

ООО «НОВЫЕ ОБЛАЧНЫЕ ТЕХНОЛОГИИ»

Программное обеспечение «МойОфис Стандартный»

Приложение. Руководство оператора. Справочник макрокоманд

СОДЕРЖАНИЕ

1 МАКРОКОМАНДЫ ПО МОЙОФИС.....	1209
2 РАБОТА С МАКРОКОМАНДАМИ	1210
2.1 Редактор макрокоманд	1210
2.1.1 Окно редактора макрокоманд	1210
2.1.2 Создание макрокоманд.....	1211
2.1.3 Выполнение макрокоманд	1211
2.1.4 Редактирование макрокоманд.....	1212
2.1.5 Удаление макрокоманд	1212
2.1.6 Отладка макрокоманд.....	1212
2.2 Пример подготовки и запуска макрокоманды	1215
2.2.1 Редактирование и запуск макрокоманды в текстовом документе	1215
2.2.2 Описание примера работы макрокоманды.....	1216
2.3 Преобразование макрокоманд на языке программирования VBA.....	1217
3 ОБЪЕКТНАЯ МОДЕЛЬ МОЙОФИС	1218
4 СПРАВОЧНИК ФУНКЦИЙ DOCUMENTAPI	1219
4.1 Диаграммы.....	1219
4.1.1 Таблица DocumentAPI.Charts.....	1219
4.1.1.1 Метод Charts:getChartsCount.....	1220
4.1.1.2 Метод Charts:getChart	1220
4.1.1.3 Метод Charts:getChartIndexByDrawingIndex	1221
4.1.2 Таблица DocumentAPI.Chart	1221
4.1.2.1 Метод Chart:getType	1221
4.1.2.2 Метод Chart:setType.....	1221
4.1.2.3 Метод Chart:getRangesCount	1221
4.1.2.4 Метод Chart:getRange	1222
4.1.2.5 Метод Chart:getTitle	1222
4.1.2.6 Метод Chart:setRange.....	1222
4.1.2.7 Метод Chart:setRect.....	1222
4.1.2.8 Метод Chart:isEmpty	1222
4.1.2.9 Метод Chart:isSolidRange	1223
4.1.2.10 Метод Chart:is3D	1223
4.1.2.11 Метод Chart:getDirectionType	1223
4.1.2.12 Метод Chart:getChartLabels	1223
4.1.2.13 Метод Chart:getRangeAsString	1224
4.1.2.14 Метод Chart:applySettings.....	1224

4.1.3 Таблица DocumentAPI.ChartLabelsDetectionMode.....	1225
4.1.4 Таблица DocumentAPI.ChartLabelsInfo	1225
4.1.5 Таблица DocumentAPI.ChartRangeInfo	1226
4.1.6 Таблица DocumentAPI.ChartRangeType.....	1227
4.1.7 Таблица DocumentAPI.ChartSeriesDirectionType	1227
4.1.8 Таблица DocumentAPI.ChartType	1228
4.2 Именованные диапазоны.....	1229
4.2.1 Таблица DocumentAPI.NamedExpressions	1229
4.2.1.1 Метод NamedExpressions:get.....	1230
4.2.1.2 Метод NamedExpressions:enumerate	1230
4.2.1.3 Метод NamedExpression:addExpression.....	1230
4.2.1.4 Метод NamedExpressions:removeExpression.....	1230
4.2.2 Таблица DocumentAPI.NamedExpression	1231
4.2.2.1 Метод NamedExpression:getName.....	1231
4.2.2.2 Метод NamedExpression:getExpression	1231
4.2.2.3 Метод NamedExpression:getCellRange	1231
4.2.3 Таблица DocumentAPI.NamedExpressionsValidationResult	1232
4.3 Макрокоманды	1232
4.3.1 Таблица DocumentAPI.Scripts	1233
4.3.1.1 Метод Scripts:getScript.....	1233
4.3.1.2 Метод Scripts:setScript	1233
4.3.1.3 Метод Scripts:removeScript.....	1233
4.3.1.4 Метод Scripts:enumerate.....	1234
4.3.2 Таблица DocumentAPI.Script.....	1234
4.3.2.1 Таблица DocumentAPI.Scripting.....	1234
4.3.2.1.1 Метод Scripting:runScript.....	1234
4.3.2.2 Метод Script:getName	1234
4.3.2.3 Метод Script:setName.....	1235
4.3.2.4 Метод Script:getBody	1235
4.3.2.5 Метод Script:setBody.....	1235
4.3.3 Функция DocumentAPI.createScripting	1235
4.4 Разделы (секции) документа	1236
4.4.1 Таблица DocumentAPI.Sections.....	1237
4.4.1.1 Метод Sections:enumerate	1237
4.4.2 Таблица DocumentAPI.Section	1237
4.4.2.1 Метод Section:setPageProperties	1237
4.4.2.2 Метод Section:getPageProperties.....	1237
4.4.2.3 Метод Section:setPageOrientation	1238

4.4.2.4	Метод Section:getPageOrientation.....	1238
4.4.2.5	Метод Section:getRange	1238
4.4.2.6	Метод Section:getHeaders	1239
4.4.2.7	Метод Section:getFooters.....	1239
4.4.3	Таблица DocumentAPI.HeadersFooters	1239
4.4.3.1	Метод HeadersFooters:enumerate.....	1240
4.4.4	Таблица DocumentAPI.HeaderFooter	1240
4.4.4.1	Метод HeaderFooter:getType	1240
4.4.4.2	Метод HeaderFooter:getBlocks	1241
4.4.4.3	Метод HeaderFooter:getRange	1241
4.4.5	Таблица DocumentAPI.HeaderFooterType	1241
4.4.6	Таблица DocumentAPI.PageOrientation	1241
4.4.7	Таблица DocumentAPI.Insets	1242
4.4.8	Таблица DocumentAPI.PageProperties	1242
4.4.8.1	Метод PageProperties: __eq.....	1243
4.5	Блоки, параграфы.....	1244
4.5.1	Таблица DocumentAPI.Blocks	1244
4.5.1.1	Метод Blocks:getBlock.....	1244
4.5.1.2	Метод Blocks:getParagraph	1244
4.5.1.3	Метод Blocks:getTable	1245
4.5.1.4	Метод Blocks:getShape.....	1245
4.5.1.5	Метод Blocks:getField	1245
4.5.1.6	Метод Blocks:enumerate.....	1245
4.5.1.7	Метод Blocks:enumerateParagraphs	1245
4.5.1.8	Метод Blocks:enumerateTables	1246
4.5.1.9	Метод Blocks:enumerateShapes	1246
4.5.1.10	Метод Blocks:enumerateFields.....	1246
4.5.2	Таблица DocumentAPI.Block.....	1246
4.5.2.1	Методы toParagraph, toTable, toShape, toField	1247
4.5.2.2	Метод Block:getRange	1247
4.5.2.3	Метод Block.remove	1247
4.5.2.4	Метод Block:getSection	1248
4.5.3	Таблица DocumentAPI.Field	1248
4.5.4	Таблица DocumentAPI.Paragraphs	1248
4.5.4.1	Метод Paragraphs:setListSchema	1248
4.5.4.2	Метод Paragraphs:setListLevel	1249
4.5.4.3	Метод Paragraphs:increaseListLevel	1249
4.5.4.4	Метод Paragraphs:decreaseListLevel.....	1249

4.5.4.5	Метод Paragraph:enumerate	1249
4.5.5	Таблица DocumentAPI.Paragraph	1250
4.5.5.1	Метод Paragraph:getParagraphProperties	1251
4.5.5.2	Метод Paragraph:setParagraphProperties	1252
4.5.5.3	Метод Paragraph:getListSchema	1252
4.5.5.4	Метод Paragraph:setListSchema	1252
4.5.5.5	Метод Paragraph:getListLevel	1253
4.5.5.6	Метод Paragraph:setListLevel	1253
4.5.5.7	Метод Paragraph:increaseListLevel	1253
4.5.5.8	Метод Paragraph:decreaseListLevel	1254
4.5.6	Таблица DocumentAPI.ListSchema	1254
4.5.7	Таблица DocumentAPI.ParagraphProperties	1255
4.5.8	Таблица DocumentAPI.LineSpacing	1258
4.5.9	Таблица DocumentAPI.LineSpacingRule	1258
4.5.10	Таблица DocumentAPI.Alignment	1260
4.6	Рецензирование документов	1261
4.6.1	Таблица DocumentAPI.TrackedChange	1262
4.6.1.1	Метод TrackedChange:getRange	1262
4.6.1.2	Метод TrackedChange:getType	1263
4.6.1.3	Метод TrackedChange:getInfo	1263
4.6.2	Таблица DocumentAPI.TrackedChangeInfo	1263
4.6.2.1	Метод TrackedChangeInfo: __eq	1264
4.6.3	Таблица DocumentAPI.DateTime	1264
4.6.3.1	Метод DateTime: __eq	1264
4.6.4	Таблица DocumentAPI.TrackedChangeType	1264
4.6.5	Таблица DocumentAPI.Comments	1265
4.6.5.1	Метод Comments:enumerate	1265
4.6.6	Таблица DocumentAPI.Comment	1266
4.6.6.1	Метод Comment:getRange	1266
4.6.6.2	Метод Comment:getText	1266
4.6.6.3	Метод Comment:getInfo	1266
4.6.6.4	Метод Comment:isResolved	1267
4.6.6.5	Метод Comment:getReplies	1267
4.7	Работа с закладками	1267
4.7.1	Таблица DocumentAPI.Bookmarks	1269
4.7.1.1	Метод Bookmarks:getBookmarkRange	1269
4.7.1.2	Метод Bookmarks:removeBookmark	1269
4.8	Таблицы и ячейки	1269

4.8.1 Доступ к таблицам	1269
4.8.2 Доступ к ячейкам	1271
4.8.2.1 Таблица DocumentAPI.CellFormat	1273
4.8.2.2 Таблица DocumentAPI.AccountingCellFormatting	1276
4.8.2.3 Таблица DocumentAPI.PercentageCellFormatting	1276
4.8.2.4 Таблица DocumentAPI.NumberCellFormatting.....	1277
4.8.2.5 Таблица DocumentAPI.CurrencyCellFormatting.....	1278
4.8.2.6 Таблица DocumentAPI.CurrencySignPlacement	1279
4.8.2.7 Таблица DocumentAPI.DateTimeCellFormatting.....	1279
4.8.2.8 Таблица DocumentAPI.DatePatterns	1280
4.8.2.9 Таблица DocumentAPI.TimePatterns	1281
4.8.2.10 Таблица DocumentAPI.FractionCellFormatting	1281
4.8.2.11 Таблица DocumentAPI.ScientificCellFormatting	1282
4.8.3 Форматирование ячеек	1282
4.8.4 Форматирование границ ячеек	1283
4.8.4.1 Таблица DocumentAPI.Borders.....	1285
4.8.4.2 Таблица DocumentAPI.RangeBorders	1286
4.8.4.3 Таблица DocumentAPI.LineProperties.....	1286
4.8.4.3.1 Поле LineProperties.style	1287
4.8.4.3.2 Поле LineProperties.width	1287
4.8.4.3.3 Поле LineProperties.color	1288
4.8.4.3.4 Поле LineProperties.headLineEndingProperties.....	1288
4.8.4.3.5 Поле LineProperties.tailLineEndingProperties	1288
4.8.4.4 Таблица DocumentAPI.LineEndingProperties	1288
4.8.4.5 Таблица DocumentAPI.LineStyle.....	1289
4.8.4.6 Таблица DocumentAPI.LineEndingStyle	1290
4.8.5 Объединение и разделение ячеек таблицы.....	1291
4.8.6 Таблица DocumentAPI.Table	1291
4.8.6.1 Метод Table:setName	1292
4.8.6.2 Метод Table:getName.....	1292
4.8.6.3 Метод Table:getRowCount.....	1293
4.8.6.4 Метод Table:getColumnsCount	1293
4.8.6.5 Метод Table:getCell.....	1293
4.8.6.6 Метод Table:getCellRange.....	1293
4.8.6.7 Метод Table:insertColumnAfter	1294
4.8.6.8 Метод Table:insertColumnBefore.....	1294
4.8.6.9 Метод Table:insertRowAfter	1295
4.8.6.10 Метод Table:insertRowBefore	1296

4.8.6.11	Метод Table:removeColumn	1296
4.8.6.12	Метод Table:removeRow	1296
4.8.6.13	Метод Table:setColumnWidth	1297
4.8.6.14	Метод Table:setRowHeight	1297
4.8.6.15	Метод Table:duplicate.....	1298
4.8.6.16	Метод Table:remove	1298
4.8.6.17	Метод Table:moveTo	1298
4.8.6.18	Метод Table:setShowZeroValue.....	1299
4.8.6.19	Метод Table:getShowZeroValue	1299
4.8.6.20	Метод Table:setVisible	1299
4.8.6.21	Метод Table:isVisible	1299
4.8.6.22	Метод Table:getFrozenRange	1300
4.8.6.23	Метод Table:freeze.....	1300
4.8.6.24	Метод Table: __eq.....	1300
4.8.6.25	Метод Table:setPrintArea	1301
4.8.6.26	Метод Table:setPrintAreas	1301
4.8.6.27	Метод Table:getPrintAreas	1301
4.8.6.28	Метод Table:getCharts	1302
4.8.6.29	Метод Table:getImages	1302
4.8.6.30	Метод Table:getMediaObjects	1302
4.8.6.31	Метод Table:getNamedExpressions.....	1303
4.8.6.32	Группировка строк и колонок таблицы	1303
4.8.6.33	Управление видимостью строк / колонок	1303
4.8.7	Таблица DocumentAPI.Cell.....	1303
4.8.7.1	Метод Cell:getRange.....	1304
4.8.7.2	Метод Cell:setBorders.....	1304
4.8.7.3	Метод Cell:setFormula.....	1304
4.8.7.4	Метод Cell:getFormat	1304
4.8.7.5	Метод Cell:setFormat.....	1305
4.8.7.6	Метод Cell:getFormattedValue	1307
4.8.7.7	Метод Cell:setFormattedValue	1308
4.8.7.8	Метод Cell:unmerge.....	1308
4.8.7.9	Метод Cell:getHyperlink.....	1308
4.8.7.10	Метод Cell:setContent.....	1308
4.8.7.11	Метод Cell:getBorders	1308
4.8.7.12	Метод Cell:getRawValue	1309
4.8.7.13	Метод Cell:getCustomFormat.....	1309
4.8.7.14	Метод Cell:setCustomFormat	1309

4.8.7.15	Метод Cell:setBool.....	1309
4.8.7.16	Метод Cell:setNumber	1309
4.8.7.17	Метод Cell:setText	1310
4.8.7.18	Метод Cell:getFormulaAsString	1310
4.8.7.19	Метод Cell:getCellProperties	1310
4.8.7.20	Метод Cell:setCellProperties	1310
4.8.7.21	Метод Cell:getParagraphProperties	1311
4.8.7.22	Метод Cell:setParagraphProperties	1311
4.8.7.23	Метод Cell:getPivotTable	1311
4.8.8	Таблица DocumentAPI.CellProperties	1311
4.8.9	Таблица DocumentAPI.VerticalAlignment	1313
4.8.10	Таблица DocumentAPI.Hyperlink	1314
4.8.10.1	Метод Hyperlink: __eq	1314
4.8.11	Таблица DocumentAPI.TextLayout.....	1315
4.8.12	Таблица DocumentAPI.TextOrientation.....	1315
4.8.12.1	Метод TextOrientation:getAngle	1316
4.8.13	Таблица DocumentAPI.CellPosition	1316
4.8.13.1	Поле CellPosition.column	1316
4.8.13.2	Поле CellPosition.row	1316
4.8.13.3	Метод CellPosition:toString.....	1317
4.8.13.4	Метод CellPosition: __eq.....	1317
4.8.14	Таблица DocumentAPI.CellRange	1317
4.8.14.1	Метод CellRange:enumerate	1317
4.8.14.2	Метод CellRange:getBeginRow.....	1318
4.8.14.3	Метод CellRange:getBeginColumn	1318
4.8.14.4	Метод CellRange:getLastRow	1318
4.8.14.5	Метод CellRange:getLastColumn.....	1318
4.8.14.6	Метод CellRange:setBorders	1319
4.8.14.7	Метод CellRange:insertCurrentDateTime	1319
4.8.14.8	Метод CellRange:getCellProperties.....	1319
4.8.14.9	Метод CellRange:setCellProperties	1320
4.8.14.10	Метод CellRange:merge.....	1320
4.8.14.11	Метод CellRange:unmerge.....	1320
4.8.15	Таблица DocumentApi.DateTimeFormat	1320
4.8.16	Таблица DocumentAPI.CellRangePosition	1321
4.8.16.1	Метод CellRangePosition:toString.....	1322
4.8.16.2	Метод CellRangePosition: __eq.....	1322
4.8.17	Таблица DocumentAPI.FrozenRangePosition.....	1322

4.8.17.1	Конструкторы.....	1322
4.8.17.2	Метод FrozenRangePosition:create	1323
4.8.17.3	Метод FrozenRangePosition:createFrozenArea.....	1323
4.8.17.4	Метод FrozenRangePosition:createFrozenRows	1323
4.8.17.5	Метод FrozenRangePosition:createFrozenCols	1323
4.8.17.6	Метод FrozenRangePosition:isRowsCols	1324
4.8.17.7	Метод FrozenRangePosition:isArea.....	1324
4.8.17.8	Метод FrozenRangePosition:isRows	1324
4.8.17.9	Метод FrozenRangePosition:isCols	1324
4.8.17.10	Метод FrozenRangePosition: __eq	1325
4.8.18	Таблица DocumentAPI.TableRangeInfo	1325
4.9	Сводные таблицы.....	1325
4.9.1	Таблица DocumentAPI.PivotTablesManager	1326
4.9.1.1	Метод PivotTablesManager:create.....	1326
4.9.2	Таблица DocumentAPI.PivotTable.....	1327
4.9.2.1	Метод PivotTable:remove	1327
4.9.2.2	Метод PivotTable:getSourceRangeAddress.....	1327
4.9.2.3	Метод PivotTable:getSourceRange.....	1327
4.9.2.4	Метод PivotTable:getPivotRange	1328
4.9.2.5	Метод PivotTable:changeSourceRange	1328
4.9.2.6	Метод PivotTable:isRowGrandTotalEnabled	1328
4.9.2.7	Метод PivotTable:isColumnGrandTotalEnabled.....	1329
4.9.2.8	Метод PivotTable:getPivotTableCaptions	1329
4.9.2.9	Метод PivotTable:getPivotTableLayoutSettings	1329
4.9.2.10	Метод PivotTable:getUnsupportedFeatures.....	1329
4.9.2.11	Метод PivotTable:getFieldsList	1330
4.9.2.12	Метод PivotTable:getRowFields.....	1330
4.9.2.13	Метод PivotTable:getColumnFields	1330
4.9.2.14	Метод PivotTable:getValueFields.....	1331
4.9.2.15	Метод PivotTable:getPageFields	1331
4.9.2.16	Метод PivotTable:getFieldCategories.....	1331
4.9.2.17	Метод PivotTable:getFieldItems	1331
4.9.2.18	Метод PivotTable:getFieldItemsByName.....	1332
4.9.2.19	Метод PivotTable:getFilter	1332
4.9.2.20	Метод PivotTable:getFilters.....	1332
4.9.2.21	Метод PivotTable:update	1332
4.9.2.22	Метод PivotTable:createPivotTableEditor.....	1333
4.9.3	Таблица DocumentAPI.PivotTableCaptions	1333

4.9.4 Таблица DocumentAPI.PivotTableLayoutSettings	1333
4.9.5 Таблица DocumentAPI.PivotTableReportLayout	1334
4.9.6 Таблица DocumentAPI.ValueFieldsOrientation.....	1335
4.9.7 Таблица DocumentAPI.PageFieldOrder	1335
4.9.8 Таблица DocumentAPI.PivotTableUnsupportedFeature	1335
4.9.9 Таблица DocumentAPI.PivotTableFieldCategories	1336
4.9.9.1 Метод PivotTableFieldCategories:enumerate	1336
4.9.10 Таблица DocumentAPI.PivotTableFunction	1336
4.9.11 Таблица DocumentAPI.PivotTableFilters	1337
4.9.11.1 Метод PivotTableFilters:enumerate	1337
4.9.12 Таблица DocumentAPI.PivotTableFilter	1337
4.9.12.1 Метод PivotTableFilter:getFieldName.....	1338
4.9.12.2 Метод PivotTableFilter:getCount.....	1338
4.9.12.3 Метод PivotTableFilter:getName.....	1338
4.9.12.4 Метод PivotTableFilter:isHidden.....	1339
4.9.12.5 Метод PivotTableFilter:setHidden	1339
4.9.13 Таблица DocumentAPI.PivotTableField	1339
4.9.14 Таблица DocumentAPI.PivotTableFieldProperties	1340
4.9.15 Таблица DocumentAPI.PivotTableCategoryField.....	1340
4.9.16 Таблица DocumentAPI.PivotTableValueField.....	1340
4.9.17 Таблица DocumentAPI.PivotTablePageField.....	1341
4.9.18 Таблица DocumentAPI.PivotTableItems	1341
4.9.18.1 Метод PivotTableItems:enumerate	1341
4.9.19 Таблица DocumentAPI.PivotTableItem	1342
4.9.19.1 Метод PivotTableItem:getName	1342
4.9.19.2 Метод PivotTableItem:getAlias	1342
4.9.19.3 Метод PivotTableItem:getItemType	1342
4.9.19.4 Метод PivotTableItem:isCollapsed.....	1343
4.9.20 Таблица DocumentAPI.PivotTableItemType	1343
4.9.21 Таблица DocumentAPI.PivotTableEditor.....	1343
4.9.21.1 Метод PivotTableEditor:addField.....	1344
4.9.21.2 Метод PivotTableEditor:moveField.....	1344
4.9.21.3 Метод PivotTableEditor:removeField.....	1344
4.9.21.4 Метод PivotTableEditor:reorderField	1345
4.9.21.5 Метод PivotTableEditor:enableField	1345
4.9.21.6 Метод PivotTableEditor:disableField	1345
4.9.21.7 Метод PivotTableEditor:setSummarizeFunction	1346
4.9.21.8 Метод PivotTableEditor:setFilter.....	1346

4.9.21.9	Метод PivotTableEditor:setFilters	1347
4.9.21.10	Метод PivotTableEditor:setCaptions	1347
4.9.21.11	Метод PivotTableEditor:setLayoutSettings	1347
4.9.21.12	Метод PivotTableEditor:setGrandTotalSettings	1348
4.9.21.13	Метод PivotTableEditor:apply	1348
4.9.22	Таблица DocumentAPI.PivotTableUpdateResult	1348
4.9.23	Таблица DocumentAPI.PivotTableFieldCategory	1349
4.10	Графические объекты	1350
4.10.1	Таблица DocumentAPI.MediaObjects	1353
4.10.1.1	Метод MediaObjects:enumerate	1354
4.10.2	Таблица DocumentAPI.MediaObject	1355
4.10.2.1	Метод MediaObject:toImage	1355
4.10.2.2	Метод MediaObject:getFrame	1355
4.10.3	Таблица DocumentAPI.Image	1356
4.10.3.1	Метод Image:getFrame	1356
4.10.3.2	Метод Image:remove	1357
4.10.4	Таблица DocumentAPI.Images	1357
4.10.4.1	Метод Images:enumerate	1357
4.10.5	Таблица DocumentAPI.AbsoluteFrame	1358
4.10.5.1	Метод AbsoluteFrame:moveTo	1358
4.10.5.2	Метод AbsoluteFrame:getTopLeft	1359
4.10.5.3	Метод AbsoluteFrame:scale	1359
4.10.5.4	Метод AbsoluteFrame:setDimensions	1360
4.10.5.5	Метод AbsoluteFrame:getDimensions	1360
4.10.6	Таблица DocumentAPI.ScaleFrom	1360
4.10.7	Таблица DocumentAPI.InlineFrame	1361
4.10.7.1	Метод InlineFrame:setPosition	1362
4.10.7.2	Метод InlineFrame:getPosition	1362
4.10.7.3	Метод InlineFrame:setDimensions	1363
4.10.7.4	Метод InlineFrame:getDimensions	1363
4.10.7.5	Метод InlineFrame:setWrapType	1363
4.10.7.6	Метод InlineFrame:getWrapType	1363
4.10.8	Таблица DocumentAPI.TextWrapType	1364
4.10.9	Таблица DocumentAPI.TextAnchoredPosition	1364
4.10.9.1	Метод TextAnchoredPosition: __eq	1365
4.10.10	Таблица DocumentAPI.HorizontalTextAnchoredPosition	1365
4.10.10.1	Метод HorizontalTextAnchoredPosition: __eq	1366
4.10.11	Таблица DocumentAPI.VerticalTextAnchoredPosition	1366

