object = FalseExpression
End If
max_age = IIf(johns_age > bills_age, johns_age, bills_age)
```

XI.4.3. Оператор Choose

Оператор choose позволяет выбрать из списка значений, основываясь на значении индекса.

Синтаксис:

```
Choose (Индекс, Выбор1[, Выбор2, ... [,Выбор_n]])
```

Если индек равен1, то возвращается первый элемент Выбор1. Если индекс равен 2, то возвращается второй элемент Выбор2. И так далее!

XI.4.4. For....Next

Справка OOo содержит прекрасное полное описание, прочтите её.

Повторение блока операторов заданное число раз.

Синтаксис:

```
For counter=start To end [Step step]
  statement block
  [Exit For]
  statement block
Next [counter]
```

Числовой счетчик (counter) первоначально получает значение «start». Если не задан шаг изменения счетчика (step), то счетчик увеличивается на 1 до тех пор, пока не превзойдет значение «end». Если задано значение шага «step», то этот шаг «step» добавляется до тех пор, пока не превзойдет значение «end». Блоки операторов выполняются один раз для каждого значения счетчика.

Счетчик «counter» для оператора «Next» не обязателен, Next автоматически ссылается на ближайший действующий оператор «For».

Можно досрочно прекратить выполнение цила, использовав оператор «Exit For». Этот оператор прекращает выполнения цикла ближайшего действующего оператора «For».

Пример:

Следующий пример заполняет массив случайными целыми числами. Этот массив затем сортируется двумя вложенными циклами (loops).

Лист. 359: Заполнение массива случайными числами

```
Sub ForNextExampleSort
  Dim iEntry(10) As Integer
  Dim iCount As Integer, iCount2 As Integer, iTemp As Integer
  Dim bSomethingChanged As Boolean

  ' заполняем этот массив целыми числами со значениями между -10 и 10
  For iCount = LBound(iEntry()) To Ubound(iEntry())
    iEntry(iCount) = Int((20 * Rnd) -10)
  Next iCount

  ' Сортируем массив
  For iCount = LBound(iEntry()) To Ubound(iEntry())

    ' Предполагаем, что массив отсортирован
    bSomethingChanged  = False
    For iCount2 = iCount + 1 To Ubound(iEntry())
      If iEntry(iCount) > iEntry(iCount2) Then
        iTemp = iEntry(iCount)
        iEntry(iCount) = iEntry(iCount2)
        iEntry(iCount2) = iTemp
        bSomethingChanged  = True
      End If
    Next iCount2
    'Если массив уже отсортирован, то прекращаем выполнение цикла!
    If Not bSomethingChanged Then Exit For
  Next iCount
  For iCount = 1 To 10
    Print iEntry(iCount)
  Next iCount
End Sub
```

XI.4.5. Do Loop

Справка OOo содержит прекрасное полное описание, прочтите его.

Конструкция Loop имеет несколько различных форм и используется для продолжения выполнения блока кода программы до тех пор, пока условие равно true. Наиболее обычная форма проверяет условие до того, как цикл начинается, и до тех пор, пока условие равно true, в цикле повторяется выполнение блока текста программы. Если условие равно false, то цикл не выполняется ни одного раза.

```
Do While condition
```

```
  Block
Loop
```

Похожая, но не такая частая форма проверяет условие до того, как начинается выполнение цикла, и до тех пор, пока это условие равно false, в цикле будет повторно выполняться блок кода программы. Если это условие равно true с самого начала, то цикл не выполняется ни одного раза.

```
Do Until condition
  Block
Loop
```

Можно также поместить проверку условия в конце цикла, в этом случае блок кода программы будет обязательно выполнен хотя бы один раз. Чтобы всегда выполнить цикл хотя бы один раз, а затем выполнять цикл до тех пор, пока условие равно true, используется следующая конструкция:

```
Do
  Block
Loop While condition
```

Чтобы всегда выполнить цикл хотя бы один раз и затем продолжать выполнение до тех пор, пока условие остается равным false, используйте следующую конструкцию:

```
Do
  Block
Loop Until condition
```

В цикле «Do Loop» можно немедленно выйти из цикла с помощью оператора «Exit Do».

XI.4.6. Оператор Select Case

Оператор Select Case похож на операторы «case» и «switch» в других языках. Он имитирует несколько блоков «Else If» в операторе «If». Указывается единственное выражения для условия и его результат сравнивается с несколькими значениями , как показано ниже:

Синтаксис:

```
Select Case ВыражениеУсловия 'condition_expression
  Case ВыражениеПроверки1 'case_expression1
    Блок операторов 1
  Case ВыражениеПроверки2 'case_expression2
    Блок операторов 2
  Case Else
    Блок операторов 3
End Select
```

ВыражениеУсловия — это выражение, результат которого будет сравниваться в каждом операторе Case. Я не забочусь о конкретном типе

значений и об ограничениях при сравнениях, единственное — что типы должны быть совместимы с типами выражений в операторах Case. Выполняется блок операторов, для которого первым совпадут значения выражений Select Case и Case. Если ни одно из выражений не подошло, то выполняется (необязательный) блок после Case Else.

XI.4.6.A. Выражения в операторах Case

Выражения в операторах case обычно являются константами, например, «Case 4» или «Case "hello"». Можно указать несколько значений через запятую: «Case 3, 5, 7». Если нужно проверить на попадание в интервал, то можно использовать ключевое слово «To», например, «Case 5 To 10». Можно указать открытый односторонний интервал — коротко так: «Case < 10» или с ключевым словом «Is» вот так:«Case Is < 10».

> ***Внимание*** *Будьте осторожны при использовании интервала в операторе Case. Справка on-line несколько раз предлагает некорректные примеры, такие как «Case i > 2 and i < 10». Это трудно понять и трудно правильно запрограммировать.*

XI.4.6.B. Простой, но неправильный пример

Я видел много некорректных примеров, поэтому я потрачу время и покажу несколько примеров, которые не работают. Начну с очень простого примера:

Правильно	*Правильно*	*Неверно*

`i = 2`
`Select Case  i`
`  Case  2`
	`i = 2`
	`Select Case  i`
	`  Case  Is = 2`
		`i = 2`
		`Select Case  i`
		`  Case  i = 2`

Неверный пример приводит к ошибке, потому что «Case i = 2» означает «Case Is = (i = 2)». Выражение (i=2) дает результат True, который преобразуется в число -1 так что получается «Case Is = -1» в данном примере.

Если Вы поняли этот простой неверный пример, то Вы готовы к более трудным.

XI.4.6.C. Неверный пример с интервалом

Следующий неверный пример был в Справке on-line.

```
Case Is > 8 and iVar < 11
```

Это не работает, так как он вычисляется как :

```
Case Is > (8 and (iVar < 11))
```

Выражение (iVar<11) вычисляется как true или false. Напомню, что true= -1 и false=0. Оператор and тогда применяется побитно к числам 8 и -1 (это true) или 0 (это false), что дает результат или 8 или 0. Это выражение дает одно из двух выражений.

Если iVar меньше чем 11 :

```
Case Is > 8
```

Если iVar больше или равно 11 :

```
Case Is > 0
```

XI.4.6.D. Неверный пример с интервалом

Я видел также следующий неверный пример.

```
Case  i > 2 and i < 10
```

Это не работает, потому что вычисляется как

```
Case  Is = (i > 2 and i < 10)
```

XI.4.6.E. Интервалы, правильный способ

Оператор

```
Case    Выражение
```

вероятно, правильный, если его можно записать

```
Case Is = (Выражение)
```

Мое первое решение таково:

```
Case  Iif(БулевскоеВыражение, i, i+1)  ' здесь i — выражение из Select Case,
так что i+1 обязательно не совпадет
```

Я гордился собой, пока не получил следующее прекрасное решение от Bernard Marcelly:

```
Case i XOR NOT (БулевскоеВыражение)
```

После первого затруднения я понял, какое замечательное это решение. Не поддавайте искушению упростить его, заменив на «i and ()», потому что это даст ошибку, если i равно 0. Я сам делал такую ошибку.

Лист. 360: Операторы Case

```
Sub DemoSelectCase
  Dim i%
  i = Int((15 * Rnd) -2)
  Select Case i%
    Case 1 To 5
```

```
      Print "Число от 1 до 5"
    Case 6, 7, 8
      Print "Число от 6 до 8"
    Case IIf(i > 8 and i < 11, i, i+1)
      Print "Больше 8"
    Case i% XOR NOT(i% > 8 and i% < 11 )
      Print i%, "Число 9 или 10"
    Case Else
      Print "Вне интервала от 1 до 10"
    End Select
End Sub
```

XI.4.7. Цикл While...Wend

Ничего особенного нет в конструкции While...Wend, её синтаксис такой:

Синтаксис:

```
While Condition
  Код программы
Wend
```

> ***Совет*** *Эта конструкция имеет ограничения, которых нет в конструкции Do While...Loop и при этом не дает никаких преимуществ. Вы не можете ни использовать оператор Exit, ни выйти из цикла с помощью GoTo.*

XI.4.8. Переход к процедуре GoSub

Оператор GoSub вызывает переход управления к метке указанной процедуры (subroutine) внутри текущей процедуры. Нельзя передать управление вовне текущей процедуры. Когда выполнится оператор Return , выполнение будет продолжено с того места, откуда был сделан первоначальный вызов (GoSub). Если оператор Return выполняется, когда не было предыдущего оператора GoSub, то происходит ошибка. Другими словами, Return не заменяет Exit Sub или Exit Function. Обычно предполагается, что использование функций и процедур создаёт более понятный код программы, чем использование операторов GoSub и GoTo.

Лист. 361: Переходы

```
Option Explicit
Sub ExampleGoSub
  Dim i As Integer
  GoSub Line2
  GoSub Line1
  MsgBox "i = " + i, 0, "GoSub Example"
  Exit Sub
```

```
Line1:
  i = i + 1
  Return
Line2:
  i = 1
  Return
End Sub
```

Совет *GoSub — это отстатки от устаревших вариантов языка, сохраненные для совместимости. GoSub очень не рекомендуется, потому что она ведет к созданию нечитаемого кода программы. Предпочтительны процедуры или функции.*

XI.4.9. Оператор GoTo

Оператор GoTo вызывает передачу управления на указанную метку внутри текущей процедуры. Нельзя передать управление вовне текущей процедуры.

Лист. 362: GoTo

```
Sub ExampleGoTo
  Dim i As Integer
  GoTo Line2
Line1:
  i = i + 1
  GoTo TheEnd
Line2:
  i = 1
  GoTo Line1
TheEnd:
  MsgBox "i = " + i, 0, "GoTo Example"
End Sub
```

Совет *Оператор GoTo является остатком от устаревших вариантов языка, сохраненный для совместимости. Оператор GoTo строго не рекомендуется, потому что он приводит к созданию нечитаемого кода программы. Предпочтительны процедуры и функции.*

XI.4.10. Конструкция On ... GoTo

Синтаксис:

```
On N GoSub Label1[, Label2[, Label3[,...]]]
On N GoTo Label1[, Label2[, Label3[,...]]]
```

Эти операторы передают управление на метку в зависимости от результата числового выражения *N*. Если (*N=0*), то передачи управления не происходит. Числовое выражение *N* должно быть в интервале от 0 до 255. Это похоже на операторы других языков «вычисляемый goto,» «case,» и «switch,» Не пытайтесь передать управление вовне текущей процедуры или функции.

Лист. 363: Передача управления на метку

```
Option Explicit
Sub ExampleOnGoTo
  Dim i As Integer
  Dim s As String
  i = 1
  On i+1 GoSub Sub1, Sub2
  s = s & Chr(13)
  On i GoTo Line1, Line2
  REM следующий оператор exit приведет к выходу из процедуры, когда
продолжение выполнения не требуется
  Exit Sub
Sub1:
  s = s & "In Sub 1" : Return
Sub2:
  s = s & "In Sub 2" : Return
Line1:
  s = s & "At Label 1" : GoTo TheEnd
Line2:
  s = s & "At Label 2"
TheEnd:
  MsgBox s, 0, "On GoTo Example"
End Sub
```

XI.4.11. Операторы Exit

Операторы Exit используются для выхода из циклов Do Loop, For Next, функции Function или процедуры Sub. Попытка выйти, если нет такой конструкции, содержащей данный код программы, вызовет ошибку. Например, попытка выйти из цикла For loop, если не находитесь внутри этого цикла. Варианты операторов следующие:

Exit DO Управление передается после соответствующего оператора Loop

Exit For Управление передается после соответствующего оператора Next

Exit Function Немедленный выход из текущей функции

Exit Sub Немедленный выход из текущей процедуры Sub.

Лист. 364: Выходы

```
Option Explicit
Sub ExitExample
  Dim a%(100)
  Dim i%
  REM Fill the array with 100, 99, 98, ..., 0
  For i = LBound(a()) To UBound(a())
    a(i) = 100 — i
  Next i
  Print SearchIntegerArray(a(), 0  )
  Print SearchIntegerArray(a(), 10 )
  Print SearchIntegerArray(a(), 100)
  Print SearchIntegerArray(a(), 200)
End Sub

Function SearchIntegerArray( list(), num%) As Integer
  Dim i As Integer
  SearchIntegerArray = -1
  For i = LBound(list) To UBound(list)
    If list(i) = num Then
      SearchIntegerArray = i
      Exit For
    End If
  Next i
End Function
```

XI.4.12. Обработка ошибок

Ваш макрос может породить несколько типов ошибок. Некоторые ошибки Вы могли бы распознать и проверить, например, отсутствие файлов, а некоторые Вы могли бы просто проигнорировать или обработать самостоятельно. Чтобы обработать самостоятельно (trap) ошибки в макросе, используйте оператор «On Error».

```
On [Local] {Error GoTo Labelname | GoTo 0 | Resume Next}
```

On Error позволяет указать, какие ошибки будут обрабатываться, включая возможность указать свой собственный обработчик ошибок. Если использовано «Local», то это указывает, что процедура обработки будет локальной для содержащей этот оператор функции или процедуры. Если не использовано «Local», то способ обработки ошибок будет назначен для всего модуля.

> ***Совет*** *Процедура может содержать несколько операторов On Error. Каждый оператор On Error может обрабатывать ошибки разным способом. (Справка on-line неверно утверждает, что обработка ошибок должна быть задана в начале процедуры).*

XI.4.12.A. Определяем, как обрабатывать ошибку

Чтобы игнорировать все ошибки, используйте «On Error Resume Next». Когда происходит ошибка, то оператор, вызываший ошибку, пропускается и выполняется следующий оператор.

Чтобы задать свой собственный обработчик ошибок, используйте «On Error GoTo Label». Чтобы объявить метку Label в OOo Basic, наберите какой-либо текст на строке и закончите его двоеточием. Метки строк должны быть уникальными. Когда происходит ошибка, управление передается на эту метку.

После указания метода обработки ошибок, можно это отменить с помощью оператора «On Error GoTo 0». Когда ошибка произойдет следующий раз, Ваш обработчик ошибоку уже не будет вызван. Это не то же самое, что «On Error Resume Next», это означае, что будет выполнена стандартная обработка ошибок (остановка макроса с выводом сообщения об ошибке).

XI.4.12.B. Написать обработчик ошибок

Когда происходит ошибка и выполнение передано Вашему обработчику ошибок, имеются несколько функций, которые помогут определить, что и где произошло.

Error([номер]) : Возвращает сообщение об ошибка как текстовую строку. Вы можете (необязательно) указать номер ошибки, чтобы получить сообщение об ошибке с заданным номером. Сообщение об ошибке выдается на том языке, который задан при установке OOo.

Err() : Возвращает номер ошибки для последней случившейся ошибки

Erl() : Возвращает номер строки, где произошла последняя ошибка

После того, как ошибка обработана, Вы можете указать, как продолжить выполнение:

- Ничего не делать и продолжить выполнение
- Выйти из процедуры или функции с использованием «Exit Sub» или «Exit Function».

Используйте «Resume» для повторного выполнения той же строки текста программы, вызвавшей ошибку. Будьте осторожны, если Вы не исправили ошибу, то это может вызывать бесконечный цикл.

Лист. 365: Обработка ошибок

```
Sub ExampleResume
  Dim x%, y%
  x = 4 : y = 0
  On Local Error Goto oopsy
  x = x / y
  Print x
```

```
  Exit Sub
oopsy:
  y = 2
  Resume
End Sub
```

Используйте «Resume Next» для возобновления выполнения макроса со строки, следующей за вызвавшей ошибку строкой.

Лист. 366: Продолжение работы после ошибки

```
Sub ExampleResumeNext
  Dim x%, y%
  x = 4 : y = 0
  On Local Error Goto oopsy
  x = x / y
  Print x
  Exit Sub
oopsy:
  x = 7
  Resume Next
End Sub
```

Используйте «Resume Label:» для продолжения выполнения с указанной метки.

Лист. 367: Продолжение работы с определённого места

```
Sub ExampleResumeLabel
  Dim x%, y%
  x = 4 : y = 0
  On Local Error Goto oopsy
  x = x / y
GoHere:
  Print x
  Exit Sub
oopsy:
  x = 7
  Resume GoHere:
End Sub
```

XI.4.12.C. Пример обработки ошибок

Следующий пример демонстрирует множество превосходных способов обработки ошибок.

Лист. 368: Обработка ошибок

```
'Author:  Bernard Marcelly
'email:   marcelly@club-internet.fr
Sub ErrorHandlingExample
  Dim v1 As Double
  Dim v2 As Double
  Dim v0 As Double

```

```
  On Error GoTo TreatError1
  v0 = 0 : v1= 45 : v2= -123 ' Инициализируем  каким-либо значением
  v2= v1 / v0 ' деление на ноль вызовет ошибку
  Print "Result1:", v2

  On Error Goto TreatError2 ' Меняем обработчик ошибок
  v2= 456 ' Инициализируем каким-либо значением
  v2= v1 / v0 ' деление на ноль вызовет ошибку!
  Print "Result2:", v2 ' это не будет выполнено

Label2:
  Print "Result3:", v2 ' передача управления на обработку ошибок

  On Error Resume Next ' игнорируем любую ошику
  v2= 963 ' инициализируем каким-либо значением
  v2= v1 / v0 ' деление на ноль вызовет ошибку!
  Print "Result4:", v2 ' это будет выполнено

  On Error Goto 0 ' отключаем текущий способ обработки ошибок
  REM Теперь действует стандартная обработка ошибок
  v2= 147 ' инициализируем некоторым значением
  v2= v1 / v0 ' деление на ноль вызовет ошибку !
  Print "Result5:", v2 ' это не будет выполнено
  Exit Sub

TreatError1:
  Print "TreatError1 : ", error
  v2= 0
  Resume Next ' продолжаем выполнение макроса после оператора, вызввавшего
ошибку (continue after statement on error)

TreatError2:
  Print "TreatError2 : строка ", erl, "номер ошибки", err
  v2= 123456789
  Resume Label2
End Sub
```

XI.5. Разное

Этот раздел содержит разные мысли, почему именно эти — знаю только я: я видел примеры, но ничего не нашёл о них.

****

Несколько операторов можно написать в одной строке, разделив их двоеточием «:» (colon).

****

Если части «If Then» находятся на одной строке, то не требуется закрывающий оператор «End If».

Лист. 369: If Then

```
Sub SimpleIf
  If 4 = 4 Then Print "4 = 4" : Print " Hello you"  REM это напечатает
  If 3 = 2 Then Print "3 = 2"          REM это не напечатает
End Sub
```

****

Libraries, dialogs, IDE, Import и Export of Macros.

Конструкция With Объектт ... End With

****

Копирование объекта просто копирует ссылку. Копия структуры создаёт новую копию этой структуры. См пример EqualUnoObjects. Это может вызывтаь проблемы и поэтому такой объект — структура должен быть скопирован обратно!

****

Глава XII. Операторы и их старшинство Operators и Precedence

OpenOffice.org Basic поддерживает основные числовые операторы -, +, /, *, и ^. Эти операторы используют стандартное правило старшинства, но я все же приведу это правило. Логические операторы возвращают 0 для false (ни одного бита не установлено равным 1) и -1 для true (все биты установлены в 1). Более полное описание см. в разделе Операторы и функции (operators и functions).

Старшинство	*Оператор*	*Описание*
0	AND	Побитовый для чисел и логический для Булевых
0	OR	Побитовый для чисел и логический для Булевых
0	XOR	Побитовый для чисел и логический для Булевых
0	EQV	Логическое и/или побитовое совпадение
0	IMP	Логическое следствие (неверно работает в версии 1.0.3.1)
1	=	Логический
1	<	Логический
1	>	Логический
1	<=	Логический
1	>=	Логический
1	<>	Логический
2	-	Числовое вычитание
2	+	Числовое сложение, для строк — сцепление (Concatenation)
2	&	Сцепление строк
3	*	Числовое умножение
3	/	Числовое деление
3	MOD	Числовой остаток после деления
4	^	Числовое возведение в степень

Лист. 370: Старшинство операторов

```
Sub TestPrecedence
  Dim i%
  Print 1 + 2 OR 1     REM Печатается 3
  Print 1 + (2 OR 1)   REM Печатается 4
  Print 1 + 2 and 1    REM Печатается 1
  Print 1 + 2 * 3      REM Печатается 7
  Print 1 + 2 * 3 ^2   REM Печатается 19
  Print 1 = 2 OR 4     REM Печатается 4
  Print 4 and 1 = 1    REM Печатается 4
End Sub
```

Внимание *Булевские значения имеют внутреннее представление как целые числа, False = 0 и True = -1. Это позволяет использовать числовые операторы для булевских значений, но я подозреваю, что (1 + True = False). Лучше использовать булевские операторы*

Глава XIII. Действия со строками

Имеются несколько способов действия с текстовыми строками. Ниже я привожу перечень функций и их краткие описания.

функция	*Описание*
Asc(s$)	Код ASCII первого симовла строки
Chr$(i)	Символ, соответствующий заданному коду ASCII.
CStr(Выражение)	Преобразует числовое выражение в строку
Format(Число[, f])	Форматирует число, основываясь на строке формата f
Hex(Число)	Строка, которая представляет числовое выражение в 16-чном коде
InStr([i,] s$, f$[, c])	Позиция строки f, найденная в строке s, 0 = если не найдена. Поиск может требовать полного совпадения или может не различать большие и малые буквы. Возвращает число типа длинное целое (Long), преобразуемое в тип Integer, поэтому для длинных строк может возращать отрицательное значение
LCase(s$)	Преобразовывает строку, заменяя большие буквы на малые
Left(s$, n)	Левые n символов строки s. nздесь целое, но строка может быть длиной до 64K
Len(s$)	Длина строки s.
LSet s$ = Text	Выравнивает строку влево, дополняя до длины переменной-результата пробелами справа или отбрасывая лишние символы справа. Не работала в версии 1.0.3.1, исправлена в 1.1
LTrim(s$)	Строка без левых пробелов. Не меняет саму строку
Mid(s$, i[, n])	Подстрока строки s, начиная с позиции i и длиной n.
Mid(s$, i, n, r$)	Заменяет подстроку на r с ограничениями. Я использую для удаления части текста
Oct(Number)	Строка, содержащая восьмеричное представление числа
Right(s$, n)	Правые n символов строки s. n- число типа integer, но строка может быть длиной до 64K
RSet s$ = Text	Выравнивает строку по правому краю (дополняется до длины переменной-результата пробелами слева или отбрасываются лишние символы слева).

функция	*Описание*
RTrim(s$)	Строка без начальных пробелов
Space(n)	Строка из заданного числа пробелов
Str(Expression)	Преобразует числовое выражение в строку
StrComp(x$, y$[, c])	Возвращает 1 если x>y, 0 если x=y, и -1 если x<y. Если c=1, то различает при сравнении большие и малые буквы
String(n, {i | s$})	Создаёт строку из n символов. Если задано число i типа integer, то оно считается кодом ASCII нужного символа. Если задана строка s, то первый символ этой строки повторяется n раз.
Trim(s$)	Строка без начальных и конечных пробелов
UCase(s$)	Строка, где все малые буквы заменены на большие
Val(s$)	Преобразует строку в число

В справке on-line приведен неверный пример преобразования регистра. Вот ка должен выглядеть правильный макрос

Лист. 371: Преобразование регистра

```
Sub ExampleLUCase
  Dim sVar As String
  sVar = "Las Vegas"
  Print LCase(sVar) REM Печатается "las vegas"
  Print UCase(sVar) REM Печатается "LAS VEGAS"
end Sub
```

XIII.1. Удалить символы из строки

Этот макрос удаляет символы из строки. Глупо было бы думать, что этот макрос лучше, чем встроенный метод mid(). Разница в том, что метод mid() модифицирует текущую строку, в то время как этот макрос возвращает новую строку. Мне следовало переписать макрос уже используя mid(), но только недавно узнал о нем

Лист. 372: Удаление символов из строки

```
'Удалить заданное число символов из строки
Function RemoveFromString(s$, index&, num&) As String

  If num = 0 Or Len(s) < index Then
    'Если удалять нечего или задано удаление вне исходной строки, то
    'возвращается исходная строка
```

```
    RemoveFromString = s
  ElseIf index <= 1 Then
    'Удаление с начала

    If num >= Len(s) Then
      RemoveFromString = ""
    Else
      RemoveFromString = Right(s, Len(s) – num)
    End If

  Else
    'Удаление из середины

    If index + num > Len(s) Then
      RemoveFromString = Left(s,index – 1)
    Else
      RemoveFromString = Left(s,index – 1) + Right(s, Len(s) – index – num + 1)
    End If

  End If

End Function
```

XIII.2. Удалить текст из строки

Этот макрос может использоваться для удаления частей строки путем указании заменяющей строки равной пустой строке. Первой моей мыслью было использовать метод mid(), но я вспомнил, что метод mid не может увеличивать длину строки. Поэтому я должен был написать этот макрос. Он не изменяет существующую строку, но создаёт новую строку с произведенными в ней заменами.

Лист. 373: Удаление текста из строки

```
's$ – исходная строка, которую нужно изменить
'index – длинное целое long показывающее, где должна быть сделана замена
'(счет символов  начинается с 1)
'Если  index  <= 1 то текст вставляется в начало строки
'Если index > Len(s) то текст всталяется в конец строки num – длинное целое
' long, показывающее, сколько символов нужно заменить

'Если num имеет нулевую длину, то ничего не удаляется, но новая строка
' вставляется replaces – это строка, которая должна быть вставлена в строку