4.10.11.1	Метод VerticalTextAnchoredPosition: __eq.....	1367
4.10.12	Таблица DocumentAPI.VerticalRelativeTo.....	1367
4.10.13	Таблица DocumentAPI.HorizontalRelativeTo	1368
4.10.14	Таблица DocumentAPI.VerticalAnchorAlignment	1368
4.10.15	Таблица DocumentAPI.HorizontalAnchorAlignment	1369
4.10.16	Таблица DocumentAPI.Shape	1369
4.10.16.1	Метод Shape:getShapeProperties.....	1369
4.10.16.2	Метод Shape:setShapeProperties	1369
4.10.17	Таблица DocumentAPI.ShapeProperties	1370
4.10.17.1	Поле ShapeProperties:borderProperties	1370
4.10.17.2	Поле ShapeProperties:verticalAlignment.....	1370
4.10.17.3	Поле ShapeProperties:fill	1370
4.10.17.4	Поле ShapeProperties:shapeTextLayout	1370
4.10.18	Таблица DocumentAPI.ShapeTextLayout.....	1370
4.10.19	Таблица DocumentAPI.Fill.....	1371
4.10.19.1	Метод Fill:getColor	1371
4.10.19.2	Метод Fill:getUrl.....	1371
4.10.19.3	Метод Fill:isNoFill.....	1371
4.11	Поиск в документе	1371
4.11.1	Метод DocumentAPI:createSearch.....	1372
4.11.2	Таблица DocumentAPI.Search	1372
4.11.2.1	Метод Search:findText.....	1372
4.12	Таблица DocumentAPI.document	1372
4.12.1	Метод document:getBlocks.....	1373
4.12.2	Метод document:getBookmarks	1373
4.12.3	Метод document:getScripts.....	1373
4.12.4	Метод document:getRange.....	1373
4.12.5	Метод document:isChangesTrackingEnabled.....	1373
4.12.6	Метод document:setChangesTrackingEnabled.....	1374
4.12.7	Метод document:getComments	1374
4.12.8	Метод document:setPageProperties	1374
4.12.9	Метод document:setFormulaType	1375
4.12.10	Метод document:getFormulaType	1375
4.12.11	Метод document:setPageOrientation	1375
4.12.12	Метод document:enumerateSections	1375
4.12.13	Метод document:getSections	1375
4.12.14	Метод document:setMirroredMarginsEnabled	1376
4.12.15	Метод document:areMirroredMarginsEnabled.....	1376

4.12.16	Метод document:getPivotTablesManager	1376
4.12.17	Метод document:getNamedExpressions	1376
4.13	Таблица DocumentAPI.TextProperties	1377
4.14	Таблица DocumentAPI.ScriptPosition	1378
4.15	Таблица DocumentAPI.Range	1379
4.15.1	Метод Range:getBegin	1380
4.15.2	Метод Range:getEnd	1381
4.15.3	Метод Range:extractText	1381
4.15.4	Метод Range:removeContent	1381
4.15.5	Метод Range:lockContent	1382
4.15.6	Метод Range:unlockContent	1382
4.15.7	Метод Range:isContentLocked	1382
4.15.8	Метод Range:replaceText	1383
4.15.9	Метод Range:setHyperlink	1383
4.15.10	Метод Range:getTextProperties	1384
4.15.11	Метод Range:setTextProperties	1384
4.15.12	Метод Range:enumerateBlocks	1385
4.15.13	Метод Range:enumerateTrackedChanges	1385
4.15.14	Метод Range:getComments	1385
4.15.15	Метод Range:getParagraphs	1386
4.15.16	Метод Range:getImages	1386
4.15.17	Метод Range:getInlineObjects	1387
4.16	Таблица DocumentAPI.Position	1387
4.16.1	Метод Position:insertText	1387
4.16.2	Метод Position:insertTable	1387
4.16.3	Метод Position:insertPageBreak	1388
4.16.4	Метод Position:insertLineBreak	1388
4.16.5	Метод Position:insertBookmark	1389
4.16.6	Метод Position:insertSectionBreak	1389
4.16.7	Метод Position:insertHyperlink	1389
4.16.8	Метод Position:insertImage	1390
4.16.9	Метод Position:removeBackward	1390
4.16.10	Метод Position:removeForward	1390
4.16.11	Метод Position: __eq	1390
4.17	Таблица DocumentAPI.Color	1391
4.17.1	Метод Color:getRGBAColor	1391
4.17.2	Метод Color:getThemeColorID	1391
4.17.3	Метод Color: __eq	1391

4.18 Таблица DocumentAPI.ColorRGBA	1392
4.18.1 Метод ColorRGBA: __eq	1392
4.19 Таблица DocumentAPI.ThemeColorID	1392
4.20 Таблица DocumentAPI.PrintSettings	1393
4.21 Таблица DocumentAPI.WorksheetPrinterFitType	1395
4.22 Таблица DocumentAPI.PrintDocumentResult	1395
4.23 Таблица DocumentAPI.SizeU	1395
4.23.1 Метод SizeU:toString	1396
4.24 Таблица DocumentAPI.PointU	1396
4.24.1 Метод PointU:toString	1396
4.25 Таблица DocumentAPI.RectU	1397
4.25.1 Метод RectU:toString	1397
4.26 Таблица DocumentAPI.VectorUInt	1397
4.26.1 Метод VectorUInt:size	1398
4.26.2 Метод VectorUInt:max_size	1398
4.26.3 Метод VectorUInt:empty	1398
4.26.4 Метод VectorUInt:clear	1398
4.26.5 Метод VectorUInt:push_back	1398
4.26.6 Метод VectorUInt:pop_back	1399
4.26.7 Метод VectorUInt:front	1399
4.26.8 Метод VectorUInt:back	1399
4.26.9 Метод VectorUInt: __getitem	1399
4.26.10 Метод VectorUInt: __setitem	1400
4.27 Таблица DocumentAPI.VectorString	1401
4.27.1 Метод VectorString:size	1401
4.27.2 Метод VectorString:max_size	1401
4.27.3 Метод VectorString:empty	1401
4.27.4 Метод VectorString:clear	1402
4.27.5 Метод VectorString:push_back	1402
4.27.6 Метод VectorString:pop_back	1402
4.27.7 Метод VectorString:front	1402
4.27.8 Метод VectorString:back	1403
4.27.9 Метод VectorString: __getitem	1403
4.27.10 Метод VectorString: __setitem	1403
5 СПРАВОЧНИК ФУНКЦИЙ EDITORAPI	1404
5.1 Функция EditorAPI.getSelection	1404
5.2 Функция EditorAPI.setSelection	1404
5.3 Функция EditorAPI.messageBox	1405

5.4	Функция EditorAPI.showPrintDialog.....	1405
5.5	Функция EditorAPI.printDocument.....	1406
5.6	Функция EditorAPI.isPrinterAvailable.....	1406
5.7	Функция EditorAPI.getActiveWorksheet.....	1406
5.8	Функция EditorAPI.setActiveWorksheet.....	1407
6	ФУНКЦИИ ДЛЯ РАБОТЫ СО СТРОКАМИ В ФОРМАТЕ ЮНИКОД (UTF-8).....	1408
6.1	Функция utf8.char.....	1408
6.2	Функция utf8.codes.....	1408
6.3	Функция utf8.codepoint.....	1409
6.4	Функция utf8.charpattern.....	1409
6.5	Функция utf8.upper.....	1409
6.6	Функция utf8.lower.....	1410
6.7	Функция utf8.substr.....	1410
6.8	Функция utf8.compare.....	1411
6.9	Функция utf8.islower.....	1412
6.10	Функция utf8.isupper.....	1412
6.11	Функция utf8.isdigit.....	1413
6.12	Функция utf8.isalpha.....	1413
6.13	Функция utf8.len.....	1413
6.14	Функция utf8.offset.....	1414
6.15	Функция utf8.next.....	1414
7	ФУНКЦИИ ДЛЯ РАБОТЫ С РЕГУЛЯРНЫМИ ВЫРАЖЕНИЯМИ.....	1416
7.1	Функция Re.create.....	1416
7.2	Функция Re.match.....	1416
7.2.1	Флаги, используемые в Re.match.....	1416
7.3	Функция Re.search.....	1418
7.4	Функция Re.replace.....	1418
7.5	Флаги, используемые для замены.....	1419
8	КЛАСС MATCHES.....	1421
8.1	Метод getFirst.....	1421
8.2	Метод getLength.....	1421
8.3	Метод getSize.....	1421
8.4	Метод getString.....	1422
8.5	Метод _tostring.....	1422

1 МАКРОКОМАНДЫ ПО МОЙОФИС

Макрокоманды ПО МойОфис представляют собой программы небольшого размера, с помощью которых автоматизируется выполнение продолжительных или часто встречающихся операций.

При работе с документом периодически возникают ситуации, когда приходится повторять одну и ту же последовательность действий. Для оптимизации работы можно создать макрокоманду, которая будет автоматически выполнять эти действия. Для разработки макрокоманд в ПО МойОфис используется язык программирования Lua.

Справочное руководство по языку программирования Lua на русском языке опубликовано по ссылке: http://lua.org.ru/contents_ru.html.

В языке программирования Lua таблицы – это единственная структура данных. Все структуры, которые предлагают другие языки программирования, в том числе массивы, объекты и другие, представлены в языке программирования Lua в виде таблиц.

Структура данных, представляющая текущий открытый документ в ПО МойОфис, в терминах языка программирования Lua является таблицей `DocumentAPI.Document` со своим набором методов. Иные структуры данных для работы с отдельными ячейками, областями, диаграммами, свойствами текста и т. д. также являются таблицами с необходимым набором полей и методов.

Для управления содержимым документа используется объектная модель документа ПО МойОфис. В данном случае термин «объектная модель» обозначает всю совокупность структур данных для управления содержимым текстового или табличного документа ПО МойОфис.

Для управления текстовым или табличным документом ПО МойОфис используются одни и те же методы объектной модели. К примеру, объект `DocumentAPI.Cell` позволяет управлять как отдельной ячейкой электронной таблицы, так и ячейкой таблицы в текстовом документе.

Для работы с текстом макрокоманды используется редактор макрокоманд в составе текстового или табличного редактора ПО МойОфис. Редактор макрокоманд также предоставляет возможность исполнения макрокоманд и доступ к информации об ошибках их исполнения.

Текст макрокоманды сохраняется в текущий открытый документ ПО МойОфис. Макрокоманда в ПО МойОфис может быть сохранена в документы с форматами DOCX, XODT, ODT, XLSX, XODS, ODS.

2 РАБОТА С МАКРОКОМАНДАМИ

В данном разделе описаны действия по созданию, выполнению и отладке макрокоманд в редакторе документов «МойОфис Стандартный».

2.1 Редактор макрокоманд

2.1.1 Окно редактора макрокоманд

Для работы с макрокомандами используется редактор макрокоманд. Чтобы открыть окно редактора, в приложении «Редактор текста» или «Редактор таблиц» выберите пункт командного меню **Инструменты > Макрокоманды > Редактор макрокоманд**.

Окно редактора макрокоманд содержит (см. Рисунок 1189):

1. Список макрокоманд документа.
2. Кнопки для создания **+** и удаления **–** макрокоманд.
3. Область ввода текста макрокоманд.
4. Кнопки выполнения (см. раздел 2.1.3) и отладки (см. раздел 2.1.6) макрокоманд. Кнопки становятся активными, изменяя цвет, после ввода текста макрокоманд в области **3** (см. раздел 2.1.4).
5. Область вывода результата выполнения макрокоманд, а также отображения информации в процессе отладки макрокоманд.

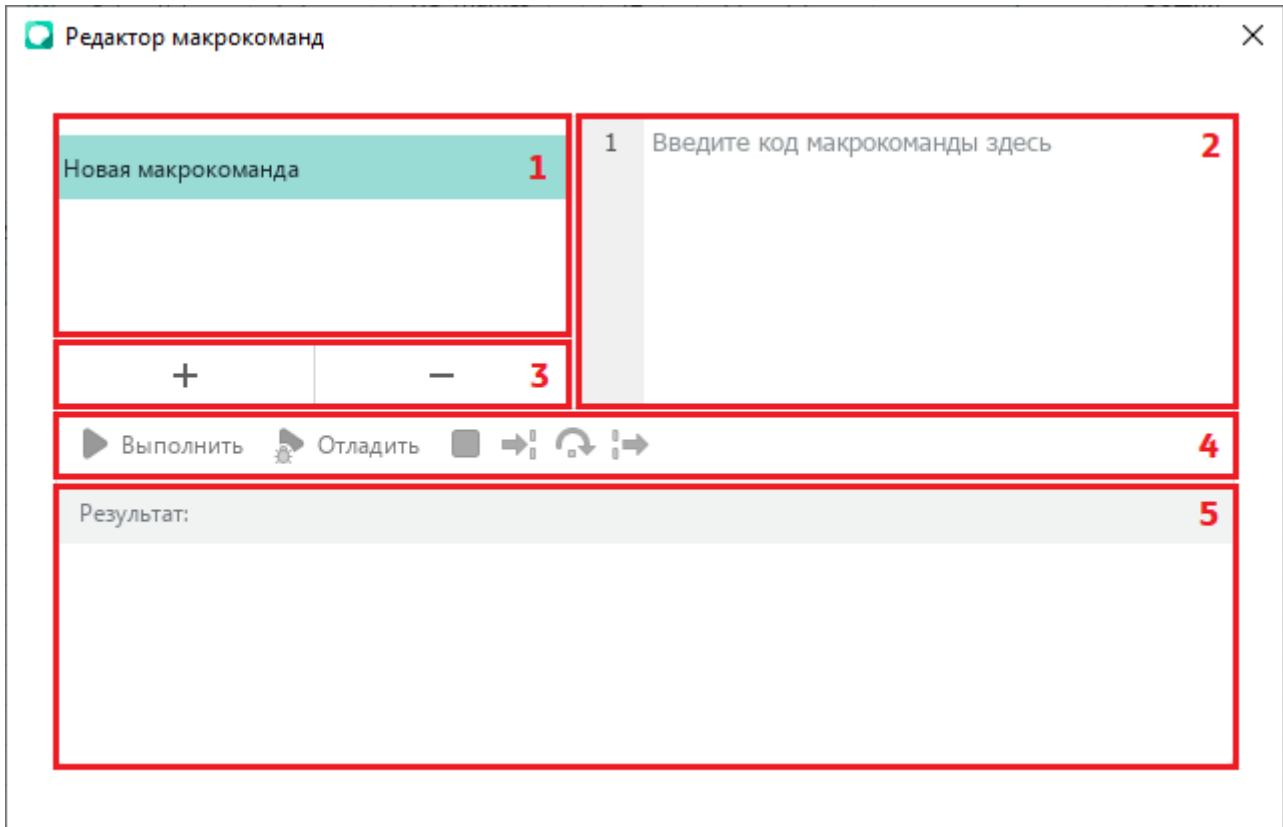


Рисунок 1189 – Окно редактора макрокоманд

2.1.2 Создание макрокоманд

Для создания макрокоманды выполните следующие действия (см. Рисунок 1189):

1. Нажмите кнопку **+** в области **2**.
2. Введите наименование макрокоманды в соответствующей строке перечня макрокоманд в области **1**. Чтобы сохранить название, нажмите клавишу **Enter** или щелкните мышью по любой области окна **Редактор макрокоманд**.
3. Введите текст макрокоманды в области ввода **3**.

2.1.3 Выполнение макрокоманд


Чтобы выполнить макрокоманду, выберите ее в перечне макрокоманд и нажмите кнопку **▶ Выполнить** (см. Рисунок 1189).

Результат выполнения макрокоманды отображается в области **5**.

2.1.4 Редактирование макрокоманд




Чтобы редактировать макрокоманду, выберите ее в перечне макрокоманд и внесите необходимые изменения в ее текст в области **3** (см. Рисунок 1189).

2.1.5 Удаление макрокоманд

Чтобы удалить макрокоманду, выберите ее в перечне макрокоманд и нажмите кнопку  в области **2** (см. Рисунок 1189).

2.1.6 Отладка макрокоманд

Для отладки макрокоманды выполните следующие действия:

1. Выберите наименование макрокоманды в перечне макрокоманд.
2. Установите (при необходимости) в тексте макрокоманд точки останова отладчика, щелкнув мышью справа от номера строки макрокоманды. Строка точки останова будет отмечена значком . Для удаления точки останова щелкните мышью на значок .
3. Нажмите кнопку  **Отладить**. Запустится режим отладки и окно редактора макрокоманд изменит свой вид (см. Рисунок 1190).

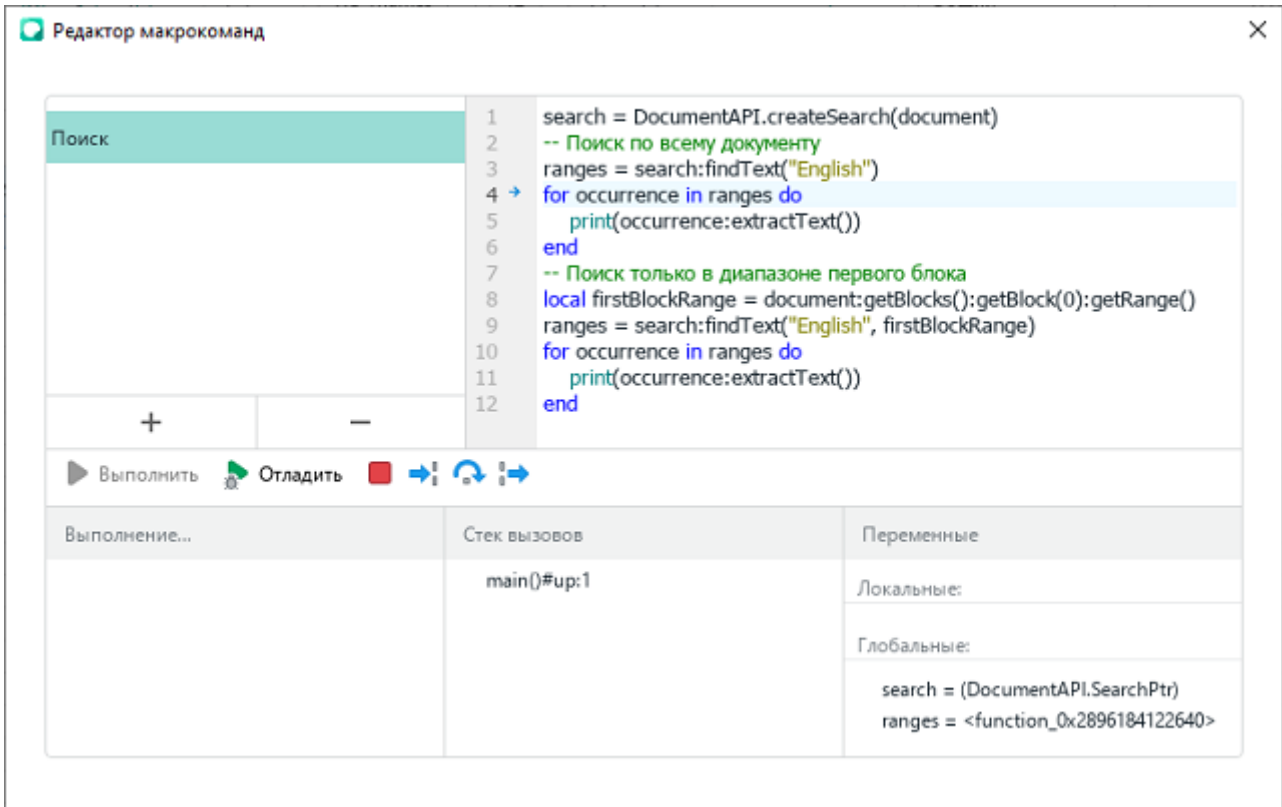






Рисунок 1190 – Окно редактора макрокоманд в начале отладки

Процесс отладки макрокоманды остановится на первой точке останова. Если точки останова отсутствуют, то отладка начнется с остановки на первой строке макрокоманды.


Для продолжения отладки нажмите одну из следующих кнопок: (см. Рисунок 1190):

-  – для выполнения одного шага отладки или захода в тело функции, если таковая есть в текущей позиции отладки;
-  – для выполнения одного шага отладки без захода в тело функции;
-  – для продолжения выполнения макрокоманды до момента выхода из функции, в которой отладчик находится в текущей позиции.

Для прерывания процесса отладки нажмите кнопку  (см. Рисунок 1190), отладка прервется и на экран будет выведено сообщение: «Выполнение макрокоманды прервано пользователем».

В процессе отладки в нижней части окна редактора макрокоманд отображаются следующие области (см. Рисунок 1191):

- **Выполнение...** – окно для вывода сообщений во время отладки, например, командой print;

- **Стек вызовов** – окно стека вызовов;
- **Переменные** – окно вывода значений локальных и глобальных переменных, доступных на текущем шаге выполнения макрокоманды. Если отображаемая переменная представляет из себя таблицу или массив, то при нажатии кнопки , расположенной рядом с именем переменной, доступен просмотр содержимого переменной в развернутом виде.

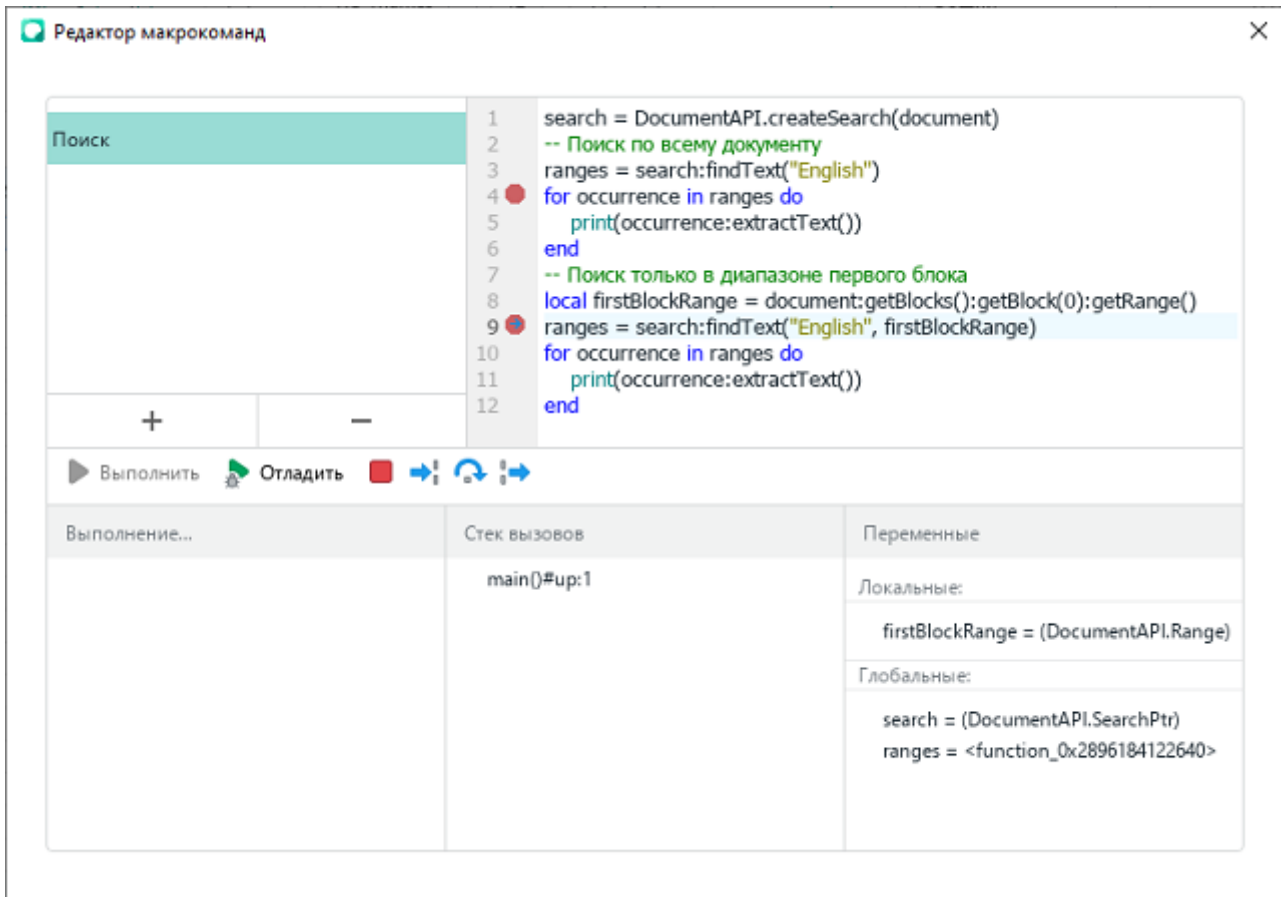


Рисунок 1191 – Окно редактора макрокоманд в процессе отладки макрокоманд

Отладка завершается при достижении конца макрокоманды.

2.2 Пример подготовки и запуска макрокоманды

2.2.1 Редактирование и запуск макрокоманды в текстовом документе

Следующий пример описывает создание и запуск макрокоманды, которая выводит в первой строке текстового документа строку «*HELLO, WORLD!*».