Function ReplaceInString(s$, index&, num&, replaces$) As String

  If index <= 1 Then
    'помещаем это в начало строки

    If num < 1 Then
      ReplaceInString = replaces + s
    ElseIf num > Len(s) Then
      ReplaceInString = replaces
```

```
    Else
      ReplaceInString = replaces + Right(s, Len(s) — num)
    End If

  ElseIf index + num > Len(s) Then
    ReplaceInString = Left(s,index — 1) + replaces
  Else
    ReplaceInString  = Left(s,index — 1) + replaces + Right(s, Len(s) —
index — num + 1)
  End If

End Function
```

XIII.3. Печать кодов ASCII символов строки

Это может показаться странным макросом, но я использовал его для определения, как текст был размещен в документе. Этот макрос будет печатать целую строку как набор кодов ASCII.

Лист. 374: Печать кодов

```
Sub PrintAll
  PrintAscii(ThisComponent.text.getString())
End Sub
Sub PrintAscii(TheText As String)
  If Len(TheText) < 1 Then Exit Sub
  Dim msg$, i%
  msg = ""
  For i = 1 To Len(TheText)
    msg = msg + Asc(Mid(TheText,i,1)) + " "
  Next i
  Print msg
End Sub
```

XIII.4. Удалить все экземпляры заданной подстроки из исходной строки

Лист. 375: Удаление подстроки

```
'Это удаляет все экземпляры подстроки bad$ из исходной строки s$
'Это изменяет исходную строку s$
Sub RemoveFromString(s$, bad$)
  Dim i%
  i = InStr(s, bad)
  Do While i > 0
    Mid(s, i, Len(bad), "")
    i = InStr(i, s, bad)
  Loop
End Sub
```

Глава XIV. Работа с числами

Функция	*Описание*
Abs(Number)	Абсолютное значение числа, возвращает тип двойное с плавающей точкой Double.
Asc(s$)	Код ASCII первого символа строки s, тип Integer
Atn(x)	Угол в радианах, имеющий заданный тангенс x
Blue(color)	Возвращает компонент «голубой» (Blue) заданного кода цвета color
CByte(Expression)	Преобразует строку или число в байт
CDbl(Expression)	Преобразует строку или число в число двойной точности с плавающей точкой (Double).
CInt(Expression)	Преобразует строку или число в целое число (Integer).
CLng(Expression)	Преобразует строку или число в длинное целое (Long).
Cos(x)	Вычисляет косинус угла, заданного в радианах
CSng(Expression)	Преобразует строку или число в одинарное число с плавающей точкой
CStr(Expression)	Преобразует строку или число в строку (String).
Exp(Expression)	Число e = 2.718282 возведенное в заданную степень
Fix(Expression)	Вовзращает целую часть числа после отбрасывания дробной части (-1.2 => -1)
Format(number [, f$])	Форматирует число на основе строки форматов f
Green(color)	Возвращает компоненту «зеленый» (Green) для заданного кода цвета color
Hex(Number)	Шестнадцатиричное представление числа, тип строковый (String)
Int(Number)	Округляет число до целого в сторону уменьшения. См . также: Fix().
IsNumeric (Var)	Проверяет, является ли выражение числом
Log(Number)	Натуральный логарифм числа
Oct(Number)	Восьмеричное представление числа, тип строка (String)
Randomize [Number]	Инициализирует генератор случайных чисел
Red(color)	Возвращает компонент «красный» (Red) заданного кода цвета color
RGB (Red, Green, Blue)	Код цвета типа длинного целого (Long) для заданных компонентов red (красный), green (зеленый) и blue

Функция	*Описание*
	(голубой)
Rnd [(Expression)]	Возращает случайное число между 0 и 1.
Sgn (Number)	Возвращает 1, -1 или 0 для чисел положительных, отрицательных и нулевых соответственно
Sin(x)	Вычисляет синус угла, заданноо в радианах
Sqr(Number)	Квадратный корень из числового выражения
Tan(x)	Тангенс угла, заданного в радианах
n = TwipsPerPixelX	Число twips , представляющих ширину пикселя
n = TwipsPerPixelY	Число twips, представляющих высоту пикселя
Val(s$)	Преобразует строку в число

Глава XV. Работа с датами

Функция	*Описание*
CDate(Expression)	Преобразует строку или число в дату
CDateFromIso(String)	Возвращает внутреннее представление даты, полученное преобразованием строки, содержащей дату в формате ISO
CDateToIso(Number)	Возвращает дату в формате ISO, полученную из составной даты (serial date), созданной функциями DateSerial или DateValue.
Date	Возвращает текущую системную дату
Date = s$	Устанавливает текущую системную дату
DateSerial(y%, m%, d %)	Возвращает дату, полученную из заданного года y, месяца m и дня d
DateValue([date])	Длинное целое (Long), полученное из даты и пригодное для вычисления разницы дат
Day(Number)	Номер дня месяца, полученный для даты/времени, созданной функциями DateSerial или DateValue.
GetSystemTicks()	Возвращает системные такты (ticks), предоставляемые операционной системой
Hour(Number)	Час, извлеченный из времени/даты, созданного функциями TimeSerial или TimeValue.
Minute(Number)	Минуты, извлеченные из времени/даты, созданного функциями TimeSerial или TimeValue.
Month(Number)	Месяц. извлеченный из даты/времени, созданной функциями DateSerial или DateValue.
Now	Текущая системная дата и время, в формате значения даты (Date)
Second(Number)	Секунды, извлеченные из времени/даты, созданного функциями TimeSerial или TimeValue.
Time	Текущее системное время
Timer	Число секунд, прошедшее с полуночи
TimeSerial (h, m, s)	Время (Serial time), созданное на основе часа h, минут m, секунд s
TimeValue (s$)	Время (Serial time), полученное из форматированной строки

Функция	*Описание*
Wait millisec	Выполняется пауза на заданное число миллисекунд
WeekDay(Number)	День недели, полученный из даты/времени, созданной функциями DateSerial или DateValue.
Year(Number)	Год, извлеченный из даты/времени, созданной фунциями DateSerial или DateValue.

Функция	*Описание*
ChDir (s$)	Изменяет текущую директорию (папку) или устройство (диск)
ChDrive(s$)	Изменяет текущее устройство (диск)
Close #n% [, #n2%[,...]]	Закрывает файлы, открытые оператором Open
ConvertFromURL(s$)	Преобразует адрес URL файла в системное имя файла
ConvertToURL(s$)	Преобразует системное имя файла в адрес URL файла.
CurDir([s$])	Возвращает текущую директорию (папку) на указанном устройстве (диске)
Dir [s$ [, Attrib%]]	Выдает перечень содержимого директории (папки)
EOF(n%)	Достиг ли указатель позиции в файле конца файла?
FileAttr (n%, Attribut%)	Возвращает атрибут файла для открытого файла
FileCopy from$, to$	Копирует файл
FileDateTime(s$)	Строка, содержащая дату и время файла
FileExists(s$)	Определяет, существует ли файл или директория (папка)
FileLen (s$)	Длина файла в байтах
FreeFile	Следующий доступный номер файла. Используется во избежание одновременного использования одного и того же номера файла
Get [#]n%, [Pos], v	Прочесть запись или байты из файла
GetAttr(s$)	Возвращает маску битов, которая идентифицирует тип файла
Input #n% v1[, v2[, [,...]]	Прочесть данные из открытого файла последовательного типа
Kill f$	Удалить файл с диска
Line Input #n%, v$	Прочитать строку из файла последовательного типа, занеся её в переменную v
Loc (FileNumber)	Возвращает текущую позицию в открытом файле
Lof (FileNumber)	Возвращает текущий размер открытого файла
MkDir s$	Создать новую директорию (папку)
Name old$, new$	Переименовать существующий файл или

Функция	*Описание*
	директорию (папку)
Open s$ [#]n%	Открыть файл. Многие параметры не показаны, этот оператор очень гибкий
Put [#] n%, [pos], v	Записать запись в файл последовательного типа
Reset	Закрыть все открытые файлы, выгрузив буферы на диск
RmDir f$	Удалить директорию (папку)
Seek[#]n%, Pos	Передвинуть указател позиции в файле
SetAttr f$, Attribute%	Установить атрибуты файла
Write [#]n%, [Exprs]	Записать данные в файл последовательного типа

Глава XVII. Операторы в выражениях, операторы программы, функции

XVII.1. Оператор вычитания (-)

Обзор:

Вычесть два числовых значениях. Используется стандартное правило старшинства математических операторов, показанное на странице 349.

Синтаксис:

```
Результат = Expression1 — Expression2
```

Параметры:

Результат : Результат вычитания

Expression1, Expression2 : Любые числовые выражения.

Пример:

Лист. 376: Оператор вычитания

```
Sub SubtractionExample
  Print 4 — 3         '1
  Print 1.23e2 — 23   '100
End Sub
```

XVII.2. Оператор умножения (*)

Обзор:

Умножает два числовых значения. Используется стандартное правило старшинства математических операторов, показанное на странице 349.

Синтаксис:

```
Результат = Expression1 * Expression2
```

Параметры:

Результат : результат умножения.

Expression1, Expression2 : Любые числовые выражения

Пример:

Лист. 377: Оператор умножения

```
Sub MultiplictionExample
  Print 4 * 3         '12
```

```
  Print 1.23e2 * 23  '2829
End Sub
```

XVII.3. Оператор сложения (+)

Обзор:

Складывает два числовых значения. Хотя это работает с булевскими значениями, потому что они могут быть представлены как числа, я очень не рекомендую этого. Проверка показываетс, что это похоже на результат логического ИЛИ, но я не советую, так как при преобразовании типов могут быть проблемы. Операции выполняются как для целых чисел и затем преобразуются обратно в булевские. Тут будут пробелмы.

Используется стандартное правило старшинства математических операторов, показанное на странице 349.

Синтаксис:

```
Результат = Expression1 + Expression2
```

Параметры:

Результат : результат сложения.

Expression1, Expression2 : Любые числовые выражения

Пример:

Лист. 378: Оператор сложения

```
Sub SubtractionExample
  Print 4 + 3        '7
  Print 1.23e2 + 23  '146
End Sub
```

XVII.4. Оператор возведения в степень (^)

Обзор:

Возводит число в степень. Пусть уравнение $x = y^z$ представляет этот оператор. Если z целое (integer), то x будет результатом умножения y на себя z раз. Используется стандартное правило старшинства математических операторов, показанное на странице 349.

Синтаксис:

```
Результат = Expression ^ Exponent
```

Параметры:

Результат : Результат возведения в степень

Expression: Любое числовое выражение.

Exponent: Любое числовое выражение.

Пример:

Лист. 379: Возведение в степень

```
Sub ExponentiationExample
  Print 2 ^ 3      '8
  Print 2.2 ^ 2    '4.84
  Print 4 ^ 0.5    '2
End Sub
```

XVII.5. Оператор деления (/)

Обзор:

Деление двух числовых значений. Будьте внимательны с результатом, потому что деление может не дать точного целого числа, когда Вы ожидаете этого. Используйте функцию Int(), если это важно. Используется стандартное правило старшинства математических операторов, показанное на стр. 349.

Синтаксис:

```
Результат = Expression1 / Expression2
```

Параметры:

Результат : результат деления.

Expression1, Expression2 : Любые числовые выражения

Пример:

Лист. 380: Оператор деления

```
Sub DivisionExample
  Print 4 /2  '2
  Print 11/2  '5.5
End Sub
```

XVII.6. Оператор AND

Обзор:

Выполняет логическое И для булевских значений побитовое И для числовых значений. Побитовое И для чисел с двойной точностью

с плавающей точкой производит сначала преобразование их в целые (integer). Числовое переполнение возможно. Используется стандартное правило старшинства математических операторов, показанное на странице 349 и в следующей таблице операторов.

x	*y*	*x И y*
ИСТИНА	ИСТИНА	ИСТИНА
ИСТИНА	ЛОЖЬ	ЛОЖЬ
ЛОЖЬ	ИСТИНА	ЛОЖЬ
ЛОЖЬ	ЛОЖЬ	ЛОЖЬ
1	1	1
1	0	0
0	1	0
0	0	0

Синтаксис:

```
Результат = Expression1 AND Expression2
```

Параметры:

Результат : результат операции

Expression1, Expression2 : Числовые или булевские выражения

Пример:

Лист. 381: Оператор AND

```
Sub AndExample
  Print (3 and 1)        'Prints 1
  Print (True and True)  'Prints -1
  Print (True and False) 'Prints 0
End Sub
```

XVII.7. Функция Abs

Обзор:

Возвращает абсолютную величину числового выражения. Если параметр является строкой, она сначала преобразуется в число, вероятно, используя функцию Val. Если число неотрицательное, то возвращается оно, если нет, то возвращается это число с обратным знаком.

Синтаксис:

```
Abs (Number)
```

Возвращаемое значение:

Число с двойной точкой с плавающей точкой (Double)

Параметры:

Number: Любое числовое выражение

Пример:

Лист. 382: Функция Abs

```
Sub AbsExample
  Print Abs(3)       '3
  Print Abs(-4)      '4
  Print Abs("-123") '123
End Sub
```

XVII.8. Функция создания массива Array

Обзор:

Создает массив типа Variant из списка параметров. Это наиболее быстрый метод создать массив констант.

> ***Внимание*** *Если Вы присвоите полученный массив типа Variant другому массиву типа не Variant, то не сможете сохранитьт данные, когда будете менять размерность массива. Я считаю это ошибкой ООо, что можно присваивать массивVariant массиву, который не является Variant*

См. также функцию DimArray

Синтаксис:

```
Array (СписокАргументов)
```

Возвращаемое значение:

Массив типа Variant, содержащий список аргументов

Параметры:

СписокАргументов: список значений, разделенных запятыми, из которых нужно создать массив.

Пример:

Лист. 383: Пример

```
Sub ArrayExample
  Dim a(5) As Integer
  Dim b() As Variant
  Dim c() As Integer
  REM Array() имеет тип  variant
  REM b имеет размерность от 0 до 9, пока  b(i) = i+1
  b = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
  PrintArray("b после начального присвоения", b())
  REM b имеет размерность от 1 до 3, все ещё b(i) = i+1
  ReDim Preserve b(1 To 3)
  PrintArray("b после применения ReDim", b())
  REM Следующий оператор НЕ верен, потому что массив уже объявлен
  REM с другим числом элементов, но Вы можете задать  ReDim для него,
пробуйте!
  REM a = Array(0, 1, 2, 3, 4, 5)

  REM c имеет размерность от 0 до 5
  REM Это вызывает у меня сомнения, потому что  "Hello" является строкой, но
  REM это работает в версии 1.0.2
  c = Array(0, 1, 2, "Hello", 4, 5)
  PrintArray("c, Variant присвоен массиву типа  Integer ", c())
  REM Забавно, это разрешено, но массив с не будет содержать данных!
  ReDim Preserve c(1 To 3) As Integer
  PrintArray("c после применения ReDim", c())
End Sub

Sub PrintArray (lead$, a() As Variant)
  Dim i%, s$
  s$ = lead$ + Chr(13) + LBound(a()) + " to " + UBound(a()) + ":" + Chr(13)
  For i% = LBound(a()) To UBound(a())
    s$ = s$ + a(i%) + " "
  Next
  REM Я использую MsgBox а не Print, потому что можно использовать
встроенный Chr(13)
  MsgBox s$
End Sub
```

XVII.9. Функция Asc

Обзор:

Возвращает код ASCII (American Standard Code for Information Interchange) первого символав строки, остальные символы игнорируются. Ошибка происходит, если строка имеет нулевую длину. Разрешены 16-битовые

символы Unicode. Эта функция является по существу универсальным (относительно языка) вариантом функции CHR$.

Синтаксис:

```
Asc (Text As String)
```

Возвращаемое значение:

Целое Integer

Параметры:

Text: Любое текстовое выражение

Пример:

Лист. 384: Функция Asc

```
Sub AscExample
  Print Asc("ABC")    '65
End Sub
```

XVII.10. Функция ATN (арктангенс)

Обзор:

Арктангенс числового выражения возвращает значение в интервале от -π/2 до π/2 (в радианах). Эта функция обратная к тангенсу (Tan). Она относится к тригонометрическим функциям

Синтаксис:

```
ATN(Number)
```

Возвращаемое значение:

Двойное с плавающей точкой (Double)

Параметры:

Number: числовое выражение

Пример:

Лист. 385: Функция ATN

```
Sub ExampleATN
  Dim dLeg1 As Double, dLeg2 As Double
  dLeg1 = InputBox("Введите длину прилегающего катета: ","Adjacent")
  dLeg2 = InputBox("Введите длину противолежащего катета: ","Opposite")
  MsgBox "Угол равен " + Format(ATN(dLeg2/dLeg1), "##0.0000") _
```

```
    + " радианов" + Chr(13) + "Угол равен " _
    + Format(ATN(dLeg2/dLeg1) * 180 / Pi, "##0.0000") + " градусов"
End Sub
```

XVII.11. Оператор Beep

Обзор:

Создает системный звук «биип»

Синтаксис:

```
Beep
```

Пример:

Лист. 386: Оператор Beep

```
Sub ExampleBeep
  Beep
  Beep
End Sub
```

XVII.12. Функция Blue

Обзор:

Цвета представляются в виде длинного целого (long). Возвращает компонент «голубой» (blue) для заданного кода цвета color. См. также RGB, Red, и Green.

Синтаксис:

```
Blue (Color As Long)
```

Возвращаемое значение:

Целое (Integer) в интервале от 0 до 255.

Параметры:

Color: Длинное целое (Long) выражение, представляющее цвет.

Пример:

Лист. 387: Функция Blue

```
Sub ExampleColor
  Dim lColor As Long
  lColor = RGB(255,10,128)
  MsgBox "Цвет " & lColor & " состоит из:" & Chr(13) &_
```

```
    "Red = " & Red(lColor) & Chr(13)&_
    "Green= " & Green(lColor) & Chr(13)&_
    "Blue= " & Blue(lColor) & Chr(13) , 64,"Colors"
End Sub
```

XVII.13. Ключевое слово ByVal

Обзор:

Параметры передаются в процедуры и функции пользователя по ссылке. Если процедура или функция меняют параметр, то он меняется и в вызвавшей прогтамме. Это может вызвать странные результаты, если параметр был константой или если вызвавшая процедура не ожидает такого измененя. Ключевое слово ByVal указывает, что параметр должен передаваться по значению, а не по ссылке.

Синтаксис:

```
Sub Name(ByVal ParmName As ParmType)
```

Пример:

Лист. 388: ByVal

```
Sub ExampleByVal
  Dim j As Integer
  j = 10
  ModifyParam(j)
  Print j  REM 9
  DoNotModifyParam(j)
  Print j  REM 9
End Sub
Sub ModifyParam(i As Integer)
  i = i – 1
End Sub
Sub DoNotModifyParam(ByVal i As Integer)
  i = i – 1
End Sub
```

XVII.14. Ключевое слово Call

Обзор:

Передает управление в процедуру (sub), функцию или процедуру DLL. Оператор Call необязателен, за исключением вызова DLL, в последнем случае DLL должна быть сначала объявлена (it must first be defined). Параметры могут быть заключены в круглые скобки, причем когда функция используется в выражении, то заключать параметры в скобки обязательно.

См также: Declare

Синтаксис:

```
[Call] Name [Parameter]
```

Параметры:

Name: Имя вызываемой процедуры, функции или DLL

Параметры: Тип и число параметров зависят от вызываемой процедуры, функции, DLL

Пример:

Лист. 389: Call

```
Sub ExampleCall
  Call CallMe "Этот текст будет выведен"
End Sub
Sub CallMe(s As String)
  Print s
End Sub
```

XVII.15. Функция CBool

Обзор:

Преобразовать параметр в булевский тип. Если выражение числовое, то 0 соответствует False и любое другое соответствует True. Если выражение имеет тип строки (String), то «true» и «false» (независимо от регистра букв) соответствуют True и False. Строки с любыми другими значениями вызывают ошибку. Это полезно, чтобы потребовать булевского типа результата. Если я использую функцию, которая возвращает числовое значение, например, InStr, то я могу написать

```
If CBool(InStr(s1, s2)) Then
```

вместо

```
If InStr(s1, s2) <> 0 Then
```

Синтаксис:

```
CBool (Expression)
```

Возвращаемое значение:

Boolean

Параметры:

Expression: числовое или булевское выражение

Пример:

Лист. 390: Функция CBool

```
Sub ExampleCBool
  Print CBool(1.334e-2)         'True
  Print CBool("TRUE")           'True
  Print CBool(-10 + 2*5)        'False
  Print CBool("John" <> "Fred")'True
End Sub
```

XVII.16. Функция CByte

Обзор:

Преобразует строковое или числовое выражение в тип байт (Byte). Строковые выражения преобразуются в числовые, а числа двойной длины с плавающей точкой округляются. Если выражение слишком велико или отрицательно, происходит ошибка.

Синтаксис:

```
Cbyte( expression )
```

Возвращаемое значение:

Байт Byte

Параметры:

Expression : Строковое или числовое выражение.

Пример:

Лист. 391: Функция CByte

```
Sub ExampleCByte
  Print Int(CByte(133))        '133
  Print Int(CByte("AB"))       '0
  'Print Int(CByte(-11 + 2*5))'Ошибка, вне допустимого интервала
  Print Int(CByte(1.445532e2))'145
  Print CByte(64.9)            'A
  Print CByte(65)              'A
  Print Int(CByte("12"))       '12
End Sub
```

XVII.17. Функция CDate

Обзор:

Преобразует в дату. Числовые выражения содержат: целая часть — дату, считая от 31.12.1899, дробная часть — время. Строки должны иметь формат, как определено для функций DateValue и TimeValue. Другими словавми, форматирование строки зависит от языка установки OOo (локали). CDateFromIso не зависит от локали, поэтому надежнее для программирования для международных целей.

Синтаксис:

```
CDate (Expression)
```

Возвращаемое значение:

Дата Date

Параметры:

Expression: Любое строковое или числовое выражение для конвертации

Пример:

Лист. 392: Функция CDate

```
sub ExampleCDate
  MsgBox cDate(1000.25) REM 09.26.1902 06:00:00
  MsgBox cDate(1001.26) REM 09.27.1902 06:14:24
  Print DateValue("06/08/2002")
  MsgBox cDate("06/08/2002 15:12:00")
end sub
```

XVII.18. Функция CDateFromIso

Обзор:

Возвращает внутреннее представление даты — число — преобразованное из строки, содержащей дату в формате ISO

Синтаксис:

```
CDateFromIso(String)
```

Возвращаемое значение:

Внутреннее представление даты в виде числа

Параметры:

String : строка, содержащая дату в формате ISO. Год может иметь как две, так и четыре цифры

Пример:

Лист. 393: Функция CDateFromIso

```
sub ExampleCDateFromIso
  MsgBox cDate(37415.70)            REM 08 June 2002 16:48:00
  Print CDateFromIso("20020608")    REM 08 June 2002
  Print CDateFromIso("020608")         REM 08 June 1902
  Print Int(CDateFromIso("20020608"))  REM 37415
end sub
```

XVII.19. Функция CDateToIso

Обзор:

Возвращает дату в формате ISO, полученную из числового представления даты (serial date number), созданного функциямиDateSerial или DateValue.

Синтаксис:

```
CDateToIso(Number)
```

Возвращаемое значение:

Строка (String)

Параметры:

Number : Целое (Integer), содержащее числовое представление даты (serial date number).

Пример:

Лист. 394: Функция CDateToIso

```
Sub ExampleCDateToIso
  MsgBox "" & CDateToIso(Now) ,64," = дата ISO"
End Sub
```

XVII.20. Функция CDbl

Обзор:

Преобразует любое числовое выражение или строковое выражение в двойное число с плавающей точкой. Строки должны быть форматированы

на основе языка установки OOo (локали). В США «12.34» будет работать, но это может вызвать ошибку в другой стране.

Синтаксис:

```
CDbl (Expression)
```

Возвращаемое значение:

Двойное с плавающей точкой (Double)

Параметры:

Expression : Любое строковое или числовое выражение для преобразования

Пример:

Лист. 395: Функция CDbl

```
Sub ExampleCDbl
  Msgbox CDbl(1234.5678)
  Msgbox CDbl("1234.5678")
End Sub
```

XVII.21. Оператор ChDir — нежелателен

Обзор:

Изменяет текущую директорию (папку) или устройство (диск). Если нужно изменить только текущий диск, введите только букву диска с двоеточием после нее. Оператор ChDir нежелателен, не используйте его.

Синтаксис:

```
ChDir(Text)
```

Параметры:

Text: любое строковое выражение, которое задает папку или диск.

Пример:

Лист. 396: Оператор ChDir

```
Sub ExampleChDir
  Dim sDir as String
  sDir = CurDir
  ChDir( "C:\temp" )
  msgbox CurDir
  ChDir( sDir )
```

```
  msgbox CurDir
End Sub
```

XVII.22. Оператор ChDrive — нежелателен

Обзор:

Изменить текущее устройство (диск). Буква диска должна быть в верхнем регистре (большой буквой). Можно использовать оператор OnError, чтобы обработать возможные ошибки. Этот оператор нежелателен, не используйте его.

Синтаксис:

```
ChDrive(Text)
```

Параметры:

Text: Строковое выражение, содержащее только одну (большую) букву диска. Допустим адрес URL.

Пример:

Лист. 397: Операторо ChDrive

```
Sub ExampleCHDrive
  On Local Error Goto NoDrive
  ChDrive "Z" REM работает, только если диск 'Z' существует
  Print "Завершено"
  Exit Sub
NoDrive:
  Print "Извините, этот диск не существует"
  Resume Next
End Sub
```

XVII.23. Функция Choose

Обзор:

Возвращает выбранное значение из заданного списка аргументов. Это быстрый способ выбрать значение из списка. Если индекс вне интервала (меньше 1 или больше n), то возвращается null

Синтаксис:

```
Choose (Index, Selection_1[, Selection_2, ... [,Selection_n]])
```

Возвращаемое значение:

Тип может быть любым — тем типом, который имеет Selection_i

Параметры:

Index: Числовое выражение, определяющее, которое значение нужно вернуть.

Selection_i: Значение, которое нужно вернуть.

Пример:

В этом примере переменная «o» не имеет типа, поэтому он принимает тип выражения Selection_i. Selection_1 имеет тип строки (String), а Selection_2 имеет тип Double. Если бы я задал для переменной «o» конкретный тип, то возвращаемое значение было бы преобразовано к этому типу.

Лист. 398: Функция Choose