Чтобы создать и запустить макрокоманду, необходимо выполнить следующие действия:

1. Запустить приложение «МойОфис Текст» и открыть редактор макрокоманд. Для этого в командном меню выбрать пункт **Инструменты > Макрокоманды > Редактор макрокоманд**.
2. Нажать кнопку **+** для создания новой макрокоманды. Указать имя макрокоманды. По умолчанию новой макрокоманде присваивается имя «Без имени».
3. Ввести текст макрокоманды:

```
range = document:getRange()
startPos = range:getBegin()

textProp = range:getTextProperties()
textProp.italic = true
textProp.allCapitals = true
range:setTextProperties(textProp)
startPos:insertText("Hello, World!")
```

4. Запустить макрокоманду нажатием на кнопку **▶ Выполнить**. В случае успешного завершения работы макрокоманды редактор макрокоманд выведет сообщение «Макрос выполнен успешно».
5. Закрыть окно редактора макрокоманд, чтобы увидеть изменения в текстовом документе. В первой строке документа отобразится текст «*HELLO, WORLD!*».
6. Сохранить текстовый документ. Для этого выбрать в командном меню пункт **Файл > Сохранить / Файл > Сохранить как** или нажать сочетание клавиш **Ctrl+S**.

При сохранении необходимо выбрать тип файла «Текстовый документ» одного из форматов: DOCX, XODT, ODT (для электронных таблиц - XLSX, XODS, ODS).

2.2.2 Описание примера работы макрокоманды

Ниже приведено описание макрокоманды, текст которой приведен в разделе 2.2.1.

С помощью последовательности вызовов устанавливается курсор в начало документа:

```
range = document:getRange()  
startPos = range:getBegin()
```

Таблица `DocumentAPI.Document` представляет текущий открытый текстовый документ.

Таблица `DocumentAPI.Range` используется для того, чтобы предоставить доступ к любой части (фрагменту) содержимого документа.

В данном случае, переменная `range` содержит весь документ целиком. Вызов `range:getBegin()` устанавливает курсор в начало фрагмента, а в данном случае – в начало самого документа.

Следующая последовательность вызовов настраивает форматирование для документа:

```
textProp = range:getTextProperties()  
textProp.italic = true  
textProp.allCapitals = true  
range:setTextProperties(textProp)
```

В результате выполнения `range:getTextProperties` переменной `textProp` присваивается экземпляр `TextProperties`, содержащий настройки форматирования текущего фрагмента документа.

Таблица `DocumentAPI.TextProperties` позволяет управлять такими характеристиками как наименование и размер шрифта, цвет, начертание и т.п.

В данном примере устанавливаются две настройки форматирования:

- свойство `textProp.italic` принимает значение **true**, что равносильно нажатию кнопки **К (Курсив)** в пользовательском интерфейсе текстового редактора;
- свойство `textProp.allCapitals` принимает значение **true**, что равносильно нажатию кнопки **АВ (Все прописные)** в пользовательском интерфейсе текстового редактора.

Следующий вызов `range:setTextProperties(textProp)` применяет новые настройки форматирования для документа. Теперь эти настройки форматирования будут применяться автоматически для вводимого текста.

Последний вызов вставляет в начало документа текст «Hello, World!»:

```
startPos.insertText("Hello, World!")
```

При вставке текста автоматически применяются настройки форматирования, и итоговый текст отображается как «*HELLO, WORLD!*» прописными буквами курсивом.

2.3 Преобразование макрокоманд на языке программирования VBA

Макрокоманды для пакета Microsoft Office, написанные на языке программирования VBA, не предназначены для исполнения в приложениях ПО «МойОфис Стандартный». Макрокоманды на языке программирования VBA предназначены для исполнения только в пакете Microsoft Office под управлением операционной системы семейства Microsoft Windows.

Однако большинство макрокоманд на языке программирования VBA возможно реализовать на языке программирования Lua с использованием объектной модели ПО «МойОфис Стандартный».

ПО «МойОфис Стандартный» является кроссплатформенным решением (решением не только для работы в операционных системах семейства Microsoft Windows), поэтому при реализации макрокоманд на основе языка программирования VBA следует принимать во внимание следующие ограничения, связанные с операционными системами семейства Microsoft Windows:

- невозможность обращения к внешним приложениям с помощью технологий Component Object Model (COM);
- невозможность использования внешних динамических библиотек DLL.

Также в настоящее время в редакторе макрокоманд ПО «МойОфис Стандартный» существует временное ограничение по работе в тексте макрокоманд с визуальными элементами, такими как выпадающие списки, переключатели и некоторыми другими.

3 ОБЪЕКТНАЯ МОДЕЛЬ МОЙОФИС

МойОфис SDK предоставляет разработчику возможности для управления содержимым текстового и табличного документа. Функции управления сосредоточены в следующей группе таблиц:

- DocumentAPI – содержит таблицы и функции для представления всех элементов документа, которые поддерживает «МойОфис Стандартный»: абзацы, таблицы, рисунки, колонтитулы, операции для работы с текстом, цветом и т.д.;
- EditorAPI – содержит функции для управления редакторами текста и таблиц.

Вышеописанные таблицы составляют объектную модель МойОфис SDK (см. Рисунок 1192).

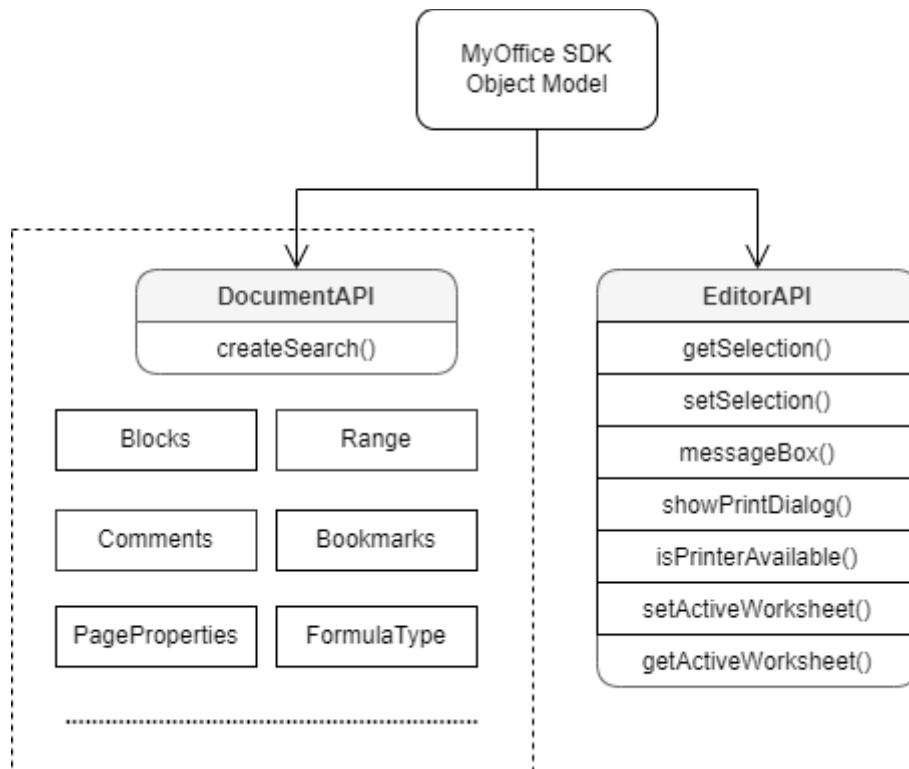


Рисунок 1192 – Объектная модель МойОфис SDK

4 СПРАВОЧНИК ФУНКЦИЙ DOCUMENTAPI

4.1 Диаграммы

Работа с диаграммами реализована только в табличных документах. На рисунке 1193 изображена объектная модель таблиц, относящихся к работе с диаграммами.

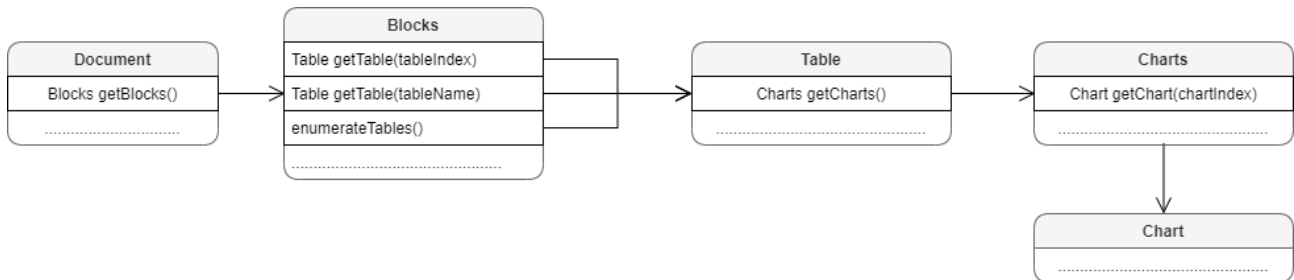


Рисунок 1193 – Объектная модель таблиц для работы с диаграммами

Доступ к списку диаграмм производится через таблицу `DocumentAPI.Table`, соответствующую листу табличного документа.

Пример:

```
local sheetDocumentPage = document:getBlocks():getTable(0)
local charts = sheetDocumentPage:getCharts()
print(charts:getChartsCount())
```

4.1.1 Таблица `DocumentAPI.Charts`

Таблица `DocumentAPI.Charts` обеспечивает доступ к списку диаграмм (см. Рисунок 1194) табличного документа. Доступ к списку диаграмм осуществляется с помощью метода `Table.getCharts()`.

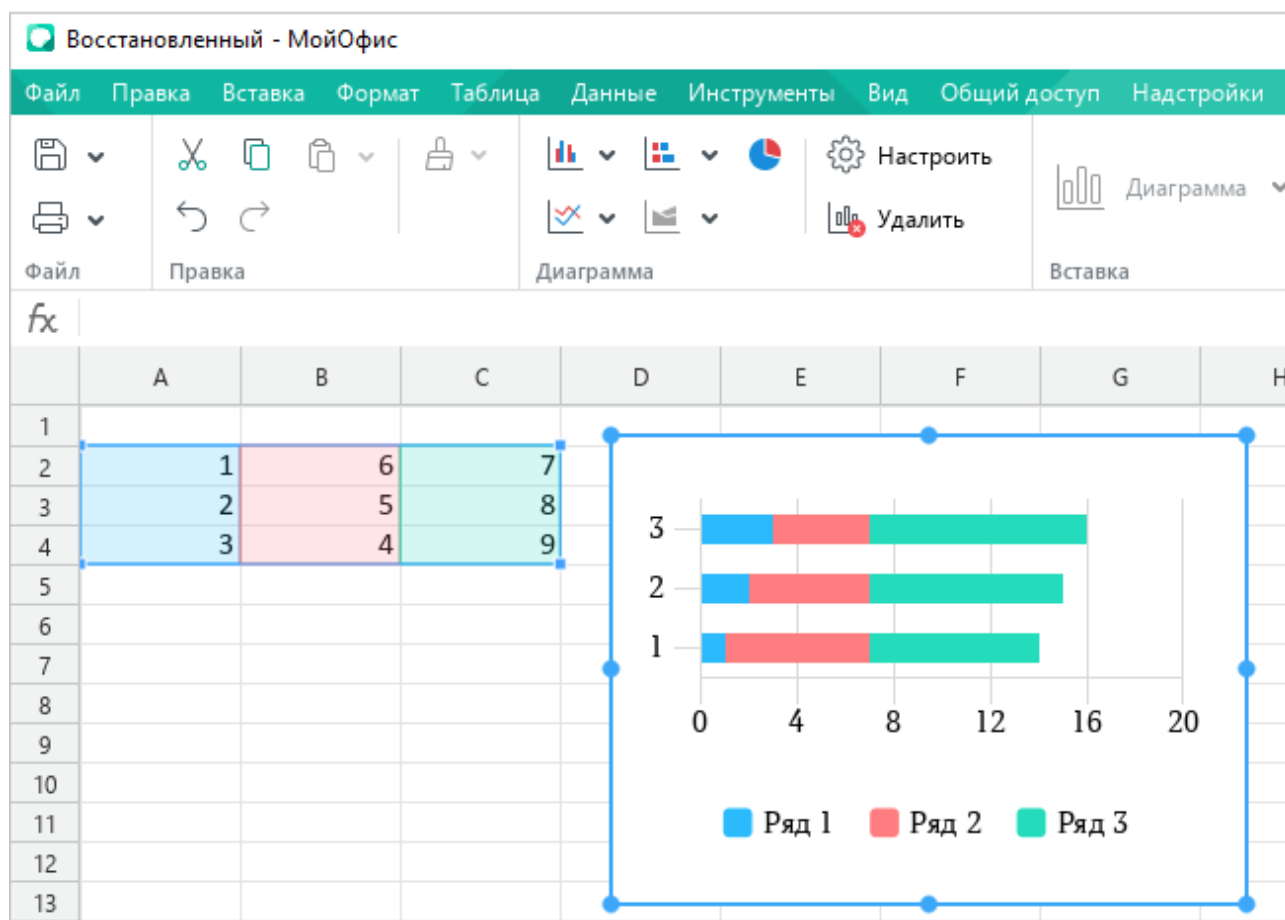


Рисунок 1194 – Пример отображения диаграммы в «Редакторе таблиц»

4.1.1.1 Метод Charts:getChartsCount

Метод возвращает общее количество диаграмм в табличном документе.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChartsCount())
```

4.1.1.2 Метод Charts:getChart

Метод возвращает диаграмму DocumentAPI.Chart по индексу chartIndex в коллекции диаграмм.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRangeAsString())
```

4.1.1.3 Метод `Charts:getChartIndexByDrawingIndex`

Метод возвращает индекс диаграммы по индексу отрисовки `drawingIndex`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChartIndexByDrawingIndex(0))
```

4.1.2 Таблица `DocumentAPI.Chart`

Таблица `DocumentAPI.Chart` представляет диаграмму в табличном документе и описывает все ее элементы (заголовки, легенда, тип, данные, диапазон и т.д.).

4.1.2.1 Метод `Chart:getType`

Метод возвращает тип диаграммы `DocumentAPI.ChartType`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getType())
```

4.1.2.2 Метод `Chart:setType`

Метод устанавливает тип диаграммы `DocumentAPI.ChartType`. Параметр `chartType` – новый тип диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
charts:getChart(0):setType(DocumentAPI.ChartType_LineStacked)
print(charts:getChart(0):getType())
```

4.1.2.3 Метод `Chart:getRangesCount`

Метод возвращает количество серий диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRangesCount())
```

4.1.2.4 Метод Chart:getRange

Метод возвращает диапазон ячеек `DocumentAPI.ChartRangeInfo` с исходными данными диаграммы. Параметр `rangesIndex` – индекс диапазона.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRange(0).rangeType)
```

4.1.2.5 Метод Chart:getTitle

Метод возвращает заголовок диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getTitle())
```

4.1.2.6 Метод Chart:setRange

Метод задает диапазон `DocumentAPI.CellRangePosition` ячеек с исходными данными для диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
charts:getChart(0):setRange(cellRangePosition)
```

4.1.2.7 Метод Chart:setRect

Метод задает область расположения диаграммы, параметр `rect` – новая область.

Важно – Метод устаревший (`deprecated`), оставлен для обратной совместимости и не рекомендован к использованию.

4.1.2.8 Метод Chart:isEmpty

Метод возвращает `true`, если диаграмма не содержит значений.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):isEmpty())
```

4.1.2.9 Метод Chart:isSolidRange

Метод возвращает true, если диапазон исходных данных диаграммы может быть выделен одним прямоугольником и не имеет промежутков.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):isSolidRange())
```

4.1.2.10 Метод Chart:is3D

Метод возвращает true, если диаграмма трехмерная.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):is3D())
```

4.1.2.11 Метод Chart:getDirectionType

Метод возвращает направление DocumentAPI.ChartSeriesDirectionType серий диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getDirectionType())
```

4.1.2.12 Метод Chart:getChartLabels

Метод возвращает коллекцию меток диаграммы типа DocumentAPI.ChartLabelsInfo.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local chart = charts:getChart(0)
local chartLabelsInfo = chart:getChartLabels()
print(chartLabelsInfo.categoriesMode, chartLabelsInfo.seriesNameMode,
chartLabelsInfo.isOneColumnRowChart)
```

4.1.2.13 Метод Chart:getRangeAsString

Метод возвращает диапазон ячеек диаграммы в формате строки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRangeAsString())
```

4.1.2.14 Метод Chart:applySettings

Метод позволяет обновить параметры текущей выбранной диаграммы.

Вызов:

```
applySettings(cellRange, directionType, title, labelsInfo)
```

Параметры:

- cellRange – обновленный диапазон исходных данных диаграммы
DocumentAPI.CellRange;
- directionType – направление серий
DocumentAPI.ChartSeriesDirectionType;
- title – заголовок диаграммы (тип – строка);
- labelsInfo – информация о метках диаграммы DocumentAPI.ChartLabelsInfo.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()

local cellRange = tbl:getCellRange("B3:C4")
local directionType = DocumentAPI.ChartSeriesDirectionType_ByRow
local title = 'Title'
local chartLabelsInfo =
```

```
DocumentAPI.ChartLabelsInfo(DocumentAPI.ChartLabelsDetectionMode_FirstRow,  
DocumentAPI.ChartLabelsDetectionMode_FirstRow, false)  
  
charts:getChart(0):applySettings(cellRange, directionType, title,  
chartLabelsInfo)
```

4.1.3 Таблица DocumentAPI.ChartLabelsDetectionMode

Таблица `DocumentAPI.ChartLabelsDetectionMode` описывает режимы автоматического определения меток диаграмм. Описание полей таблицы представлено в таблице 30.

Таблица 30 – Описание полей таблицы `DocumentAPI.ChartLabelsDetectionMode`

Поле	Описание
<code>DocumentAPI.ChartLabelsDetectionMode_Unknown</code>	Неопределенный тип.
<code>DocumentAPI.ChartLabelsDetectionMode_FirstRow</code>	Метка на первой строке.
<code>DocumentAPI.ChartLabelsDetectionMode_FirstColumn</code>	Метка на первой колонке.
<code>DocumentAPI.ChartLabelsDetectionMode_NoLabels</code>	Не отрисовывать метки.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
local charts = tbl:getCharts()  
local chart = charts:getChart(0)  
local chartLabels = chart:getChartLabels()  
print(chartLabels.categoriesMode, chartLabels.seriesNameMode)
```

4.1.4 Таблица DocumentAPI.ChartLabelsInfo

Таблица `DocumentAPI.ChartLabelsInfo` описывает настройки автоматического определения меток диаграммы. Инициализируется конструктором, в который передаются параметры:

- `categoriesMode` – режим автоматического определения меток для категорий, тип `DocumentAPI.ChartLabelsDetectionMode`;
- `seriesNameMode` – режим автоматического определения меток для серий, тип `DocumentAPI.ChartLabelsDetectionMode`;
- `oneColumnRow` – передается `true`, если диапазон диаграммы содержит только одну строку или одну колонку.

Описание полей таблицы представлено в таблице 31.

Таблица 31 – Описание полей таблицы DocumentAPI.ChartLabelsInfo

Поле	Описание	Тип
categoriesMode	Режим автоматического определения меток для категорий	DocumentAPI.ChartLabelsDetectionMode
seriesNameMode	Режим автоматического определения меток для серий	DocumentAPI.ChartLabelsDetectionMode
isOneColumnRowChart	Поле содержит true, если диапазон диаграммы содержит только одну строку или одну колонку	Boolean

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local chart = charts:getChart(0)
local chartLabelsInfo = chart:getChartLabels()
print(chartLabelsInfo.categoriesMode, chartLabelsInfo.seriesNameMode,
chartLabelsInfo.isOneColumnRowChart)
```

4.1.5 Таблица DocumentAPI.ChartRangeInfo

Таблица DocumentAPI.ChartRangeInfo описывает серию диаграммы.

Инициализируется конструктором, в который передаются следующие параметры:

- tableRangeInfo – диапазон ячеек, тип DocumentAPI.TableRangeInfo;
- color – цвет серии диаграммы, тип DocumentAPI.ColorRGBA;
- hidden – видимость серии, тип Boolean;
- rangeType – тип диапазона исходных данных диаграммы, тип DocumentAPI.ChartRangeType.

Описание полей таблицы представлено в таблице 32.

Таблица 32 – Описание полей таблицы DocumentAPI.ChartRangeInfo

Поле	Описание	Тип
tableRangeInfo	Исходный диапазон ячеек для серии	DocumentAPI.TableRangeInfo
rangeColor	Цвет для отрисовки серии	DocumentAPI.ColorRGBA
isHidden	Задаёт видимость серии диаграммы	Boolean
rangeType	Тип диапазона диаграммы	DocumentAPI.ChartRangeType

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)
print(rangeInfo.tableRangeInfo, rangeInfo.rangeColor, rangeInfo.isHidden,
rangeInfo.rangeType)
```

4.1.6 Таблица DocumentAPI.ChartRangeType

Таблица DocumentAPI.ChartRangeType описывает тип диапазона исходных данных диаграммы. Описание полей таблицы представлено в таблице 33.

Таблица 33 – Описание полей таблицы DocumentAPI.ChartRangeType

Поле	Описание
DocumentAPI.ChartRangeType_Series	Серии
DocumentAPI.ChartRangeType_SeriesName	Имена серий
DocumentAPI.ChartRangeType_Categories	Категории
DocumentAPI.ChartRangeType_DataPoint	Разметка данных

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)

rangeTypes = {"Series", "SeriesName", "Categories", "DataPoint" }
print(rangeTypes[rangeInfo.rangeType + 1])
```

4.1.7 Таблица DocumentAPI.ChartSeriesDirectionType

Таблица DocumentAPI.ChartSeriesDirectionType описывает направление серий диаграмм. Описание полей таблицы представлено в таблице 34.

Таблица 34 – Описание полей таблицы DocumentAPI.ChartSeriesDirectionType

Поле	Описание
DocumentAPI.ChartSeriesDirectionType_Unknown	Неопределенный тип
DocumentAPI.ChartSeriesDirectionType_ByRow	Серии направлены по строкам
DocumentAPI.ChartSeriesDirectionType_ByColumn	Серии направлены по колонкам

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getDirectionType())
```

4.1.8 Таблица DocumentAPI.ChartType

Таблица DocumentAPI.ChartType описывает все поддерживаемые типы диаграмм. Описание полей таблицы представлено в таблице 35.

Таблица 35 – Описание полей таблицы DocumentAPI.ChartType

Поле	Описание
DocumentAPI.ChartType_Unknown	Неопределенный тип.
DocumentAPI.ChartType_Bar	Линейчатая диаграмма с группировкой.
DocumentAPI.ChartType_BarStacked	Линейчатая диаграмма с накоплением.
DocumentAPI.ChartType_BarPercentStacked	Линейчатая нормированная диаграмма с накоплением.
DocumentAPI.ChartType_Column	Гистограмма с группировкой.
DocumentAPI.ChartType_ColumnStacked	Гистограмма с накоплением.
DocumentAPI.ChartType_ColumnPercentStacked	Нормированная гистограмма с накоплением.
DocumentAPI.ChartType_Line	Стандартный график.
DocumentAPI.ChartType_LineStacked	График с накоплением.
DocumentAPI.ChartType_LinePercentStacked	Нормированный график с накоплением.
DocumentAPI.ChartType_LineWithMarker	Стандартный график с маркерами.
DocumentAPI.ChartType_LineWithMarkerStacked	График с накоплением и маркерами.
DocumentAPI.ChartType_LineWithMarkerPercentStacked	Нормированный график с накоплением и маркерами.
DocumentAPI.ChartType_Area	Стандартная диаграмма с областями.
DocumentAPI.ChartType_AreaStacked	Диаграмма с областями с накоплением.
DocumentAPI.ChartType_AreaPercentStacked	Нормированная диаграмма с областями с накоплением.
DocumentAPI.ChartType_PieAreaPercentStacked	Круговая диаграмма.
DocumentAPI.ChartType_PieExploded	Круговая диаграмма с отделенными

Поле	Описание
	секторами.
DocumentAPI.ChartType_Scatter	Диаграмма рассеяния.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getType())
```

4.2 Именованные диапазоны

Именованный диапазон – это диапазон ячеек или формула, которым присвоено имя. Преимуществом именованного диапазона является его информативность. Именованные диапазоны упрощают работу с ячейками, также их удобно использовать при работе с формулами. На данный момент доступна возможность работы с именованными диапазонами, представляющими собой ссылки на диапазоны ячеек. Доступ к именованным диапазонам осуществляется посредством методов `Document:getNameExpressions()` и `Table:getNameExpressions()` (см. Рисунок 1195).

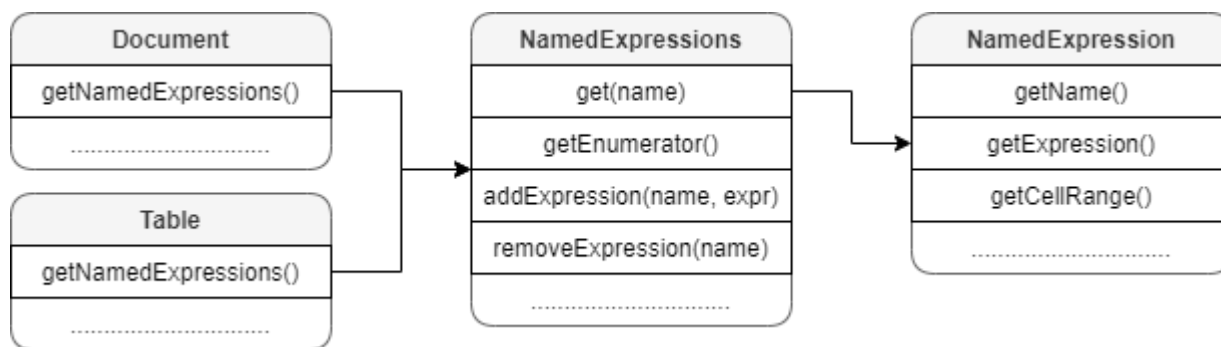


Рисунок 1195 – Таблицы для работы с именованными диапазонами

4.2.1 Таблица DocumentAPI.NamedExpressions

Таблица для представления списка именованных диапазонов. Может быть получена с помощью методов `Document:getNameExpressions()`, `Table:getNameExpressions()`.

4.2.1.1 Метод `NamedExpressions:get`

Возвращает именованный диапазон `NamedExpression` по имени `name`, если он существует.

Пример:

```
local namedExpression = namedExpressions:get("Продажи")
if (namedExpression) then
    print(namedExpression:getName()) -- Продажи
else
    print("No named expression was found")
end
```

4.2.1.2 Метод `NamedExpressions:enumerate`

Позволяет получить доступ ко всему списку именованных диапазонов.

Пример:

```
local namedExpressions = sheet:getNamedExpressions()
for namedExpression in namedExpressions:enumerate() do
    print(namedExpression)
end
```

4.2.1.3 Метод `NamedExpression:addExpression`

Добавляет новый диапазон в список именованных диапазонов, возвращает результат операции `NamedExpressionsValidationResult`.

Пример:

```
local expressionName = "Покупки"
local expressionValue = "=формула покупки!$E$6:$E$14"
local validationResult = namedExpressions:addExpression(expressionName,
expressionValue)
if (validationResult == DocumentAPI.NamedExpressionsValidationResult_Success)
then
    print("Named expression was added")
end
```

4.2.1.4 Метод `NamedExpressions:removeExpression`

Удаляет именованный диапазон по заданному имени, возвращает результат операции `NamedExpressionsValidationResult`.

Пример:

```
local namedExpression = namedExpressions:get(expressionName)
if (namedExpression) then
    local validationResult = namedExpressions:removeExpression(expressionName)
    if (validationResult ==
DocumentAPI.NamedExpressionsValidationResult_Success) then
        print("Named expression was removed")
    end
end
end
```

4.2.2 Таблица DocumentAPI.NamedExpression

Класс описывает структуру именованного диапазона.

Пример:

```
local namedExpressions = sheet:getNamedExpressions()
for namedExpression in namedExpressions:enumerate() do
    print(namedExpression:getName())
    print(namedExpression:getExpression())
    cellRange = namedExpression:getCellRange()
    print(cellRange:getBeginRow(), cellRange:getLastRow())
end
```

4.2.2.1 Метод NamedExpression:getName

Возвращает имя именованного диапазона. Пример см. в DocumentAPI.NamedExpression.

4.2.2.2 Метод NamedExpression:getExpression

Возвращает текст диапазона (формулы). Пример см. в DocumentAPI.NamedExpression.

4.2.2.3 Метод NamedExpression:getCellRange

Возвращает диапазон ячеек DocumentAPI.CellRange именованного диапазона. Пример см. в разделе 4.2.2.

4.2.3 Таблица DocumentAPI.NamedExpressionsValidationResult

Таблица DocumentAPI.NamedExpressionsValidationResult описывает результат операций NamedExpressions:addExpression(), NamedExpressions:removeExpression(). Описание полей таблицы представлено в таблице 36.

Таблица 36 – Описание полей таблицы DocumentAPI.NamedExpressionsValidationResult

Поле	Описание
DocumentAPI.NamedExpressionsValidationResult_Success	Операция выполнена успешно
DocumentAPI.NamedExpressionsValidationResult_WrongName	Неправильный формат имени
DocumentAPI.NamedExpressionsValidationResult_isUsedInFormula	Имя уже используется в формуле

4.3 Макрокоманды

Таблица DocumentAPI.Scripts предоставляет доступ к списку макрокоманд документа. На рисунке 1196 изображена объектная модель таблиц, относящихся к работе с макрокомандами.

Таблица DocumentAPI.Scripts предназначена для доступа к списку макрокоманд, доступна через метод document:getScripts(), таблица DocumentAPI.Scripting служит для запуска макрокоманд, доступна через DocumentAPI.createScripting(document).

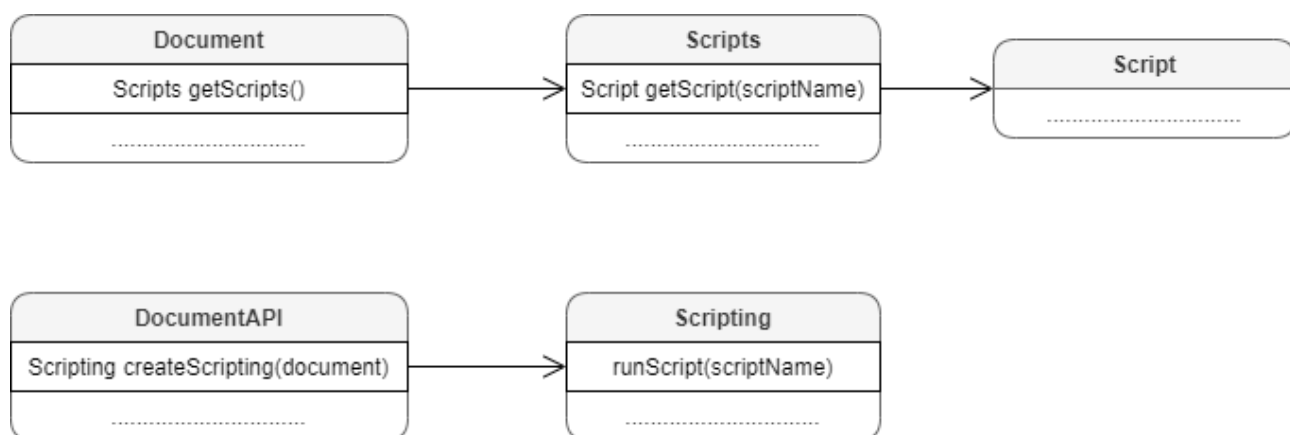


Рисунок 1196 – Объектная модель таблиц для работы с макрокомандами

4.3.1 Таблица DocumentAPI.Scripts

Таблица `DocumentAPI.Scripts` предоставляет доступ к списку макрокоманд документа. Коллекцию макрокоманд `DocumentAPI.Scripts` можно получить из документа посредством вызова метода `document:getScripts()`.

Пример:

```
local scripts = document:getScripts()
for script in scripts:enumerate() do
    print(script:getName())
    print(script:getBody())
end
```

4.3.1.1 Метод Scripts:getScript

Метод возвращает таблицу `DocumentAPI.Script`, описывающую макрокоманду. В качестве аргумента используется имя макрокоманды.

Пример:

```
local scripts = document:getScripts()
local script = scripts:getScript("Enumerate scripts for document")
print(script:getName())
```

4.3.1.2 Метод Scripts:setScript

Метод добавляет макрокоманду в текущий документ. Если макрокоманда с таким именем уже существует, будет обновлено ее содержимое.

Пример:

```
local scripts = document:getScripts()
local script_name = "Enumerate scripts for document"
local script_code = "local scripts = document:getScripts()\nfor script in\nscripts:enumerate() do\nprint(script:getName())\nend"
scripts:setScript(script_name, script_code)
```

4.3.1.3 Метод Scripts:removeScript

Метод удаляет макрокоманду из текущего документа. В качестве аргумента используется имя макрокоманды.

Пример:

```
local scripts = document:getScripts()  
scripts:removeScript("Enumerate scripts for document")
```

4.3.1.4 Метод `Scripts:enumerate`

Метод возвращает коллекцию макрокоманд для их дальнейшего перечисления.

Пример:

```
for script in document:getScripts():enumerate() do  
    print(script:getName())  
end
```

4.3.2 Таблица `DocumentAPI.Script`

Таблица `DocumentAPI.Script` предназначена для управления отдельной макрокомандой. Таблица содержит поля `Name` и `Body`.

4.3.2.1 Таблица `DocumentAPI.Scripting`

Таблица `DocumentAPI.Scripting` может быть получена путем вызова `DocumentAPI.createScripting()` и содержит метод `runScript`, который используется для запуска макрокоманды.

4.3.2.1.1 Метод `Scripting:runScript`

Метод предназначен для запуска макрокоманды, хранящейся в документе. В качестве аргумента передается имя макрокоманды.

Пример:

```
scripting = DocumentAPI.createScripting(document)  
scripting:runScript("Enumerate scripts for document")
```

4.3.2.2 Метод `Script:getName`

Метод возвращает имя макрокоманды.

Пример:

```
local scripts = document:getScripts()  
local sc = scripts:getScript("Enumerate scripts for document")  
print(sc:getName())
```

4.3.2.3 Метод Script:setName

Метод устанавливает имя для макрокоманды.

Пример:

```
local scripts = document:getScripts()  
local sc = scripts:getScript("Enumerate scripts for document")  
sc:setName("Enumerate scripts for current document")
```

4.3.2.4 Метод Script:getBody

Метод возвращает текст макрокоманды в виде строки.

Пример:

```
local scripts = document:getScripts()  
local script = scripts:getScript("Enumerate scripts for document")  
local scriptBody = script:getBody()  
print(scriptBody)
```

4.3.2.5 Метод Script:setBody

Метод устанавливает текст макрокоманды, полностью заменяя уже имеющийся текст.

Пример:

```
local scripts = document:getScripts()  
local script = scripts:getScript("Enumerate scripts for document")  
script:setBody("local scripts = document:getScripts()\nfor script in\nscripts:enumerate() do\nprint(script:getName())\nend")
```

4.3.3 Функция DocumentAPI.createScripting

Функция DocumentAPI.createScripting возвращает таблицу

DocumentAPI.Scripting. В качестве параметра используется текущий документ.

Пример:

```
scripting = DocumentAPI.createScripting(document)
```

4.4 Разделы (секции) документа

На рисунке 1197 изображена объектная модель таблиц, относящихся к работе с секциями текстового документа.

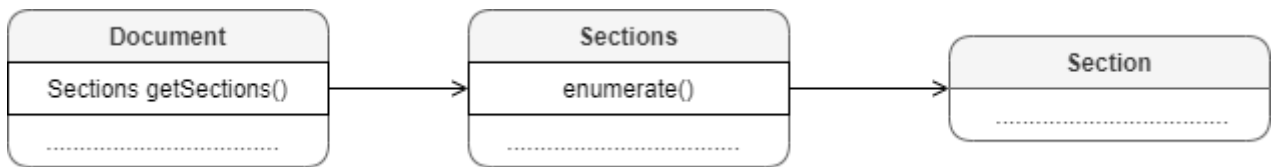


Рисунок 1197 – Объектная модель таблиц для работы с секциями

Секция в текстовом документе – это раздел, который содержит страницы с одинаковыми параметрами, а также одинаковыми верхними и нижними колонтитулами.

Доступ к секциям текстового документа может быть осуществлен одним из следующих способов:

- получение таблицы DocumentAPI.Sections с помощью вызова document:getSections();
- перечисление всех доступных секций DocumentAPI.Section с помощью вызова document:enumerateSections();
- получение секции DocumentAPI.Section вызовом метода Block.getSection() для блока, который входит в секцию.

Примеры:

```
local sections = document:getSections()
for section in sections:enumerate() do
    local properties = section:getPageProperties()
    print(properties.width)
    print(properties.height)
end

local sections = document:enumerateSections()
for section in sections do
    print(section:getPageProperties().width)
end

local section = document:getBlocks():getBlock(0):getSection()
local properties = section:getPageProperties()
```

4.4.1 Таблица DocumentAPI.Sections

Таблица DocumentAPI.Sections представляет интерфейс для доступа к коллекции секций документа. Описание секции см. в разделе 4.4.2.

4.4.1.1 Метод Sections:enumerate

Метод возвращает коллекцию секций документа.

Пример:

```
local sections = document:getSections()
for section in sections:enumerate() do
    local properties = section:getPageProperties()
    print(properties.width)
    print(properties.height)
end
```

4.4.2 Таблица DocumentAPI.Section

Таблица DocumentAPI.Section представляет собой раздел в документе.

4.4.2.1 Метод Section:setPageProperties

Метод устанавливает параметры DocumentAPI.PageProperties страниц, находящихся в разделе.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local properties = section:getPageProperties()
properties.width = 100
properties.height = 200
properties.margins.left = 10
section:setPageProperties(properties)
```

4.4.2.2 Метод Section:getPageProperties

Метод возвращает параметры страниц раздела DocumentAPI.PageProperties.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()  
local properties = section:getPageProperties()  
print(properties.width)  
print(properties.height)  
print(properties.margins.left)  
print(properties.margins.top)
```

4.4.2.3 Метод Section:setPageOrientation

Метод задает ориентацию страниц раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()  
section:setPageOrientation(DocumentAPI.PageOrientation_Landscape)  
local orientation = section:getPageOrientation()  
print(orientation)
```

4.4.2.4 Метод Section:getPageOrientation

Метод возвращает ориентацию страниц раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()  
local orientation = section:getPageOrientation()  
print(orientation)
```

4.4.2.5 Метод Section:getRange

Метод возвращает диапазон DocumentAPI.Range в документе, соответствующий данному разделу.

Пример:

```
local sections = document:enumerateSections()  
for section in sections do  
    print(section:getRange():extractText())  
end
```

4.4.2.6 Метод `Section:getHeaders`

Метод возвращает коллекцию `DocumentAPI.HeadersFooters` верхних колонтитулов данного раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    if (header:getType() == DocumentAPI.HeaderFooterType_Header) then
        print("Header") else print("Footer")
    end
end
```

4.4.2.7 Метод `Section:getFooters`

Метод возвращает коллекцию `DocumentAPI.HeadersFooters` нижних колонтитулов данного раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local footers = section:getFooters()
for footer in footers:enumerate() do
    if (footer:getType() == DocumentAPI.HeaderFooterType_Header) then
        print("Header") else print("Footer")
    end
end
```

4.4.3 Таблица `DocumentAPI.HeadersFooters`

Таблица `DocumentAPI.HeadersFooters` представляет коллекцию верхних и нижних колонтитулов раздела (см. Рисунок 1198). Доступ к колонтитулам осуществляется посредством методов `Section:getHeaders()`, `Section:getFooters()`.

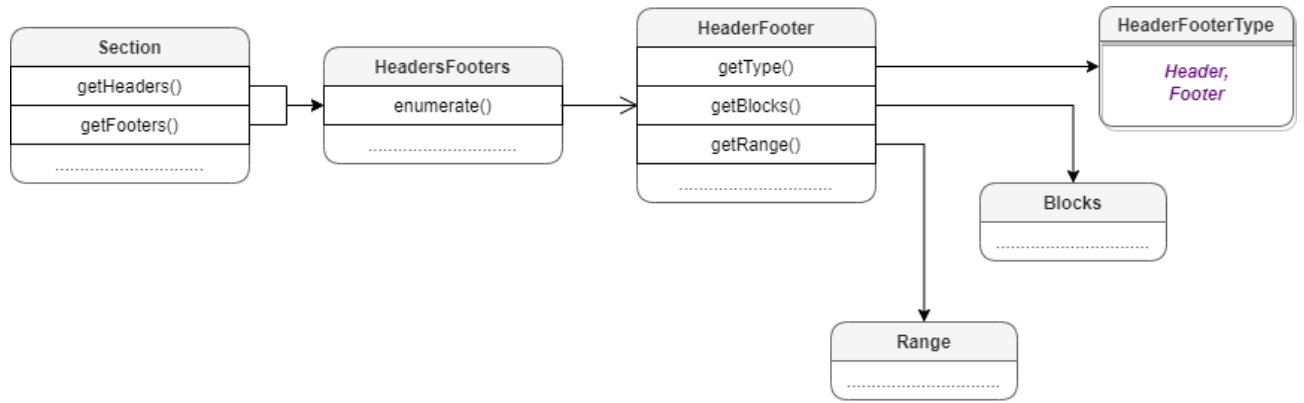


Рисунок 1198 – Таблицы колонтитулов

4.4.3.1 Метод HeadersFooters:enumerate

Метод возвращает коллекцию колонтитулов.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    if (header:getType() == DocumentAPI.HeaderFooterType_Header) then
        print("Header") else print("Footer")
    end
end
end
```

4.4.4 Таблица DocumentAPI.HeaderFooter

Таблица DocumentAPI.HeaderFooter определяет колонтитул текстового документа.

4.4.4.1 Метод HeaderFooter:getType

Метод предоставляет информацию о типе колонтитула (DocumentAPI.HeaderFooterType).

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    if (header:getType() == DocumentAPI.HeaderFooterType_Header) then
        print("Header") else print("Footer")
    end
end
end
```

4.4.4.2 Метод `HeaderFooter:getBlocks`

Метод предоставляет доступ к блокам (`DocumentAPI.Blocks`), которые содержатся в колонтитуле.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    for block in header:getBlocks():enumerate() do
        print(block:getRange():extractText())
    end
end
end
```

4.4.4.3 Метод `HeaderFooter:getRange`

Метод предоставляет диапазон (`DocumentAPI.Range`) с содержанием верхнего или нижнего колонтитулов.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    print(header:getRange():extractText())
end
```

4.4.5 Таблица `DocumentAPI.HeaderFooterType`

Типы колонтитулов представлены в таблице 37.

Таблица 37 – Типы колонтитулов

Наименование константы	Описание
<code>DocumentAPI.HeaderFooterType_Header</code>	Верхний колонтитул
<code>DocumentAPI.HeaderFooterType_Footer</code>	Нижний колонтитул

4.4.6 Таблица `DocumentAPI.PageOrientation`

Типы ориентации страницы представлены в таблице 38. Данная константа может быть использована для получения / установки ориентации страниц для секции или документа.

Таблица 38 – Типы ориентации страницы

Наименование константы	Описание
DocumentAPI.PageOrientation_Landscape	Альбомная ориентация страницы
DocumentAPI.PageOrientation_Portrait	Портретная ориентация страницы

Примеры:

```
local section = document:getBlocks():getBlock(0):getSection()
section:setPageOrientation(DocumentAPI.PageOrientation_Landscape)
print(section:getPageOrientation())

local section =
document:setPageOrientation(DocumentAPI.PageOrientation_Portrait)
local section = document:getBlocks():getBlock(0):getSection()
print(section:getPageOrientation())
```

4.4.7 Таблица DocumentAPI.Insets

Таблица DocumentAPI.Insets предназначена для задания полей, например, страницы. DocumentAPI.Insets представлено в таблице 39. Используется в поле margins таблицы DocumentAPI.PageProperties.

Таблица 39 – Описание полей таблицы DocumentAPI.Insets

Поле	Тип	Описание
DocumentAPI.Insets.left	number	Левая граница поля
DocumentAPI.Insets.top	number	Верхняя граница поля
DocumentAPI.Insets.right	number	Правая граница поля
DocumentAPI.Insets.bottom	number	Нижняя граница поля

Пример:

```
local insets = DocumentAPI.Insets()
insets.left = 10.0
print(insets.left)
```

4.4.8 Таблица DocumentAPI.PageProperties

Таблица DocumentAPI.PageProperties предоставляет такие свойства страницы как высота, ширина, размеры полей. Описание полей приведено в таблице 40. Используется в

```
Document.setPageProperties(),  
Section.setPageProperties().
```

```
Section.getPageProperties(),
```

Таблица 40 – Описание полей таблицы DocumentAPI.PageProperties

Поле	Описание
DocumentAPI.PageProperties.height	Высота страницы
DocumentAPI.PageProperties.width	Ширина страницы
DocumentAPI.PageProperties.margins	Поля страницы, тип – Insets

Примеры:

```
local section = document:getBlocks():getBlock(0):getSection()  
local pageProperties = section:getPageProperties()  
pageProperties.width = 100  
pageProperties.height = 200  
pageProperties.margins.left = 10  
section:setPageProperties(pageProperties)  
  
local pageProperties = DocumentAPI.PageProperties()  
pageProperties.width = 100  
pageProperties.height = 200  
document:setPageProperties(pageProperties)  
  
local pageProperties = DocumentAPI.PageProperties(100, 200)  
document:setPageProperties(pageProperties)
```

4.4.8.1 Метод PageProperties: __eq

Метод позволяет использовать оператор сравнения == для определения эквивалентности содержимого двух структур DocumentAPI.PageProperties.

Пример:

```
local pageProperties1 = DocumentAPI.PageProperties(100, 200)  
document:setPageProperties(pageProperties1)  
  
local pageProperties2 = DocumentAPI.PageProperties(100, 200)  
document:setPageProperties(pageProperties2)  
  
print(pageProperties1:__eq(pageProperties2)) - true
```

4.5 Блоки, параграфы

4.5.1 Таблица DocumentAPI.Blocks

Таблица `DocumentAPI.Blocks` обеспечивает доступ к блокам `DocumentAPI.Block` документа или диапазона документа (см. Рисунок 1199). Таблица `DocumentAPI.Blocks` может быть получена вызовом метода `Document:getBlocks` или `HeaderFooter:getBlocks`.

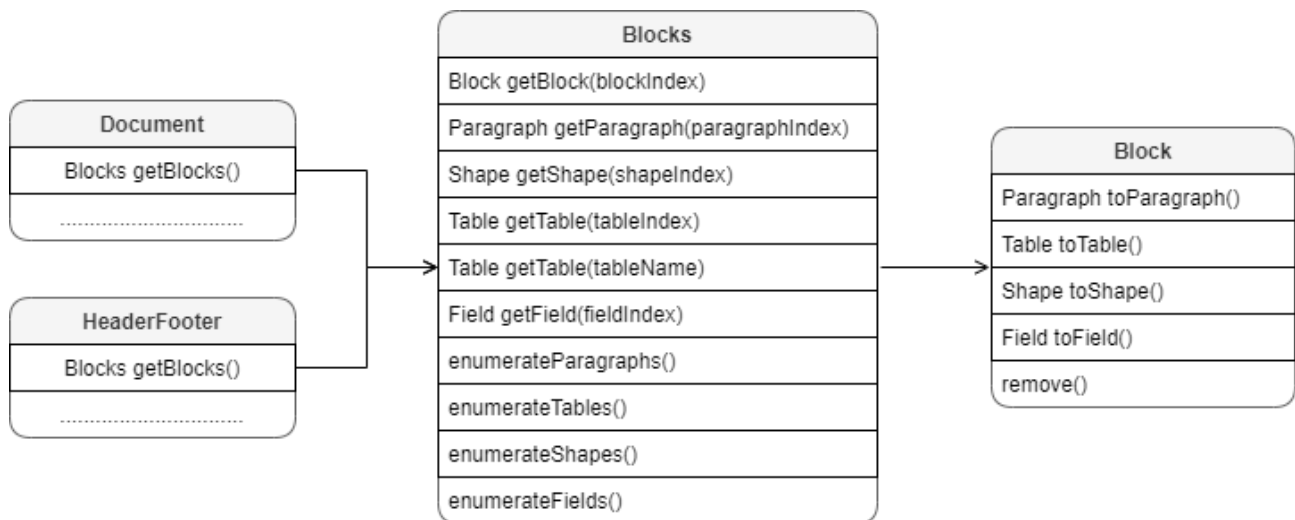


Рисунок 1199 – Объектная модель таблицы `DocumentAPI.Blocks`

4.5.1.1 Метод `Blocks:getBlock`

Возвращает объект типа `DocumentAPI.Block` по заданному индексу. Нумерация индексов начинается с нуля.

Пример:

```
local block = document:getBlocks():getBlock(0)
```

4.5.1.2 Метод `Blocks:getParagraph`

Возвращает абзац с указанным индексом. Нумерация индексов начинается с нуля.

Пример:

```
local para = document:getBlocks():getParagraph(0)
```

4.5.1.3 Метод `Blocks:getTable`

Для табличного документа возвращает лист (`worksheet`), для текстового документа возвращает таблицу. Параметры поиска – индекс или имя таблицы. Нумерация листов начинается с нуля.

Пример:

```
local table = document:getBlocks():getTable(0)
```

В качестве параметра метода также можно указать имя таблицы.

Пример:

```
local table = document:getBlocks():getTable("Sheet1")
```

4.5.1.4 Метод `Blocks:getShape`

Возвращает фигуру `DocumentAPI.Shape` по заданному индексу.

Пример:

```
local shape = document:getBlocks():getShape(0)
```

4.5.1.5 Метод `Blocks:getField`

Возвращает объект типа `DocumentAPI.Field` по заданному индексу.

Пример:

```
local field = document:getBlocks():getField(0)
```

4.5.1.6 Метод `Blocks:enumerate`

Позволяет перечислить объекты типа `DocumentAPI.Block`.

Пример:

```
for block in document:getBlocks():enumerate() do  
    print(block:getRange():extractText())  
end
```

4.5.1.7 Метод `Blocks:enumerateParagraphs`

Позволяет реализовать перечисление абзацев `DocumentAPI.Paragraph`.

Пример:

```
for paragraph in document:getBlocks():enumerateParagraphs() do
  print(paragraph:getRange():extractText())
end
```

4.5.1.8 Метод `Blocks:enumerateTables`

Позволяет перечислить объекты типа `DocumentAPI.Table`.