```
Sub ExampleChoose
  Dim sReturn As String
  Dim sText As String
  Dim i As Integer
  Dim o
  sText = InputBox ("Введите целое число от 1 до 3:","Example")
  i = Int(sText)
  o = Choose(i, "One", 2.2, "Three")
  If IsNull(o) Then
    Print "Извините, '" + sText + "' неверно"
  Else
    Print "Получено '" + o + "' типа " + TypeName(o)
  End If
end Sub
```

XVII.24. Функция Chr

Обзор:

Возвращает символ, соответствующий заданному коду символа (ASCII или Unicode).

Эта функция используется для создания специальных последовательностей символов, таких как управляющие коды для принтера, знаки табуляции, новой строки, возврата каретки и т.д. Это также позволяет вставить двойную кавычку в строку. Иногда эта функция записывается как «Chr$()».

См. также функцию Asc.

Синтаксис:

```
Chr(Expression)
```

Возвращаемое значение:

Строка (String)

Параметры:

Expression: Числовые выражения, представляющие правильные 8-битовые коды ASCII (0-255) или 16-битовые коды Unicode.

Пример:

Лист. 399: Функция Chr

```
sub ExampleChr
  REM Вывести "Line 1" и "Line 2" на отдельных строках.
  MsgBox "Line 1" + Chr$(13) + "Line 2"
End Sub
```

XVII.25. Функция CInt

Обзор:

Преобразует любое числовое выражение или строковое выражение в тип целое (Integer). Строки должны быть отформатированы согласно языку установки OOo (локали). В США «12.34» будет работать.

См. также: Fix

Синтаксис:

```
CInt(Expression)
```

Возвращаемое значение:

Целое (Integer)

Параметры:

Expression : Любое строковое или числовое выражение для преобразования

Пример:

Лист. 400: Функция CInt

```
Sub ExampleCInt
  Msgbox CInt(1234.5678)
  Msgbox CInt("1234.5678")
End Sub
```

XVII.26. Функция CLng

Обзор:

Преобразует любое числовое выражение или строковое выражение в тип длинное целое (Long). Строки должны быть отформатированы в соответствии с языком установки OOo (локали). В США «12.34» будет работать.

Синтаксис:

```
CLong(Expression)
```

Возвращаемое значение:

Длинное целое (Long)

Параметры:

Expression : Любое строковое или числовое выражение для преобразования

Пример:

Лист. 401: Функция CLng

```
Sub ExampleCLng
  Msgbox CLng(1234.5678)
  Msgbox CLng("1234.5678")
End Sub
```

XVII.27. Оператор Close

Обзор:

Закрыть файлы, предварительно открытые оператором Open. Несколько файлов можно закрыть одним оператором.

См. также операторы/фукнции Open, EOF, Kill, и FreeFile

Синтаксис:

```
Close #FileNumber As Integer[, #FileNumber2 As Integer[,...]]
```

Параметры:

FileNumber : Целое (Integer) выражение, задающее предварительно открытый файл.

Пример:

Лист. 402: Оператор Close

```
Sub ExampleCloseFile
  Dim iNum1 As Integer, iNum2 As Integer
  Dim sLine As String, sMsg As String
  'Следующий доступный номер файла!
  iNum1 = FreeFile
  Open "c:\data1.txt" For Output As #iNum1
  iNum2 = FreeFile
  Open "c:\data2.txt" For Output As #iNum2
  Print #iNum1, "Текст в файле один с номером " + iNum1
  Print #iNum2, "Текст в файле два с номером " + iNum2
  Close #iNum1, #iNum2
  Open "c:\data1.txt" For Input As #iNum1
  iNum2 = FreeFile
  Open "c:\data2.txt" For Input As #iNum2
  sMsg = ""
  Do While not EOF(iNum1)
    Line Input #iNum1, sLine
    If sLine <> "" Then sMsg = sMsg+"File: "+iNum1+":"+sLine+Chr(13)
  Loop
  Close #iNum1
  Do While not EOF(iNum2)
    Line Input #iNum2, sLine
    If sLine <> "" Then sMsg = sMsg+"File: "+iNum2+":"+sLine+Chr(13)
  Loop
  Close #iNum2
  Msgbox sMsg
End Sub
```

XVII.28. Оператор константы Const

Обзор:

Константы помогают легче читать программу путем присвоения имен константам и путем создания одного места для определения констант. Константы могут включать определение типа, хотя это не обязательно. Константа объявляется один раз и не может изменяться.

Синтаксис:

```
Const Text [As type] = Expression[, Text2 [As type] = Expression2[, ...]]
```

Параметры:

Text: Любое имя константы, соответствующее требованиям к именам переменных.

Пример:

Лист. 403: Оператор Const

```
Sub ExampleConst
  Const iVar As String = 1964
  Const sVar = "Program", dVar As Double = 1.00
  Msgbox iVar & " " & sVar & " " & dVar
End Sub
```

XVII.29. Функция ConvertFromURL

Обзор:

Преобразует адрес URL файла в системное имя файла

Синтаксис:

```
ConvertFromURL(filename)
```

Возвращаемое значение:

String

Параметры:

Filename: имя файла в виде адреса URL

Пример:

Лист. 404: Функция ConvertFromURL

```
Sub ExampleFromUrl
  Dim s$
  s = "file:///c:/temp/file.txt"
  MsgBox s & " => " & ConvertFromURL(s)
  s = "file:///temp/file.txt"
  MsgBox s & " => " & ConvertFromURL(s)
End Sub
```

XVII.30. Функция ConvertToURL

Обзор:

Преобразует системное имя файла в адрес URL.

Синтаксис:

```
ConvertToURL(filename)
```

Возвращаемое значение:

Строка (String)

Параметры:

Filename: Имя файла в форми системного имени файла

Пример:

Лист. 405: Функция ConvertToURL

```
Sub ExampleToUrl
  Dim s$
  s = "c:\temp\file.txt"
  'file:///c:/temp/file.txt
  MsgBox s & " => " & ConvertToURL(s)
  s = "\temp\file.txt"
  'file:///temp/file.txt
  MsgBox s & " => " & ConvertToURL(s)
End Sub
```

XVII.31. Функция косинуса Cos

Обзор:

Косинус числового выражения возвращает значение в интервале от -1 до 1. Функция относится к тригонометрическим функциям.

Синтаксис:

```
Cos(Number)
```

Возвращаемое значение:

Двойное с плавающей точкой (Double)

Параметры:

Number: любое числовое выражение

Пример:

Лист. 406: Функция Cos

```
Sub ExampleCos
  Dim dLeg1 As Double, dLeg2 As Double, dHyp As Double
  Dim dAngle As Double
  dLeg1 = InputBox("Введите длину прилегающего катета: ","Adjacent")
  dLeg2 = InputBox("Вседите длину протоволежащего катета: ","Opposite")
  dHyp = Sqr(dLeg1 * dLeg1 + dLeg2 * dLeg2)
  dAngle = Atn(dLeg2 / dLeg1)
  MsgBox "Прилегающий катет = " + dLeg1 + Chr(13) + _
```

```
      "Противолежащий катет = " + dLeg2 + Chr(13) + _
      "Гипотенуза = " + Format(dHyp, "##0.0000") + Chr(13) + _
      "Угол = "+Format(dAngle*180/Pi, "##0.0000")+" градусовв"+Chr(13)+_
      "Cos = " + Format(dLeg1 / dHyp, "##0.0000") + Chr(13) + _
      "Cos = " + Format(Cos(dAngle), "##0.0000")
End Sub
```

XVII.32. Функция создания диалога CreateUnoDialog

Обзор:

Создать объект UNO Basic, который представляет собой управляющий диалог UNO в процессе выполнения макроса.

Диалоги определены в библиотеках диалогов. Чтобы вывести диалог, должен быть создан «живой» диалог из нужной библиотеки.

Синтаксис:

```
CreateUnoDialog( oDlgDesc )
```

Возвращаемое значение:

Object: Диалог, который нужно выполнить!

Параметры:

oDlgDesc : Описание диалога, предварительно определенное в библиотеке.

Пример:

Лист. 407: Функция CreateUnoDialog

```
Sub ExampleCreateDialog
  Dim oDlgDesc As Object, oDlgControl As Object
  DialogLibraries.LoadLibrary("Standard")
  ' Получить описание диалога из библиотеки диалогов
  oDlgDesc = DialogLibraries.Standard
  Dim oNames(), i%
  oNames = DialogLibraries.Standard.getElementNames()
  i = lBound( oNames() )
  while( i <= uBound( oNames() ) )
    MsgBox "How about " + oNames(i)
    i = i + 1
  wend
  oDlgDesc = DialogLibraries.Standard.Dialog1
  ' создаем "живой" диалог
  oDlgControl = CreateUnoDialog( oDlgDesc )
  ' выводим "живой" диалог
  oDlgControl.execute
End Sub
```

XVII.33. Функция CreateUnoService

Обзор:

Создает (экземпляр) сервиса UNO с помощью ServiceManager.

Синтаксис:

```
oService = CreateUnoService( UNO service name )
```

Возвращаемое значение:

Запрашиваемый сервис.

Параметры:

Строка с именем запрашиваемого сервиса

Пример:

Этот пример предоставлен автором Laurent Godard. Как и он, я не смог придумать, как вставить текств в электронное письмо, потому что в версии OOo 1.1.1 нельзя вставлять текст. Сервис электронной почты был создан для пересылки приложений, надеюсь, это будет исправлено или изменено в будущем.

Лист. 408: Функция CreateUnoService

```
Sub SendSimpleMail()
  Dim vMailSystem, vMail, vMessage

  vMailSystem=createUnoService("com.sun.star.system.SimpleSystemMail")
  vMail=vMailSystem.querySimpleMailClient()
  'если Вы хотите знать, что ещё можно сделать с этим, см.
  'http://api.openoffice.org/docs/common/ref/com/sun/star/system/XSimpleMail
Message.html
  vMessage=vMail.createsimplemailmessage()
  vMessage.setrecipient("Andrew.Pitonyak@qwest.com")
  vMessage.setsubject("Это моя тестовая тема письма")

  'Приложения к письму заданы последовательностью, котоная на языке  basic
означает массив
  'Можно использовать  ConvertToURL() для создания адреса URL!
  Dim vAttach(0)
  vAttach(0) = "file:///c|/macro.txt"
  vMessage.setAttachement(vAttach())

  'DEFAULTS ссылается на настроенный в данный момент системный почтовый
клиент
  'NO_USER_INTERFACE не показывает этот интерфейс, только выполняет его!
  'NO_LOGON_DIALOG никакого диалога ввода логина и пароля, но будет вызвана
ошибка, если логин и пароль потребуются
```

```
vMail.sendSimpleMailMessage(vMessage,
com.sun.star.system.SimpleMailClientFlags.NO_USER_INTERFACE)
End Sub
```

XVII.34. Функция CreateUnoStruct

Обзор:

Создает экземпляр структуры UNO заданного типа

Для com.sun.star.beans.PropertyValue лучше использовать следующую конструкцию!

```
Dim oStruct As New com.sun.star.beans.PropertyValue
```

Синтаксис:

```
oStruct = CreateUnoStruct( UNO type name )
```

Параметры:

Строка с именем запрашиваемой структуры.

Возвращаемое значение:

Запрошенная структура

Пример:

Лист. 409: Функция CreateUnoStruct

```
oStruct = CreateUnoStruct("com.sun.star.beans.PropertyValue")
'Хотите ли Вы выбрать определенный принтер
'Dim mPrinter(0) As New com.sun.star.beans.PropertyValue
'mPrinter(0).Name="Name"
'mPrinter(0).value="Other printer"
'oDoc.Printer = mPrinter()
'Вы могли бы сделать это таким способом после того, как mPrinter(0)
' определен!
'mPrinter(0) = CreateUnoStruct("com.sun.star.beans.PropertyValue")
```

XVII.35. Функция CSng

Обзор:

Преобразует любое числовое или строковое выражение в тип одинарного числа с плавающей точкой (Single). Строка должна быть отформатирована в соответствии с языком установки OOo (локали). В США «12.34» будет работать.

Синтаксис:

```
CSng(Expression)
```

Возвращаемое значение:

одинарное с плавающей точкой (Single)

Параметры:

Expression : Любое строковое или числовое выражение для преобразования

Пример:

Лист. 410: Функция CSng

```
Sub ExampleCLng
  Msgbox CSng(1234.5678)
  Msgbox CSng("1234.5678")
End Sub
```

XVII.36. Функция CStr

Обзор:

Преобразует любое выражение в строку. Эта функция обычно используется для преобразования числа в строку.

Исходный тип	*Результат*
Boolean	«True» or «False»
Date	Строка с датой и временем
Null	Ошибка (run-time error)
Empty	Пустая строка «»
Any Number	Число в виде строки. Завершающие нули после десятичной точки опущены.

Синтаксис:

```
CStr (Expression)
```

Возвращаемое значение:

Строка (String)

Параметры:

Expression: Любое строковое или числовое выражение для преобразования

Пример:

Лист. 411: Функция CStr

```
Sub ExampleCSTR
Dim sVar As String
Msgbox CDbl(1234.5678)
Msgbox CInt(1234.5678)
Msgbox CLng(1234.5678)
Msgbox CSng(1234.5678)
sVar = CStr(1234.5678)
MsgBox sVar
end sub
```

XVII.37. Функций CurDir

Обзор:

Возвращает строку, содержащую текущий путь для указанного устройства (диска). Если параметр опущен или имеет нулевую длину, то возвращается путь для текущего диска. Эта функция сейчас не работает! У меня она всегда возвращает путь к программе OOo

Синтаксис:

```
CurDir [(Text As String)]
```

Возвращаемое значение:

Строка String

Параметры:

Text: Необязательная, зависимая от регистра букв, строка, определяющая существующий диск, для которого нужно выдать текущий путь. Это должен быть один символ.

Пример:

Лист. 412: Функций CurDir

```
Sub ExampleCurDir
  MsgBox CurDir("C")
  MsgBox CurDir("P")
  MsgBox CurDir()
end sub
```

XVII.38. Функций Date

Обзор:

Возвращает или изменяет текущую системную дату. Формат даты зависит от языка установки OOo (локали).

Синтаксис:

```
Date
Date = Text As String
```

Возвращаемое значение:

Строка String

Параметры:

Text: Новая системная дата, как она определена в установке локали.

Пример:

Лист. 413: Функций Date

```
Sub ExampleDate
  MsgBox "Дата равна " & Date
End Sub
```

XVII.39. Функция DateSerial

Обзор:

Преобразует год, месяц и день в объект даты Date. Внутреннее представление даты — двойное число с плавающей точкой, где 0 соответствует 31 декабря 1899 года. Можно считать это число равным числу дней от этой базовой даты. Отрицательные числа соответствуют датам раньше, а положительные — позже этой базовой даты.

См. также DateValue, Date, и Day.

> ***Внимание*** *Две цифры года считаются годом 19xx. Это не так для функции DateValue.*

Синтаксис:

```
DateSerial (year, month, day)
```

Возвращаемое значение:

Дата Date

Параметры:

Year: Целое — год. Значения между 0 и 99 считаются годами от 1900 до 1999. Все другие значения года должны содержать четыре цифры.

Month: целое — месяц. Допустимые знчения от 1 до 12.

Day: Целое — день месяца. Допустимые значения от 1 до 28, 29, 30 или 31 (в зависимости от месяца и года).

Пример:

Лист. 414: Функция DateSerial

```
Sub ExampleDateSerial
  Dim lDate as Long, sDate as String, lNumDays As Long
  lDate = DateSerial(2002, 6, 8)
  sDate = DateSerial(2002, 6, 8)
  MsgBox lDate REM возвращает 37415
  MsgBox sDate REM возвращает 06/08/2002
  lDate = DateSerial(02, 6, 8)
  sDate = DateSerial(02, 6, 8)
  MsgBox lDate REM возвращает 890
  MsgBox sDate REM возвращает 06/08/1902
end sub
```

XVII.40. Функция DateValue

Обзор:

Преобразует строку с датой в одинарное число с плавающей точкой, пригодное для определения числа дней между двумя датами.

См. также DateSerial, Date, и Day

> ***Внимание*** *Две цифры года считаются годом 20xx. Это не так для функции DateSerial.*

Синтаксис:

```
DateValue [(date)]
```

Возвращаемое значение:

Длинное целое (Long)

Параметры:

Date: Строковое представление даты

Пример:

Лист. 415: Функция DateValue

```
Sub ExampleDateValue
  Dim s(), i%, sMsg$, l1&, l2&
  REM Все эти строки соответствуют 8 июня 2002 года
  s = Array("06.08.2002", "6.08.02", "6.08.2002", "June 08, 2002", _
    "Jun 08 02", "Jun 08, 2002", "Jun 08, 02", "06/08/2002")
  REM Если Вы используете эти значения, возникнет ошибка,
  REM что противоречит встроенной Справке
  REM s = Array("6.08, 2002", "06.08, 2002", "06,08.02", "6,08.2002",
"Jun/08/02")
  sMsg = ""
  For i = LBound(s()) To UBound(s())
    sMsg = sMsg + DateValue(s(i)) + "<=" + s(i) + Chr(13)
  Next
  MsgBox sMsg
  Print "я женился " + (DateValue(Date) — DateValue("6/8/2002")) + " дней
назад"
end sub
```

XVII.41. Функция Day

Обзор:

Возвращает день месяца по дате (serial date number), созданной функциями DateSerial или DateValue.

Синтаксис:

```
Day (Number)
```

Возвращаемое значение:

Целое (Integer)

Параметры:

Number: Номер составной даты (Serial date number), такой, как возвращает функция DateSerial

Пример:

Лист. 416: Функция Day

```
Sub ExampleDay
  Print Day(DateValue("6/8/2002")  REM 8
```

```
  Print Day(DateSerial(02,06,08))  REM 8
end sub
```

XVII.42. Оператор объявления Declare

Обзор:

Используется для объявления и для определения процедуры в DLL (Dynamic Link Library) для выполнения из OpenOffice.org Basic. Ключевое слово ByVal должен использоваться, если параметры нужно передавать по значению, а не по ссылке.

См. также: FreeLibrary, Call

Синтаксис:

```
Declare {Sub | Function} Name Lib «Libname» [Alias «Aliasname"] [Parameter]
[As Type]
```

Parameter/Element:

Name: Имя, отличное от имен, определенных в DLL, используемое для вызова процедуры из Basic.

Aliasname: Имя процедуры, как оно определено в DLL.

Libname: Имя файла или системное имя DLL. Эта библиотека автоматически загружается, когда данная функция используется первый раз.

Argumentlist: Список параметров, представляющих аргументы, которые передаются процедуре, когда она вызвана. Тип и число параметров зависит от выполняемой процедуры.

Type: Определяет тип данных того значения, которое возвращается процедурой-функцией. Можно опустить эту часть, если определение типа введено после имени.

Пример:

Лист. 417: Оператор объявления Declare

```
Declare Sub MyMessageBeep Lib "user32.dll" Alias "MessageBeep" ( long )
Sub ExampleDeclare
  Dim lValue As Long
  lValue = 5000
  MyMessageBeep( lValue )
  FreeLibrary("user32.dll" )
End Sub
```

XVII.43. Оператор DefBool

Обзор:

Устанавливает тип переменной (булевский — DefBool) по умолчанию дя тех переменных, имена которых начинаются с символа, попадающего в указанный в операторе интервал, действующий в тех случаях, когда не указано ключевое слово типа и не задан символ, определяющий тип, после имени переменной. Можно задать, что все переменные с именем, начинающимся на «b», например, автоматически получат тип Boolean.

См. также: DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, и DefVar.

Синтаксис:

```
DefBool CharacterRange1[, CharacterRange2[,...]]
```

Параметры:

CharacterRange: Буквы или интервалы букв, задающие имена переменных, для которых устанавливается тип данных по умолчанию.

Пример:

Лист. 418: Оператор DefBool

```
'Prefix definition for variable types:
DefBool b
DefDate t
DefDbL d
DefInt i
DefLng l
DefObj o
DefVar v
DefBool b-d,q
Sub ExampleDefBool
  cOK = 2.003
  zOK = 2.003
  Print cOK  REM True
  Print zOK  REM 2.003
End Sub
```

XVII.44. Оператор DefDate

Обзор:

Установить тип переменной (дата — DefDate) по умолчанию, в зависимости от первой буквы имени переменной, попадающей в интервал, указанный в операторе, если только не задан в конце имени переменной символ, определяющий её тип, и не задан тип переменной ключевым словом.

Можно указать, что все переменные, чье имя начинается с «t», например, автоматически получают тип Date.

См. также: DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, и DefVar.

Синтаксис:

```
DefDate Characterrange1[, Characterrange2[,...]]
```

Параметры:

Characterrange: Буквы или интервалы букв, определяющие интервал имен переменных, для которых устанавливается по умолчанию тип data

Пример:

```
См. выше Лист. 418 ExampleDefBool
```

XVII.45. Оператор DefDbl

Обзор:

Устанавливает тип переменной (двойное с плавающей точкой — DefDbl), согласно первым буквам имени переменной, попадающим в указанный в операторе интервал, если только не указан в конце имени переменной символ, задающий её тип, и не указан тип переменной ключевым словом. Можно указать, что все переменные, чьи имена начинаются на «d», например, автоматически получают тип Double.

См. также: DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, и DefVar.

Синтаксис:

```
DefDbl Characterrange1[, Characterrange2[,...]]
```

Параметры:

Characterrange: Буквы или интервалы букв, определяющие имена переменных, для которых устанавливается по умолчанию тип Double.

Пример:

```
См. выше Лист. 418 ExampleDefBool
```

XVII.46. Оператор DefInt

Обзор:

Устанавливает тип переменной (целый Integer — DefInt) по умолчанию, который присваивается всем переменным, первые буквы которых попадают

в интервал, указанный в операторе, если только имя переменной не оканчивается символом, определяющим её тип, и тип переменной не задан ключевым словом. Можно указать, что все переменные, чьи имена начинаются на «i», например, автоматически получают тип Integer.

См. также: DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, и DefVar.

Синтаксис:

```
DefInt Characterrange1[, Characterrange2[,...]]
```

Параметры:

Characterrange: Буквы или интервалы букв, определяющие имена переменных, для который по умолчанию устанавливается тип целое Integer.

Пример:

```
См. выше Лист. 418 ExampleDefBool
```

XVII.47. DefLng Statement

Обзор:

Установить тип переменной (длинное целое Long — DefLng) по умолчанию для всех переменных, чьи имена начинаются с символа, попадающего в указанный в операторе интервал, если только после имени переменной не стоит символ, определяющий её тип, и тип переменной не задан ключевым словом. Можно указать, что все переменные, начинающиеся с буквы «l», например, автоматически получают тип Long.

См. также: DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, и DefVar.

Синтаксис:

```
DefLng Characterrange1[, Characterrange2[,...]]
```

Параметры:

Characterrange: Буквы, определяющие интервал имен переменных, для которых по умолчанию присваивается тип Long.

Пример:

```
См. выше Лист. 418 ExampleDefBool
```

XVII.48. Оператор DefObj

Обзор:

Установить тип переменной (объект Obiect — DefObj) по умолчанию, если первая буква имени переменной попадает в интервал, заданный оператором, если только в конце имени переменной не приписан символ, определяющий её тип, и тип не задан ключевым словом. Можно установить, что все переменные, чьи имена начинаются с «o», например, автоматически получают тип Object.

См. также: DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, и DefVar.

Синтаксис:

```
DefObj Characterrange1[, Characterrange2[,...]]
```

Параметры:

Characterrange: Буквы или интервалы букв, определяющие имена переменных, для который по умолчанию устанавливается тип Obiect.

Пример:

```
См. выше Лист. 418 ExampleDefBool
```

XVII.49. Оператор DefVar

Обзор:

Установить типа переменной (Variant — DefVar) по умолчанию для переменных, первая буква имени которых попадает в интервал, заданный в операторе, если только после имени переменной не указан символ, определяющий её тип, и тип переменной не задан ключевым словом. Можно указать, что все переменные, чье имя начинается с «v», например, автоматически получают тип Variant.

См. также: DefBool, DefDate, DefDbL, DefInt, DefLng, DefObj, и DefVar.

Синтаксис:

```
DefVar Characterrange1[, Characterrange2[,...]]
```

Параметры:

Characterrange: Буквы или интервалы букв, задающие имена переменных, которым по умолчанию присваивается тип Variant.

Пример:

```
См. выше Лист. 418 ExampleDefBool
```

XVII.50. Оператор Dim

Обзор:

Объявляет переменные. Тип каждой переменной объявляется отдельно, типом по умолчанию является Variant. Следующий пример объявляет переменные a, b, и c как имеющие тип Variant и d как имеющую тип Date.

```
Dim a, b, c, d As Date
```

Тип переменных может также задаваться путем использования символа, дописываемого в конец имени переменной. Это подробнее описано в разделе, посвященном типам переменных.

Dim используется для объявления переменных, локальных внутри процедур. Глобальные переменные вне процедур определяются операторами PUBLIC или PRIVATE.

Если только не задан оператор «Option Explicit» в начале модуля, то для переменных — не массивов — допускается использование без объявления, тогда их тип variant.

Поддерживаются и одномерные, и многомерные массивы (arrays)

См. также: Public, Private, ReDim

Синтаксис:

```
[ReDim]Dim Name_1 [(start To end)] [As Type][, Name_2 [(start To end)] [As
Type][,...]]
```

Параметры/ключевые слова:

Name_i: Имя переменой или массива

Start, End: Целые константы (Integer) от -32768 до 32767 включительно, задающие интервал индекса массива. На уровне процедур ReDim позволяет использовать числовые выражения, так что границы интервала индексов массива могут быть изменены во время выполнения макроса.

Type: ключевое слово, используемое для объявления типа данных, хранящихся в переменной. Поддерживаемые типы переменных: Boolean, Currency, Date, Double, Integer, Long, Object, Single, String, или Variant.

Пример:

Лист. 419: Оператор Dim

```
Sub ExampleDim
  Dim s1 As String, i1 As Integer, i2%
  Dim a1(5) As String REM 0 to 6
  Dim a2(3, -1 To 1) As String REM (0 to 3, -1 to 1)
  Const sDim as String = " Dimension:"
  For i1 = LBound(a2(), 1) To UBound(a2(), 1)
    For i2 = LBound(a2(), 2) To UBound(a2(), 2)
      a2(i1, i2) = Str(i1) & ":" & Str(i2)
    Next
  Next
  For i1 = LBound(a2(), 1) To UBound(a2(), 1)
    For i2 = LBound(a2(), 2) To UBound(a2(), 2)
      Print a2(i1, i2)
    Next
  Next
End Sub
```

XVII.51. Функция DimArray

Обзор:

Создает массив типа Variant. Этот оператор работает точно как Dim(СписокАргументов). Если нет ни одного аргумента, то создается пустой массив (empty array). Если есть параметры, то создается по одному измерению для каждого из параметров.

См. также: Array

Синтаксис:

```
DimArray (СписокАргументов)
```

Возвращаемое значение:

Массив типа Variant

Параметры:

Argument list: Необязательный разделенный запятыми список целых чисел (integers).

Пример:

```
DimArray(2,2,4) — это то же самое, что и DIM a(2,2,4)
```

XVII.52. Функция Dir

Обзор:

Возращает имя файла, директории (папки) или всех файлов и папок внутри директории (папки) или диска, которые соответствуют указанному пути поиска. Возможные способы использования включают проверку, что файл или папка существуют, или вывод списка файлов и папок внутри нужной папки.

Если ни один файл и ни одна папка не подошли, возвращается строка нулевой длины. Повторный вызов функции Dir до тех пор, пока не будет возвращена строка нулевой длины, позволяет перебрать все подходящие значения (файлы, папки).

Атрибуты диска и папки являются исключительными значениями, так что если запросить один из них, то это будет единственным значением, которое Вы получите. Я не смог определить, какой из них более старший, потому что в версии 1.0.3, атрибут volume ничего не делает.

Артибуты являются частью атрибутов, доступных в GetAttr.

См. также: GetAttr

Синтаксис:

```
Dir [(Text As String[, Attrib As Integer])]
```

Возвращаемое значение:

Строка (String)

Параметры:

Text: строка, которая задает путь поиска, директорию (папку) или файл. Допускается указание адреса URL.

Attrib: Целое выражение (Integer) для атрибутов файла. Функция Dir возвратит только файлы и папки, которые соответствуют заданным атрибутам. Нужно сложить числовые значения атрибутов, чтобы задать их комбинацию.

Атрибут Attribute	*Описание*
0	Обычные файлы
2	Скрытые файлы (Hidden)
4	Системные файлы (System)

Атрибут *Attribute*	*Описание*
8	Возвращает имя тома (name of the volume (Exclusive — не комбинируется с другими атрибутами).
16	Возвращает директорию (папку) (Exclusive — не комбинируется с другими атрибутами).

Пример:

Лист. 420: Функция Dir

```
Sub ExampleDir
  REM Выводит все файлы и папки
  Dim sFile as String, sPath As String
  Dim sDir as String, sValue as String
  Dim iFIle as Integer
  sFile= "Файлы: "
  sDir="Папки:"
  iFile = 0
  sPath = CurDir
  REM  0 : Обычные файлы
  REM  2 : Скрытые файлы
  REM  4 : Системные файлы
  REM  8 : Возвращает имя тома
  REM 16 : Возвращает только имя папки
  REM Этот пример выведет ТОЛЬКО папки, потому что включен атрибут 16 !
  REM Удалите атрибут  16 — и Вы получите файлы
  sValue = Dir$(sPath, 0 + 2 + 4 + 16)
  Do
    If sValue <> "." and sValue <> ".." Then
      If (GetAttr( sPath + getPathSeparator() + sValue) and 16) > 0 Then
        REM здесь папки
        sDir = sDir & chr(13) & sValue
      Else
        REM здесь файлы
        If iFile Mod 3 = 0 Then sFile = sFile + Chr(13)
        iFile = iFile + 1
        sFile = sFile + sValue &"; "
      End If
    End If
    sValue = Dir$
  Loop Until sValue = ""
  MsgBox sDir,0,sPath
  MsgBox "" & iFile & " " & sFile,0,sPath
End Sub
```

> ***Совет*** *Метод getPathSeparator() включен, даже хотя он не включен в Справку. Я ещё не нашел, где это*

описано, но он используется в распространяемых инструментах и справках к OpenOffice.

Внимание *Некоторые операционные системы включают папки «.» и «..», которые ссылаются на текущую папку и на родительскую (вышестоящую) папку соответственно. Если Вы пишете программу, которая перебирает все дерево папок, содержащихся в данной папке, то Вы, вероятно, не захотите переходить по этим двум ссылкам, иначе получится бесконечный цикл*

Внимание *Когда Вы получаете список папок (атрибут 16), то файлы не включаются в список, хотя в Справке некоторые примеры показывают, что файлы будут выдаваться.*

XVII.53. Операторы Do...Loop

Обзор:

Конструкция для повторения операторов

См. также: Loop Control, страница 337.

Синтаксис1:

```
Do [{While | Until} condition = True]
statement block
[Exit Do]
statement block
Loop
```

Синтаксис2:

```
Do
statement block
[Exit Do]
statement block
Loop [{While | Until} condition = True]
```

XVII.54. Оператор End

Обзор:

Обозначает конец программы, блока, процедуры (subroutine) или функции
См. также: Exit

Синтаксис:

Форма	*Действие*
End	Оператор End сам по себе заканчивает выполнение программы. Его можно вписывать в любом месте. Он необязателен
End Function	Обозначает конец функции
End If	Обозначает конец блока If...Then...Else
End Select	Обозначает конец блока Select Case
End Sub	Обозначает конец процедуры

Пример:

Лист. 421: Оператор End

```
Sub ExampleEnd
  Dim s As String
  s = InputBox ("Введите целое число:","White Space Checker")
  If IsWhiteSpace(Val(s)) Then
    Print "ASCII " + s + " является пробелом (white space)"
  Else
    Print "ASCII " + s + " не является пробелом (white space)"
  End If
  End
  Print "Это место программы никогда не выполнится"
End Sub
Function IsWhiteSpace(iChar As Integer) As Boolean
  Select Case iChar
  Case 9, 10, 13, 32, 160
    IsWhiteSpace = True
  Case Else
    IsWhiteSpace = False
  End Select
End Function
```

XVII.55. Функция Environ

Обзор:

Возвращает значение переменной окружения (environment). Переменные окружения зависят от операционной системы. На компьютере Macintosh Эта функция возвращает пустую строку.

Синтаксис:

```
Environ (Environment As String)
```

Возвращаемое значение:

Строка (String)

Параметры:

Environment: Переменная окружения, чье значение возвращается

Пример:

Лист. 422: Функция Environ

```
Sub ExampleEnviron
  MsgBox "Path = " & Environ("PATH")
End Sub
```

XVII.56. Функция EOF

Обзор:

Используйте EOF, чтобы избежать попыток прочитать данные после конца файла. Когда конец файла достигнут, функция EOF возвращает значение True (-1).

См. также Open, Close, Kill, и FreeFile

Синтаксис:

```
EOF (intexpression As Integer)
```

Возвращаемое значение:

Булевское (Boolean)

Параметры:

Intexpression: Целое (Integer) выражение, которое задает номер открытого файла.

Пример:

Лист. 423: Функция EOF

```
'Этот пример получен доработкой Справки on-line
'Пример из Справки on-line не работает
Sub ExampleEof
  Dim iNumber As Integer
  Dim aFile As String
  Dim sMsg as String, sLine As String
  aFile = "c:\DeleteMe.txt"
  iNumber = Freefile
  Open aFile For Output As #iNumber
  Print #iNumber, "Первая строка текста"
  Print #iNumber, "Другая строка текста"
  Close #iNumber
  iNumber = Freefile
  Open aFile For Input As iNumber
  While Not Eof(iNumber)
    Line Input #iNumber, sLine
    If sLine <>"" Then
      sMsg = sMsg & sLine & chr(13)
    End If
  Wend
  Close #iNumber
  Msgbox sMsg
End Sub
```

XVII.57. Функция EqualUnoObjects

Обзор:

Проверяет, являются ли два объекта UNO экземплярами одного и того же объекта UNO.

Синтаксис:

```
EqualUnoObjects( oObj1, oObj2 )
```

Возвращаемое значение:

Булевское (Boolean)

Пример:

Лист. 424: Функция EqualUnoObjects

```
Sub ExampleEqualUnoObjects
  Dim oIntrospection, oIntro2, Struct2
  REM Копия объектов -> тот же самый экземпляр
  oIntrospection = CreateUnoService( "com.sun.star.beans.Introspection" )
  oIntro2 = oIntrospection
```

```
  Print EqualUnoObjects( oIntrospection, oIntro2 )
  REM Структуры копируются по значению -> новый экземпляр
  Dim Struct1 As New com.sun.star.beans.Property
  Struct2 = Struct1
  Print EqualUnoObjects( Struct1, Struct2 )
End Sub
```

XVII.58. Оператор EQV

Обзор:

Вычисляет логическую эквивалентность двух выражений. При побитовом сравнении оператор EQV устанавливает в 1 соответствующий бит результата, если либо этот бит равен 1 в обоих выражениях, либо этот бит равен 0 в обоих выражениях. Используется стандартное правило старшинства математических операторов, как показано на странице 349 , результат оператора показан в следующей таблице:

x	*y*	*x EQV y*
ИСТИНА	ИСТИНА	ИСТИНА
ИСТИНА	ЛОЖЬ	ЛОЖЬ
ЛОЖЬ	ИСТИНА	ЛОЖЬ
ЛОЖЬ	ЛОЖЬ	ИСТИНА
1	1	1
1	0	0
0	1	0
0	0	1

Синтаксис:

```
Result = Expression1 EQV Expression2
```

Параметры:

Result: Числовая переменная, содержащая результат сравнения.

Expression1, expression2 : Сравниваемые выражения

Пример:

Лист. 425: Оператор EQV

```
Sub ExampleEQV
  Dim vA as Variant, vB as Variant, vC as Variant, vD as Variant
```

```
  Dim vOut as Variant
  vA = 10: vB = 8: vC = 6: vD = Null
  vOut = vA > vB  EQV vB > vC  REM возвращает -1
  Print vOut
  vOut = vB > vA  EQV vB > vC  REM возвращает -1
  Print vOut
  vOut = vA > vB  EQV vB > vD  REM возвращает 0
  Print vOut
  vOut = (vB > vD EQV vB > vA) REM возвращает -1
  Print vOut
  vOut = vB EQV vA REM возвращает -1
End Sub
```

XVII.59. Функция Erl

Обзор:

Возвращает номер строки, в которой произошла ошибка при выполнении программы

См. также: Err

Синтаксис:

```
Erl
```

Возвращаемое значение:

Целое (Integer)

Пример:

Лист. 426: Функция Erl

```
Sub ExampleErl
  On Error GoTo ErrorHandler
  Dim iVar as Integer
  iVar = 0
  iVar = 4 / iVar
  Exit Sub
ErrorHandler:
  REM Error 11 : Division by Zero
  REM In line: 8
  REM ....
  MsgBox "Error " & err & ": " & error$ + chr(13) + _
    "In line : " + Erl + chr(13) + Now , 16 ,"An error occured"
End Sub
```

XVII.60. Функция Err

Обзор:

Возвращает номер ошибки, — последней ошибки при выполнении

См. также: Erl

Синтаксис:

```
Err
```

Возвращаемое значение:

Целое (Integer)

Пример:

Лист. 427: Функция Err

```
Sub ExampleErr
  On Error GoTo ErrorHandler
  Dim iVar as Integer
  iVar = 0
  iVar = 4 / iVar
  Exit Sub
ErrorHandler:
  REM Error 11 : Division by Zero
  REM In line: 8
  REM ....
  MsgBox "Error " & err & ": " & error$ + chr(13) + _
    "In line : " + Erl + chr(13) + Now , 16 ,"An error occured"
End Sub
```

XVII.61. Оператор Error не работает так, как описано

Обзор:

Симулирует возникновение ошибки при выполнении программы. Он НЕ работает!

Синтаксис:

```
Error errornumber As Integer
```

Возвращаемое значение:

Целое (Integer)

Параметры:

Errornumber: Целое (Integer) выражение, которое задает номер ошибки, которую нужно симулировать.

XVII.62. Функция Error

Обзор:

Возвращает сообщение об ошибке, соответствующей заданному номеру (коду) ошибки

Синтаксис:

```
Error (Expression)
```

Возвращаемое значение:

Строка (String)

Параметры:

Expression: Необязательное — целое (integer), содержащее номер (код) ошибки. Если не задано, возвращается сообщение об ошибке — последней произошедшей ошибке выполнения макроса.

Пример:

Лист. 428: Функция Error

```
Sub ExampleError
  On Error GoTo ErrorHandler
  Dim iVar as Integer
  iVar = 0
  iVar = 4 / iVar
  Exit Sub
ErrorHandler:
  REM Error 11 : Division by Zero
  REM In line: 8
  REM ....
  MsgBox "Error " & err & ": " & error$ + chr(13) + _
    "In line : " + Erl + chr(13) + Now , 16 ,"An error occured"
End Sub
```

XVII.63. Оператор Exit

Обзор:

Оператор Exit используется для выхода из конструкций Do...Loop, For...Next, Function или Subroutine. Другими словами, можено немедленно выйти из любой из этих конструкций. Если я нахожусь внутри процедуры (Sub) и обнаружил, что аргументы неправильные, то я могу немедленно покинуть эту процедуру.

См. также: End

Синтаксис:

Форма	*Действие*
Exit Do	Выход из ближайшего действующего блока Do...Loop.
Exit For	Выход из ближайшего действующего блока For...Next.
Exit Function	Выход из функции и продолжение выполнения после вызова функции
Exit Sub	Выход из процедуры и продолжение выполнения после вызова процедуры

Пример:

Лист. 429: Оператор Exit

```
Sub ExampleExit
  Dim sReturn As String
  Dim sListArray(10) as String
  Dim siStep as Single
  REM Build array ("B", "C", ..., "L")
  For siStep = 0 To 10 REM Заполняем массив тестовыми данными
    sListArray(siStep) = chr(siStep + 66)
  Next siStep
  sReturn = LinSearch(sListArray(), "M")
  Print sReturn
  Exit Sub  REM На самом деле это бесполезный оператор, выход из процедуры
будет все равно!
End Sub
REM Возвращает индекс элемента или (LBound(sList()) — 1), если элемент
не найден
Function LinSearch( sList(), sItem As String ) as integer
  Dim iCount As Integer
  REM LinSearch ищет в текстовом массиве TextArray:sList() значение
TextEntry:
  For iCount=LBound(sList()) To UBound(sList())
    If sList(iCount) = sItem Then
      LinSearch = iCount
      Exit Function  REM Вероятно, это хорошее использование оператора Exit
здесь!
    End If
  Next
  LinSearch = LBound(sList()) — 1
End Function
```

XVII.64. Функция Exp

Обзор:

Возвращает число e = 2.718282 (базу натуральных логарифмов), возведенное в заданную степень

См. также: Log

Синтаксис:

```
Exp (Number)
```

Возвращаемое значение:

Двойное с плавающей точкой (Double)

Параметры:

Number: Любое числовое выражение

Пример:

Лист. 430: Функция Exp

```
Sub ExampleExp
  Dim d as Double, e As Double
  e = Exp(1)
  Print "e = " & e
  Print "ln(e) = " & Log(e)
  Print "2*3 = " & Exp(Log(2.0) + Log(3.0))
  Print "2^3 = " & Exp(Log(2.0) * 3.0)
end sub
```

XVII.65. Функция FileAttr

Обзор:

Первой целью функции FileAttr было определение способа доступа к файлу, открытому с помощью оператора Open. Установка второго параметра равным 1 запрашивает способ доступа, составленный из слагаемых, перечисленных в следующей таблице

Значение	*Описание типа доступа*
1	INPUT (файл открыт для ввода)
2	OUTPUT (файл открыт для вывода)
4	RANDOM (файл открыт для произвольного доступа)

Значение	*Описание типа доступа*
8	APPEND (Файл открыт для дописывания в конец)
32	BINARY (Файл открыт в бинарном режиме)

Второй целью функции FileAttr было определение MS-DOS атрибутов файла, открытого оператором Open. Это значение зависит от операционной системы. Установка второго параметра равным 2 запрашивает это значение атрибутов файла.

> ***Внимание*** *Атрибуты файла зависят от операционной системы. Если операционная система 32-битовая, то невозможно использовать функцию FileAttr для определения атрибутов файла, потому что возвращается нуль.*

См. также: Open

Синтаксис:

```
FileAttr (FileNumber As Integer, Attribute As Integer)
```

Возвращаемое значение:

Целое (Integer)

Параметры:

FileNumber: Номер, использованный в операторе Open при открытии файла

Attribute: Целое (Integer), указывающее, какие данные нужны. Значение 1 запрашивает тип доступа к файлу, значение 2 запрашивает атрибуты файла.

Пример:

Лист. 431: Функция FileAttr

```
Sub ExampleFileAttr
  Dim iNumber As Integer
  iNumber = Freefile
  Open "file:///c|/data.txt" For Output As #iNumber
  Print #iNumber, "Random Text"
  MsgBox AccessModes(FileAttr(#iNumber, 1 )),0,"Access mode"
  MsgBox FileAttr(#iNumber, 2 ),0,"File attribute"
  Close #iNumber
End Sub
Function AccessModes(x As Integer) As String
```

```
  Dim s As String
  s = ""
  If (x and 1) <> 0 Then s = "INPUT"
  If (x and 2) <> 0 Then s = "OUTPUT"
  If (x and 4) <> 0 Then s = s & " RANDOM"
  If (x and 8) <> 0 Then s = s & " APPEND"
  If (x and 32) <> 0 Then s = s & " BINARY"
  AccessModes = s
End Function
```

XVII.66. Оператор FileCopy

Обзор:

Копирует файл. Нельзя копировать файл, который сейчас открыт

Синтаксис:

```
FileCopy TextFrom As String, TextTo As String
```

Параметры:

TextFrom: Строка, указывающая имя исходного файла

TextTo: Строка, указывающая имя выходного файла

Пример:

Лист. 432: Оператор FileCopy

```
Sub ExampleFilecopy
  Filecopy "c:\Data.txt", "c:\Temp\Data.sav"
End Sub
```

XVII.67. Функция FileDateTime

Обзор:

Возвращает строку, которая показывает дату и время создания или последнего изменения файла, причем эта строка возвращается в формате, зависящем от операционной системы, на моем компьютере это «MM/DD/YYYY HH:MM:SS». Возможно применение функции DateValue к этой строке.

Синтаксис:

```
FileDateTime(Text As String)
```

Возвращаемое значение:

String

Параметры:

Text: Спецификация файла (не разрешены символы-шаблоны * ? и т.п.) Допускается адрес URL

Пример:

Лист. 433: Функция FileDateTime

```
Sub ExampleFileDateTime
  REM 04/23/2003 19:30:03
  MsgBox FileDateTime("file://localhost/C|/macro.txt")
End Sub
```

XVII.68. Функция FileExists

Обзор:

Определяет, существует ли файл или директория (папка)

Синтаксис:

```
FileExists(FileName As String | DirectoryName As String)
```

Возвращаемое значение:

Булевское (Boolean)

Параметры:

FileName | DirectoryName: Спецификация файла или папки (не разрешены символы-шаблоны * ? и т.п.).

Пример:

Лист. 434: Функция FileExists

```
Sub ExampleFileExists
  MsgBox FileExists("C:\autoexec.bat")
  MsgBox FileExists("file://localhost/c|/macro.txt")
  MsgBox FileExists("file:///d|/private")
End Sub
```

XVII.69. Функция FileLen

Обзор:

Определяет длину файла. Если файл сейчас открыт, то возвращается длина файла, какой она была до его открытия. Чтобы определить текущую длину открытого файла, используйте функцию Lof.

Синтаксис:

```
FileLen(FileName As String)
```

Возвращаемое значение:

Длинное целое (Long)

Параметры:

FileName: Спецификация файла (не разрешены символы-шаблоны * ? и т.п.) Допускается адрес URL

Пример:

Лист. 435: Функция FileLen

```
Sub ExampleFileExists
  MsgBox FileLen("C:\autoexec.bat")
  MsgBox FileLen("file://localhost/c|/macro.txt")
End Sub
```

XVII.70. Функция FindObject

Обзор:

Получает имя переменной и возвращает ссылку на тот же объект. См. также FindPropertyObject. Выполнение следующего макроса демонстрирует, что это не слишком хорошо работает.

Лист. 436: Функция FindObject

```
Sub TestTheThing
  Dim oTst As Object
  Dim oDoc As Object
  oTst = FindObject("oDoc")

  REM yes
  If oTst IS oDoc Then Print "oTst и oDoc are the same"

  oDoc = ThisComponent
  oTst = FindObject("oDoc")
  REM no  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  If oTst IS oDoc Then Print "oTst и oDoc are the same"
  REM no  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  If oTst IS ThisComponent Then Print "oTst и ThisComponent are the same"
  REM yes
  If oDoc IS ThisComponent Then Print "oDoc и ThisComponent are the same"

  oDoc = ThisComponent
  oTst = FindObject("ThisComponent")
  REM yes
```

```
  If oTst IS oDoc Then Print "oTst и oDoc are the same"
  REM yes
  If oTst IS ThisComponent Then Print "oTst и ThisComponent are the same"
  REM yes
  If oDoc IS ThisComponent Then Print "oDoc и ThisComponent are the same"

  REM это покажет ThisComponent
  RunSimpleObjectBrowser(oTst)
  oDoc = ThisComponent
  oTst = ThisComponent.DocumentInfo
  oTst = FindPropertyObject(oDoc, "DocumentInfo")
  If IsNull(oTst) Then Print "Is Null"
  If oTst IS ThisComponent.DocumentInfo Then Print "They are the same"
'  RunSimpleObjectBrowser(oTst)

End Sub
```

XVII.71. Функция FindPropertyObject

Обзор:

Это тоже не слишком хорошо работает. Другими словами, считайте эту функцию неработоспособной!

Объект содержит объекты данных. Например, документ электронной таблицы имеет свойство DrawPages, на которое можно ссылаться напрямую с помощью команды ThisComponent.DrawPages. Можно использовать FindPropertyObject для получения ссылки на этот объект:

```
obj = FindPropertyObject(ThisComponent, «DrawPages»)
```

Теперь можно получить доступ к объекту DrawPages с помощью переменной obj. Я считаю это возможным источником ошибок!

Пример:

Лист. 437: Функция FindPropertyObject

```
Sub StrangeThingsInStarBasic

Dim oSBObj1 As Object
Dim oSBObj2 As Object
Dim oSBObj3 As Object

  Set oSBObj1 = Tools
  RunSimpleObjectBrowser(oSBObj1)
  'Имеется также свойство Name!!
  Print oSBObj1.Name ' @SBRTL ?? что это такое?

  'P.S....
  Set oSBObj2 = FindObject("Gimmicks")
  Print oSBObj2.Name ' @SBRTL again...
```

```
  'можно изменить свойство name, но это ничего не дает:
'  oSBObj2.Name = "Ciao" : Print oSBObj2.Name
'  oSBObj2.Name = "@SBRTL" : Print oSBObj2.Name

  'требуется, чтобы сейчас библиотека Gimmicks уже была загружена
  GlobalScope.BasicLibraries.LoadLibrary("Gimmicks")

  'другой устаревший, нерекомендуемый, недокументированный, практически
неработающий кусок кода программы....