Пример:

```
for table in document:getBlocks():enumerateTables() do
  print(table:getName())
end
```

4.5.1.9 Метод `Blocks:enumerateShapes`

Позволяет перечислить объекты типа `DocumentAPI.Shape`.

Пример:

```
for shape in document:getBlocks():enumerateShapes() do
  print(shape:getShapeProperties())
end
```

4.5.1.10 Метод `Blocks:enumerateFields`

Позволяет перечислить объекты типа `DocumentAPI.Field`.

Пример:

```
for field in document:getBlocks():enumerateFields() do
  print(field:getRange():extractText())
end
```

4.5.2 Таблица `DocumentAPI.Block`

Таблица `DocumentAPI.Block` является базовой для всех блоков документа. От нее наследуются таблицы `Paragraph`, `Table`, `Shape`, `Field` (см. Рисунок 1200).

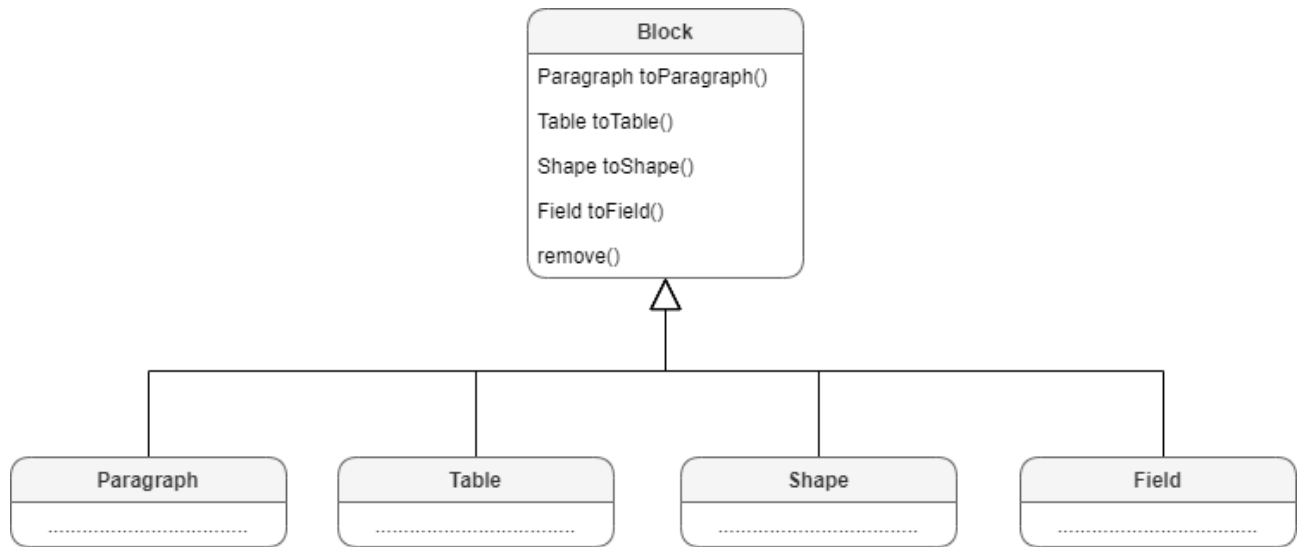


Рисунок 1200 – Объектная модель таблицы DocumentAPI.Block

4.5.2.1 Методы toParagraph, toTable, toShape, toField

Преобразует объект DocumentAPI.Block в объект соответствующего типа.

Пример:

```
local paragraph = document:getBlocks():getBlock(0):toParagraph()
local para_props = paragraph:getParagraphProperties()
```

4.5.2.2 Метод Block.getRange

Возвращает диапазон DocumentAPI.Range, в котором содержится данный блок.

Пример:

```
local range = document:getBlocks():getBlock(0):getRange()
print(range:extractText())
```

4.5.2.3 Метод Block.remove

Удаляет блок из документа. Текущий экземпляр объекта DocumentAPI.Block становится недействительным.

Пример:

```
document:getBlocks():getBlock(0):remove()
```

4.5.2.4 Метод `Block.getSection`

Метод возвращает раздел `DocumentAPI.Section`, содержащий блок.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local pageProperties = section:getPageProperties()
```

4.5.3 Таблица `DocumentAPI.Field`

Таблица `Field` предназначена для реализации некоторых полей, например, содержания.

4.5.4 Таблица `DocumentAPI.Paragraphs`

Таблица `DocumentAPI.Paragraphs` предоставляет доступ к коллекции абзацев типа `DocumentAPI.Paragraph` (см. Рисунок 1201). Коллекция абзацев может быть получена из таблицы `DocumentAPI.Range` посредством использования вызова `Range:getParagraphs()`.

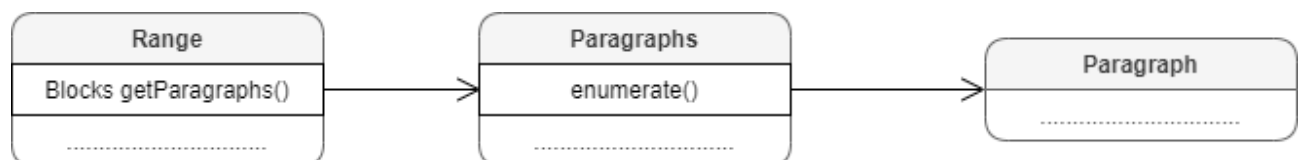


Рисунок 1201 – Объектная модель для работы со списком абзацев

Пример для текстового документа:

```
local paragraphs = document:getRange():getParagraphs()
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local paragraphs = cell:getRange():getParagraphs()
```

4.5.4.1 Метод `Paragraphs:setListSchema`

Метод устанавливает тип маркированного или нумерованного списка `DocumentAPI.ListSchema`. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()  
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
```

4.5.4.2 Метод Paragraphs:setListLevel

Метод устанавливает глубину вложенности элемента списка. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()  
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)  
paragraphs:setListLevel(1)
```

4.5.4.3 Метод Paragraphs:increaseListLevel

Метод увеличивает уровень списка на единицу. В случае, если максимальный уровень уже установлен, увеличения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()  
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)  
paragraphs:increaseListLevel()
```

4.5.4.4 Метод Paragraphs:decreaseListLevel

Метод уменьшает уровень списка на единицу. В случае, если минимальный уровень уже установлен, уменьшения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()  
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)  
paragraphs:decreaseListLevel()
```

4.5.4.5 Метод Paragraphs:enumerate

Метод позволяет перечислить коллекцию абзацев.

Пример для текстового документа:

```
local paragraphs = document:getRange():getParagraphs()
for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    print(para_props.alignment)
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    print(para_props.alignment)
end
```

4.5.5 Таблица DocumentAPI.Paragraph

Таблица DocumentAPI.Paragraph предоставляет доступ к свойствам абзаца (см. Рисунок 1202).

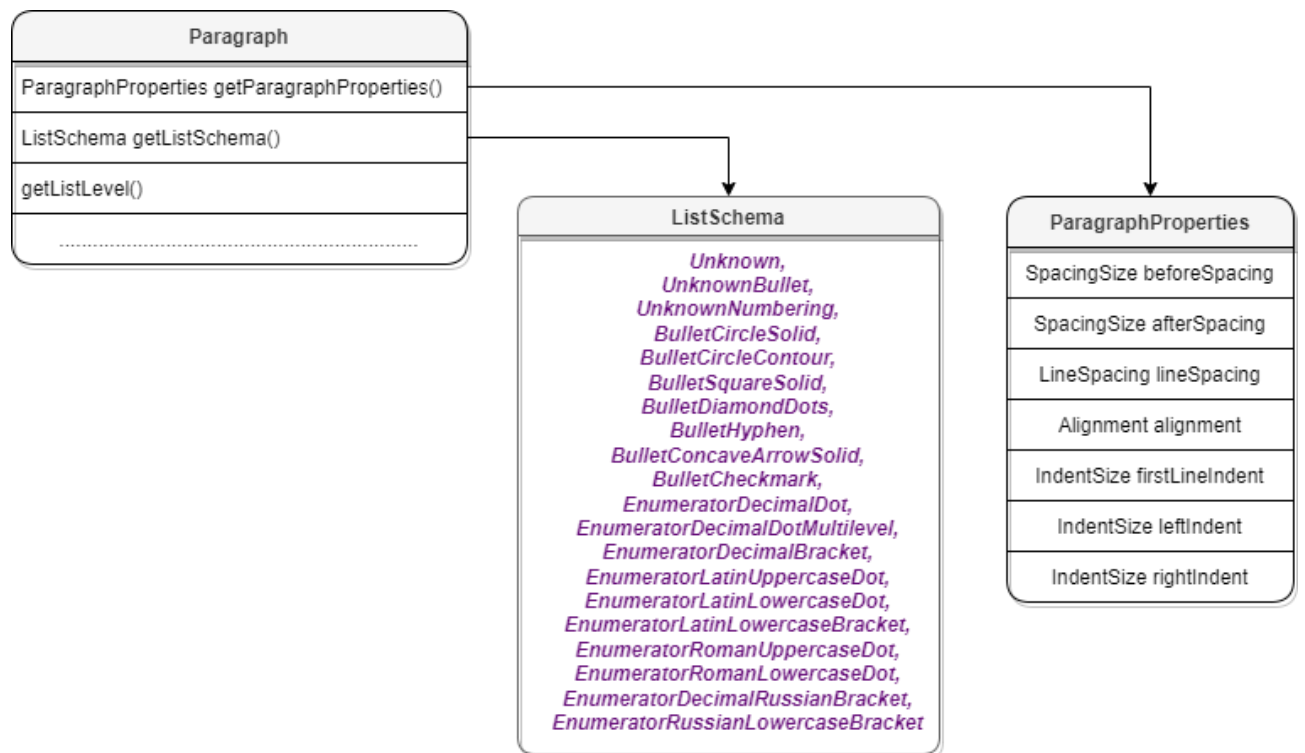


Рисунок 1202 – Объектная модель таблиц для работы со свойствами параграфа

4.5.5.1 Метод Paragraph:getParagraphProperties

Метод предоставляет доступ к таблице свойств форматирования абзаца DocumentAPI.ParagraphProperties, таким как выравнивание текста, межстрочные интервалы, отступы и т. д.

Пример для текстового документа:

```

local para = document:getBlocks():getParagraph(0)
local para_props = para:getParagraphProperties()
print(para_props.afterSpacing)

```

Пример для табличного документа:

```

local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    print(para_props.afterSpacing)
end

```

4.5.5.2 Метод Paragraph:setParagraphProperties

Метод предназначен для обновления таблицы свойств форматирования абзаца DocumentAPI.ParagraphProperties.

Пример для текстового документа:

```
local para = document:getBlocks():getParagraph(0)
local para_props = para:getParagraphProperties()
para_props.alignment = DocumentAPI.Alignment_Right
para:setParagraphProperties(para_props)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    para_props.alignment = DocumentAPI.Alignment_Right
    para:setParagraphProperties(para_props)
end
```

4.5.5.3 Метод Paragraph:getListSchema

Метод возвращает схему форматирования абзаца DocumentAPI.ListSchema либо значение nil, если схема нумерации не установлена для абзаца. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
local schema = paragraph:getListSchema()
```

4.5.5.4 Метод Paragraph:setListSchema

Метод позволяет установить тип маркированного или нумерованного списка DocumentAPI.ListSchema. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
```

4.5.5.5 Метод Paragraph:getListLevel

Метод позволяет получить глубину вложенности элемента списка. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
local level = paragraph:getListLevel()
```

4.5.5.6 Метод Paragraph:setListLevel

Метод позволяет установить глубину вложенности элемента списка.

Значение может быть равным nil, если схема нумерации не установлена для абзаца. В этом случае будет установлено минимальное значение. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
local level = paragraph:setListLevel(1)
```

4.5.5.7 Метод Paragraph:increaseListLevel

Метод позволяет увеличить на единицу глубину вложенности элемента списка. В случае, если максимальный уровень уже установлен, увеличения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
paragraph:increaseListLevel()
```

4.5.5.8 Метод Paragraph:decreaseListLevel

Метод позволяет уменьшить на единицу глубину вложенности элемента списка. В случае, если минимальный уровень уже установлен, уменьшения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
paragraph:decreaseListLevel()
```

4.5.6 Таблица DocumentAPI.ListSchema

Типы схем форматирования списков, которые могут быть применены к абзацам текста представлены в таблице 41. Данные константы используются в методах Paragraph:getListSchema(), Paragraph:setListSchema().

Таблица 41 – Типы схем форматирования списков

Наименование константы	Описание
DocumentAPI.ListSchema_Unknown	Схема не определена
DocumentAPI.ListSchema_UnknownBullet	Список без маркера
DocumentAPI.ListSchema_UnknownNumbering	Нумерация без номера
DocumentAPI.ListSchema_BulletCircleSolid	Список с маркерами в виде заполненного круга
DocumentAPI.ListSchema_BulletCircleContour	Список с маркерами в виде окружности
DocumentAPI.ListSchema_BulletSquareSolid	Список с маркерами в виде квадрата
DocumentAPI.ListSchema_BulletDiamondDots	Список с маркерами в виде четырех ромбов
DocumentAPI.ListSchema_BulletHyphen	Список с маркерами в виде дефиса
DocumentAPI.ListSchema_BulletConcaveArrowSolid	Список с маркерами в виде вогнутой стрелки.
DocumentAPI.ListSchema_BulletCheckmark	Список с маркерами в виде галочки.
DocumentAPI.ListSchema_EnumeratorDecimalDot	Десятичная нумерация с точкой.
DocumentAPI.ListSchema_EnumeratorDecimalDotMultiLevel	Многоуровневая десятичная нумерация с точкой
DocumentAPI.ListSchema_EnumeratorDecimalBracket	Десятичная нумерация со скобкой
DocumentAPI.ListSchema_EnumeratorLatinUppercaseDot	Нумерация латинскими прописными буквами с точкой

Наименование константы	Описание
DocumentAPI.ListSchema_EnumeratorLatinLowercaseDot	Нумерация латинскими строчными буквами с точкой
DocumentAPI.ListSchema_EnumeratorLatinLowercaseBracket	Нумерация латинскими строчными буквами со скобкой
DocumentAPI.ListSchema_EnumeratorRomanUppercaseDot	Нумерация римскими прописными цифрами с точкой
DocumentAPI.ListSchema_EnumeratorRomanLowercaseDot	Нумерация римскими строчными цифрами с точкой
DocumentAPI.ListSchema_EnumeratorDecimalRussianBracket	Десятичная нумерация через запятую со скобкой
DocumentAPI.ListSchema_EnumeratorRussianLowercaseBracket	Нумерация с русскими строчными буквами со скобкой

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
```

4.5.7 Таблица DocumentAPI.ParagraphProperties

Таблица DocumentAPI.ParagraphProperties предназначена для управления свойствами форматирования (см. Рисунок 1203). Таблица DocumentAPI.ParagraphProperties используется в методах Paragraph:getParagraphProperties и Paragraph:setParagraphProperties.

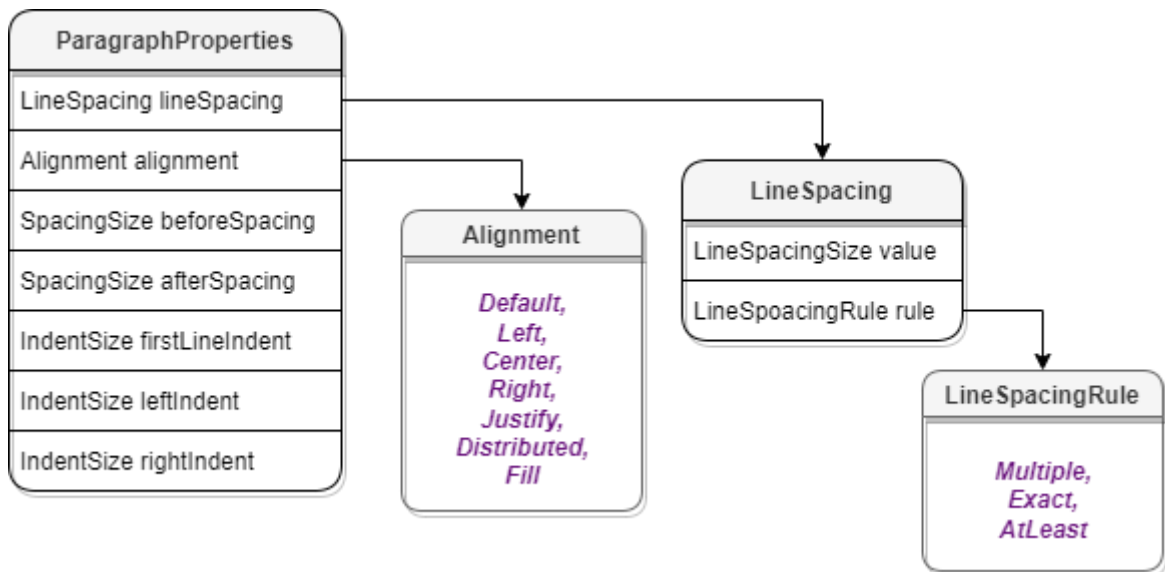
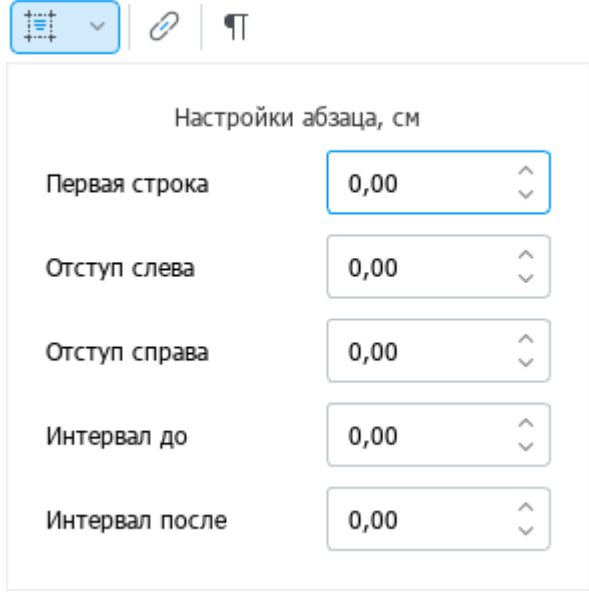


Рисунок 1203 – Объектная модель таблиц для работы со свойствами параграфа

Описание полей таблицы DocumentAPI.ParagraphProperties представлено в таблице 42.

Таблица 42 – Описание полей таблицы DocumentAPI.ParagraphProperties

Поле	Описание
ParagraphProperties.beforeSpacing	<p>Установка величины расстояния до абзаца.</p> <p>При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне Настройки абзаца, (см. рисунок выше), в поле Интервал до (подробнее см. в приложении «Руководство оператора. Часть 1. Редактор текста»).</p>
ParagraphProperties.afterSpacing	<p>Установка величины расстояния после абзаца.</p> <p>При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне Настройки абзаца, в поле Интервал после (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).</p>  <p>The screenshot shows a dialog box titled 'Настройки абзаца, см' (Paragraph Settings, cm). It contains five input fields with up and down arrows: 'Первая строка' (First line), 'Отступ слева' (Left indent), 'Отступ справа' (Right indent), 'Интервал до' (Interval before), and 'Интервал после' (Interval after). The 'Интервал после' field is highlighted with a blue border and contains the value '0,00'.</p>
ParagraphProperties.lineSpacing	<p>Расстояние между строк одного абзаца (межстрочный интервал), LineSpacing.</p>
ParagraphProperties.alignment	<p>Выравнивание текстового фрагмента по горизонтали. Список допустимых значений находится в разделе 4.5.10.</p>
ParagraphProperties.firstLineIndent	<p>Расстояние от левого поля документа до первой строки в абзаце с учетом отступа слева.</p> <p>При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне Настройки абзаца, (см. рисунок выше), в поле Первая строка (подробнее см. в</p>

Поле	Описание
	приложении «Руководство оператора. Часть 1. Редактор текста»).
ParagraphProperties.leftIndent	Расстояние от левого поля документа до абзаца (отступ слева). При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне Настройки абзаца , (см. рисунок выше), в поле Отступ слева (подробнее см. в приложении «Руководство оператора. Часть 1. Редактор текста»).
ParagraphProperties.rightIndent	Расстояние от правого поля документа до абзаца. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне Настройки абзаца , (см. рисунок выше), в поле Отступ справа (подробнее см. в приложении «Руководство оператора. Часть 1. Редактор текста»).

Пример для текстового документа:

```

local para = document:getBlocks():getParagraph(0)
local para_props = para:getParagraphProperties()
--
para_props.afterSpacing = 28.3 -- значение соответствует 1 см
para_props.beforeSpacing = 28.3 -- значение соответствует 1 см
para_props.alignment = DocumentAPI.Alignment_Center
para_props.firstLineIndent = 28.3 -- значение соответствует 1 см
para_props.leftIndent = 28.3 -- значение соответствует 1см
para_props.lineSpacing = DocumentAPI.LineSpacing(5.0,
DocumentAPI.LineSpacingRule_Multiple)
para_props.rightIndent = 28.3 -- значение соответствует 1см
--
para:setParagraphProperties(para_props)

```

Пример для табличного документа:

```

local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()

```



```

para_props.afterSpacing = 28.3 -- значение соответствует 1 см
para_props.beforeSpacing = 28.3 -- значение соответствует 1 см
para_props.alignment = DocumentAPI.Alignment_Center
para_props.firstLineIndent = 28.3 -- значение соответствует 1 см
para_props.leftIndent = 28.3 -- значение соответствует 1см
para_props.lineSpacing = DocumentAPI.LineSpacing(5.0,
DocumentAPI.LineSpacingRule_Multiple)
para_props.rightIndent = 28.3 -- значение соответствует 1см
para:setParagraphProperties(para_props)
end

```

4.5.8 Таблица DocumentAPI.LineSpacing

Таблица DocumentAPI.LineSpacing задает межстрочный интервал абзаца. Поля таблицы приведены в таблице 43. Для управления значением межстрочного интервала используются значения, представленные в разделе 4.5.9.

Таблица 43 – Параметры межстрочного интервала

Поле	Описание
DocumentAPI.LineSpacing.value	Значение межстрочного интервала.
DocumentAPI.LineSpacing.rule	Правило формирования межстрочного интервала DocumentAPI.LineSpacingRule.

Пример:

```

-- Конструктор
local lineSpacing = DocumentAPI.LineSpacing(1.5,
DocumentAPI.LineSpacingRule_Multiple)
-- Обращение к полям
lineSpacing.value = 1
lineSpacing.rule = DocumentAPI.LineSpacingRule_Exact

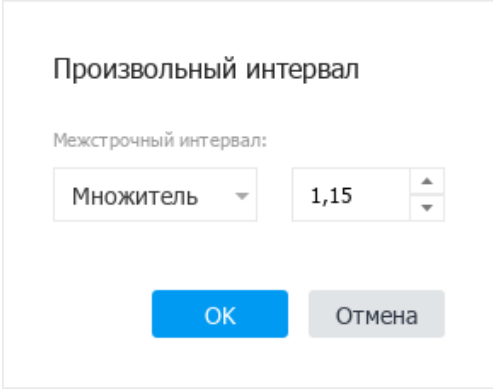
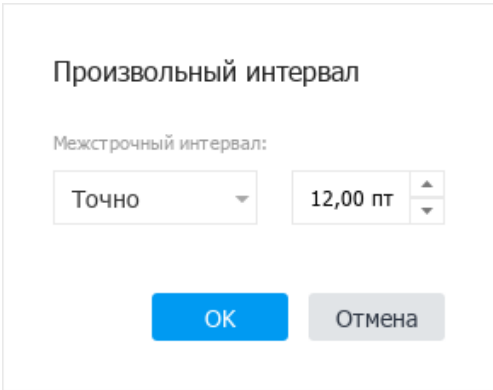
```

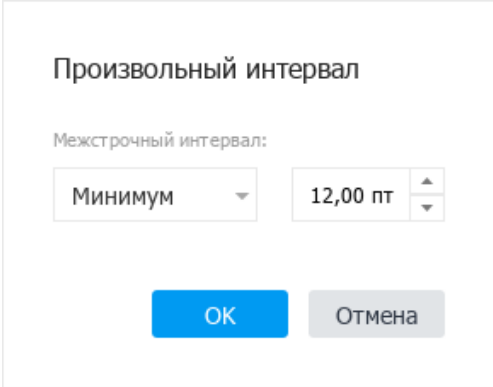
4.5.9 Таблица DocumentAPI.LineSpacingRule

В таблице 44 представлены варианты правил формирования межстрочного интервала текстового абзаца.

Таблица 44 – Виды межстрочного интервала

Наименование константы	Описание
DocumentAPI.LineSpacingRule_Multiple	Установка произвольного межстрочного интервала с использованием множителя.