  'userfields — это модуль в библиотеке Gimmicks
  Set oSBObj3 = FindPropertyObject(oSBObj2, "Userfields")
  Print (oSBObj3 Is Gimmicks.Userfields)

  'требуется, чтобы сейчас библиотека Gimmicks уже была загружена
  GlobalScope.BasicLibraries.LoadLibrary("Gimmicks")

  'функция StartChangesUserfields находится в модуле Userfields
  'вызов по полному имени!
  oSBObj2.Userfields.StartChangesUserfields
End Sub
```

XVII.72. Функция Fix

Обзор:

Возвращает целую часть числового выражения, полученную отбрасыванием дробной части.

См. тажке: CInt, Int

Синтаксис:

```
Fix(Expression)
```

Возвращаемое значение:

Двойное с плавающей точкой (Double)

Параметры:

Expression: Число, для которого нужно вычислить целую часть.

Пример:

Лист. 438: Функция Fix

```
sub ExampleFix
  Print Fix(3.14159)   REM returns 3.
  Print Fix(0)         REM eturns 0.
  Print Fix(-3.14159)  REM returns -3.
End Sub
```

XVII.73. Конструкция For...Next

Обзор:

Конструкция для повторения операторов с автоматически изменяющимся счетчиком

См. также: For....Next on Page 336.

Синтаксис:

```
For counter=start To end [Step step]
БлокОператоров
  [Exit For]  ' это если нужно досрочно выйти из цикла
БлокОператоров
Next [counter]
```

XVII.74. Функция Format

Обзор:

Преобразует число в строку, отформатированную согласно (необязательной) строке формата. Несколько форматов могут находиться в одной строке формата. Каждый отдельный формат отделяется точкой с запятой «;». Первый формат используется для положительных числе, второй — для отрицательных, третий — для нуля. Если имеется только один формат, то он применяется для всех чисел.

Код	*Описание*
0	Если число имеет цифру в позиции, соответствующей 0 в формате, то выводится цифра, иначе выводится 0. Это означает, что ведущие и завершающие нули выводятся, ведущие цифры не отбрасываются, а завершающие цифры дробной части округляются.
#	Это работает так же, как и 0, но ведущие и завершающие нули не выводятся.
.	указывает место для десятичной точки и определяет число цифр слева и справа от десятичной точки.
%	Умножает число на 100 и вставляет знак процента (%) там, где он указан в строке формата
E- E+ e-	Если строка формата содержит по меньшей мере один знак для цифр (0 или #) справа от данного символа, то число форматируется в «научной записи». Буква E или e

Код	*Описание*
e+	вставляется между числом и «экспонентой» (то есть показателем степени числа 10, на которое умножается число перед буквой E). Количество мест для цифр (0 или #) справа от данного символа определяет число цифр в экспоненте. Если экспонента отрицательная, то минус вставляется сразу перед экспонентой. Если экспонента отрицательная, то плюс выводится только для вариантов формата E+ или e+.
,	Запятая показывает место для разделителя тысяч. Она отделяет тысячи от сотен в числах с 4 и более цифрами. Разделитель тысяч выводится, если запятая в строке формата окружена знаками для цифр (0 или #).
- + $ ( ) space	плюс (+), минус (-), знак доллара ($), пробел или скобки, указанные прямо в строке формата, выводятся тем же символом.
\	Обратный слэш указывает, что следующий символ в строке формата нужно вывести тем же символом, то есть без обработки. Другими словами, это не позволяет рассматривать следующий символ в качестве возможного специального символа. Обратный слэш сам не выводится, для вывода (одного) обратного слэша нужно указать его дважды (\\) в строке формата. Символы, перед которыми нужно обязательно указать обратный слэш для вывода этих символов, — это символы формата даты и времени (a, c, d, h, m, n, p, q, s, t, w, y, /, :), формата чисел (#, 0, %, E, e, точка, запятая) и формата строк (@, &, <, >, !). Можно также заключить подобные символы в двойные кавычки.
General Number	(Обычное число) Числа выводятся так, как они записаны в строке формата
Currency	(Денежная сумма) Знак доллара помещается перед числом; отрицательные числа заключаются в круглые скобки. Выводятся две цифры после точки. (на самом деле, это зависит от языка установки OOo — локали)
Fixed	(Число с фиксированной точкой) По меньшей мере одна цифра выводится перед десятичной точкой. После точки выводятся две цифры
Standard	(Стандартное) Выводит числа с разделителем тысяч, зависящим от локали. После точки выводятся две цифры.

Код	*Описание*
Scientific	(Научное) Выводит числа в «научной записи». После точки выводятся две цифры.

> ***Внимание*** *Не делается никакого преобразования, если параметр не числовой (например, строковый), и возвращается пустая строка.*

> *Внимание* *Предупреждение В версии OOo 1.0.3.1 Format(123.555, «.##») выводит «.12356», что я считаю ошибкой. Изменение формата на «#.##» исправляет эту ошибку. Всегда используйте подходящий ведущий символ формата «#» или «0»*

Научная запись сейчас не работает

Денежная сумма выводится со знаком доллара справа

Сейчас невозможно избежать специальных символов (Currently not able to escape special characters) — нельзя обойти обработку специальных символов даже путем использования обратного слэша?

Синтаксис:

```
Format (Number [, Format As String])
```

Возвращаемое значение:

Строка (String)

Параметры:

Number: Числовое выражение для преобразования в форматированную строку

Format: Нужный формат. если он опущен, то функция Format работает как функция Str

Пример:

Лист. 439: Функция Format

```
Sub ExampleFormat
  MsgBox Format(6328.2, "##,##0.00")          REM = 6,328.20
  MsgBox Format(123456789.5555, "##,##0.00")  REM = 123,456,789.56
  MsgBox Format(0.555, ".##")                 REM .56
  MsgBox Format(123.555, "#.##")              REM 123.56
  MsgBox Format(0.555, "0.##")                REM 0.56
  MsgBox Format(0.1255555, "%#.##")           REM %12.56
  MsgBox Format(123.45678, "##E-####")        REM 12E1
  MsgBox Format(.0012345678, "0.0E-####")     REM 1.2E3 (broken)
  MsgBox Format(123.45678, "#.e-###")         REM 1e2
  MsgBox Format(.0012345678, "#.e-###")       REM 1e3 (broken)
  MsgBox Format(123.456789, "#.## is ###")    REM 123.45 is 679 (strange)
  MsgBox Format(8123.456789, "General Number")REM 8123.456789
  MsgBox Format(8123.456789, "Fixed")         REM 8123.46
  MsgBox Format(8123.456789, "Currency")      REM 8,123.46$ (broken)
  MsgBox Format(8123.456789, "Standard")      REM 8,123.46
  MsgBox Format(8123.456789, "Scientific")    REM 8.12E03
  MsgBox Format(0.00123456789, "Scientific")  REM 1.23E03 (broken)
End Sub
```

XVII.75. Функция FreeFile

Обзор:

Возвращает следующий доступный номер файла для открытия файла. Это позволяет убедиться, что номер файла, который Вы будете использовать, пока не был использован.

См. также Open, EOF, Kill, и Close.

Синтаксис:

```
FreeFile
```

Возвращаемое значение:

Целое (Integer)

Пример:

```
См. пример для оператора Close.
```

XVII.76. Функция FreeLibrary

Обзор:

Освобождает DLL, загруженную оператором Declare. Эта DLL будет автоматически загружена вновь, если его функции будут вызваны. Только те DLL, которые загружены в процессе выполнения программы на Basic, могут быть освобождены таким способом.

См. также: Declare

Синтаксис:

```
FreeLibrary (LibName As String)
```

Параметры:

LibName: Имя DLL.

Пример:

Лист. 440: Функция FreeLibrary

```
Declare Sub MyMessageBeep Lib "user32.dll" Alias "MessageBeep" ( long )
Sub ExampleDeclare
  Dim lValue As Long
  lValue = 5000
  MyMessageBeep( lValue )
  FreeLibrary("user32.dll" )
End Sub
```

XVII.77. Оператор Function

Обзор:

Объявляет функцию, заданную пользователем, в отличие от процедуры. Фукнции могут возвращать значения, а процедуры не могут.

См. также: Sub

Синтаксис:

```
Function Name[(VarName1 [As Type][, VarName2 [As Type][,...]]]) [As Type]
БлокОператоров
  [Exit Function]
БлокОператоров
End Function
```

Возвращаемое значение:

Тот тип, который объявлен для функции

Пример:

Лист. 441: Оператор Function

```
Function IsWhiteSpace(iChar As Integer) As Boolean
  Select Case iChar
  Case 9, 10, 13, 32, 160
    IsWhiteSpace = True
  Case Else
    IsWhiteSpace = False
  End Select
End Function
```

XVII.78. Оператор Get

Обзор:

Читает запись из файла (relative file) или последовательность байтов из двоичного файла (binary file) в переменную. Если параметр позиции в файле опущен, то данные читаются из текущей позиции в этом файле. Для файлов, открытых как двоичный файл, позиция является позицией байта в этом файле.

См. также: PUT

Синтаксис:

```
Get [#] FileNumber As Integer, [Position], Variable
```

Параметры:

FileNumber: Целое (Integer) выражение, которое определяет номер файла. Я использую функцию FreeFile для получения такого числа

Position: Для файлов, открытых в режиме произвольного доступа (Random), это номер записи, которую нужно прочесть.

Variable: Переменная, в которую нужно прочесть данные. Переменная типа Object не может здесь использоваться.

Пример:

Лист. 442: Оператор Get

```
'?? Это не работает!
Sub ExampleRandomAccess2
  Dim iNumber As Integer, aFile As String
  Dim sText As Variant REM обязательно тип variant
  aFile = "c:\data1.txt"
  iNumber = Freefile
  Open aFile For Random As #iNumber Len=5
  Seek #iNumber,1 REM позиция начала файла
```

```
  Put #iNumber,, "1234567890" REM заполняем строку текстом
  Put #iNumber,, "ABCDEFGHIJ"
  Put #iNumber,, "abcdefghij"
  REM Вот как файл выглядит сейчас!
  REM 08 00 0A 00 31 32 33 34  35 36 37 38 39 30 08 00   ....1234567890..
  REM 0A 00 41 42 43 44 45 46  47 48 49 4A 08 00 0A 00   ..ABCDEFGHIJ....
  REM 61 62 63 64 65 66 67 68  69 6A 00 00 00 00 00 00   abcdefghij
  REM
  Seek #iNumber,1
  Get #iNumber,,sText
  Print "При открытии:" & sText
  Close #iNumber
  iNumber = Freefile
  Open aFile For Random As #iNumber Len=5
  Get #iNumber,,sText
  Print "повторное открытие: " & sText
  Put #iNumber,,"ZZZZZ"
  Get #iNumber,1,sText
  Print "следующее чтение "& sText
  Get #iNumber,1,sText
  Put #iNumber,20,"Это текст в записи 20"
  Print Lof(#iNumber)
  Close #iNumber
End Sub
```

XVII.79. Функция GetAttr

Обзор:

Возвращает битовую маску, идентифицирующую тип файла. Атрибуты соответствуют тем, которые используются в функции Dir.

Attribute	*Description*
0	Обычный файл
1	Файл только для чтения
2	Скрытый файл
4	Системный файл
8	Имя тома
16	Директория (папка)
32	Архивный бит (файл был изменён после последнего сохранения backup)

См. также: Dir

> ***Внимание*** *Не работает в версии OOo 1.0.3.1. Проверяйте для той версии, которую используете Вы*

Синтаксис:

```
GetAttr (Text As String)
```

Возвращаемое значение:

Целое (Integer)

Параметры:

Text: Строковое выражение, содержащее однозначное определение файла. Допускается адрес URL.

Пример:

Лист. 443: Функция GetAttr

```
Sub ExampleGetAttr
  REM Должно выводиться " Read-Only Hidden System Archive"
  REM Выводит " Read-Only"
  Print FileAttributeString(GetAttr("C:\IO.SYS"))
  REM Должно выводить " Archive" и выводить "Normal"
  Print FileAttributeString(GetAttr("C:\AUTOEXEC.BAT"))
  REM "Папка"
  Print FileAttributeString(GetAttr("C:\WINDOWS"))
End Sub
Function FileAttributeString(x As Integer) As String
  Dim s As String
  If (x = 0) Then
    s = "Normal"
  Else
    s = ""
    If (x and 16) <> 0 Then s = "Directory"
    If (x and 1) <> 0 Then s = s & " Read-Only"
    If (x and 2) <> 0 Then s = " Hidden"
    If (x and 4) <> 0 Then s = s & " System"
    If (x and 8) <> 0 Then s = s & " Volume"
    If (x and 32) <> 0 Then s = s & " Archive"
  End If
  FileAttributeString = s
End Function
```

XVII.80. Функция GetProcessServiceManager

Обзор:

Получает центральный менеджер сервисов UNO (service manager). Это нужно, если требуется установить сервис с использованием CreateInstance с аргументами[1].

Синтаксис:

```
oServiceManager = GetProcessServiceManager()
```

Пример:

Лист. 444: Функция GetProcessServiceManager

```
oServiceManager = GetProcessServiceManager()
oIntrospection =
oServiceManager.createInstance("com.sun.star.beans.Introspection");
rem Это то же самое, что и следующий оператор:
oIntrospection = CreateUnoService("com.sun.star.beans.Introspection")
```

XVII.81. Функция GetSolarVersion

Обзор:

Возвращает внутреннее встроенное число — текущую версию OpenOffice.org. Вы можете написать свой макрос для обхода ошибок, связанных с разными версиями OOo. К сожалению, значение функции GetSolarVersion часто остается тем же самым, хотя версия меняется. Версия 1.0.3.1 возвращает «641», а версия 1.1RC3 возвращает 645, но этого не всегда достаточно для детальных различий. Следующий макрос возвращает реальную версию OOo.

Лист. 445: Функция GetSolarVersion

```
Function OOOVersion() As String
  'Получить версию выполняемого OOO
  'Author : Laurent Godard
  'e-mail : listes.godard@laposte.net
  '
  Dim oSet, oConfigProvider
  Dim oParm(0) As New com.sun.star.beans.PropertyValue
  Dim sProvider$, sAccess$
  sProvider = "com.sun.star.configuration.ConfigurationProvider"
  sAccess   = "com.sun.star.configuration.ConfigurationAccess"
  oConfigProvider = createUnoService(sProvider)
```

[1] Поищите пример получше ! Найдите пример, где есть аргументы этой функции!

```
  oParm(0).Name = "nodepath"
  oParm(0).Value = "/org.openoffice.Setup/Product"
  oSet = oConfigProvider.createInstanceWithArguments(sAccess, oParm())

  OOOVersion=oSet.getbyname("ooSetupVersion")
End Function
```

Синтаксис:

```
s = GetSolarVersion()
```

Возвращаемое значение:

Строка (String)

Пример:

Лист. 446: Функция GetSolarVersion

```
Sub ExampleGetSolarVersion
  REM Для версии of 1.0.3.1 возвращается "641"
  Print GetSolarVersion()
End Sub
```

XVII.82. Функция GetSystemTicks

Обзор:

Возвращает системные такты (system ticks), обеспечиваемые операционной системой. Число системных тактов, возвращаемых за определенный период времени, всегда зависит от операционной системы.

Синтаксис:

```
GetSystemTicks()
```

Возвращаемое значение:

Длинное целое (Long)

Пример:

Этот пример пытается измерить, сколько тактов происходит в секунду. Для Windows XP и версии 1.0.3.1 OpenOffice.org оказывается 1000 тактов в секунду

Лист. 447: Функция GetSystemTicks

```
Sub ExampleGetSystemTicks
  Dim lTick As Long, lMillisToWait As Long
  Dim lSecsToWait As Long, lTicksPerSec As Long
  lSecsToWait = 60
```

```
  lMillisToWait = lSecsToWait * 1000
  lTick = GetSystemTicks()
  wait(lMillisToWait)
  lTick = (GetSystemTicks() — lTick)
  lTicksPerSec = lTick / lSecsToWait
  MsgBox "В каждой секунде содержится примерно " & lTicksPerSec & " тактов
в секунду"
End Sub
```

XVII.83. Ключевое слово GlobalScope

Обзор:

Макросы и диалоги Basic организованы в библиотеки. Библиотека может содержать модули и (или) диалоги. В языке Basic контейнет библиотеки называется «BasicLibraries» , а контейнер диалогов называется «DialogLibraries». Хотя оба контейнера библиотек существуют в обоих уровнях: приложения и документа — в языке basic они автоматически загружаются, но это не так для документа. Для вызова этих контейнеров библиотек из документа нужно использовать ключевое слово GlobalScope.

Синтаксис:

```
GlobalScope
```

Пример:

Лист. 448: GlobalScope

```
' calling Dialog1 in the document library Standard
oDlgDesc = DialogLibraries.Standard.Dialog1
' calling Dialog2 in the application library Library1
oDlgDesc = GlobalScope.DialogLibraries.Library1.Dialog2
```

XVII.84. Оператор GoSub

Обзор:

Передает выполнение макроса на метку внутри текущей процедуры или функции. Операторы, следующие за этой меткой, выполняются до тех пор, пока не встретится оператор Return, после чего выполнение программы возвращается к месту после оператора GoSub.

См. также: Раздел по порядку выполнения программы, где объясняется, почему нужно избегать оператора GoSub.

> ***Совет*** *GoSub осталась как напоминание о старых версиях Basic и сохранена для совместимости.*

> *GoSub строго рекомендуется избегать, потому что приводит к созданию нечитаемого кода программы. Предпочтительно использовать процедуры и функции (Subs, Functions).*

Синтаксис:

```
Sub/Function
  REM Здесь любые операторы
  GoSub Label
  REM Здесь любые операторы
  GoSub Label
  Exit Sub/Function
Label:
  Блок операторов
  Return
End Sub/Function
```

Пример:

Лист. 449: Оператор GoSub

```
Sub ExampleGoSub
  Print "Перед выполнением gosub"
  GoSub SillyLabel
  Print "После оператора gosub"
  Exit Sub
SillyLabel:
  Print "После метки Silly Label"
  Return
End Sub
```

XVII.85. Оператор GoTo

Обзор:

Передает управление на метку внутри текущей процедуры или функции. Будут выполняться операторы после метки.

См. также: Раздел по порядку выполнения программы, где объясняется, почему нужно избегать использования GoTo

> ***Совет*** *GoTo осталось как напоминание об устаревших версиях языка Basic, оставленное для совместимости. . GoTo строго не рекомендуется для использования, так как оно приводит к созданию нечитаемого текста программы.*

Предопчтительнее процедуры и функции (Subs, Functions)

Синтаксис:

```
Sub/Function
  REM Здесь любые операторы
  GoTo Label
  REM Здесь любые операторы
  GoTo Label
  Exit Sub/Function
Label:
  Блок операторов
End Sub/Function
```

Пример:

Лист. 450: Оператор GoTo

```
Sub ExampleGoTo
  Print "Перед выполнением goto"
  GoTo SillyLabel
  Print "После оператора goto — это никогда не выполнится"
  Exit Sub        REM Это тоже никогда не выполнится
SillyLabel:
  Print "После метки Silly Label"
End Sub
```

XVII.86. Функция Green

Обзор:

Цвета представлены как длинные целые числа (long). Функция возвращает компонент «зеленый» (green) цвета, заданного кодом color. См. также RGB, Red, и Blue.

Синтаксис:

```
Green(Color As Long)
```

Возвращаемое значение:

Целое (Integer) в интервале от 0 до 255.

Параметры:

Color value: Длинное целое (Long) выражение, представляющее код цвета color.

Пример:

Лист. 451: Функция Green

```
Sub ExampleColor
  Dim lColor As Long
  lColor = RGB(255,10,128)
  MsgBox "Цвет " & lColor & " состоит из:" & Chr(13) &_
    "Red = " & Red(lColor) & Chr(13)&_
    "Green= " & Green(lColor) & Chr(13)&_
    "Blue= " & Blue(lColor) & Chr(13) , 64,"Colors"
End Sub
```

XVII.87. Функция HasUnoInterfaces

Обзор:

Проверяет, поддерживает ли объект Basic UNO указанные интерфейсы UNO. Возвращает true, только если ВСЕ указанные интерфейсы UNO подерживаются.

Синтаксис:

```
HasUnoInterfaces( oTest, UNO-Interface-Name 1 [, UNO-Interface-Name 2, ...])
```

Возвращаемое значение:

Булевское (Boolean)

Параметры:

oTest : Объект Basic UNO, который нужно проверить.

UNO-Interface-Name: список имен интерфейсов UNO.

Пример:

Лист. 452: Функция HasUnoInterfaces

```
Sub CloseOpenDocument
  If HasUnoInterfaces(oDoc, "com.sun.star.util.XCloseable") Then
    oDoc.close(true)
  Else
    oDoc.dispose
  End If
End Sub
```

XVII.88. Функция Hex

Обзор:

Возвращает строку, которая представляет собой шестнадцатеричное значение числа. Если параметр не является числом, он преобразуется в число, если это возможно.

Синтаксис:

```
Hex(Number)
```

Возвращаемое значение:

Строка (String)

Параметры:

Number: Числовое выражение, которое нужно преобразовать в шестнадцатеричное число. Может быть строковым.

Пример:

Лист. 453: Функция Hex

```
Sub ExampleHex
  Dim i1%, i2%, iNum%, s$, sFormat$, sTemp$
  iNum = 0
  s = ""
  For i1=0 To 15
    For i2=0 To 15
      s = s & " " & PrependChar(Hex(iNum), "0", 2)
      iNum = iNum + 1
    Next
    s = s & Chr(13)
  Next
  MsgBox s, 64, "Hex Table"
  Print Hex("64")
End Sub
Function PrependChar(s$, sPrependString$, iTotLen%) As String
  If Len(s) < iTotLen Then
    PrependChar = String(iTotLen – Len(s), sPrependString) & s
  Else
    PrependChar = s
  End If
End Function
```

XVII.89. Функция Hour

Обзор:

Возвращает час из значения типа «время» (time), созданного функциями TimeSerial или TimeValue.

Синтаксис:

```
Hour(Number)
```

Возвращаемое значение:

Целое (Integer)

Параметры:

Number: Числовое выражение, которое содержит значение типа «время» (serial time value).

Пример:

Лист. 454: Функция Hour

```
Sub ExampleHour
  Print "Текущий час равен " & Hour( Now )
  Print Hour(TimeSerial(14,08,12))
  Print Hour(TimeValue("14:08:12"))
End Sub
```

XVII.90. Оператор If

Обзор:

Определяет один или несколько блоков операторов, которые должны выполняться в том случае, когда заданное условие равно True. Хотя можно использовать операторы GoTo или GoSub для передачи управления из блока If , нельзя передать управление внутрь блока If.

Синтаксис:

```
If condition=true Then
   Блок операторов
[ElseIf condition=true Then]
   Блок операторов
[Else]
   Блок операторов
End If
```

Синтаксис:

```
If condition=True Then Оператор
If condition=False Then Оператор
```

Пример:

Лист. 455: Оператор If

```
Sub ExampleIf
  Dim i%
  i% = 4
  If i < 5 Then
    Print "i меньше чем 4"
    If i = 4 Then Print "i равно 4"
    If i < 3 Then
      Print "i меньше чем 3"
    End If
  ElseIf i = 5 Then
    Print "i равен 5"
  Else
    Print "i больше чем 5"
  End If
End Sub
```

XVII.91. Функция IIF

Обзор:

Возвращает один из двух возможных результатов в зависимости от логического значения вычисленного выражения. Хотя мне нравится эта команда, я видел примеры поведения этой функции, которая оставила меня в сомнении относительно правильного выполнения этой функции в версии OOo 1.0.3.1.

Синтаксис:

```
IIf (Expression, ExpressionTrue, ExpressionFalse)
```

Возвращаемое значение:

ExpressionTrue или ExpressionFalse

Параметры:

Expression: Условное выражение, которое нужно вычислить

ExpressionTrue: Возвращается, если Expression равно true.

ExpressionFalse : Возвращается, если Expression равно false.

Пример:

Лист. 456: Функция IIF

```
Sub IIfExample
  Print IIf(3>4,"Yes", "No")  REM No
  Print IIf(4>2,"Yes", "No")  REM Yes
End Sub
```

XVII.92. Оператор Imp

Обзор:

Вычисляет логическую функцию «следствие» (implication) для двух выражений. В науке логике говорят, что *x* влечет (implies) *y, если y* равно true всегда, когда *x* равно true. Если *x* равно false, то значение *y* неважно и выражение считается равным true. Рассмотрим утверждение: если «Вы большой» (*x*), то «Вы не будете ограблены» (*y*). Если Вы большой и Вы ограблены (*x*=True, *y*=False), то все утверждение неверно. Если Вы большой и не ограблены (*x*=True, *y*=True), то все утверждение верно. Еслы Вы не большой (*x*=False), то ничто не сделает все утверждение неверным

x	***y***	***x IMP y***
ИСТИНА	ИСТИНА	ИСТИНА
ИСТИНА	ЛОЖЬ	ЛОЖЬ
ЛОЖЬ	ИСТИНА	ИСТИНА
ЛОЖЬ	ЛОЖЬ	ИСТИНА
1	1	1
1	0	0
0	1	1
0	0	1

Внимание *В версии OOo 1.0.3.1, этот оператор выполняется неверно для случая (выражение 0 Imp 1 дает 0 вместо 1)*

Внимание *Imp, кажется, не работает с константами:*
vC = -1 : vD = 0
Print -1 Imp 0 REM -1 (неверно)
Print vC Imp vDREM 0 (теперь верно)

Совет *Используйте (Not x Or y) вместо (x Imp y). потому что эти выражения логически эквивалентны, а второе всегда возвращает верный результат. Выражение (2 Imp 1) вернет Вам -3 !*

Синтаксис:

```
Result = Expression1 Imp Expression2
```

Параметры:

Expression1, Expression2 : Числовые или булевские выражения.

Пример:

Лист. 457: Оператор Imp

```
Sub ExampleImp
  Dim bFalse As Boolean, bTrue As Boolean
  Dim i1%, i2%
  bFalse = False : bTrue  = True
  Print bTrue  Imp bTrue  REM -1
  Print bTrue  Imp bFalse  REM  0
  Print bFalse Imp bTrue  REM  0
  Print bFalse Imp bFalse  REM -1
  i1 = 1 : i2 = 0
  Print i1 Imp i1  REM -1
  Print i1 Imp i2  REM -2
  Print i2 Imp i1  REM -2
  Print i2 Imp i2  REM -1
  Print 1 Imp 0    REM -1
End Sub
```

XVII.93. Оператор Input

Обзор:

Оператор Input используется для последовательного чтения числовых или строковых записей из открытого файла и присвоения полученных данных одной или нескольким переменным. Возврат каретки (код Asc=13), новая

строка (код Asc=10), и запятая действуют как разделители. Если прочтено число, то пробел также действует как разделитель. Чтение нечислового значения в числовую переменную устанавливает значение этой переменной равное нулю.

Невозможно прочесть запятые или двойные кавычки с использованием оператора Input#. Если без этого не обойтись, то используйте оператор Line Input.

См. также: Open, Line Input#, Close, Eof, Get

Синтаксис:

```
Input #FileNumber var1[, var2[, var3[,...]]]
```

Параметры:

FileNumber: Номер файла, из которого читаются данные.

var: Числовые и (или) строковые переменные, в которые будут помещены прочтенные данные.

XVII.94. Функция InputBox

Обзор:

Просит пользователя ввести данные в окно диалога. Если пользователь выбрал отказ (cancel), то возвращаетя строка нулевой длины. если не указана позиция диалогового окна, то оно размещается в центре экрана.

Синтаксис:

```
InputBox (Msg [, Title[, Default[, x_pos, y_pos As Integer]]]])
```

Возвращаемое значение:

Строка (String)

Параметры:

Msg: Строка сообщения, выводимая в диалоговом окне.

Title: Выводимый в заголовке диалогового окна текст.

Default: Строка, задающая значение ввода по умолчанию и выводимая вначале в окне диалога.

x_pos: Абсолютная позиция по горизонтали в «twips» — целое (integer).

y_pos: Абсолютная позиция по вертикали в «twips» — целое (integer)

Пример:

Лист. 458: Функция InputBox

```
Sub ExampleInputBox
  Dim s$
  s = InputBox ("Prompt:","Title", "default")
  MsgBox ( s , 64, "Confirmation of phrase")
End Sub
```

XVII.95. Функция InStr

Обзор:

Возвращает позицию в строке, где найдено значение, совпадающее с другой строкой. Функция Instr возвращает позицию, начиная с которой идет совпадающая часть. Если строка не найдена, возвращается 0.

> ***Внимание*** *В версии OOo 1.1.2 возвращается значение типа целое (integer), но возвращаемое значение может быть больше, так как строка может быть длины 64K. Если значение слишком велико, то возвращается отрицательное число (65536-позиция).*

Лист. 459: Функция InStr

```
Sub BugInStr
  Dim b$, i&
  b$ = String(40000, "a") & "|"         REM character 40,001 is an "|"
  i = instr(b, "|")                     REM -25535
  MsgBox cstr(i) & " or " & (65536 + i) REM -25535 or 40001
End Sub
```

Синтаксис:

```
InStr([Start As Integer,] Text1 As String, Text2 As String[, Compare])
```

Возвращаемое значение:

Целок (Integer)

Параметры:

Start: Необязательная позиция начала поиска в строке Text1. Если не задано, считается равным1 — первым символом строки Text1.

Text1: Строковое выражение, в котором нужно искать.

Text2: Строковое выражение, которое ищется.

Compare: Если равно 1, то при поиске не различаются большие и малые буквы, значение по умолчанию 0 — сравнение двоичных котов, то есть точное совпадение.

Пример:

Лист. 460: Функция InStr

```
Sub ExampleInStr
  Dim s$
  s = "SbxInteger getTruck(SbxLong)"
  RemoveFromString(s, "Sbx")
  Print s
End Sub

REM Этот макрос удаляет все экземпляры строки bad$ из строки s$
REM Он модифицирует строку s$
Sub RemoveFromString(s$, bad$)
  Dim i%
  i = InStr(s, bad)
  Do While i > 0
    Mid(s, i, Len(bad), "")
    i = InStr(i, s, bad)
  Loop
End Sub
```

Внимание *Нельзя использовать параметр «Compare», если не задан явно параметр «Start»*

XVII.96. Функция Int

Обзор:

Возвращает наибольшее целое число, которое не больше, чем значение параметра. Это означает, что аргумент округляется до ближайшего целого в сторону уменьшения.Отрицательные числа поэтому становятся больше по абсолютной величине, а положительные числа — становятся меньше по абсолютной величине.

См. также: CInt, Fix

Синтаксис:

```
Int (Number)
```

Возвращаемое значение:

Двойное с плавающей точкой (Double) !да, именно так!

Параметры:

Number: Любое числовое выражение

Пример:

Лист. 461: Функция Int

```
Sub ExampleINT
  Print " " & Int(3.14159) & " " & Fix(3.14)      REM 3 3
  Print " " & Int(0) & " " & Fix(0)               REM 0 0
  Print " " & Int(-3.14159) & " " & Fix(-3.1415) REM -4 -3
  Print " " & Int(2.8) & " " & Fix(2.8)           REM 2 2
End Sub
```

> ***Внимание*** *Функция Ing округляет -3.4 до -4. Используйте функцию Fix, если нужно просто отбросить дробную часть, Fix(-3.4)=-3*

XVII.97. Функция IsArray

Обзор:

Проверяет, является ли переменная массивом

Синтаксис:

```
IsArray(Var)
```

Возвращаемое значение:

Булевское (boolean)

Параметры:

Var: Любая переменная, которую нужно проверить, объявлена ли она как массив.

Пример:

Лист. 462: Функция IsArray

```
Sub ExampleIsArray
  Dim sDatf(10) as String, i
  Print IsArray(sDatf())  'True
  Print IsArray(i())    'False
End Sub
```

XVII.98. Функция IsDate

Обзор:

Проверяет, можно ли число или строку преобразовать в дату

Синтаксис:

```
IsDate(Expression)
```

Возвращаемое значение:

Булевское (boolean)

Параметры:

Expression: Любое числовое или строковое выражение для проверки

Пример:

Лист. 463: Функция IsDate

```
Sub ExampleIsDate
  Print IsDate("12.12.1997") 'True
  Print IsDate("12121997")   'False
End Sub
```

XVII.99. Функция IsEmpty

Обзор:

Првоеряет переменную типа Variant, содержит ли она значение Empty, что показывает, что данная переменная ещё не была инициализирована

См. также: «Object, Variant, Empty и Null» на странице 326.

Синтаксис:

```
IsEmpty(Var)
```

Возвращаемое значение:

Булевское (boolean)

Параметры:

Var: Любая переменная, которую нужно проверить

Пример:

Лист. 464: Функция IsEmpty

```
Sub ExampleIsEmpty
  Dim v1 as Variant, v2 As Variant, v3 As Variant
  v2 = Null : v3 = "hello"
  Print IsEmpty(v1)  ' True
  Print IsEmpty(v2)  ' False
  Print IsEmpty(v3)  ' False
  v2 = Empty      '  не работает в версиях OOo 1.0.3.1 и 1.1.1.
  Print IsEmpty(v2)  ' true —  уже работает в версии 2.3
End Sub
```

XVII.100. Функция IsMissing

Обзор:

Проверяет, была ли процедура или функция вызвана с указанием или без указания данного необязательного параметра. Параметр должен быть объявлен с ключевым словом «Optional», чтобы функция могла работать. В версии OOo 1.0.3.1 было несколько незначительных ошибок, как уже упоминалось в разделе по необязательным параметрам на странице 332.

Синтаксис:

```
IsMissing(var)
```

Возвращаемое значение:

Булевское (boolean)

Параметры:

var: Переменная, которую нужно проверить

Пример:

Лист. 465: Функция IsMissing

```
Function FindCreateNumberFormatStyle (sFormat As String, Optional doc,
Optional locale)
  Dim oDoc As Object
  Dim aLocale As New com.sun.star.lang.Locale
  Dim oFormats As Object
  'Если параметр doc не был указан, используется ThisComponent
  oDoc = IIf(IsMissing(doc), ThisComponent, doc)
  oFormats = oDoc.getNumberFormats()
  ....
End Function
```

XVII.101. Функция IsNull

Обзор:

Проверяет, содержит ли переменная типа Variant или Object специальное значение Null, показывающее, что переманная не содержит данных. Не инициализированная переменная типа Object равна NULL,
а не инициализированная переменная типа Variant равна Empty, хотя и ей можно задать значение NULL.

См. также: IsEmpty, а также макрос GetSomeObjInfo

Синтаксис:

```
IsNull(Var)
```

Возвращаемое значение:

Булевское (boolean)

Параметры:

var: Переменная, которую нужно проверить

Пример:

Лист. 466: Функция IsNull

```
Sub ExampleIsNull
  Dim v1 as Variant, v2 As Variant, v3 As Variant, o As Object
  v2 = Null : v3 = "hello"
  Print IsNull(v1)  ' False
  Print IsNull(v2)  ' True
  Print IsNull(v3)  ' False
  v3 = Null
  Print IsNull(v3)  ' True
  Print IsNull(o)   ' True
End Sub
```

XVII.102. Функция IsNumeric

Обзор:

Проверяет выражение и возвращает True, если оно числовое или может быть преобразовано в числовое.

Синтаксис:

```
IsNumeric(Var)
```

Возвращаемое значение:

Булевское (boolean)

Параметры:

Var: Любое выражение для проверки

Пример:

Лист. 467: Функция IsNumeric

```
Sub ExampleIsNumeric
  Dim v1, v2, v3
  v1 = "abc" : v2 = "123" : v3 = 4
  Print IsNumeric(v1)    ' False
  Print IsNumeric(v2)    ' True
  Print IsNumeric(v3)    ' True
  Print IsNumeric("123x")' False
End Sub
```

XVII.103. Функция IsObject

Обзор:

Согласно документации в Интернете, эта функция проверяет, является ли заданная переменная типа Object объектом OLE. Я проверил приведенный там макрос и выполнил тест, причем он вернул true и для обычного объекта. Функция IsObject возвращает true и в том случае, когда переменная типа variant содержит объект

См. также: макрос GetSomeObjInfo

Синтаксис:

```
IsObject(ObjectVar)
```

Возвращаемое значение:

Булевское (boolean)

Параметры:

ObjectVar: Any variable to be tested.

Пример:

Лист. 468: Функция IsObject

```
Sub ExampleIsObject
  Dim o As Object, s AS String
  Print IsObject(o)  ' True
```

```
  Print IsObject(s)  ' False
End Sub
```

XVII.104. Функция IsUnoStruct

Обзор:

Возвращает true, если указанный объект является структурой UNO.

См. также: макрос GetSomeObjInfo

Синтаксис:

```
IsUnoStruct(var)
```

Возвращаемое значение:

Булевская (boolean)

Параметры:

var: переменная типа object для проверки

Пример:

Лист. 469: Функция IsUnoStruct

```
Sub ExampleIsUnoStruct
  Dim o As Object, s AS String
  Dim aProperty As New com.sun.star.beans.Property
  Print IsUnoStruct(o)                    ' False
  Print IsUnoStruct("com.sun.star.beans.Property")  ' False — параметр
является строкой
  Print IsUnoStruct(aProperty)              ' True
End Sub
```

XVII.105. Функция Kill

Обзор:

Удаляет файл с диска. Допустима любая форма записи имени файла, но символы-шаблоны (* ?) использовать нельзя.

Синтаксис:

```
Kill(file_name)
```

Возвращаемое значение:

Не возвращает ничего

Параметры:

file_name : Имя файла, который нужно удалить

Пример:

Лист. 470: Функция Kill

```
Sub ExampleKill
  Kill "C:\datafile.dat"
End Sub
```

XVII.106. Функция LBound

Обзор:

Возвращает нижнюю границу индексов массива. Индексы массива не обязаны начинаться с 0.

Синтаксис:

```
LBound(ArrayName [, Dimension])
```

Возвращаемое значение:

Целок (Integer)

Параметры:

ArrayName: Имя массива, для которого возвращается нижняя граница индексов.

[Dimension] : необязательный — целое (Integer), которое указывает измерение, для которого нужна граница индексов. Если не задано, то используется первое измерение. Имеет смысл для многомерных массивов.

Пример:

Лист. 471: Функция LBound

```
Sub ExampleUboundLbound
  Dim a1(10 to 20) As String, a2 (10 to 20,5 To 70) As String
  Print "(" & LBound(a1()) & ", " & UBound(a1()) & ")"      ' (10, 20)
  Print "(" & LBound(a2()) & ", " & UBound(a2()) & ")"      ' (10, 20)
  Print "(" & LBound(a2(),1) & ", " & UBound(a2(),1) & ")"  ' (10, 20)
  Print "(" & LBound(a2(),2) & ", " & UBound(a2(),2) & ")"  ' (5, 70)
End Sub
```

XVII.107. Функция LCase

Обзор:

Возвращает строку, в которой все буквы заменены на маленькие (нижний регистр). Исходная строка не меняется.

Синтаксис:

```
LCase (String)
```

Возвращаемое значение:

Строка (String)

Параметры:

String: Строка, которая должна быть преобразована, все большие буквы в малые (нижний регистр)

Пример:

Лист. 472: Функция LCase

```
Sub ExampleLCase
  Dim s$
  s = "Las Vegas"
  Print LCase(s) REM Returns "las vegas"
  Print UCase(s) REM Returns "LAS VEGAS"
end Sub
```

XVII.108. Функция Left

Обзор:

Возвращает левые n символов указанной строки

> ***Внимание*** *В версии OOo 1.1RC2 второй параметр функции Left имеет тип целое (integer), хотя строка может иметь длину 64K.*

Синтаксис:

```
Left(String, Integer)
```

Возвращаемое значение:

Строка (String)

Параметры:

String: Любое строковое выражение

Integer: Число символов, которое нужно возвратить. Если указан 0, то возвращается строка нулевой длины

Пример:

Лист. 473: Функция Left

```
Print Left("123456789", 2)  'печатается 12
```

XVII.109. Функция Len

Обзор:

Возвращает число символов в строке, равное числу байтов, необходимое для хранения переменной.

Синтаксис:

```
Len(Text As String)
```

Возвращаемое значение:

Длинное целое (Long)

Параметры:

Text: Любое строковое выражение или переменная другого типа.

Пример:

Лист. 474: Функция Len

```
Sub ExampleLen
  Dim s$, i%
  s = "123456"
  i = 7
  Print Len(s)    '6
  Print Len(i)    '1
  Print Len(1134) '4
  Print Len(1.0/3)'17
End Sub
```

XVII.110. Ключевое слово Let

Обзор:

Необязательное ключевое слово, указывающее, что значение выражения присваивается переменной. Используется редко

Синтаксис:

```
[Let] VarName=Expression
```

Возвращаемое значение:

Не возвращается ничего

Параметры:

VarName: Переменная, которой присваивается значение

Пример:

Лист. 475: Ключевое слово Let

```
Sub ExampleLet
  Dim s$
  Let s = "Las Vegas"
End Sub
```

XVII.111. Оператор Line Input

Обзор:

Читает строки из файла последовательного типа в переменную. Предварительно нужно открыть файл оператором Open. Файл читается построчно, концом строки является или символ возврата каретки (код Asc=13) или новой строки (код Asc=10). Эти символы конца строки не заносятся в итоговую строку.

Синтаксис:

```
Line Input #FileNumber As Integer, Var As String
```

Возвращаемое значение:

Ничего не возвращается

Параметры:

FileNumber: Номер открытого файла, из которого читаются данные.

var: Имя переменной, куда сохраняются данные.

XVII.112. Функция Loc

Обзор:

Функция Loc возвращает текущую позицию в открытом файле. Если функция Loc используется для открытого файла с произвольным доступом, то

она возвращает номер последней прочтенной или записанной записи. Для файла последовательного типа функция Loc возвращает позицию в файле, деленную на 128. Для двоичных (binary) файлов возвращается позиция последнего прочитанного или записанного байта.

Синтаксис:

```
Loc(FileNumber)
```

Возвращаемое значение:

Длинное целое (Long)

Параметры:

FileNumber: Числовое выражение, содержащее номер файла используемого открытого файла.

XVII.113. Функция Lof

Обзор:

Функция Lof возвращает размер открытого файла в байтах. Чтобы получить размер файла, который ещё не открыт, используйте функцию FileLen

Синтаксис:

```
Lof(FileNumber)
```

Возвращаемое значение:

Длинное целое (Long)

Параметры:

FileNumber: Числовое выражение, содержащее номер файла для уже открытого файла

Пример[1]:

Лист. 476: Функция Lof

```
Sub ExampleRandomAccess
  Dim iNumber As Integer
  Dim sText As Variant REM must be a Variant
  Dim aFile As String
  aFile = "c:\data.txt"
  iNumber = Freefile
  Open aFile For Random As #iNumber Len=32
  Seek #iNumber,1 REM Позиция на начало файла
```

[1] Прим. пер.: этот макрос у меня не сработал.

```
  Put #iNumber,, "Это первая строка текста в файле" REM Заполняем текстом
  Put #iNumber,, "Это вторая строка текста в файле"
  Put #iNumber,, "Это треться строка текста в файле"
  Seek #iNumber,2
  Get #iNumber,,sText ' (Прим. пер.): здесь у меня ошибка выполнения 57
  Print sText
  Close #iNumber
  iNumber = Freefile
  Open aFile For Random As #iNumber Len=32
  Get #iNumber,2,sText
  Put #iNumber,,"А это новая строка в файле"
  Get #iNumber,1,sText
  Get #iNumber,2,sText
  Put #iNumber,20,"А вот это текст записи номер 20"
  Print Lof(#iNumber)
  Close #iNumber
End Sub
```

XVII.114. Функция Log

Обзор:

Возвращает натуральный логарифм числа. Натуральный логарифм — это логарифм по основанию e = 2.718282... Если нужно вычислить логарифм по другому основанию (n) от какого-то числа (x), достаточно поделить натуральные логарифмы: Log_n(x) = Log(x) / Log(n)

Синтаксис:

```
Log(Number)
```

Возвращаемое значение:

Двойное с плавающей точкой (Double)

Параметры:

Number: Числовое выражение, для которого нужно вычислить натуральный логарифм.

Пример:

Лист. 477: Функция Log

```
Sub ExampleLogExp
  Dim a as Double
  Dim const b1=12.345e12
  Dim const b2=1.345e34
  a=Exp( Log(b1)+Log(b2) )
  MsgBox "" & a & chr(13) & (b1*b2) ,0,"Multiplication by logarithm
function"
End Sub
```

XVII.115. Оператор Loop

Обзор:

Оператор Loop используется для повторения операторов до тех пор, пока условие равно true, или до тех пор, пока условие не станет равным true. См. рассмотрение циклов do loops на стр. 337.

Синтаксис 1:

```
Do [{While | Until} condition = True]
блок операторов
[Exit Do]
блок операторов
Loop
```

Синтаксис 2:

```
Do
блок операторов
[Exit Do]
блок операторов
Loop [{While | Until} condition = True]
```

Пример:

Лист. 478: Оператор Loop

```
Sub ExampleDoLoop
  Dim sFile As String, sPath As String
  sPath = "c:\" : sFile = Dir$( sPath ,22)
  If sFile <> "" Then
  Do
    MsgBox sFile
    sFile = Dir$
    Loop Until sFile = ""
  End If
End Sub
```

XVII.116. Оператор LSet

Обзор:

Оператор Lset позволяет выровнять по левому краю строку внутри другой строковой переменной, уже имеющей нужную длину. Длина строки-результата не изменяется, недостающие позиции справа заполняются пробелами, лишние символы справа отбрасываются. Это не работало до версии 1.1.x.

Оператор Lset также позволяет копировать переменную определенного пользователем типа в другую переменную такого же типа. При этом все байты одной переменной копируются в другую переменную без учета внутренней структуры переменных. Я не проверял этот оператор для определенного пользователем типа данных в версии 1.1.1

Синтаксис:

```
LSet Var As String = Text
LSet Var1 = Var2
```

Параметры:

Var: Любая строковая переменная, в которую заданная строка помещается с выравниванием по левому краю

Text: Строка, которую нужно выровнять по левому краю в строковую переменную

Var1: Имя переменной типа, определенного пользователем, куда копируются данные

Var2: Имя переменной типа, определенного пользователем, откуда копируются данные

Пример:

Лист. 479: Оператор LSet

```
Sub ExampleLSet
  Dim sVar As String, sExpr As String
  sVar = String(40,"*")
  sExpr = "SBX"
  REM Left-align "SBX" within the 40-character reference string
  LSet sVar = sExpr
  Print ">"; sVar; "<"    REM ">SBX                                    <"
  sVar = String(5,"*")
  sExpr = "123456789"
  LSet sVar = sExpr
  Print ">"; sVar; "<"    REM ">12345<"
End Sub
```

XVII.117. Функция LTrim

Обзор:

Удаляет начальные пробелы в строковом выражении

Синтаксис:

```
LTrim(Text)
```

Возвращаемое значение:

Строковое (String)

Параметры:

Text: Любое строковое выражение

Пример:

Лист. 480: Функция LTrim

```
Sub ExampleSpaces
  Dim sText2 As String,sText As String,sOut As String
  sText2 = " <*Las Vegas*> "
  sOut = "'"+sText2 +"'"+ Chr(13)
  sText = Ltrim(sText2)     REM sText = <*Las Vegas*> "
  sOut = sOut + "'"+sText +"'" + Chr(13)
  sText = Rtrim(sText2)     REM sText = " <*Las Vegas*> "
  sOut = sOut +"'"+ sText +"'" + Chr(13)
  sText = Trim(sText2)     REM sText = " <*Las Vegas*> "
  sOut = sOut +"'"+ sText +"'"
  MsgBox sOut
End Sub
```

XVII.118. Ключевое слово Private

Обзор:

Ключевое слово Private используется для объявления переменной вне процедуры, при этом переменная может использоваться только в том же модуле всеми процедурамм и функциями. Если переменная объявлена с использованием ключевого слова Dim, то она считается private. Синтаксис аналогичен ключевому слову Dim.

См. также: Dim, Public

Синтаксис:

```
Private Name_1 [(start To end)] [As VarType][, Name_2 [(start To end)] [As
VarType][,...]]
```

Пример:

Лист. 481: Ключевое слово Private

```
Private iPriv As Integer
Sub ExamplePublic
  iPriv = 1
  Call CalledSub
End Sub
Sub CalledSub
```

```
  Print iPriv  REM 1
End Sub
```

XVII.119. Ключевое слово Public

Обзор:

Ключевое слово Public используется для объявления переменной вне процедуры как глобальной переменной для всех модулей. Если переменная объявлена с использованием ключевого слова Dim, то она считается private, а не Public. Синтаксис аналогичен синтаксису Dim

См. также: Dim, Private

Синтаксис:

```
Public Name_1 [(start To end)] [As VarType][, Name_2 [(start To end)] [As
VarType][,...]]
```

Пример:

Лист. 482: Ключевое слово Public

```
Public iPub As Integer
Sub ExamplePublic
  iPub = 1
  Call CalledSub
End Sub
Sub CalledSub
  Print iPub  REM 1
End Sub
```

XVII.120. Функция Red

Обзор:

Цвета представлены длинными целыми (long) Функция возвращает компонент «красный» (red) для заданного кода цвета color. См. также RGB, Blue, и Green.

Синтаксис:

```
Red(Color As Long)
```

Возвращаемое значение:

Целое (Integer) в интервале от 0 до 255.

Параметры:

Color value: Длинное целое (Long) выражение, представляющее цвет кода color.

Пример:

Лист. 483: Функция Red

```
Sub ExampleColor
  Dim lColor As Long
  lColor = RGB(255,10,128)
  MsgBox "The color " & lColor & " consists of:" & Chr(13) &_
    "Red = " & Red(lColor) & Chr(13)&_
    "Green= " & Green(lColor) & Chr(13)&_
    "Blue= " & Blue(lColor) & Chr(13) , 64,"Colors"
End Sub
```

XVII.121. Оператор RSet

Обзор:

Оператор Rset позволяет выровнять по правому краю строку в переменную, уже имеющую определенную длину. Недостающие позиции заполняются пробелами слева, лишние символы слева отбрасываются. Этот оператор не работал в версии OOo 1.0.3.1, исправлен в версии OOo 1.1.1.

Синтаксис:

```
RSet Var As String = Text
```

Параметры:

Var: Любая строковая переменная (String), в которую помещается выровненная по правому краю строка. Длина переменной Var не изменяется.

Text: Строка, которую нужно выровнять по правому краю строковой переменной.

Пример:

Лист. 484: Оператор RSet

```
Sub ExampleRSet
  Dim sVar As String, sExpr As String
  sVar = String(40,"*")
  sExpr = "SBX"
  RSet sVar = sExpr
  Print ">"; sVar; "<"    REM  ">                                        SBX<"
  sVar = String(5,"*")
  sExpr = "123457896"
  RSet sVar = sExpr
```

```
  Print ">"; sVar; "<"    REM ">12345<"
End Sub
```

XVII.122. Функция Shell

Обзор:

Запускает внешнее приложение (программу). Необязательный параметр — стиль окна для запущенного приложения — состоит из следующих числовых кодов:

Стиль окна	*Описание*
0	Активно скрытое окно программы
1	Активно окно программы стандартного размера
2	Активно свернутое окно программы
3	Активно развернутое на весь экран (maximized) окно программы
4	Окно программы стандартного размера, не становится активным
6	Окно программы свернутое, не становится активным, то окно, которое было до этого активным, остается активным
10	Вывод на весь экран без рамки окна (Full-Screen)

Программа запускается в продолжает выполняться в фоновом режиме (in the background), кроме случая, когда последний параметр (bsync) задан равным True. Это означает, что управление немедленно возвращается из команды Shell. Если bSync задан равным True, то выполнение макроса и все задачи OpenOffice приостанавливаются до завершения запущенного приложения.

Тип возвращаемого значения не описан. Мои эксперименты показали, что возвращается значение типа длинное целое (LONG), причем у меня оно всегда равнялось нулю. Если программа не существует, то возникает ошибка и макрос останавливается.

Синтаксис:

```
Shell (Pathname As String[, Windowstyle As Integer][, Param As String][, bSync])
```

Возвращаемое значение:

Ничего не возвращается

Параметры:

Pathname: Полный путь и имя программы, которую нужно запустить

Windowstyle: Указывает стиль окна, в котором выполняется запущенное приложение

Param: Любое строковое выражение, которое задает командную строку, которая должна быть передана запущенному приложению

bSync: Если равно False (или опущен), то управление немедленно возвращается в макрос. Если равно True, то оператор Shell не возвращает управление в макрос до тех пор, пока приложение не завершится.

Пример:

Лист. 485: Функция Shell

```
Sub ExampleShell
  Dim vRC As Variant
  REM Тип окна 2 выводит на экран окно обычного размера на передний план
  REM
  vRC = Shell("C:\andy\TSEProWin\g32.exe", 2, "c:\Macro.txt")
  Print "Управление только что возвращено в макрос и код возврата равен  " & vRC
  REM Два случая, когда в именах есть пробелы, пробел заменяется кодом %20
  Shell("file:///C|/Andy/My%20Documents/oo/tmp/h.bat",2)
  Shell("C:\Andy\My%20Documents\oo\tmp\h.bat",2)
End Sub
```

Автор Antal Attila [atech@nolimits.ro] представил следующий пример использования аргумента bsync.

Лист. 486: Аргумент bsync

```
Sub Main()
  ' Сначала нужно создать файл на диске со следующим содержимым:
  ' Под  Windows (с именем C:\tmp\Test.bat):
  '       echo %1
  '       pause
  ' Под Linux (с именем /home/guest/Test.sh):
  '       echo $1
  '       sleep 100000
  ' Пример аргумента с ожиданием звершения Sync
  ' вызываем мой метод запуска приложения оператором shell с аргументом
  ' bSync=TRUE
  ' Выполнение макроса basic будет остановлено до тех пор, пока терминал
  '(Linux)
  ' или командная строка (Windows) будет завершен(а) (любая клавиша или
  'Ctrl+C)
  ' Под Windows
  shellRunner("file://C:/tmp/", "Test", "Helo World", TRUE)
  ' или Под Linux
  shellRunner("file:///home/guest/", "Test", "Helo World", TRUE)
  ' сигнализируем о завершении выполнения
```

```
  Print "Конец"

  ' Пример аргумента без ожидания завершения Async
  ' Вызываем мой метод запуска приложения оперетором shell с аргументом
  ' bSync=FALSE
  ' Выполнение макроса basic будет при этом продолжаться
  ' Под Windows
  shellRunner("file://C:/tmp/", "Test", "Helo World", FALSE)
  ' или Под Linux
  shellRunner("file:///home/guest/", "Test", "Helo World", FALSE)
  ' сигнализируем о завершении выполнения
  Print "Конец"
End Sub

Sub shellRunner(dirPath$, script$, prms$, sync as Boolean)
  Dim filePath$, ef$, ed$, isWindows as Boolean

  ' loking for OS type
  If instr(mid(dirPath,8),":/")>0 or instr(dirPath,8),"\")>0 Then
    isWindows=TRUE
  Else
    isWindows=FALSE
  End If

  ' Преобразование адреса  url в путь к файлу
  filePath = convertFromURL(dirPath)

  ' Создание командной строки
  If isWindows Then
      ef = "command.com /C "+filePath+script+".bat"
  Else
      ef = "xvt -e sh "+filePath+script+".sh"
  End If

  ' Выполнение команды оболочки операционной системы shell
  Shell(ef, 1, prms, sync)
End Sub
```

XVII.123. Функция UBound

Обзор:

Возвращет верхнюю границу индексов массива

Синтаксис:

```
UBound(ArrayName [, Dimension])
```

Возвращаемое значение:

Целое (Integer)

Параметры:

ArrayName: Имя массива, для которого нужно вернуть верхнюю границу индексов

[Dimension] : Целое (Integer), которое задает номер измерения, для которого возвращается верхняя граница индексов массива. Если не указано, то используется первое измерение. Имеет смысл задавать для многомерных массивов.

Пример:

Лист. 487: Функция UBound

```
Sub ExampleUboundLbound
  Dim a1(10 to 20) As String, a2 (10 to 20,5 To 70) As String
  Print "(" & LBound(a1()) & ", " & UBound(a1()) & ")"    ' (10, 20)
  Print "(" & LBound(a2()) & ", " & UBound(a2()) & ")"    ' (10, 20)
  Print "(" & LBound(a2(),1) & ", " & UBound(a2(),1) & ")"  ' (10, 20)
  Print "(" & LBound(a2(),2) & ", " & UBound(a2(),2) & ")"  ' (5, 70)
End Sub
```

XVII.124. Функция UCase

Обзор:

Возвращает строку, в которой маленькие буквы заменены на большие. Исходная строка не изменяется

Синтаксис:

```
UCase (String)
```

Возвращаемое значение:

Строковое (String)

Параметры:

String: строковое выражение, в котором маленькие буквы нужно заменить на большие

Пример:

Лист. 488: Функция UCase

```
Sub ExampleUCase
  Dim s$
  s = "Las Vegas"
  Print LCase(s) REM Returns "las vegas"
  Print UCase(s) REM Returns "LAS VEGAS"
end Sub
```

XVII.125. Имена файлов и адреса URL

XVII.125.1. Адреса URL

На компьютере под windows «c:\autoexec.bat» являетвся типичным способом ссылки на файл. Эта ссылка может иметь вид адерса URL «file:///c|/autoexec.bat». Общий принцип такого преобразования адресов состоит в том, чтобы начать адрес URL с комбинации «file:///», затем заменить «:» на «|» и заменять «\» на «/». Если нужно вставить имя компьютера или IP-адрес, вставляйте их между вторым и третьим слэшами «/» например, так «file://localhost/c|/autoexec.bat/».

XVII.125.2. Пути с пробелами и другими специальными символами

Пробелы и специальные символы можно вставлять в адрес URL в виде «escape — последовательностей». Берется код ASCII символов, которые нужно вставить, преобразованный в шестнадцатеричное представление, перед которым ставится символ процента «%», и все это помещается туда, куда нужно вставить специальные символы. Примером может быть вставка пробелов в путь к файлу: «c:\My%20Documents\info.sxw» и «file///c|/My %20Documents/info.sxw».

XVII.125.3. Привязка к домашней папке (Home Directory) под операционными системами типа Unix

Благодарности автору Andrew McMillan [andrew@catalyst.net.nz], который упомянул, что под Unix/Linux можно привязать путь к файлу к домашней папке следующим образом:

```
file://~/document.sxw — document.sxw в домашней папке.
file://~person/document.sxw — document.sxw в домашней папке пользователя
«person».
```

Глава XVIII. Другие языки

XVIII.1. C#

В языке C# нужно использовать COM-объекты для доступа к объектам OOo. Все возвращается как объект, включая целые числа и тому подобное. Следующий простой пример на C# должен продемонстрировать, как вызвать набор различных методов с различными возвращаемыми значениями. Это не является полным примером. Если нужны ещё и другие примеры, сообщите мне.

Лист. 489: Простой пример на C# для доступа к книжным закладкам (bookmarks).