Наименование константы	Описание
	<p>При вызове необходимо указать значение множителя, например:</p> <pre>pPr.lineSpacing = DocumentAPI.LineSpacing(1.15, DocumentAPI.LineSpacingRule_Multiple)</pre> <p>В данном примере используется значение множителя 1.15.</p> <p>Действие команды аналогично ручной настройке межстрочного интервала в диалоге «Произвольный интервал» (см. приложение «Руководство оператора. Часть 1. Редактор текста»).</p> 
DocumentAPI.LineSpacingRule_Exact	<p>Установка произвольного межстрочного интервала с использованием точного значения.</p> <p>При вызове необходимо указать точное значение, например:</p> <pre>pPr.lineSpacing = DocumentAPI.LineSpacing(12.0, DocumentAPI.LineSpacingRule_Exact)</pre> <p>В данном примере используется точное значение 12.0.</p> <p>Действие команды аналогично ручной настройке межстрочного интервала в диалоге «Произвольный интервал» (см. приложение «Руководство оператора. Часть 1. Редактор текста»).</p> 
DocumentAPI.LineSpacingRule_AtLeast	<p>Установка произвольного межстрочного интервала с использованием минимального значения.</p> <p>При вызове необходимо указать минимальное значение, например:</p> <pre>pPr.lineSpacing = DocumentAPI.LineSpacing(12.0, DocumentAPI.LineSpacingRule_AtLeast)</pre>

Наименование константы	Описание
	<p>В данном примере используется точное значение 12.0.</p> <p>Действие команды аналогично ручной настройке межстрочного интервала в диалоге «Произвольный интервал» (см. приложение «Руководство оператора. Часть 1. Редактор текста»).</p> 




Пример:

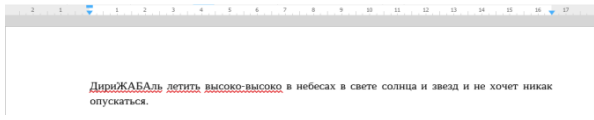
```
p = document:getBlocks():getParagraph(0)
pPr = p:getParagraphProperties()
pPr.lineSpacing = DocumentAPI.LineSpacing(5.0,
DocumentAPI.LineSpacingRule_Multiple)
p:setParagraphProperties(pPr)
```

4.5.10 Таблица DocumentAPI.Alignment

В таблице 45 представлены варианты выравнивания текста по горизонтали в текстовом редакторе или содержимого ячеек в табличном редакторе.

Таблица 45 – Варианты выравнивания по горизонтали

Наименование константы	Описание
DocumentAPI.Alignment_Default	Выравнивание по умолчанию.
DocumentAPI.Alignment_Left	<p>По левому краю</p> 
DocumentAPI.Alignment_Center	<p>По центру</p> 
DocumentAPI.Alignment_Right	<p>По правому краю</p> 

Наименование константы	Описание
DocumentAPI.Alignment_Justify	По ширине 

Пример:

```
local para = document:getBlocks():getParagraph(0)
local props = para:getParagraphProperties()
props.alignment = DocumentAPI.Alignment_Center
para:setParagraphProperties(props)
```

4.6 Рецензирование документов

Средства рецензирования документа доступны в текстовом редакторе, они позволяют выполнять следующие действия:

- пометать изменения, вносимые пользователем в текстовый документ (DocumentAPI.TrackedChange);
- ассоциировать текстовый комментарий с фрагментом текстового документа (DocumentAPI.Comments).

Данные механизмы используются на стадии рецензирования или согласования документа с последующим внесением замечаний. Функции объектной модели для работы со средствами рецензирования позволяют получить детальную информацию о каждом изменении: автор изменения, дата внесения изменения, оригинальный текст, измененный текст.

Для включения или отключения режима рецензирования используется метод `document:setChangesTrackingEnabled()`. Для проверки текущего статуса данного режима используется метод `document:isChangesTrackingEnabled()`.

Пример:

```
document:setChangesTrackingEnabled(true)
print(document:isChangesTrackingEnabled())
```

Инструменты рецензирования применяются к диапазону документа, по этой причине методы доступа к ним находятся в таблице `DocumentAPI.Range` (см. Рисунок 1204).

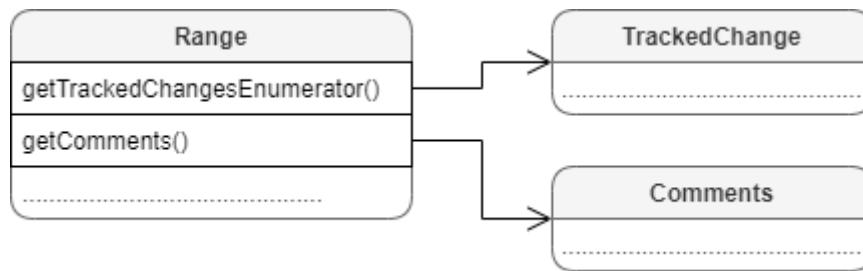


Рисунок 1204 – Инструменты рецензирования документа

4.6.1 Таблица DocumentAPI.TrackedChange

Таблица DocumentAPI.TrackedChange представляет отслеживаемое изменение в диапазоне документа (см. Рисунок 1205).

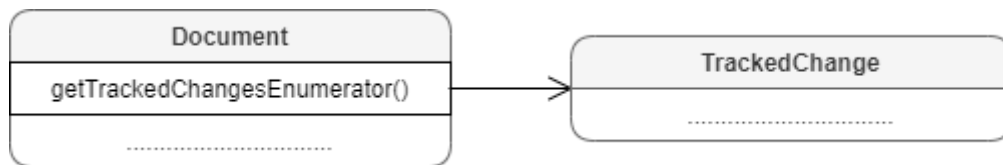


Рисунок 1205 – Объектная модель таблиц для работы с отслеживаемыми изменениями

Для получения списка отслеживаемых изменений используется метод Range.enumerateTrackedChanges().

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()
for change in changesList do
    print(change:getRange():extractText())
end
```

4.6.1.1 Метод TrackedChange:getRange

Метод возвращает объект DocumentAPI.Range, который соответствует измененному диапазону внутри абзаца.

Пример:

```
local tracked_changes = document:getRange():enumerateTrackedChanges()
for tracked_change in tracked_changes do
```

```
print(tracked_change:getRange():extractText())  
end
```

4.6.1.2 Метод TrackedChange:getType

Метод позволяет получить информацию о типе отслеживаемого изменения DocumentAPI.TrackedChangeType.

Пример:

```
local tracked_changes = document:getRange():enumerateTrackedChanges()  
for tracked_change in tracked_changes do  
    print(tracked_change:getType())  
end
```

4.6.1.3 Метод TrackedChange:getInfo

Метод позволяет получить информацию об отслеживаемых изменениях (DocumentAPI.TrackedChangeInfo).

Пример:

```
local tracked_changes = document:getRange():enumerateTrackedChanges()  
for tracked_change in tracked_changes do  
    print(tracked_change:getInfo().author.name)  
end
```

4.6.2 Таблица DocumentAPI.TrackedChangeInfo

Таблица DocumentAPI.TrackedChangeInfo содержит информацию об отслеживаемых изменениях. Описание полей таблицы представлено в таблице 46.

Таблица 46 – Описание полей таблицы DocumentAPI.TrackedChangeInfo

Поле	Тип	Описание
DocumentAPI.TrackedChangeInfo.author	UserInfo	Автор изменений.
DocumentAPI.TrackedChangeInfo.timeStamp	DateTime	Дата и время изменений.

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()  
for change in changesList do  
    local trackedChangeInfo = change:getInfo()
```

```
local author = trackedChangeInfo.author
local ts = trackedChangeInfo.timeStamp
local ts_msg = string.format("%d/%d/%d - %d:%d:%d", ts.day, ts.month, ts.year,
ts.hour, ts.minute, ts.second)
print(author.name, ts_msg)
end
```

4.6.2.1 Метод TrackedChangeInfo:__eq

Метод используется для определения эквивалентности двух отслеживаемых изменений.

4.6.3 Таблица DocumentAPI.DateTime

Таблица DocumentAPI.DateTime предоставляет дату и время с точностью до секунды. Используется для поля TrackedChangeInfo.timeStamp. Описание полей таблицы DocumentAPI.DateTime представлено в таблице 47.

Таблица 47 – Описание полей таблицы DocumentAPI.DateTime

Поле	Тип	Описание
DocumentAPI.DateTime.year	Дата	Год
DocumentAPI.DateTime.month	Дата	Месяц
DocumentAPI.DateTime.day	Дата	День
DocumentAPI.DateTime.hour	Время	Часы
DocumentAPI.DateTime.minute	Время	Минуты
DocumentAPI.DateTime.second	Время	Секунды

4.6.3.1 Метод DateTime:__eq

Метод используется для определения эквивалентности двух значений времени.

4.6.4 Таблица DocumentAPI.TrackedChangeType

Типы отслеживаемых изменений представлены в таблице 48.

Таблица 48 – Типы отслеживаемых изменений

Наименование константы	Описание
DocumentAPI.TrackedChangeType_Added	Добавленные изменения.

Наименование константы	Описание
DocumentAPI.TrackedChangeType_Removed	Удаленные изменения.

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()
for change in changesList do
  if DocumentAPI.TrackedChangeType_Added == change:getType() then action =
"Добавлено: " else action = "Удалено: " end
  print(action)
end
```

4.6.5 Таблица DocumentAPI.Comments

Таблица DocumentAPI.Comments содержит коллекцию комментариев диапазона (см. Рисунок 1206).

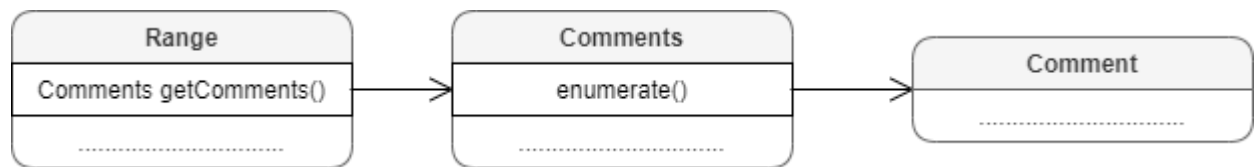


Рисунок 1206 – Объектная модель таблиц для работы с комментариями

Для получения списка комментариев используется метод Range.getComments().

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
  local commentInfo = comment:getInfo()
  local name = commentInfo.author.name
  print("Комментарий " .. name .. ": ", comment:getText())
end
```

4.6.5.1 Метод Comments:enumerate

Метод возвращает коллекцию комментариев всего документа.

Пример:

```
local comments = document:getComments()
for comment in comments:enumerate() do
```



```
print(comment:getText())  
end
```

4.6.6 Таблица DocumentAPI.Comment

Таблица DocumentAPI.Comment предоставляет доступ к следующим свойствам комментария:

- диапазон текста DocumentAPI.Range, который описывает комментарий;
- текст комментария;
- информация о комментарии DocumentAPI.TrackedChangeInfo;
- признак того, что комментарий принят;
- список ответов на комментарий DocumentAPI.Comments.

4.6.6.1 Метод Comment:getRange

Метод возвращает диапазон документа DocumentAPI.Range, которому соответствует комментарий.

Пример:

```
local commentsList = document:getRange():getComments()  
for comment in commentsList:enumerate() do  
    print("Диапазон комментария: ", comment:getRange():extractText())  
end
```

4.6.6.2 Метод Comment:getText

Метод возвращает текст комментария.

Пример:

```
local commentsList = document:getRange():getComments()  
for comment in commentsList:enumerate() do  
    print("Текст комментария: ", comment:getText())  
end
```

4.6.6.3 Метод Comment:getInfo

Метод предоставляет доступ к информации о комментарии DocumentAPI.TrackedChangeInfo (автор изменения, дата и т. д.).

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    local commentInfo = comment:getInfo()
    local name = commentInfo.author.name
    print("Автор комментария:", name)
end
```

4.6.6.4 Метод `Comment:isResolved`

Метод возвращает значение `true`, если комментарий принят.

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    print("Комментарий принят:", comment:isResolved())
end
```

4.6.6.5 Метод `Comment:getReplies`

Метод предоставляет доступ к ответам на комментарии. Ответы находятся в такой же таблице `DocumentAPI.Comments`, как и сами комментарии документа.

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    local commentReplies = comment:getReplies()
    for reply in commentReplies:enumerate() do
        local name = reply.author.name
        print("Ответ на комментарий " .. name .. ": ", reply:getText())
    end
end
```

4.7 Работа с закладками

Основной таблицей для работы с закладками является `DocumentAPI.Bookmarks`. Список закладок документа возвращает метод `document:getBookmarks()`. Метод `Bookmarks.getBookmarkRange()` возвращает диапазон текста, метод

`Bookmarks.removeBookmark()` удаляет закладку по имени. Для создания закладки используется метод `Position.insertBookmark()`.

Доступны следующие операции с закладками:

- вставка закладки в указанное местоположение;
- удаление закладки с заданным именем;
- поиск закладки по имени;
- замена текстового содержимого закладки;
- вставка текста в закладку;
- удаление содержимого закладки;
- получение текстового содержимого закладки;
- вставка таблицы в закладку.

Вставка закладки в указанное местоположение

```
-- Вставка новой закладки с именем Signers в начало документа
local sig_pos = document:getRange():getBegin()
sig_pos:insertBookmark("Signers")
```

Удаление закладки с заданным именем

```
-- Удаление закладки "Signers"
document:getBookmarks():removeBookmark("Signers")
```

Поиск закладки по имени

```
-- Поиск закладки "Signers" по имени
local sig_rng = document:getBookmarks():getBookmarkRange("Signers")
```

Замена текстового содержимого закладки

```
-- Замена содержимого закладки на текст "Lua"
local bookmarks = document:getBookmarks()
local bookmarkRange = bookmarks:getBookmarkRange("bm_1")
bookmarkRange.replaceText("Lua")
```

Вставка текста в закладку

```
sig_rng:getBegin():insertText("Лист")
```

Удаление содержимого закладки

```
sig_rng:removeContent()
```

Получение текстового содержимого закладки

```
local msg = sig_rng:extractText()  
print(msg)
```

Вставка таблицы в закладку

```
-- Вставка таблицы в закладку "Signers"  
local tbl_id = sig_rng:getEnd():insertTable(3, 3, "signers_list")
```

4.7.1 Таблица DocumentAPI.Bookmarks

Предоставляет доступ к операциям с закладками в документе.

4.7.1.1 Метод Bookmarks:getBookmarkRange

Возвращает объект DocumentAPI.Range для дальнейшей работы с содержимым закладки (bookmark).

Пример:

```
local bookmarks = document:getBookmarks()  
local bookmark = bookmarks:getBookmarkRange("Bookmark")  
bookmark:replaceText("Lua")
```

4.7.1.2 Метод Bookmarks:removeBookmark

Удаляет закладку по ее названию.

Пример:

```
document:getBookmarks():removeBookmark("Bookmark")
```

4.8 Таблицы и ячейки

4.8.1 Доступ к таблицам

Доступ к таблицам DocumentAPI.Table осуществляется из DocumentAPI.Blocks (см. Рисунок 1207). В табличном документе таблицами являются листы документа.

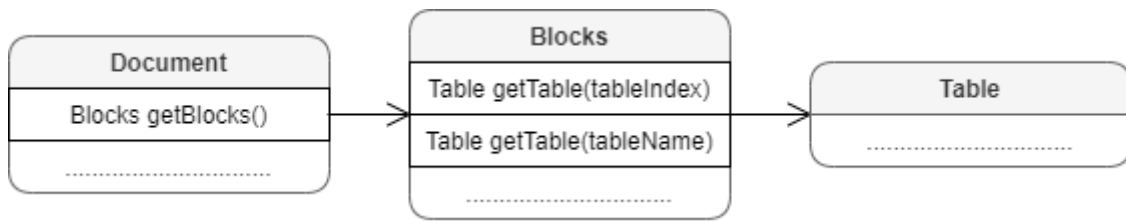


Рисунок 1207 – Объектная модель для работы с таблицами

Для получения таблицы используется метод `Blocks:getTable()`. В качестве аргумента используется индекс или имя таблицы.

Примеры:

```
local table = document:getBlocks():getTable(0)
local table = document:getBlocks():getTable("Таблица1")
```

Для перечисления таблиц текстового документа или листов табличного документа можно использовать метод `Blocks:enumerateTables()`.

```
for sheet in document:getBlocks():enumerateTables() do
    print(sheet:getName())
end
```

Для табличного документа также доступен вариант перечисления листов документа посредством использования метода `Blocks:enumerate()` с дальнейшим преобразованием блока в таблицу.

```
for sheet in document:getBlocks():enumerate() do
    print(sheet:toTable():getName())
end
```

Для вставки таблицы в текстовый документ или листа в табличный документ используется метод `Position:insertTable()`. В качестве аргументов передаются размеры и имя таблицы.

Пример:

```
local position = document:getRange():getEnd()
position:insertTable(4, 3, "Таблица")
```

В табличном документе таблицы являются листами документа. Для листов табличного документа доступны следующие операции:

- добавление листа, см. описание метода `Position.insertTable()` в разделе 4.16.2;
- переименование листа, см. описание метода `Table.setName()` в разделе 4.8.6.1;
- копирование и перемещение листа, см. описание метода `Table.duplicate()` в разделе 4.8.6.15;
- удаление листа, см. описание метода `Table.remove()` в разделе 4.8.6.16;
- перечисление листов документа, см. варианты реализации в разделе 4.8;
- обращение к листу документа, см. варианты реализации в разделе 4.8;
- скрытие и отображение листов, см. описание метода `Table.setVisible()` в разделе 4.8.6.20.

4.8.2 Доступ к ячейкам

Доступ к ячейкам таблицы возможен двумя способами (см. Рисунок 1208):

- непосредственно из таблицы, используя метод `Table.getCell()`;
- из диапазона ячеек методом перечисления `CellRange.enumerate()`.

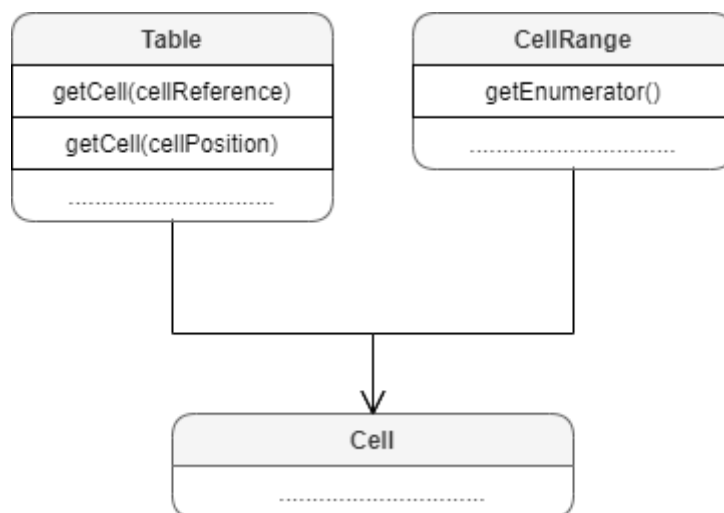


Рисунок 1208 – Объектная модель для работы с ячейками таблиц

Для получения содержимого ячейки, заполнения данных, а также для форматирования ячейки используется объект `DocumentAPI.Cell`, представляющий ячейку таблицы с указанным адресом. Метод `Table.getCell()` возвращает экземпляр таблицы `Cell`.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")
```

Второй вариант доступа к ячейке - перечисление диапазона ячеек методом `CellRange.enumerate()`.

Пример:

```
local table = document:getBlocks():getTable(0)
local rng = table:getCellRange("B3:C4")
for cell in rng:enumerate() do
    print(cell:getFormattedValue())
end
```

Для установки значений ячеек используются методы `Cell:setText`, `Cell:setNumber`, `Cell:setFormula`, `Cell:setBool`.

Примеры:

```
local sheet = document:getBlocks():getTable("Лист2")

--setText, текстовое значение
sheet:getCell("A1"):setText("Текст")

--setNumber, числовое значение с фиксированной точкой
sheet:getCell("B2"):setNumber(10)

--setNumber, числовое значение с плавающей точкой
sheet:getCell("B3"):setNumber(1,0)

--setFormula, текст формулы
sheet:getCell("B4"):setFormula( "=SUM(B2:B3)" )

--setBool, логическое значение
sheet:getCell("B4"):setBool(false)
```

Для установки даты и времени используется функция `Cell:setFormattedValue`. Данная функция пытается определить тип значения, переданного в качестве аргумента (число, дата и т.д.) и применяет необходимое форматирование.

Пример:

```
local sheet = document:getBlocks():getTable("Лист1")

--setFormattedValue, дата
sheet:getCell("B5"):setFormattedValue("22.07.2020")
```

```
--setFormattedValue, время
sheet:getCell("B6"):setFormattedValue("12:39")
```

При необходимости есть возможность явно указать формат вводимого значения DocumentAPI.CellFormat (процентный, денежный, экспоненциальный и т.д.), для этого используется функция Cell.SetFormat().

Пример:

```
local sheet = document:getBlocks():getTable("Лист1")
local value = 12
local cell = sheet:getCell("B1")
-- Установка формата данных
cell:setFormat(DocumentAPI.CellFormat_Accounting)
cell:setNumber(value)
```

Для получения значения ячейки используется метод Cell.getFormattedValue().

Пример:

```
local sheet = document:getBlocks():getTable("Лист1")
local value = sheet:getCell("B1"):getFormattedValue()
```

4.8.2.1 Таблица DocumentAPI.CellFormat

По умолчанию при создании документа всем ячейкам присваивается формат «Общий». Полный список форматов представлен в таблице 49.

Таблица 49 – Поддерживаемые форматы ячеек таблицы

Наименование константы	Описание
DocumentAPI.CellFormat_General	<p>Формат ячейки «Общий».</p> <p>В этом формате в ячейке отображаются первые 9 символов числа, остальные доступны для просмотра в строке формул. Для дробных чисел в формате «Общий» незначащие нули в дробной части не отображаются.</p> <p>Числа, состоящие более чем из 12 символов, переводятся в экспоненциальную форму после завершения ввода в ячейку.</p>
DocumentAPI.CellFormat_Percentage	<p>Формат ячейки «Процентный».</p> <p>Этот формат используется для представления чисел как процентов. При применении формата «Процентный» введенное число умножается на</p>

Наименование константы	Описание
	100 и обозначается знаком «%».
DocumentAPI.CellFormat_Number	<p>Формат ячейки «Числовой».</p> <p>Если в ячейке с форматом «Числовой» содержится дробное число, то можно указать количество знаков, отображаемых в данном числе после разделителя.</p>
DocumentAPI.CellFormat_Text	Формат ячейки «Текстовый».
DocumentAPI.CellFormat_Currency	<p>Формат ячейки «Денежный».</p> <p>Этот формат используется для представления чисел со знаком или кодом валюты.</p>
DocumentAPI.CellFormat_Accounting	<p>Формат ячейки «Финансовый».</p> <p>Этот формат применяется для чисел, используемых в бухгалтерских документах. В формате «Финансовый» введенное число автоматически дополняется названием валюты, которая соответствует настройкам системы компьютера.</p> <p>Отрицательные числа в формате «Финансовый» заключаются в круглые скобки в ячейке, а в строке формул остаются в том виде, в котором они были введены.</p>
DocumentAPI.CellFormat_Date	<p>Формат ячейки «Дата».</p> <p>Этот формат автоматически присваивается числам, введенным в определенном виде, например, ДД.ММ.ГГГГ.</p>
DocumentAPI.CellFormat_Time	<p>Формат ячейки «Время».</p> <p>Этот формат автоматически присваивается числам, введенным в определенном виде, например, ЧЧ:ММ.</p>
DocumentAPI.CellFormat_Fraction	<p>Формат ячейки «Дробный».</p> <p>Этот формат используется для представления дробных чисел в виде обыкновенных дробей (то есть дробная часть заменяется на числитель и знаменатель).</p>
DocumentAPI.CellFormat_Scientific	<p>Формат ячейки «Экспоненциальный».</p> <p>Экспоненциальный (или научный) формат</p>

Наименование константы	Описание
	<p>используется для представления больших чисел в короткой форме. Все введенные числа длиной более 12 символов автоматически переводятся в этот формат.</p> <p>В ячейке число в формате «Экспоненциальный» представлено следующим образом:</p> <ul style="list-style-type: none"> – целая часть, всегда состоящая из одной цифры; – разделитель целой и дробной части; – дробная часть, по умолчанию состоящая из двух цифр; – показатель степени числа 10 в виде E<знак показателя степени> <показатель степени>.
DocumentAPI.CellFormat_Custom	Пользовательский формат.

Использование данных констант позволяет установить выбранный формат. При этом будет использованы параметры формата по умолчанию.

Примеры использования:

```

local table = document:getBlocks():getTable(0)
local cell_B1 = table:getCell("B1")
cell_B1:setFormat(DocumentAPI.CellFormat_General)
local cell_B2 = table:getCell("B2")
cell_B2:setFormat(DocumentAPI.CellFormat_Percentage)
local cell_B3 = table:getCell("B3")
cell_B3:setFormat(DocumentAPI.CellFormat_Number)

```

Результат выполнения данного примера приведен на рисунке 1209.

	A	B
1	CellFormat.General	1
2	CellFormat.Percentage	100,00%
3	CellFormat.Number	1,00

Рисунок 1209 – Результат установки формата

Пример форматирования ячейки также приведен в разделе 4.8.2, описывающем установку значений ячеек.

4.8.2.2 Таблица DocumentAPI.AccountingCellFormatting

Таблица содержит параметры финансового формата ячеек таблицы и используется в качестве аргумента метода `Cell:setFormat()`.