```
using System;
using System.Reflection;

namespace ootest
{
  class Class1
  {
    // obj — Привлекаем методы этого объекта
    // method — имя метода, который будет использоваться
    // par — Массив аргументов типа Object
    static object Invoke(object obj,string method, params object[] par)
    {
      return
obj.GetType().InvokeMember(method,BindingFlags.InvokeMethod,null,obj,par);
    }

    static void PrintImpName(object obj)
    {
      object x = Invoke(obj, "getImplementationName", new object[0]);
      System.Console.WriteLine(x.ToString());
    }

    /// <summary>
    /// Главная точка входа в приложение
    /// </summary>
    [STAThread]
    public static void Main(string[] args)
    {
      // Получаем копию менеджера сервисов
object
usm=Activator.CreateInstance(Type.GetTypeFromProgID("com.sun.star.ServiceMan
ager"));

      // создаем копию рабочего стола. Помните, что рабочий стол только
один.
      object desk=Invoke(usm,"createInstance","com.sun.star.frame.Desktop");
      PrintImpName(desk);

      // Типичные пример загружает новый документ OOo Calc
      //object
calcDoc=Invoke(desk,"loadComponentFromURL","private:factory/scalc","_blank",
0,new object[0]);
```

```
      // Что насчет получения текущего компонента? Этот пример не получает
аргументов

      object oDoc = Invoke(desk,"getCurrentComponent", new object[0]);
      PrintImpName(oDoc);
      //System.Threading.Thread.Sleep(1000);

      object x = Invoke(oDoc, "supportsService",
"com.sun.star.text.TextDocument");
      if (x is bool && (bool) x)
      {
        System.Console.WriteLine("Текущий компонент является текстовым
документом");
      }
      else
      {
        System.Console.WriteLine("Текущий компонент — НЕ текстовый
документ");
        System.Threading.Thread.Sleep(10000);
        return;
      }
      object oBookmarks = Invoke(oDoc, "getBookmarks", new object[0]);
      x = Invoke(oBookmarks, "getCount", new object[0]);
      int nCount = (int) x;
      for (int n = 0; n < nCount; ++n)
      {
        object oMark = Invoke(oBookmarks, "getByIndex", n);
        x = Invoke(oMark, "getName", new object[0]);
        System.Console.WriteLine((string) x);
      }

      System.Threading.Thread.Sleep(5000);
    }
  }
}
```

XVIII.2. Программисты на Visual Basic

Моя книга содержит многочисленные примечания по различиям между Visual Basic и StarBasic; я не буду повторять эту информацию здесь — если захотите её прочесть, купите мою книгу.

Версия enterprise продукта StarOffice (коммерческой версии OOo, см. `http://www.staroffice.com`) способна выполнять макросы office в StarOffice.

Фирма Novell разрабатывает бесплатный (free) способ выполнения макросов Excel прямо в OOo. Часть этих возможностей уже доступна в «OpenOffice Novell Edition», который является частью дистрибутива SUSE Linux.

`https://reverendted.wordpress.com/2006/07/03/openofficeorg-and-excel-vba-macros/`

Имеется много доступных ссылок на другие источники

`http://www.oooforum.org/forum/viewforum.phtml?f=9`

`http://www.oooforum.org/forum/viewtopic.phtml?t=8833&sid=0d667c7f6452b204aef49b352cee2007`

XVIII.2.1. Активная рабочая книге ActiveWorkbook

Чтобы получить «active workbook» из OOo Basic, используйте переменную ThisComponent. Если Вы не используете OOo Basic, то необходимо получить текущую компоненту из объекта рабочего стола (desktop object). К сожалению, Basic IDE (средство отладки для языка Basic) и Справка по OOo также являются компонентами рабочего стола, поэтому необходимо проверить, поддерживиет ли этот компонент интерфейс Xmodel. Некоторые примеры в данной статье используют коды, такие как
StarDesktop.getCurrentComponent()
или
StarDesktop.CurrentComponent,
для получения текущего документа. Этот способ программирования часто приводит к ошибке, когда макрос выполняется в среде отладки IDE, потому что IDE сам является текущим компонентом, но последний документ, который был текущим, нужно получать по ссылке ThisComponent.

Документы OpenOffice.org поддерживают модель, которая содержит данные, видимый курсор (view) для вывода этих данных и контроллер (controller), который взаимодействует с пользователем. Именно контроллер «знает», что пользователь делает сейчас, поэтому, вообще говоря, если нужно узнать текущее состояние дел, следует получить текущий контроллер документа и спросить его. Например, именно текущий контроллер знает о текущем выделенном тексте, активном листе (sheet) и активной ячейке (cell).

XVIII.2.2. Активный лист и активная ячейка (ActiveSheet и ActiveCell)

Чтобы найти активный лист (sheet), нужно вызвать getActiveSheet() для объекта, который поддерживает сервис SpreadSheetView. На самом деле достаточно, чтобы объект поддерживал интерфейс XspreadsheetView, но сервис SpreadsheetView поддерживает следующий интерфейс:

`http://api.openoffice.org/docs/common/ref/com/sun/star/sheet/SpreadsheetView.html`

Получите видимый курсор данной электронной таблицы путем вызова ThisComponent.getCurrentController(). Текущий контроллер — это то, что взаимодействует с пользователем и поэтому «знает», что сейчас выделено. Текущий контроллер для документа OOo Writer не будет поддерживать сервис SpreadSheetView.

Пробема с текущей ячейкой (cell) состоит в том, что нельзя получить текущую ячейку, по крайней мере, ту ячейку, которая очевидным образом доступна. Обычно Вы получаете текущее выделение, которое может быть ячейкой интервалом ячеек или множественным несвязанным выделением. Раздел 6.5 описывает выделение текста в документе OOo Calc.

Если выделено несколько областей и нужно узнать, которая ячейка содержит курсор, то необходимо выполнить некоторую хитрость, потому что курсор недоступен напрямую. Можно просто передвинуть ячейку на одну

позицию влево и затем обратно, но это может привести к ошибке и Вы потеряете текущее выделение. Более элегантным решением, которое работает, создано автором Paolo Mantovani и продемонстрировано в методе RetrieveTheActiveCell, который показан в разделе Получить активную ячейку игнорировать остальное на странице 138.

Приложения

Приложение 1. Дополнения переводчика

Пакетное преобразование множества файлов Word в формат OpenOffice см. пример макроса здесь:

http://xmlhack.ru/texts/06/from-microsoft-to-openoffice/from-microsoft-to-openoffice.html

Пример макроса, исправляющего оглавление.

Оглавление сформировано автоматически, для его настройки нужно долго разбираться с настройкой указателя-оглавления (так, убрать лишнюю табуляцию смог только выделением кнопочки Т и затем клавишей Del), изменением стилей абзаца (конкретно — заголовков), но все равно мне пришлось использовать Ctrl+F — больше параметров — выделять по стилям — указать нужный стиль , например, Заголовок1, кнопка Найти все, затем меню Формат — Абзац — вкладка Нумерация — Стиль нумерации, там повторить нумерацию по умолчанию (у меня это оказалось WW8Num4). Без этой перенастройки получалось так, что после Копировать — Вставить во вставленном куске текста нумерация заголовков начиналась заново.

В процессе экспериментов я написал макрос, который просматривает оглавление и меняет в нем позиции табуляции (в каждом параграфе/абзаце оставляется только одна позиция табуляции 14 см). Макрос в OpenOffice 2.3 работает, хотя и не заносится символ заполнения остатка строки. Вот текст макроса для тех, кому это интересно:

Лист. 490: Исправление оглавления

```
sub Main
Dim s As String
Dim oCurs as Variant
Dim sCurStyle As String
document   = ThisComponent.CurrentController.Frame
dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")
oCurs = ThisComponent.Text.CreateTextCursor()
oText = ThisComponent.getText()
oNumText=oText.CreateEnumeration()
oCurs.GoToStart(False)
'msgbox oCurs.dbg_methods              'Методы данного объекта
'msgbox oCurs.dbg_properties           'Свойства данного объекта
'msgbox typename(oCurs.dbg_properties)
str1 = oCurs.dbg_properties
ind1=instr(str1,"paraTabStops")
str2=mid(str1,ind1-20)
msgbox str2
 Do
  If NOT oCurs.gotoEndOfParagraph(True) Then Exit Do
  sCurStyle = oCurs.ParaStyleName
  strCur=oCurs.String
  if (left(sCurStyle,9) = "Contents ") and (sCurStyle <> "Contents Heading")
then
'            неработающий способ:
'   dim args2(0) as new com.sun.star.beans.PropertyValue
'   args2(0).Name = "Tabstops.TabStops"
```