Описание полей таблицы `DocumentAPI.AccountingCellFormatting` представлено в таблице 50.

Таблица 50 – Описание полей таблицы `DocumentAPI.AccountingCellFormatting`

Поле	Описание
<code>DocumentAPI.AccountingCellFormatting.decimalPlaces</code>	Количество десятичных позиций
<code>DocumentAPI.AccountingCellFormatting.symbol</code>	Символ денежной единицы
<code>DocumentAPI.AccountingCellFormatting.localeCode</code>	Идентификатор кода языка (MS-LCID)
<code>DocumentAPI.AccountingCellFormatting.fillSymbol</code>	Символ заполнения
<code>DocumentAPI.AccountingCellFormatting.useThousandsSeparator</code>	Использовать разделитель для тысячных
<code>DocumentAPI.AccountingCellFormatting.currencySignPlacement</code>	Тип размещения знака валюты <code>CurrencySignPlacement</code>

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

cell:setFormat(DocumentAPI.CellFormat_Accounting)
local accountingCellFormatting = DocumentAPI.AccountingCellFormatting()
accountingCellFormatting.decimalPlaces = 3
accountingCellFormatting.symbol = 'Руб'

cell:setFormat(accountingCellFormatting)
print(cell:getFormattedValue())
```

4.8.2.3 Таблица DocumentAPI.PercentageCellFormatting

Содержит параметр для процентного формата ячеек таблицы, используется в качестве аргумента метода `Cell:setFormat()`. Описание полей таблицы `DocumentAPI.PercentageCellFormatting` представлено в таблице 51.

Таблица 51 – Описание полей таблицы DocumentAPI.PercentageCellFormatting

Поле	Описание
DocumentAPI.PercentageCellFormatting.decimalPlaces	Количество десятичных позиций.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

local percentageCellFormatting = DocumentAPI.PercentageCellFormatting()
percentageCellFormatting.decimalPlaces = 2

cell:setFormat(percentageCellFormatting)
print(cell:getFormattedValue())
```

4.8.2.4 Таблица DocumentAPI.NumberCellFormatting

Таблица содержит параметры для числового формата ячеек таблицы, используется в качестве аргумента метода Cell:setFormat(). Описание полей таблицы DocumentAPI.NumberCellFormatting представлено в таблице 52.

Таблица 52 – Описание полей таблицы DocumentAPI.NumberCellFormatting

Поле	Описание
DocumentAPI.NumberCellFormatting.decimalPlaces	Количество десятичных позиций.
DocumentAPI.NumberCellFormatting.useThousandsSeparator	Использовать разделитель для тысячных.
DocumentAPI.NumberCellFormatting.useRedForNegative	Использовать красный цвет для отрицательных значений.
DocumentAPI.NumberCellFormatting.useBracketsForNegative	Использовать скобки для отрицательных значений.
DocumentAPI.NumberCellFormatting.hideSign	Скрывать знак «минус» для отрицательных значений.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A2")

local numberCellFormatting = DocumentAPI.NumberCellFormatting()
numberCellFormatting.decimalPlaces = 2
```

```
numberCellFormatting.useThousandsSeparator = true
numberCellFormatting.useRedForNegative = true
numberCellFormatting.useBracketsForNegative = true
numberCellFormatting.hideSign = false

cell:setFormat(numberCellFormatting)
print(cell:getFormattedValue())
```

4.8.2.5 Таблица DocumentAPI.CurrencyCellFormatting

Таблица содержит параметры для денежного формата ячеек таблицы. Описание полей таблицы DocumentAPI.CurrencyCellFormatting представлено в таблице 53.

Таблица 53 – Описание полей таблицы DocumentAPI.CurrencyCellFormatting

Поле	Описание
DocumentAPI.CurrencyCellFormatting.decimalPlaces	Количество десятичных позиций.
DocumentAPI.CurrencyCellFormatting.symbol	Символ денежной единицы.
DocumentAPI.CurrencyCellFormatting.localeCode	Идентификатор кода языка (MS-LCID).
DocumentAPI.CurrencyCellFormatting.useThousandsSeparator	Использовать разделитель для тысячных.
DocumentAPI.CurrencyCellFormatting.useRedForNegative	Использовать красный цвет для отрицательных значений.
DocumentAPI.CurrencyCellFormatting.useBracketsForNegative	Использовать скобки для отрицательных значений.
DocumentAPI.CurrencyCellFormatting.hideSign	Скрывать знак «минус» для отрицательных значений.
DocumentAPI.CurrencyCellFormatting.currencySignPlacement	Варианты размещения знака валюты CurrencySignPlacement.

Данная таблица используется в качестве аргумента метода Cell:setFormat(), см. пример.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

local currencyCellFormatting = DocumentAPI.CurrencyCellFormatting()
```

```

currencyCellFormatting.decimalPlaces = 2
currencyCellFormatting.useThousandsSeparator = true
currencyCellFormatting.useRedForNegative = true
currencyCellFormatting.useBracketsForNegative = true
currencyCellFormatting.hideSign = false
currencyCellFormatting.currencySignPlacement =
DocumentAPI.CurrencySignPlacement_Suffix

cell:setFormat(currencyCellFormatting)
print(cell:getFormattedValue())

```

4.8.2.6 Таблица DocumentAPI.CurrencySignPlacement

Варианты размещения знака валюты представлены в таблице 54. Данный тип используется в поле `currencyFormat` таблицы `DocumentAPI.LocaleInfo`, а также в поле `currencySignPlacement` таблицы `DocumentAPI.CurrencyCellFormatting` (см. пример в разделе 4.8.2.5).

Таблица 54 – Описание полей таблицы `DocumentAPI.CurrencySignPlacement`

Поле	Описание	Пример
<code>DocumentAPI.CurrencySignPlacement_Prefix</code>	Размещение знака валюты до значения.	\$12.00
<code>DocumentAPI.AccountingCellFormatting_Suffix</code>	Размещение знака валюты после значения.	12,00 Р

4.8.2.7 Таблица DocumentAPI.DateTimeCellFormatting

Таблица содержит параметры для формата ячеек таблицы типа Дата и Время, используется в качестве аргумента метода `Cell:setFormat()`. Описание полей таблицы `DocumentAPI.DateTimeCellFormatting` представлено в таблице 55.

Таблица 55 – Описание полей таблицы `DocumentAPI.DateTimeCellFormatting`

Поле	Описание
<code>DocumentAPI.DateTimeCellFormatting.dateListID</code>	Формат даты <code>DatePatterns.</code>
<code>DocumentAPI.DateTimeCellFormatting.timeListID</code>	Формат времени <code>TimePatterns.</code>

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

local dateTimeCellFormatting = DocumentAPI.DateTimeCellFormatting()
dateTimeCellFormatting.dateListID = DocumentAPI.DatePatterns_FullDate
dateTimeCellFormatting.timeListID = DocumentAPI.TimePatterns_LongTime

cell:setFormat(dateTimeCellFormatting)
print(cell:getFormattedValue())
```

4.8.2.8 Таблица DocumentAPI.DatePatterns

Форматы даты представлены в таблице 56. Пример использования см. в разделе 3.8.2.7.

Таблица 56 – Форматы даты

Наименование константы	Описание
DocumentAPI.DatePatterns_DayMonthTextLongYearLong	'mmmm dd, yyyu' для языка en_US.
DocumentAPI.DatePatterns_FullDate	'день недели, mmmm dd, yyyu' для языка en_US.
DocumentAPI.DatePatterns_DayMonthNumberLongYearLong	'mm/dd/yyyy' для языка en_US.
DocumentAPI.DatePatterns_DayMonthNumberLongYearShort	'm/dd/yy' для языка en_US.
DocumentAPI.DatePatterns_DayMonthNumberShortYearShort	'dd-mmm' для языка en_US.
DocumentAPI.DatePatterns_DayMonthTextShort	'mmm-yy' для языка en_US.
DocumentAPI.DatePatterns_DayMonthTextShortYearShort	'mmm dd, yy' для языка en_US.
DocumentAPI.DatePatterns_DayMonthYearLongNonLocalizableHyphen	Нелокализуемый шаблон 'dd-mm-yyuu'.
DocumentAPI.DatePatterns_DayMonthYearLongNonLocalizableSlash	Нелокализуемый шаблон 'dd/mm/yyyy'.

4.8.2.9 Таблица DocumentAPI.TimePatterns

Форматы времени представлены в таблице 57. Пример использования см. в разделе 4.8.2.7.

Таблица 57 – Форматы времени

Наименование константы	Описание
DocumentAPI.TimePatterns_ShortTime	'hh:mm AM/PM' для языка en_US.
DocumentAPI.TimePatterns_LongTime	'hh:mm:ss AM/PM' для языка en_US.

4.8.2.10 Таблица DocumentAPI.FractionCellFormatting

Таблица содержит параметры для дробного формата ячеек таблицы. Данная таблица используется в качестве аргумента метода Cell:setFormat(). Описание полей таблицы DocumentAPI.FractionCellFormatting представлено в таблице 58.

Таблица 58 – Описание полей таблицы DocumentAPI.FractionCellFormatting

Поле	Описание
DocumentAPI.FractionCellFormatting.minNumeratorDigits	Количество позиций числителя.
DocumentAPI.FractionCellFormatting.minDenominatorDigits	Количество позиций знаменателя.
DocumentAPI.FractionCellFormatting.denominatorValue	Знаменатель.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("B2")

local fractionCellFormatting = DocumentAPI.FractionCellFormatting()
fractionCellFormatting.minNumeratorDigits = 2
fractionCellFormatting.minDenominatorDigits = 3
fractionCellFormatting.denominatorValue = 22

cell:setFormat(fractionCellFormatting)
print(cell:getFormattedValue())
```


4.8.2.11 Таблица DocumentAPI.ScientificCellFormatting

Таблица содержит параметры для экспоненциального формата ячеек таблицы. Данная таблица используется в качестве аргумента метода `Cell:setFormat()`. Описание полей таблицы `DocumentAPI.ScientificCellFormatting` представлено в таблице 59.

Таблица 59 – Описание полей таблицы `DocumentAPI.ScientificCellFormatting`

Поле	Описание
<code>DocumentAPI.ScientificCellFormatting.decimalPlaces</code>	Количество десятичных позиций.
<code>DocumentAPI.ScientificCellFormatting.minExponentDigits</code>	Минимальное количество позиций экспоненты.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("B2")

local scientificCellFormatting = DocumentAPI.ScientificCellFormatting()
scientificCellFormatting.decimalPlaces = 2
scientificCellFormatting.minExponentDigits = 3

cell:setFormat(scientificCellFormatting)
print(cell:getFormattedValue())
```

4.8.3 Форматирование ячеек

При работе с ячейками таблиц можно использовать следующие варианты форматирования:

- форматирование параметров ячейки `DocumentAPI.CellProperties`, например, цвет фона, угол поворота текста;
- форматирование абзаца ячейки, например, отступы абзаца, межстрочный интервал текста;
- форматирование текста, например, цвет текста, начертание;
- задание параметров границ ячеек.

Содержимое ячейки (контент), вне зависимости от того является ли оно текстом, числовым значением или формулой, также описывается экземпляром класса `Paragraph`, и обладает свойствами `DocumentAPI.ParagraphProperties`. Это дает возможность управлять настройками отображения контента как отдельного абзаца, так и группы абзацев

(например, если ячейка содержит несколько предложений текста). Для управления этим настройками используются методы `Cell.getParagraphProperties()` и `Cell.setParagraphProperties()`.

Пример установки и получения свойств параграфа ячейки:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A2") --(DocumentAPI.CellPosition(1,0))

local paraProps = cell:getParagraphProperties()
paraProps.alignment = DocumentAPI.Alignment_Center
cell:setParagraphProperties(paraProps)
```

Управление настройками текста ячейки (шрифт, цвет) производится через соответствующий ему диапазон. Класс `Cell` позволяет получить диапазон для всего контента с помощью метода `Cell.getRange()`. Далее, метод `Range.getTextProperties()` позволяет получить экземпляр класса `DocumentAPI.TextProperties`, представляющий свойства текста. После изменения значения свойств их необходимо применить к тексту ячейки с помощью метода `Range.setTextProperties()`.

Пример настроек текста ячейки:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell(DocumentAPI.CellPosition(0,1))

local textProps = cell:getRange():getTextProperties()
textProps.bold = true
textProps.italic = true
local rgba = DocumentAPI.ColorRGBA(121,112,212,255)
textProps.textColor = DocumentAPI.Color(rgba)
cell:getRange():setTextProperties(textProps)
```

4.8.4 Форматирование границ ячеек

Для оформления границ ячеек используется таблица `DocumentAPI.Borders` (см. Рисунок 1210). Она описывает свойства полей, соответствующих границам и диагоналям ячейки: `Left`, `Right`, `Top`, `Bottom`, `DiagonalDown`, `DiagonalUp`, `InnerHorizontal`, `InnerVertical`. Каждая граница ячейки описывается таблицей `DocumentAPI.LineProperties`, которая, в свою очередь, обладает свойствами

DocumentAPI.LineStyle, DocumentAPI.LineEndingProperties, DocumentAPI.Color, LineWidth.

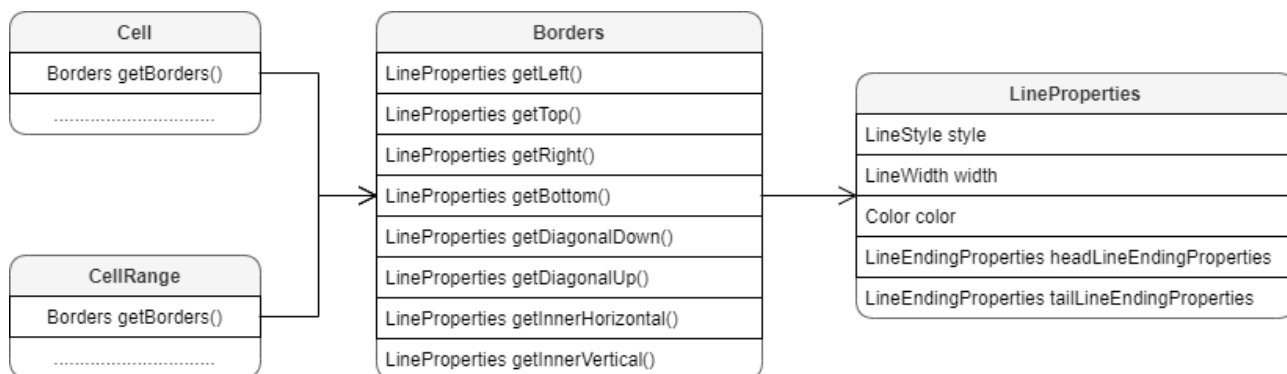


Рисунок 1210 – Таблицы для работы с границами ячеек

Для оформления границ отдельной ячейки или группы ячеек необходимо выполнить следующие действия:

- получить ячейку DocumentAPI.Cell или область ячеек DocumentAPI.CellRange;
- настроить параметры для рисования линии границы с помощью экземпляра класса DocumentAPI.LineProperties;
- настроить свойства линии: левой границы, верхней границы и т.д. с помощью экземпляра класса DocumentAPI.Borders;
- установить границы ячеек с помощью Cell.setBorders() или CellRange.setBorders().

Пример настройки границ ячеек:

```

local sheet = document:getBlocks():getTable("Лист2")
local cellRange = sheet:getCellRange("F3:H7")

--Настроить параметры для рисования линии
local lineProp = DocumentAPI.LineProperties()
lineProp.style = DocumentAPI.LineStyle_Solid
lineProp.width = 1.5
local lc = DocumentAPI.ColorRGBA(55, 146, 179, 200)
lineProp.color = DocumentAPI.Color(lc)

--Настроить положение линии – обводка по внешней границе области
local borders = DocumentAPI.RangeBorders()
    
```



```
--установка внешних границ
borders:setOuter(lineProp)

--Нарисовать границы области
cellRange:setBorders(borders)
```

4.8.4.1 Таблица DocumentAPI.Borders

Таблица DocumentAPI.Borders предназначена для оформления границ отдельной ячейки таблицы (см. Таблица 60). Параметры линии, такие как тип линии, ее ширина и цвет, задаются с помощью таблицы DocumentAPI.LineProperties.

Таблица 60 – Описание методов таблицы DocumentAPI.Borders

Метод	Описание
Borders:setLeft	Установка левой границы ячейки.
Borders:setRight	Установка правой границы ячейки.
Borders:setTop	Установка верхней границы ячейки.
Borders:setBottom	Установка нижней границы ячейки.
Borders:setDiagonalDown	Установка диагональной линии. 
Borders:setDiagonalUp	Установка диагональной линии. 
Borders:setOuter	Установка внешних границ ячейки.
Borders:setDiagonals	Установка обоих типов диагональных линий одновременно.
Borders:setInnerHorizontal	Установка внутренних горизонтальных границ ячейки.
Borders:setInnerVertical	Установка внутренних вертикальных границ ячейки.
Borders:setInner	Установка внутренних границ ячейки.
Borders:setAll	Установка всех границ ячейки.
Borders:getLeft	Получение левой границы ячейки.
Borders:getRight	Получение правой границы ячейки.
Borders:getTop	Получение верхней границы ячейки.
Borders:getBottom	Получение нижней границы ячейки.
Borders:getDiagonalDown	Получение диагональной линии.
Borders:getDiagonalUp	Получение диагональной линии.
Borders:getInnerHorizontal	Получение внутренних горизонтальных границ ячейки.

Метод	Описание
<code>Borders:getInnerVertical</code>	Получение внутренних вертикальных границ ячейки.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("C3")

LineProperties = DocumentAPI.LineProperties()
LineProperties.style = DocumentAPI.LineStyle_Dash
LineProperties.width = 1.5
LineProperties.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0, 255))

borders = DocumentAPI.Borders()
borders = borders:setLeft(LineProperties)
borders = borders:setRight(LineProperties)
borders = borders:setTop(LineProperties)
borders = borders:setBottom(LineProperties)

borders = cell:setBorders(borders)
```

4.8.4.2 Таблица `DocumentAPI.RangeBorders`

Таблица `DocumentAPI.RangeBorders` оставлена для совместимости. Вместо нее необходимо использовать таблицу `DocumentAPI.Borders`.

4.8.4.3 Таблица `DocumentAPI.LineProperties`

Таблица `DocumentAPI.LineProperties` предназначена для установки таких параметров линии, как тип, ширина, цвет (см. Рисунок 1211).

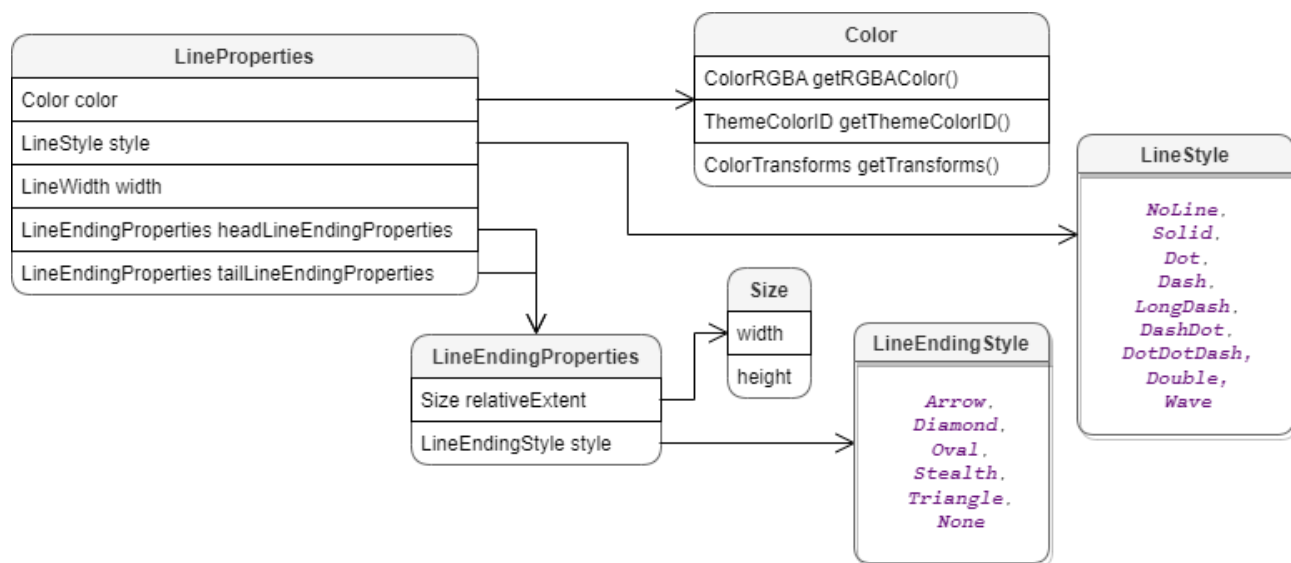


Рисунок 1211 – Свойства границ ячеек

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("C3")

lineProperties = DocumentAPI.LineProperties()
lineProperties.style = DocumentAPI.LineStyle_Solid
lineProperties.width = 1.5
lineProperties.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(55, 146, 179,
200))

borders = DocumentAPI.Borders()
borders = borders:setTop(lineProperties)
local brds = cell:setBorders(borders)
```

4.8.4.3.1 Поле LineProperties.style

Поле предназначено для установки типа линии. Допустимые значения представлены в разделе 4.8.4.5.

4.8.4.3.2 Поле LineProperties.width

Поле предназначено для установки ширины линии. Тип – числовой.

4.8.4.3.3 Поле `LineProperties.color`

Поле предназначено для установки цвета линии. Тип – `DocumentAPI.Color`.

4.8.4.3.4 Поле `LineProperties.headLineEndingProperties`

Поле предназначено для оформления начала линии

`DocumentAPI.LineEndingProperties`.

4.8.4.3.5 Поле `LineProperties.tailLineEndingProperties`

Поле предназначено для оформления конца линии

`DocumentAPI.LineEndingProperties`.

4.8.4.4 Таблица `DocumentAPI.LineEndingProperties`

Таблица `DocumentAPI.LineEndingProperties` содержит варианты оформления окончаний линий. Описание полей таблицы `DocumentAPI.LineEndingProperties` представлено в таблице 61. Используется в полях `headLineEndingProperties` и `tailLineEndingProperties` таблицы `DocumentAPI.LineProperties`.

Таблица 61 – Описание полей таблицы `DocumentAPI.LineEndingProperties`

Поле	Тип	Описание
<code>DocumentAPI.LineEndingProperties.style</code>	<code>LineEndingStyle</code>	Стиль окончания линии.
<code>DocumentAPI.LineEndingProperties.relativeExtent</code>	<code>Size</code>	Размер окончания линии относительно ее ширины.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = tbl:getCell("C3")

lineProperties = DocumentAPI.LineProperties()
lineProperties.headLineEndingProperties = DocumentAPI.LineEndingProperties()
lineProperties.headLineEndingProperties.style =
DocumentAPI.LineEndingStyle_Arrow

lineProperties.headLineEndingProperties.relativeExtent = DocumentAPI.SizeU()
lineProperties.headLineEndingProperties.relativeExtent.width = 2
```

```
lineProperties.headLineEndingProperties.relativeExtent.height = 2








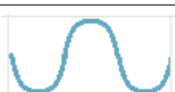
lineProperties.tailLineEndingProperties = DocumentAPI.LineEndingProperties()
lineProperties.tailLineEndingProperties.style =
DocumentAPI.LineEndingStyle_Arrow
lineProperties.tailLineEndingProperties.relativeExtent = DocumentAPI.SizeU()
lineProperties.tailLineEndingProperties.relativeExtent.width = 2
lineProperties.tailLineEndingProperties.relativeExtent.height = 2

borders = DocumentAPI.Borders()
borders = borders:setTop(lineProperties)
cell:setBorders(borders)
```

4.8.4.5 Таблица DocumentAPI.LineStyle

В таблице 62 приведены типы линий. Используется в поле style таблицы DocumentAPI.LineProperties.

Таблица 62 – Типы линий

Наименование константы	Описание
DocumentAPI.LineStyle_NoLine	Нет линии
DocumentAPI.LineStyle_Solid	
DocumentAPI.LineStyle_Dot	
DocumentAPI.LineStyle_Dash	
DocumentAPI.LineStyle_LongDash	
DocumentAPI.LineStyle_DashDot	
DocumentAPI.LineStyle_DotDotDash	
DocumentAPI.LineStyle_Double	
DocumentAPI.LineStyle_Wave	

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("C3")

lineProperties = DocumentAPI.LineProperties()
lineProperties.style = DocumentAPI.LineStyle_Wave

borders = DocumentAPI.Borders()
borders = borders:setTop(lineProperties)
cell:setBorders(borders)
```

4.8.4.6 Таблица DocumentAPI.LineEndingStyle

В таблице 63 приведены типы окончания линии. Используется в поле `style` таблицы `DocumentAPI.LineEndingProperties`.

Таблица 63 – Типы окончания линии

Наименование константы	Описание
<code>DocumentAPI.LineEndingStyle_Arrow</code>	
<code>DocumentAPI.LineEndingStyle_Diamond</code>	
<code>DocumentAPI.LineEndingStyle_Oval</code>	
<code>DocumentAPI.LineEndingStyle_Stealth</code>	
<code>DocumentAPI.LineEndingStyle_Triangle</code>	
<code>DocumentAPI.LineEndingStyle_None</code>	

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = tbl:getCell("C3")

lineProperties = DocumentAPI.LineProperties()
lineProperties.headLineEndingProperties = DocumentAPI.LineEndingProperties()
lineProperties.headLineEndingProperties.style = DocumentAPI.LineEndingStyle_Oval

borders = DocumentAPI.Borders()
```

```
borders = borders:setTop(lineProperties)  
cell:setBorders(borders)
```

4.8.5 Объединение и разделение ячеек таблицы

Допустимо объединение произвольного числа ячеек таблицы. При объединении указанный диапазон становится единой ячейкой. После завершения операции объединенная ячейка получает значение первой ячейки диапазона.