```
'   args2(0).Value =
Array(Array(14000,com.sun.star.style.TabAlign.LEFT,",",".","."))
'   dispatcher.executeDispatch(document, ".uno:Tabstops", "", 0, args2())
'           и вот здесь OOo "неожиданно завершилось"
   sTabs=oCurs.ParaTabStops
   'msgbox "sTabs="+typename(stabs)
   'msgbox " IsArray(sTabs)="+IsArray(sTabs)
   if IsArray(sTabs) then
    'msgbox " IsUNOStruct(sTabs)="+IsUNOStruct(sTabs)
    stabs(0).position=14000
    stabs(0).alignment=com.sun.star.style.TabAlign.LEFT
    stabs(0).fillchar="."
    sTabsF=sTabs(0).FillChar
    ReDIM Preserve stabs(0)
    oCurs.paraTabStops=stabs
    'oCurs.paraTabStops=Array(Array(14000,com.sun.star.style.TabAlign.LEFT,"
,",".","."))
    sTabs=oCurs.paraTabStops
    sTabsF=sTabs(0).FillChar
   endif
  endif
 Loop Until NOT oCurs.gotoNextParagraph(False)
 MsgBox "Все стили Contents NN обработаны"
end sub
```

Приложение 2. Горячие клавиши

Введение

Вы можете использовать OpenOffice.org (OOo) без требуемых координатно-указательных устройств ввода, таких как мышь или трекбол, используя встроенные сокращённые клавиатурные команды. Разнообразные и сложные задачи, такие как пристыковка и отстыковка панелей инструментов и окон, или изменение размеров или положения объектов, все это может быть выполнено только с клавиатуры.

OOo имеет общий набор сокращённых клавиатурных команд, доступных во всех компонентах, и специфический для компонента набор, непосредственно связанных с работой этого компонента (Writer, Calc, Impress, Draw и Base). Здесь описывается общий набор команд. Специфический для компонента набор клавиатурных команд подробно описан в Справке по компоненту.

Для справки о клавиатурных команд OOo, или использовании OOo только с клавиатурой, найдите в справочной системе OOo используя ключевые слова «сочетания клавиш».

Назначение сочетаний клавиш

В дополнение к использованию встроенных сочетаний клавиш (перечислены далее в этом Приложении), вы можете определить свои собственные. Вы можете назначить сочетания клавиш стандартным функциям OOo или макросу и сохранить их для использование со всем набором OpenOffice.org.

> ***Внимание*** *Будьте осторожны, повторно назначая предопределённые сочетания клавиш вашей операционной системы или OOo. Многие назначения клавиш универсально поняты, такие как F1 для Справки, и как ожидают, обеспечивают определённые результаты. Хотя вы можете легко отменить назначения сочетаний клавиш в OOo на значения по умолчанию, изменение некоторых общих сочетаний клавиш может вызвать беспорядок, расстройство и возможную потерю или разрушение данных, особенно если другие пользователи совместно используют ваш компьютер.*

Чтобы адаптировать сочетания клавиш для ваших нужд, используйте диалоговое окно Настройка, как описано ниже.

1. Выберите Сервис → Настройка → Клавиатура. Откроется диалоговое окно Настройка.

2. Чтобы иметь назначение клавиатурной команды во всех компонентах OpenOffice.org установите переключатель OpenOffice.org.

3. Затем выберите необходимую функцию из списков Категория и Функция.

4. Теперь выберите желательное сочетание клавиш в списке Комбинации клавиш и нажмите кнопку Заменить сверху справа.

5. Нажмите OK чтобы принять изменения. Теперь выбранные сочетания клавиш выполнят функцию, выбранную в шаге 3 выше всякий раз, когда они нажимаются.

Примечание *Все существующие сочетания клавиш для выбранной Функции перечислены в списке Клавиши. Если список Клавиши пуст, это указывает на то, что выбранная комбинация клавиш свободна для использования. Если бы это было не так, и вы желаете повторно назначить сочетание клавиш, которое уже используется, вы должны сначала Удалить существующее сочетание клавиш. Сочетания клавиш, которые отображаются серым в списке в диалоговом окне Настройка, такие как F1 и F10, не доступны для переназначения.*

Сохранение изменений в файле

Изменения назначенных сочетаний клавиш (и другой конфигурации) могут быть сохранены в конфигурационном файле для использования в будущем, таким образом разрешая Вам создать и применять различные конфигурации, если возникает потребность. Для сохранения сочетаний клавиш в файл:

1. После создания ваших назначенных сочетаний клавиш, нажмите кнопку Сохранить справа внизу диалогового окна Настройка.

2. В диалоговом окне Сохранить конфигурацию клавиатуры, выберите Все файлы из списка Тип файла.

3. Затем введите имя для файла конфигурации клавиатуры в поле Имя файла или выберите существующий файл из списка. Если необходимо, переместитесь чтобы найти файл из другого места.

4. Нажмите перезаписать существующий файл, в противном случае не будет никакой ответной реакции и файл будет сохранен.

Загрузка сохранённой конфигурации клавиатуры

Для загрузки сохранённого файла конфигурации клавиатуры и замены Вашей существующей конфигурации, нажмите кнопку Загрузить справа внизу диалогового окна Настройка, и затем выберите файл конфигурации в диалоговом окне Загрузить конфигурацию клавиатуры.

Восстановление сочетаний клавиш

Чтобы восстановить все комбинации клавиш в их значения по умолчанию, нажмите кнопку Восстановить справа внизу диалогового окна Настройка. Используйте эту возможность осторожно, так как никакого диалогового окна подтверждения показано не будет; значения по умолчанию будут установлены без какого-либо уведомления или вопроса к пользователю.

Выполнение макроса

Вы можете также определить комбинацию клавиш, которая будет запускать макрос. Подобные комбинации клавиш определяются только пользователем; ни одна из них не встроена.

Общие сочетания клавиш для OpenOffice.org

Таблица 19. Общие сочетания клавиш для OpenOffice.org

Сочетание клавиш	*Результат*
[ENTER]	Активизирует фокусированную кнопку в диалоговом окне.
[ESC]	Завершает действие или закрывает диалоговое окно. Если в Справке OpenOffice.org: переход на один уровень вверх.
[ПРОБЕЛ]	Переключение активного флажка в диалоговом окне.
[← ↑ → ↓]	Смена активного элемента управления в разделе диалогового окна.
[TAB]	Перемещение курсора на следующий раздел или элемент управления диалогового окна.
[SHIFT] + [TAB]	Перемещение курсора на предыдущий раздел или элемент управления диалогового окна.

Сочетание клавиш	*Результат*
[ALT]+[↓]	Открывает список активного поля со списком в диалоговом окне. Эти сочетания клавиш применимы не только к полям со списком но и к кнопкам значков с всплывающими меню. Открытый список можно закрыть, нажав клавишу [ESC].
[DEL]	Удаление выбранных элементов и их перенос в корзину.
[SHIFT]+[DEL]	Удаление выбранных элементов без их помещения в корзину.
[BACKSPACE]	Для открытой папки: переход на один уровень вверх (назад).
[CTRL]+[SHIFT]+ [ПРОБЕЛ]	Снятие прямого форматирования с выделенного текста или объектов (аналогично **Формат → Форматирование по умолчанию**).
[CTRL] + [TAB]	В позиции в начале заголовка, вставка позиции табуляции.
[ENTER] *(если выбран объект OLE)*	Активизация выбранного объекта OLE.
[ENTER] *(если выбран рисованный или текстовый объект)*	Активизация режима ввода текста.
[CTRL]+[O]	Открытие документа.
[CTRL]+[S]	Сохранение текущего документа.
[CTRL]+[N]	Создание нового документа.
[SHIFT]+[CTRL]+[N]	Открытие диалогового окна *Шаблоны и документы*.
[CTRL]+[P]	Печать документа.
[CTRL]+[Q]	Завершение работы приложения.
[CTRL]+[X]	Вырезать выделенные элементы.
[CTRL]+[C]	Копировать выделенные элементы.
[CTRL]+[V]	Вставка из буфера обмена.
[CTRL]+[SHIFT]+[V]	Открытие диалогового окна *Специальная вставка*.

Сочетание клавиш	*Результат*
[CTRL]+[A]	Выделить все.
[CTRL]+[Z]	Отмена последнего действия.
[CTRL]+[Y]	Возвращение последнего отменённого действия.
[CTRL]+[F]	Открытие диалогового окна *Найти и заменить*.
[CTRL]+[SHIFT]+[F]	Поиск согласно последнему введённому условию поиска.
[CTRL]+[SHIFT]+[J]	Осуществляет переключение между полноэкранным и нормальным режимом в Writer или Calc.
[CTRL]+[SHIFT]+[R]	Обновление отображения документа.
[SHIFT]+[CTRL]+[I]	Включает или выключает курсор в текстах, открытых в режиме *только для чтения*.
[CTRL]+[I]	К выделенной области применяется атрибут *Курсив*. Если курсор находится в слове, это слово выделяется курсивом.
[CTRL]+[B]	К выделенной области применяется атрибут *Полужирный*. Если курсор находится в слове, это слово выделяется полужирным шрифтом.
[CTRL]+[U]	К выделенной области применяется атрибут Подчёркивание. Если курсор находится в слове, это слово подчёркивается.

Сочетания клавиш, использующие функциональные клавиши

Таблица 20. Сочетания клавиш, использующие функциональные клавиши

Сочетание клавиш	*Результат*
[F1]	Запуск Справки OpenOffice.org. В Справке OpenOffice.org: переход на главную страницу справки.
[SHIFT]+[F1]	Активизирует Подробные всплывающие подсказки для указателя мыши, который превращается в вопросительный знак. Поместите указатель над элементом (командой, значком или элементом

Сочетание клавиш	Результат
	управления) для отображения подробных всплывающих подсказок.
[SHIFT]+[F2]	Показывает подробную всплывающую подсказку для элемента (команды, значка или элемента управления) выбранной в настоящее время с использованием клавиатуры.
[CTRL]+[F5] *или* [ALT]+[F5]	Закрывает текущий документ (закрывает OpenOffice.org, когда закрывается последний открытый документ).
[F6]	Устанавливает курсор в следующем подокне (например, представление документа/источника данных).
[SHIFT]+[F6]	Устанавливает курсор в предыдущем подокне.
[F10]	Активизирует первое меню (меню **Файл**).
[SHIFT]+[F10]	Открывает контекстное меню.
[CTRL]+[F11]	Открывает каталог стилей.

Сочетания клавиш в Галерее

Таблица 21. Сочетания клавиш в Галерее

Сочетание клавиш	Результат
[TAB]	Перемещение между областями.
[SHIFT]+[TAB]	Перемещение между областями (в обратном порядке).

Сочетания клавиш в области Создание темы в Галерее

Таблица 22. Сочетания клавиш в области Создание темы в Галерее

Сочетание клавиш	Результат
[↑]	Перемещение выделения вверх.
[↓]	Перемещение выделения вниз.
[CTRL]+[ENTER]	Открытие диалогового окна *Свойства*.

Сочетание клавиш	*Результат*
[SHIFT]+[F10]	Открытие контекстного меню.
[CTRL]+[U]	Обновление выбранной темы.
[CTRL]+[R]	Открытие диалогового окна *Ввести заголовок*.
[CTRL]+[D]	Удаление выбранной темы.
[INSERT]	Вставка новой темы.

Сочетания клавиш в области предварительного просмотра галереи

Таблица 23. Сочетания клавиш в области предварительного просмотра галереи

Сочетание клавиш	*Результат*
[HOME]	Переход к первой записи.
[END]	Переход к последней записи.
[←]	Выделение следующего элемента галереи слева.
[→]	Выделение следующего элемента галереи справа.
[↑]	Выделение следующего элемента галереи сверху.
[↓]	Выделение следующего элемента галереи снизу.
[PAGE UP]	Переход на один экран вверх.
[PAGE DOWN]	Переход на один экран вниз.
[CTRL]+[SHIFT]+ [INSERT]	Вставка выделенного объекта в текущий документ в качестве связанного объекта.
[CTRL]+[I]	Вставка копии выделенного объекта в текущий документ.
[CTRL]+[T]	Открытие диалогового окна Ввести заголовок.
[CTRL]+[P]	Переключение между представлением темы и представлением объекта.
[ПРОБЕЛ]	Переключение между представлением темы и представлением объекта.
[ENTER]	Переключение между представлением темы и представлением объекта.

Сочетание клавиш	Результат
[BACKSPACE]	Возврат к основному представлению (только в представлении объекта).

Выделение строк и столбцов в таблице

Таблица 24. Выделение строк и столбцов в таблице

Сочетание клавиш	Результат
[ПРОБЕЛ]	Переключение выделения строки, за исключением строк в режиме редактирования.
[CTRL]+[ПРОБЕЛ]	Переключение выделения строки.
[SHIFT]+[ПРОБЕЛ]	Выделение текущего столбца.

Сочетания клавиш для рисованных объектов

Таблица 25. Сочетания клавиш для рисованных объектов

Сочетание клавиш	Результат
Выбор панели инструментов с помощью клавиши [F6]. *Выберите значок нужной панели инструментов с помощью* [↓] *и* [→] *и нажмите* [CTRL]+[ENTER]	Вставка рисованного объекта.
Выберите документ с помощью [CTRL]+[F6] *и нажмите* [TAB]	Выбор рисованного объекта.
[TAB]	Выбор следующего рисованного объекта.
[SHIFT]+[TAB]	Выбор предыдущего рисованного объекта.
[CTRL]+[HOME]	Выбор первого рисованного объекта.
[CTRL]+[END]	Выбор последнего рисованного объекта.
[ESC]	Завершение выделения рисованных объектов.

Сочетание клавиш	*Результат*
[ESC] *(в режиме выделения маркеров)*	Выход из режима выделения маркеров и возврат к режиму выделения объектов.
[← ↑ → ↓]	Перемещение выбранной точки (функции привязки к сетке временно недоступны, однако конечные точки связаны друг с другом).
[ALT]+[← ↑ → ↓]	Перемещение выбранного рисованного объекта на один пиксел (в режиме выделения). Изменение размера рисованного объекта (в режиме выделения маркеров). Поворот рисованного объекта (в режиме вращения). Открытие диалогового окна свойств для рисованного объекта. Активизация режима выделения точек для выбранного рисованного объекта.
[ПРОБЕЛ]	Выбор точки рисованного объекта (в режиме выделения точек) или отмена выделения. Выбранная точка будет мигать раз в секунду.
[SHIFT]+[ПРОБЕЛ]	Выбор ещё одной точки в режиме выделения точек.
[CTRL]+[TAB]	Выбор следующей точки рисованного объекта (в режиме выделения точек). В режиме вращения можно выбрать центр вращения.
[SHIFT]+[CTRL]+[TAB]	Выбор предыдущей точки рисованного объекта (в режиме выделения точек).
[CTRL]+[ENTER]	Помещает новый рисованный объект стандартного размера в центр текущего представления.
[CTRL]+[ENTER] на *выделенном значке*	Активизирует первый рисованный объект в документе.
[ESC]	Выход из режима выделения точек. Рисованный объект остаётся выделенным. Изменение точки рисованного объекта (режим редактирования точек).
[Любая текстовая или цифровая клавиша]	Если выбран рисованный объект, переключается в режим редактирования и помещает курсор в конец текста в рисованном объекте. Вставляется печатаемый символ.

Приложение 3. Особенности сборки OpenOffice.org pro

OpenOffice.org pro, созданный компанией «Инфра-Ресурс» на базе OpenOffice.org, в соответствии с лицензией GNU LGPL, доступен для свободной загрузки и использования. Сборка **pro** выполнена для платформ Windows, FreeBSD 6.1 и GNU/Linux в пакетах RPM, DEB и Generic.

Создание **OpenOffice.org pro** оказалось возможным благодаря активной поддержке и конструктивной критике пользователей и, в первую очередь, участников сообщества community.i-rs.ru. Их пристальное внимание и настойчивость помогали лучше расставлять приоритеты при подготовке новой версии.

Отличия сборки OpenOffice.org pro от сборки Community build.

Общие улучшения

- Отменен мастер регистрации при первом запуске (замечены проблемы этого мастера на некоторых платформах);
- Исправлена ошибка сохранения отчества в параметрах пользователя;
- Добавлены шаблоны стандартного документооборота РФ;
- Улучшен импорт старых документов MSO. Кодовая страница импортированного документа устанавливается в локальных настройках OpenOffice.org, что позволяет открывать устаревшие документы как на кириллице, так и на других европейских языках;
- Автозамена подставляет длинное тире (**em dash**) вместо среднего (**en dash**) в русском и украинском языках;
- Увеличена скорость холодного и повторного запусков;
- Ускорена работа с файлами и каталогами за счёт оптимизации системных вызовов;
- Требуется меньше памяти для проверки орфографии;
- Частично устранены проблемы при работе с файлами, размещёнными на NFS;
- Реализована частичная поддержка **VBA**;
- Реализован улучшенный экспорт в HTML;
- Запрещено сохранение документов в старых форматах MSO, что приводило к потере информации. Сохранять документы можно только в форматах, не имеющих проблем с кодовыми страницами (ODF, StarOffice, MSO 97+);
- Добавлен словарь синонимов русского языка;
- Добавлена возможность блокировки настроек панелей инструментов, что должно облегчить работу системным администраторам, обслуживающим множество рабочих мест;
- Реализована ручная проверка выхода обновлений;

- Исправлена ошибка Быстрого запуска в **Windows**;
- Добавлен импорт документов формата **T602** (`http://www.602.cz/cz`), распространённого в Восточной Европе;
- Добавлен импорт графики **WordPerfect**.

Улучшения в программе Writer

- Русская нумерация списков в соответствии с **ГОСТ 7.32-2001**.
- Ускорен импорт документов MS Word (~15% на тестовых файлах);

Улучшения в программе Calc

- Улучшена логика работы кнопки суммирования;
- В автофильтре добавлены значения пусто/не пусто;
- При делении на 0 выводится ассоциативное сообщение об ошибке — `#DIV/0!`;
- Добавлены горячие клавиши `[CTRL]+[+]/[-]` для добавления/удаления строк;
- Добавлена натуральная сортировка;
- Сохраняются между сессиями параметры открытия CSV-файлов;
- Функция **value()** возвращает 0 для пустой ячейки;
- Ускорено открытие сводных таблиц Excel;
- Добавлена возможность изменения диапазона исходных данных сводных таблиц;
- Добавлена поддержка адресации ячеек типа R1C1 (`.uno:SheetUseR1C1`);
- Улучшена палитра по умолчанию для графиков;
- Автоматическое удаление префиксных и постфиксных пробелов в ячейках с цифрами (numeric);
- Автоматическое открытие окна для разбивки по ячейкам при вставке из буфера обмена структурированного текста (CSV);
- Открытие файлов XLS-95, сгенерированных программой 1С, происходит без потери форматирования в объединённых ячейках;
- Запрещено сохранение электронных таблиц в устаревших, не поддерживаемых разработчиками форматах на уровне исходного кода.

Улучшения в программе Impress

- Добавлена команда в контекстное меню «Сохранить изображение»;
- Добавлены дополнительные фоны для презентаций;
- Устранено падение при попытке загрузки шаблонов для фона презентаций.

Улучшения в сборке для Windows

- Создана версия Portable, которая может запускаться без инсталляции с Flash накопителя. Она предназначена для пользователей, работающих с документами в стандарте ODF (ISO/IEC 26300:2006) на рабочих местах, где ещё не установлен OpenOffice.org. OpenOffice.org 2.1 Portable не создаёт записи в реестре Windows и профиле пользователя. Все настройки сохраняются на Flash накопителе.

Улучшения в сборке для GNU/Linux и FreeBSD

- Запрос подтверждения при закрытии сессии при наличии не сохранённых документов на предмет сохранения документов и/или отмены закрытия сессии;
- Добавлена возможность использования системных диалоговых окон открытия файлов из оболочек **GNOME** и **KDE**;
- Добавлен модуль для быстрого старта в **GNOME**;
- Улучшенный рендеринг шрифта Tahoma в пользовательском интерфейсе;
- Добавлен драйвер **MS Access** файлов с ограниченным набором SQL-команд (только для чтения);
- Добавлена поддержка библиотеки мультимедиа GStreamer;
- Добавлена поддержка почтового клиента Seamonkey;
- Улучшенная поддержка печати CUPS;
- Исправлена ошибка Быстрого запуска в **GNOME**.

И ещё около 500 разных исправлений и улучшений.

Таким образом, **OpenOffice.org pro** — результат совместных усилий пользователей и разработчиков, направленных на непрерывное повышение удобства при производстве электронных документов.

Приложение 4. Алфавитный указатель (Index)

A

Abs.....367
And.....331, 349, 366
ApplicationScriptLibraryContainer 99p.
Array.....327pp.
Asc.....369, 378
ATN.....369
AVERAGE.....144

B

Beep.....370
Blue.....370
Boolean.....320
BottomLine.....146
ByVal.....333p., 371

C

Call.....372
case.....343
CBool.....372
Cbyte.....373
CDate.....374
CDateFromIso.....374
CDateToIso.....375
CDbl.....376
Cell.....
  CellAddress.....140
  CellAddressConversion.....138pp.
  CellAddressConversion.....140
  CellBackColor.....134
  getFormula.....134
  getSpreadSheet.....140
  getString.....134
  getValue.....134
  IsCellBackgroundTransparent.....134
  NumberFormat.....134, 144
  setFormula.....134, 144
  setString.....134, 144
  setValue.....134
CellAddress.....
  Column.....140
  Row.....140
CharacterProperties.....
  CharFontName.....203
  CharHeight.....203, 229
  CharLocale.....81, 204
  CharPosture.....203
    FontSlant.....203
  CharUnderline.....203
  CharWeight.....203, 210, 214
    FontWeight.....203, 210, 214
  FontSlant.....
    DONTKNOW.....203
    ITALIC.....203
    NONE.....203
    OBLIQUE.....203
    REVERSE_ITALIC.....203
    REVERSE_OBLIQUE.....203
  FontUnderline.....
    BOLD.....203
    BOLDDASH.....203
    BOLDDASHDOT.....203
    BOLDDASHDOTDOT.....203
    BOLDDOTTED.....203
    BOLDLONGDASH.....203
    BOLDWAVE.....203
    DASH.....203
    DASHDOT.....203
    DASHDOTDOT.....203
    DONTKNOW.....203
    DOTTED.....203
    DOUBLE.....203
    DOUBLEWAVE.....203

LONGDASH....203
NONE....203
SINGLE....203
SMALLWAVE....203
WAVE....203
FontWeight....
BLACK....203
BOLD....203
DONTKNOW....203
LIGHT....203
NORMAL....203
SEMIBOLD....203
SEMILIGHT....203
THIN....203
ULTRABOLD....203
ULTRALIGHT....203
ChDir....376
ChDrive....377
Choose 377
Chr....369, 378
Christian Anderson....333
CInt....379
clipboard....76
CLong....380
Close....41pp., 50, 58, 380, 403, 420
COM....5
Component....
CurrentController....144
CurrentSelection....144
DatabaseRanges....144
getCurrentController....147
getText....14
removeByName....144
ConfigurationAccess....41
ConfigurationProvider....38, 41
ConfigurationUpdateAccess....38
Const....381
Control....
button....278
check box....279
combo box....279
Controls....277
group box....280
label....278
list box....279
progress bar....280
radio button....280
text box....278
Controller....
ActiveSheet....144
getViewCursor()....203p.
select....147
StatusIndicator....35
ConvertFromURL....382
ConvertToURL....382
Cos....383
createInstanceWithArguments....41
createTextCursorByRange....221
CreateUnoDialog....276, 384
CreateUnoService....14, 385
CreateUnoStruct....386
CSng....387
CStr 387
CurDir....388
Currency....320, 325
CurrentController....35
Cursor....
compareRegionEnds....175, 179, 182p., 186pp., 191, 194pp., 212, 217
compareRegionStarts....175p., 189, 191, 198, 209, 217

CreateTextCursor.....50, 68, 70, 81, 151, 177pp., 197, 201, 207p., 218p., 257
createTextCursorByRange.....45, 60, 173pp., 191, 193, 197pp., 203pp., 209, 306
getStart().....203p.
goLeft.....169, 176, 182p., 194, 196, 198
goRight. 81, 169, 175, 179, 182p., 187p., 193pp., 198p., 201p., 212p., 217
gotoEnd.....50, 68, 169, 177p., 197, 208
gotoEndOfParagraph.....203, 207, 218p., 221
gotoEndOfParagraph().....203
gotoNextParagraph.....81, 179, 183, 194, 207, 209, 218p., 221
gotoRange.....179, 213, 230
GoToStart.....81, 169, 177pp., 197, 201, 207p., 218p.
gotoStartOfParagraph().....203
IsCollapsed.....169, 173p., 177p., 197p.

D

Date.....320, 331, 389
DateSerial.....375, 389
DateValue.....374p., 390
Day.....391
DBG_Methods.....20, 27
DBG_Properties.....20, 27
DBG_SupportedInterfaces.....20, 27
Debug.....
PrintdbgInfo.....20
ShowArray.....20
Declare.....392
DefBool.....320, 393
DefDate.....320, 394
DefDbl.....320, 394
DefInt.....320, 395
DefLng.....320, 395
DefObj.....320, 396
DefVar.....320, 396
Desktop.....
CurrentComponent.....144
LoadComponentFromURL().....43
Dialog.....
endExecute.....276
execute.....276
Dim.....319, 321p., 327, 329, 397
DimArray.....329, 367, 398
Dir.....399, 423
dispose. .42, 50, 68, 70, 236, 284, 430
dispose).....70
Do...Loop.....409
Double.....320, 325

E

Else.....335, 432
ElseIf 335, 432
email.....95
email.....
sending email.....94
Empty.....326
End.....402
End Function.....402
End If.....402
End Select.....402
End Sub.....402
Environ.....403
EOF.....380, 403, 420

EqualUnoObjects.....221, 404
EQV.....349, 405
Erl.....345, 406
Err.....345, 407
Error.....345, 407
ExampleShell.....457
Exit.....332, 343
  Exit DO.....343
  Exit For.....343
  Exit Function.....343
  Exit Sub.....343
Exit Do.....338, 409
Exit For.....336p., 409
Exit Function.....341, 409
Exit Sub.....341, 409
Exp.....410

F

False.....332
Fibonacci.....60
FieldMaster.....65pp.
File.....
  CLOSE.....58
  FileExists.....58
  FreeFile.....58
  LINE INPUT.....58
  Loc.....448
  Lof.....449
  OPEN.....58
FileAttr.....410p.
FileCopy.....412
FileDateTime.....412
FileExists.....413
FileLen.....414, 449
Find & Replace.....
  createReplaceDescriptor.....211, 214
  createSearchDescriptor.....210, 212p.
  findFirst.....210pp.
  findNext.....210pp.
  replaceAll.....211, 214
  ReplaceDescriptor.....
    ReplaceString.....211, 214
  SearchDescriptor.....
    searchAll.....214
    SearchCaseSensitive.....210pp.
    SearchRegularExpression.....214
    SearchString.....210pp.
    searchStyles.....214
    SearchWords.....210
  SetReplaceAttributes.....214
  SetSearchAttributes.....214
Fix.....416
For.....336
For...Next.....409
Format.....419
FreeFile.....380, 403, 420
Function.....332
Function 12

G

GCD.....86
Get.....422
GetAttr.....399, 424
getDocumentInfo.....63p.
getImplementationName.....25
getNumberFormats().....59
getPathSeparator().....400

GetProcessServiceManager()…….425
GetSolarVersion.....40
GetSolarVersion().....426
GetSystemTicks().....426
GlobalScope.....427
GoSub.....335, 341, 428, 432
GoTo.....58, 335, 341pp., 429, 432
gotoEndOfParagraph.....217p.
gotoNextParagraph.....217, 221
GraphicExportFilter.....62, 105
GraphicObjectFillBitmap.....52
GraphicObjectShape.....52, 223p.
Green.....429
GUI.....5

H

hasLocation.....42, 45p., 61, 114
HasUnoInterfaces.....430
Header.....
HeaderOn.....150
HeaderShared.....151
Hex.....431
Hour.....432

I

IDE.....5
IDL.....5
If 335, 432p.
IIf.....332, 336, 433
Imp.....349, 435
Input.....435p.
Input#.....436
InputBox.....436
InStr.....437
Int.....58, 438
Integer.....320, 324
Is.....331
IsArray.....18, 330, 439
IsCursorInlastPar.....221
IsDate.....330p., 440
IsEmpty.....18, 326, 330p., 440
IsMissing.....59, 330pp., 441
isModified.....42, 45, 61
IsNull.....18, 326, 330p., 442
IsNumeric.....330p., 442
IsObject.....18, 330, 443
isPlugged.....123
isReadOnly.....42, 45, 61
IsUnoStruct.....18, 330p., 444

K

Kill.....380, 403, 420, 444

L

LBound.....328, 445
LCase.....446
Left.....446
Len.....35, 58, 447
Let.....448
Line Input.....448

LineDistance.....146
loadComponentFromURL.....43
Loc.....449
Locale.....
Country.....204
Language.....204
locale).....81
locale) 81
Lof.....449
Log.....450
Long.....320, 325
Loop.....337
Do.....338
Do Until.....338
Do While.....337
Loop Until.....338
Loop While.....338
LSet.....452
LTrim.....452

M

Macro Author.....
ADPSetWordCase.....216
Andrew Pitonyak.....
AccessModes.....411
ADPWordCountCharCursor.....186
ADPWordCountStrings.....184
ADPWordCountWordCursor.....188
CalcGroupingExample.....149
ClearDefinedRange.....134
CloseOpenDocument.....430
ColumnNumberToString.....141
CreateSelectedTextIterator.....177
DecimalFeetToString.....88
ExampleNewPage.....206
ExampleShell.....57
FileAttributeString.....424
FindCreateNumberFormatStyle.....59
ForNextExampleSort.....337
GetLeftMostCursor.....175
GetRightMostCursor.....176
InsertDateField.....205
InsertDateIntoCell.....141
InsertSimpleText.....204
IsAnythingSelected.....174
IsSpreadsheetDoc.....133
IsWhiteSpace.....180, 422
IterateOverSelectedTextFramework 177
MultipleTextSelectionExample.....174
PrintableAddressOfCell.....140
PrintAscii.....355
PrintDataInColumn.....149
PrintEachCharacterWorker.....179
ProtectSpreadsheet.....150
RankChar.....180
Read_Write_Number_In_File.....57
RemoveEmptyParsWorker.....183
RemoveFromString.....352
ReplaceInString.....353
SearchSelectedText.....212
SelectedNewLinesToSpaces.....192
SelectedNewParagraphsToNewLines.....193
SetDocumentLocale.....81
SetTextAttributes.....203
StringToDecimalFeet.....88
testOptionalParameters.....332
ToFraction.....87
Bernard Marcelly.....
ErrorHandlingExample.....346
Birgit Kellner.....
AtToUnicode.....211
Christian Junker.....38, 76, 257
Database Columns.....
David Kwok.....155
David Woody.....
DrawLineInCalcDocument.....105
InsertAndPositionGraphic.....103

DisplayAllStyles....35
edA-qa mort-ora-y....
Fibonacci....60
GetDocumentType....25
GetGlobalRangeByName....
Rob Gray....152
GetSomeObjInfo....19
Hermann Kienlein....257
Laurent Godard....
OOOVersion....425
SendSimpleMail....95, 385
UnzipAFile....106
ListFonts....37
Load library....
Sunil Menon....100
Marc Messeant....
AppliquerStyle....209
Niklas Nebel....
setting_borders_in_calc....146p.
Oliver Brinzing....
CopySpreadsheetRange....77
Olivier Bietzer....
GCD....86
OOoLang....41
OOoVersion....40
Paolo Mantovani....
RetrieveTheActiveCell....138
Paul Sobolik....37
Peter Biela....31
ProgressBar....35
Ryan Nelson....
CopyPasteRange....76
Sasa Kelecvic....
ActiveSheet....144
Analize....143
CellPos....144
DefineDbRange....145
DeleteDbRange....144, 146
ExampleGetValue....133
ExampleSetValue....134
FillCells....143
GetAddress....144
SelectedCells....142
SortRange....147
SetBitMapSize....
Vance Lankhaar....101
StatusText....35
Stephan Wunderlich....
CopySpreadsheet....151
MAX....144
MIN....144
MOD....349
module....12

N

NOT....331
Null....326, 331
NumberFormat....59
addNew....59
DATE....59
FindCreateNumberFormatStyle.141, 205
queryKey....59

O

Object....320, 326
OLE....5
OleObjectFactory....57, 96
On Error....58
Local....344
On Error....344
On Error GoTo 0....345
On Error GoTo Label....345
On Error Resume Next....345
Resume....345
Resume Label:....346
Resume Next....346
On Local Error Goto 0....58
On N GoSub....343

On N GoTo.....343
OOo.....6
Open.....58, 380, 403, 420
Option Base.....328
Option Explicit.....319
Optional.....59, 331pp.
OR.....331p., 349
outline numbering.....234
Outlook.....96
Outlook.Application.....96

P

PageDescName.....206
PageNumberOffset.....206
ParaStyleName.....218
PI.....327
PRINT.....58
printdbgInfo.....6
Private.....321, 453
PropertyValue.....147
  Name.....147
  Value.....147
Public.....321, 454

R

Red.....454
ReDim.....319pp., 329p., 397
ReDim.....
  Preserve.....329p.
  ReDim.....329p.
  ReDimExample.....329
REM.....319
Return.....341
RSet.....455

S

SDK.....5
Select.....
  Case.....338
  Case Else.....338p.
  Is339
  Select Case.....338
  To.....339
Selection.....
  Columns.....144
  getRangeAddress.....144
  Rows.....144
SetExpression.....71
setModified.....43, 45
Shell.....456
SimpleCommandMail.....95
SimpleSystemMail.....95
Single.....320
Sort.....147
SortField.....147
  Field.....147
  SortAscending.....147
SpreadsheetDocument.....
  BottomLine.....146
  Column.....
    Columns.....142p.
    getByIndex.....142, 144
    getCount.....143p.
    getName.....142, 149
  Columns.....142, 149
  Count.....135

CurrentSelection.....142p.
getByName.....133p., 147
getCellByPosition.....133p., 143
getCellRangeByName.....147
getCellRangeByPosition.....146
getCount.....142
getName.....140
getRangeAddress.....142
Row.....
EndRow.....143p.
getCount.....142pp.
Rows.....143
StartRow.....143p.
Rows.....142
setString.....143
Sheets.....133pp., 146p., 149
SpreadsheetDocument.....133
SupportsService.....133
TableBorder.....146
StarDesktop.....13p.
Static.....321p.
StatusIndicator.....
start.....35
Str.....58
String.....320, 326
StyleFamilies.....150
Styles.....
CharacterStyles.....35
FrameStyles.....35
NumberingStyles.....35
PageStyles.....35
ParagraphStyles.....35
Sub.....12, 332
switch.....338, 343
SystemShellExecute.....113

T

table of contents.....236
TableBorder.....146
Tan.....369
terminate.....42
TextContent.....224
TextCursor.....169
goDown().....169
goLeft().....169
goRight().....169
gotoEnd().....169
gotoStart().....169
goUp().....169
IsCollapsed.....169
IsCollapsed().....173p.
TextCursor.....169
TextDocument.....
compareRegionEnds().....175
compareRegionStarts().....175p.
createReplaceDescriptor.....211
createTextCursorByRange.....204
createTextCursorByRange().173pp., 203
CurrentController.....203p.
findFirst.....210
findNext.....212
getCurrentSelection.....
getCount().....175
getCurrentSelection().....173pp.
getByIndex().....174p.
getCount().....174
insertControlCharacter.....193
insertString.....204
Text.....203
TextField.....
Annotation.....206
DateTime.....205
FieldMaster.....
attachTextFieldMaster.....68

Drawing....66
Illustration....66
Table....66
Text....66
User....65pp.
SheetName....151
URL....70
TextField.SetExpression....71
TextGraphicObject....104, 223p.
TextRange....169
TextSection....23, 237
Thanks....
Alain Viret....26
Andreas Bregas....327, 334
Andrew Brown....184, 188
Andrew McMillan....460
Antoine Jarrige....121
Berend Cornelius....75, 284
Bernard Marcelly....148, 212, 340, 346
Birgit Kellner....211
Chris Clementson....156
Christian Erpelding....35
Christian Junker....42, 69
Dan Juliano....106
Daniel Juliano....96
Daniel Vogelheim....189
David Woody....146
Eric VanBuggenhaut....121
Erik Anderson....113
Frank Schönheit....275
Gerrit Jasper....157p., 161
Giuseppe Castagno....228
Jim Thompson....317
Jürgen Schmidt....99
Laurent Godard....101, 106, 108
Leston Buell....305
Marc Messeant....84, 209
Mathias Bauer....29p., 75, 318
Mikhail Voitenko....47
Niklas Nebel....146
Oliver Brinzing....150, 284
Oliver Brinzing 77
Olivier Bietzer....86
Paolo Mantovani....111, 116, 138, 281, 308, 464
Paul Sobolik....36
Peter Biela....31
Rob Gray....152
Rodrigo V Nunes....66
Russ Phillips....97, 113
Ryan Nelson....76, 149
Sasa Kelecevic. 35, 133p., 142p., 145pp.
Stephan Wunderlich....151
Sunil Menon....100
Sven Jacobi....104
Tony Bloomfield....27
Vance Lankhaar....101p.
Автор Antal Attila....457
ThisComponent....13
TimeValue....374
TOC)....236
Trim....58
True....332
TypeName....18, 322

U

UBound....328, 458
UCase....459
UNO....5
Until....451

V

Val....366
Variant....320, 326, 331, 440
VarType....322

Visual Basic.....5

W

While.....451
While...Wend.....341
WritedbgInfo.....6

X

XOR.....331p., 349
XSelectionSupplier.....76

^

^ 349, 364

-

- 349

*****

* 349, 364

/

/ 349, 365

&

&349

+

+ 349, 364

<

< 331, 349
<=.....331, 349
<>.....331, 349

=

= 331, 349

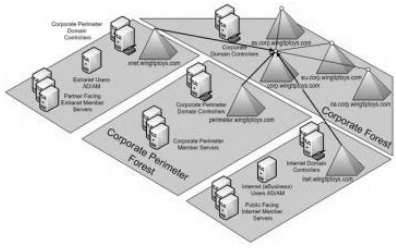
>

> 331, 349
>=.....331, 349

Для заметок:

Для заметок:

OpenOffice-Enterprise



Решение для управления офисным пакетом OpenOffice.org в масштабах предприятия. Данное решение упрощает процесс развертывания и управления OpenOffice.org в среде Microsoft Windows.

- Управление всеми компьютерами с единой консоли
- Контроль настроек с помощью Group Policy, Local Policy, реестра Windows и файлов Windows Installer (.msi).
- Автоматическое применение настроек OpenOffice.org актуальных для предприятия, в том числе, отключения мастера регистрации и автоматических обновлений.
- Доступ к более чем 25 000 настроек OpenOffice.org.
- Блокировка изменений настроек пользователем, установка настроек по умолчанию.
- Различные настройки могут быть применены к разным пользователям или компьютерам.
- Изменения применяются в течении нескольких минут.

По вопросам приобретения обращаться по адресу http://partner.i-rs.ru, mailto: partner@i-rs.ru

ООО «Инфра-Ресурс», г. Москва
http://i-rs.ru, mailto:infra@i-rs.ru

Инфра-Ресурс — Партнёрам

Для повышения качества обслуживания региональных пользователей OpenOffice.org Компания «Инфра-Ресурс» представляет специальный раздел сайта для партнёров.

OpenOffice.org — динамично развивающийся продукт. Обновления выпускаются в среднем один раз в 3-4 месяца. Высокие темпы развития OpenOffice.org нуждаются в оперативной реакции на спрос потребителя и особенно в регионах.

Партнёрский раздел сайта позволит приобретать носители с последней версией OpenOffice.org мелким оптом со скидкой для оперативной перепродажи.

Чтобы стать партнёром компании «Инфра-Ресурс», достаточно принять договор оферты и зарегистрироваться.

Постоянный адрес партнёрского раздела: **http://partner.i-rs.ru**

ООО «Инфра-Ресурс», г. Москва
http://i-rs.ru, mailto:infra@i-rs.ru

Книги издательства «ДМК Пресс» можно заказать в торгово-издательском холдинге «АЛЬЯНС-КНИГА» наложенным платежом, выслав открытку или письмо по почтовому адресу: **123242, Москва, а/я 20** или по электронному адресу: **orders@alians-kniga.ru**.

При оформлении заказа следует указать адрес (полностью), по которому должны быть высланы книги; фамилию, имя и отчество получателя.

Желательно также указать свой телефон и электронный адрес.

Эти книги вы можете заказать и в Internet-магазине: **www.alians-kniga.ru.**

Оптовые закупки: тел. (495) 258-91-94, 258-91-95; электронный адрес **books@alians-kniga.ru.**

OpenOffice.org pro. Автоматизация работы

Руководитель проекта	Н. Е. Гарбуз
Технический консультант	Р. А. Алиев
Литературная обработка текста	А. Е. Харламенков
Дизайн обложки	О. Е. Медведев

Подписано в печать 14.07.2008. Формат 70x100 1/16
Гарнитура «Arial». Печать офсетная.
Усл. печ. л. 48,0. Тираж 1 500 экз.

Издательство ДМК Пресс.
Web-сайт издательства: www.dmk-press.ru
Интернет-магазин: www.alians-kniga.ru
Электронный адрес издательства: books@dmk-press.ru