Для объединения нескольких ячеек используйте метод `CellRange.merge()`.

Пример:

```
-- Объединение ячеек A1 и A2 на первом листе табличного документа  
local tbl = document:getBlocks():getTable(0)  
tbl:getCellRange("A1:A2"):merge()
```

Допустимо разъединение только тех ячеек, которые были объединены ранее. После завершения операции данные, содержащиеся в объединенной ячейке, будут помещены в верхнюю левую ячейку диапазона.

Для разъединения ячеек используйте метод `CellRange.unmerge()`.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
-- Ячейка A1 является результатом объединения диапазона A1:A2  
tbl:getCell("A1"):unmerge()
```

4.8.6 Таблица `DocumentAPI.Table`

Таблица `DocumentAPI.Table` предоставляет доступ к листу в табличном документе или таблице в составе текстового документа (см. Рисунок 1212).

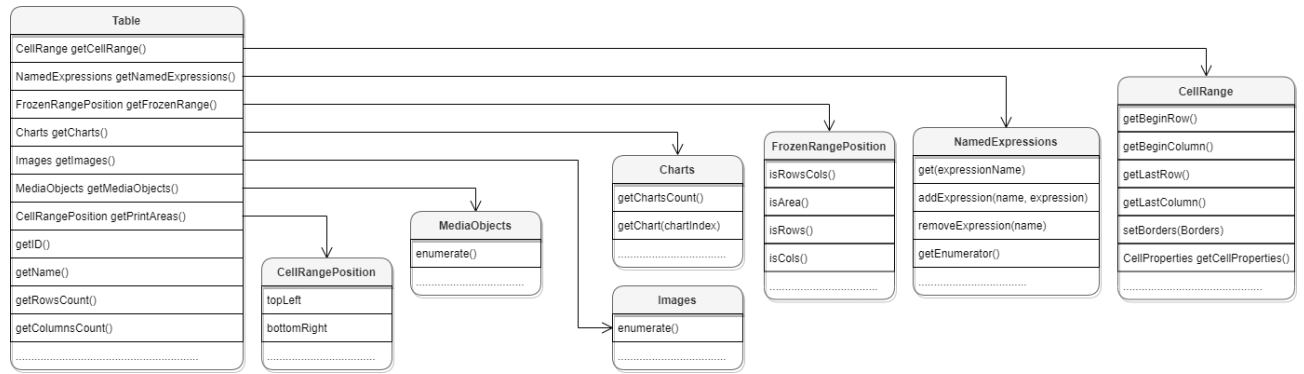


Рисунок 1212 – Структура полей таблицы DocumentAPI.Table

4.8.6.1 Метод Table:setName

Метод задает имя таблицы. В случае с табличным документом это имя будет являться заголовком листа документа. Данное значение должно быть уникальным, т.к. может использоваться для ссылки на таблицу, например, из формул.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:setName("Первый")
```

Для текстовых документов использование данного метода также допустимо, наименование таблицы нигде не отображается, но в дальнейшем его можно использовать для доступа к таблице по имени.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:setName("Первый")
tbl = document:getBlocks():getTable("Первый")
```

4.8.6.2 Метод Table:getName

Метод позволяет получить наименование листа табличного документа.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getName())
```

4.8.6.3 Метод Table:getRowCount

Метод позволяет получить количество строк таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getRowCount())
```

4.8.6.4 Метод Table:getColumnsCount

Метод позволяет получить количество столбцов таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getColumnsCount())
```

4.8.6.5 Метод Table:getCell

Метод позволяет получить доступ к отдельной ячейке таблицы. В качестве аргумента может выступать текстовое представление адреса ячейки, либо экземпляр таблицы DocumentApi.CellPosition.

Примеры:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
print(cell:getFormattedValue())

local cellPosition = DocumentAPI.CellPosition(2, 1)
local cell = tbl:getCell(cellPosition)
print(cell:getFormattedValue())
```

4.8.6.6 Метод Table:getCellRange

Метод позволяет получить доступ к диапазону ячеек таблицы DocumentAPI.CellRange.

Примеры:

```
local table = document:getBlocks():getTable(0)
local range = table:getCellRange("A1:C4")
for cell in range:enumerate() do
    print(cell:getFormattedValue())
end
```

```
local table = document:getBlocks():getTable(0)
local range = table:getCellRange(DocumentAPI.CellRangePosition(0, 0, 2, 2))
for cell in range:enumerate() do
    print(cell:getFormattedValue())
end
```

4.8.6.7 Метод Table:insertColumnAfter

Метод предназначен для вставки нового столбца после указанной позиции в таблице.

Вызов:

```
insertColumnAfter( columnIndex, copyColumnStyle, columnsCount )
```

Параметры:

- columnIndex – индекс столбца в таблице, после которого производится вставка. Индексация столбцов начинается с нуля.
- copyColumnStyle – флаг наследования стиля. Если этот параметр установлен в значение true, то новый столбец наследует настройки форматирования столбца с индексом columnIndex. Если параметр copyColumnStyle установлен в значение false, то настройки форматирования не копируются. Значение по умолчанию true.
- columnsCount – количество вставляемых столбцов. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")
-- Добавление двух столбцов в середину таблицы, без наследования настроек
форматирования
tbl:insertColumnAfter(0, false, 2)
```

4.8.6.8 Метод Table:insertColumnBefore

Метод предназначен для вставки нового столбца до указанной позиции в таблице.

Вызов:

```
insertColumnBefore( columnIndex, copyColumnStyle, columnsCount )
```

Параметры:

- columnIndex – индекс столбца в таблице, перед которым производится вставка. Индексация столбцов начинается с нуля.

- `copyColumnStyle` – флаг наследования стиля. Если этот параметр установлен в значение `true`, то новый столбец наследует настройки форматирования столбца с индексом `columnIndex`. Если параметр `copyColumnStyle` установлен в значение `false`, то настройки форматирования не копируются. Значение по умолчанию `true`.
- `columnsCount` – количество вставляемых столбцов. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")

-- Добавление двух столбцов в середину таблицы, без наследования настроек
форматирования
tbl.insertColumnBefore(1, false, 2)
```

4.8.6.9 Метод `Table:insertRowAfter`

Метод предназначен для вставки новой строки после указанной позиции в таблице.

Вызов:

```
insertRowAfter( rowIndex, copyRowStyle, rowsCount )
```

Параметры:

- `rowIndex` – индекс строки в таблице, после которой производится вставка. Индексация строк начинается с нуля.
- `copyRowStyle` – флаг наследования стиля. Если этот параметр установлен в значение `true`, то новая строка наследует настройки форматирования строки с индексом `rowIndex`. Если параметр `copyRowStyle` установлен в значение `false`, то настройки форматирования не копируются. Значение по умолчанию `true`.
- `rowsCount` – количество вставляемых строк. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")

-- Добавление двух строк в середину таблицы, без наследования настроек
форматирования
tbl.insertRowAfter(0, false, 2)
```

4.8.6.10 Метод `Table:insertRowBefore`

Метод предназначен для вставки новой строки до указанной позиции в таблице.

Вызов:

```
insertRowBefore( rowIndex, copyRowStyle, rowCount )
```

Параметры:

- `rowIndex` – индекс строки в таблице, перед которой производится вставка. Индексация строк начинается с нуля.
- `copyRowStyle` – флаг наследования стиля. Если этот параметр установлен в значение `true`, то новая строка наследует настройки форматирования строки с индексом `rowIndex`. Если параметр `copyRowStyle` установлен в значение `false`, то настройки форматирования не копируются. Значение по умолчанию `true`.
- `rowCount` – количество вставляемых строк. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2  
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")  
  
-- Добавление двух строк в середину таблицы, без наследования настроек  
форматирования  
tbl:insertRowBefore(1, false, 2)
```

4.8.6.11 Метод `Table:removeColumn`

Метод предназначен для удаления столбца таблицы, начиная с заданного индекса.

Вызов:

```
removeColumn(columnIndex, columnsCount)
```

Параметры:

- `columnIndex` – индекс столбца, начиная с которого будет удалено заданное количество столбцов. Индексация столбцов начинается с нуля.
- `columnsCount` – количество столбцов для удаления. Значение по умолчанию 1.

4.8.6.12 Метод `Table:removeRow`

Метод предназначен для удаления строки таблицы, начиная с заданного индекса.

Вызов:

```
removeRow(rowIndex, rowCount)
```

Параметры:

- `rowIndex` – индекс строки, начиная с которого будет удалено `rowCount` строк. Индексация строк начинается с нуля.
- `rowCount` – количество строк для удаления. Значение по умолчанию 1.

4.8.6.13 Метод `Table:setColumnWidth`

Метод устанавливает ширину столбца таблицы в пунктах (1/72 дюйма).

Вызов:

```
setColumnWidth( columnIndex, width )
```

Параметры:

- `columnIndex` – индекс столбца в таблице, для которого устанавливается значение ширины. Индексация столбцов начинается с нуля.
- `width` – ширина столбца в пунктах (1/72 дюйма).

Пример:

```
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")  
  
-- Установить ширину столбца в 400 pt  
tbl:setColumnWidth(1, 400)
```

4.8.6.14 Метод `Table:setRowHeight`

Метод устанавливает высоту строки таблицы в пунктах (1/72 дюйма).

Вызов:

```
setRowHeight(rowIndex, height)
```

Параметры:

- `rowIndex` – индекс строки в таблице, для которой устанавливается значение высоты. Индексация строк начинается с нуля.
- `height` – высота строки в пунктах (1/72 дюйма).
- `rowHeightRule` – точность значения (`DocumentAPI.RowHeightRule_Exact` – точно, `DocumentAPI.RowHeightRule_AtLeast` – не меньше).

Пример:

```
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")  
  
-- Установить высоту строки в 100 pt  
tbl:setRowHeight(1,100, DocumentAPI.RowHeightRule_Exact)
```

4.8.6.15 Метод Table:duplicate

Для создания копии листа в табличном документе используется метод `duplicate`. Созданная копия листа размещается после копируемого листа. Метод может быть использован только в табличном документе.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
tbl:duplicate()
```

4.8.6.16 Метод Table:remove

Для удаления таблицы в текстовом документе или листа в табличном документе используется метод `remove()`.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
tbl:remove()
```

4.8.6.17 Метод Table:moveTo

Для перемещения листа таблицы по указанному индексу в табличном документе используется метод `moveTo`. Указанный индекс должен быть меньше или равен количеству листов в документе. Индексация листов начинается с нуля. Метод может быть использован только в табличном документе.

Пример:

```
-- В табличном документе два листа с индексами 0 и 1.  
-- Поменяем их местами.  
local tbl = document:getBlocks():getTable(0)  
tbl:moveTo(1)
```

4.8.6.18 Метод Table:setShowZeroValue

Для упрощения чтения таблицы нулевые значения ячеек могут быть скрыты. Для управления скрытием/показом ячеек используется метод `setShowZeroValue`.

Пример:

```
tbl = document:getBlocks():getTable(0)
tbl:setShowZeroValue(true)
```

4.8.6.19 Метод Table:getShowZeroValue

Для проверки режима отображения нулевых значений ячеек используется метод `getShowZeroValue`.

Пример:

```
tbl = document:getBlocks():getTable(0)
tbl:setShowZeroValue(false)
print(tbl:getShowZeroValue())
```

4.8.6.20 Метод Table:setVisible

Метод управляет видимостью листа таблицы. Используется только в табличном документе.

Вызов:

```
setVisible( visible )
```

Параметр:

`visible` – параметр, задающий видимость листа. Если значение параметра `visible` равно `true`, то лист таблицы отображается в редакторе таблиц.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:setVisible(false)
```

4.8.6.21 Метод Table:isVisible

Метод возвращает значение `true`, если лист таблицы в табличном документе отображается в редакторе таблиц.

Пример:

```
local tbl = document:getBlocks():getTable(0)
if not tbl.isVisible() then
    tbl.setVisible(true)
end
```

4.8.6.22 Метод Table:getFrozenRange

Существует возможность закрепления диапазона строк и столбцов. Такие диапазоны всегда остаются видимыми на экране в случае, когда пользователь осуществляет навигацию по таблице.

Метод `getFrozenRange` возвращает закрепленный диапазон `DocumentAPI.FrozenRangePosition`.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
local tbl = document:getBlocks():getTable(0)
tbl.freeze(frozenRangePosition)
print(tbl:getFrozenRange():isCols())
```

4.8.6.23 Метод Table:freeze

Метод `freeze` закрепляет заданную область `DocumentAPI.FrozenRangePosition` таблицы.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
local tbl = document:getBlocks():getTable(0)
tbl.freeze(frozenRangePosition)
print(tbl:getFrozenRange():isCols())
```

4.8.6.24 Метод Table:__eq

Метод используется для определения эквивалентности двух таблиц.

Пример:

```
local tbl1 = document:getBlocks():getTable(0)
local tbl2 = document:getBlocks():getTable("Table Name")
if tbl1:__eq(tbl2) then
```

```
print("tbl1 и tbl2 ссылаются на общую таблицу в документе")  
end
```

4.8.6.25 Метод Table:setPrintArea

Метод служит для установки и сброса области печати DocumentAPI.CellRangePosition.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
tbl:setPrintArea(DocumentAPI.CellRangePosition(0, 0, 5, 5)) -- уставить область  
печати размером в пять строк и пять колонок, начиная с левого верхнего угла  
таблицы
```

4.8.6.26 Метод Table:setPrintAreas

Метод Table:setPrintAreas задает множественные области печати или экспорта CellRangePositions, где CellRangePositions – вектор из элементов CellRangePosition (см. описание вектора в разделе 4.26).

Пример:

```
tbl = document:getBlocks():getTable(0)  
ranges = DocumentAPI.CellRangePositions()  
ranges:push_back(DocumentAPI.CellRangePosition(0, 0, 5, 5))  
ranges:push_back(DocumentAPI.CellRangePosition(1, 2, 5, 5))  
tbl:setPrintAreas(ranges)  
  
printAreas = tbl:getPrintAreas()  
print(printAreas[0]:toString(), printAreas[1]:toString())
```

4.8.6.27 Метод Table:getPrintAreas

Метод Table:getPrintAreas возвращает текущие области печати – вектор элементов DocumentAPI.CellRangePosition. См. описание методов вектора в разделе 4.26.

Пример:

```
tbl = document:getBlocks():getTable(0)  
tbl:setPrintArea(DocumentAPI.CellRangePosition(0, 0, 5, 5))
```

```
printAreas = tbl:getPrintAreas()  
print(printAreas[0]:toString())
```

4.8.6.28 Метод Table:getCharts

Для получения списка диаграмм (DocumentAPI.Charts) таблицы используется метод Table:getCharts.

Пример:

```
for tbl in document:getBlocks():enumerateTables() do  
    print(tbl:getCharts():getChartsCount())  
end
```

4.8.6.29 Метод Table:getImages

Для получения списка изображений (DocumentAPI.Images) таблицы используется метод Table:getImages.

Пример:

```
tbl = document:getBlocks():getTable(0)  
images = tbl:getImages()  
  
for image in images:enumerate() do  
    print(image)  
end
```

4.8.6.30 Метод Table:getMediaObjects

Для получения списка медиаобъектов (DocumentAPI.MediaObjects) таблицы используется метод Table:getMediaObjects.

Пример:

```
tbl = document:getBlocks():getTable(0)  
mediaObjects = tbl:getMediaObjects()  
  
for mediaObject in mediaObjects:enumerate() do  
    print(mediaObject)  
end
```

4.8.6.31 Метод `Table:getNameExpressions`

Для получения списка именованных диапазонов `DocumentAPI:NamedExpressions` используется метод `Table:getNameExpressions()`.

4.8.6.32 Группировка строк и колонок таблицы

Следующий набор методов позволяет группировать строки и колонки таблицы. Редактор дает возможность отображать группы в виде иерархии. Совместно с данными методами можно использовать методы `Table::setVisibleColumns` и `Table::setVisibleRows` чтобы раскрывать и закрывать фрагменты иерархии групп.

Методы могут вызвать исключения `OutOfRangeException` и `IncorrectArgumentError` в случае использования индексов, выходящих за рамки таблицы.

Методы для группировки:

```
groupRows(first, rowCount)
ungroupRows(first, rowCount)
clearRowGroups(first, rowCount)
groupColumns(first, columnsCount)
ungroupColumns(first, columnsCount)
clearColumnGroups(first, columnsCount)
```

4.8.6.33 Управление видимостью строк / колонок

Метод `Table::setVisibleColumns` позволяет задавать видимость `columnsCount` столбцов, начиная с индекса `first`. Индексация столбцов начинается с нуля.

```
setVisibleColumns(first, columnsCount, visible)
```

Метод `Table:setVisibleRows` позволяет задавать видимость `rowCount` строк, начиная с индекса `first`. Индексация строк начинается с нуля.

```
setVisibleRows(first, rowCount, visible)
```

4.8.7 Таблица `DocumentAPI.Cell`

Таблица `DocumentAPI.Cell` предоставляет доступ к ячейке в таблице текстового документа или на листе табличного документа (см. Рисунок 1213).

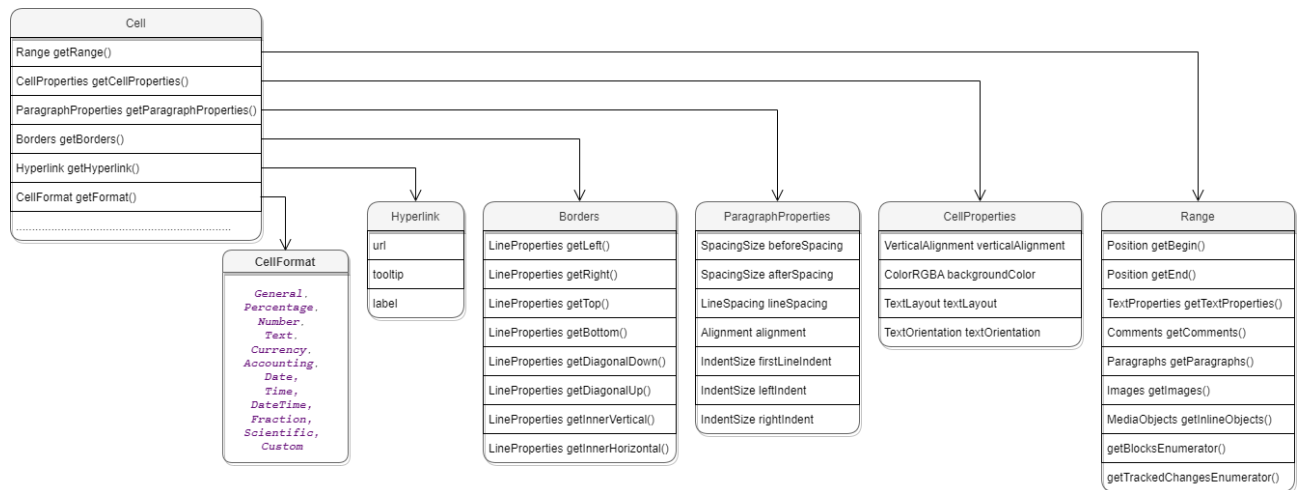


Рисунок 1213 – Объектная модель ячейки таблиц

4.8.7.1 Метод Cell:getRange

Метод возвращает объект `DocumentAPI.Range` для управления содержимым ячейки.

4.8.7.2 Метод Cell:setBorders

Метод предназначен для установки границ ячейки. Примеры использования приведены в разделе 4.8.4.1.

4.8.7.3 Метод Cell:setFormula

Метод позволяет вставить формулу в ячейку табличного документа.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:getCell("A1"):setNumber(2.3)
tbl:getCell("A2"):setNumber(3.2)
tbl:getCell("A3"):setFormula( "=SUM(A1:A2)" ) -- 5,5
```

4.8.7.4 Метод Cell:getFormat

Метод возвращает формат ячейки. Список поддерживаемых форматов ячеек приведен в разделе 4.8.2.1.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

local cellFormatting = DocumentAPI.PercentageCellFormatting()
cell:setFormat(cellFormatting)
print("Формат: ", cell:getFormat()) -- 1
```

4.8.7.5 Метод Cell:setFormat

Метод устанавливает формат ячейки. Существуют несколько вариантов использования метода.

Варианты вызова метода:

```
setFormat(cellFormat)
```

Где **cellFormat** – формат ячейки типа `DocumentAPI.CellFormat`.

```
setFormat(accountingCellFormatting)
```

Где **accountingCellFormatting** – формат ячейки типа `DocumentAPI.AccountingCellFormatting`.

```
setFormat(percentageCellFormatting)
```

Где **percentageCellFormatting** – формат ячейки типа `DocumentAPI.PercentageCellFormatting`.

```
setFormat(numberCellFormatting)
```

Где **numberCellFormatting** – формат ячейки типа `DocumentAPI.NumberCellFormatting`.

```
setFormat(currencyCellFormatting)
```

Где **currencyCellFormatting** – формат ячейки типа `DocumentAPI.CurrencyCellFormatting`.

```
setFormat(dateTimeCellFormatting, typeFormat)
```

Где **dateTimeCellFormatting** – формат ячейки типа `DocumentAPI.DateTimeCellFormatting`, **typeFormat** – формат даты/времени типа `DocumentAPI.CellFormat`.


```
setFormat(fractionCellFormatting)
```

Где **fractionCellFormatting** – формат ячейки типа

DocumentAPI.FractionCellFormatting.

```
setFormat(scientificCellFormatting)
```

Где **scientificCellFormatting** – формат ячейки типа

DocumentAPI.ScientificCellFormatting.

Примеры использования:

```
local tbl = document:getBlocks():getTable(0)
local cellA1 = tbl:getCell("A1")

cellA1:setNumber(2.3)

-- Формат: Общий
cellA1:setFormat(DocumentAPI.CellFormat_General)
print("Формат Общий: ", cellA1:getFormat()) -- 0
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,3

-- Формат: Процентный
local alCellFormatting = DocumentAPI.PercentageCellFormatting()
alCellFormatting.decimalPlaces = 1
cellA1:setFormat(alCellFormatting)
print("Формат Процентный: ", cellA1:getFormat()) -- 1
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 230,0%

-- Формат: Числовой
alCellFormatting = DocumentAPI.NumberCellFormatting()
alCellFormatting.decimalPlaces = 2
cellA1:setFormat(alCellFormatting)
print("Формат Числовой: ", cellA1:getFormat()) -- 2
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30

-- Формат: Денежный
alCellFormatting = DocumentAPI.CurrencyCellFormatting()
alCellFormatting.symbol = '$'
cellA1:setFormat(alCellFormatting)
print("Формат Денежный: ", cellA1:getFormat()) -- 4
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30$
```

```
-- Формат: ФИНАНСОВЫЙ
a1CellFormatting = DocumentAPI.AccountingCellFormatting()
a1CellFormatting.symbol = '₽'
cellA1:setFormat(a1CellFormatting)
print("Формат ФИНАНСОВЫЙ: ", cellA1:getFormat()) -- 5
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30₽

-- Формат: Дата / Время
a1CellFormatting = DocumentAPI.DateTimeCellFormatting()
a1CellFormatting.dateListID = DocumentAPI.DatePatterns_FullDate
a1CellFormatting.timeListID = DocumentAPI.TimePatterns_ShortTime
cellA1:setFormat(a1CellFormatting)
print("Формат Дата / Время: ", cellA1:getFormat()) -- 8
print("Значение ячейки: ", cellA1:getRange():extractText()) -- понедельник, 1
января 1900 г. 7:12

-- Формат: Экспоненциальный
a1CellFormatting = DocumentAPI.FractionCellFormatting()
a1CellFormatting.minNumeratorDigits = 2
cellA1:setFormat(a1CellFormatting)
print("Формат Экспоненциальный: ", cellA1:getFormat()) -- 9
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2 2/7

-- Формат: Научный
a1CellFormatting = DocumentAPI.ScientificCellFormatting()
a1CellFormatting.decimalPlaces = 5
cellA1:setFormat(a1CellFormatting)
print("Формат Научный: ", cellA1:getFormat()) -- 10
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30000E+00
```

4.8.7.6 Метод Cell:getFormattedValue

Метод позволяет получить значение ячейки в текущем формате. Список поддерживаемых форматов см. в разделе 4.8.2.1.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getCell("A6"):getFormattedValue()) -- 21.6.1972
```

4.8.7.7 Метод Cell:setFormattedValue

Анализирует переданное значение и автоматически устанавливает формат ячейки и ее значение. В случае, если распознать тип переданного значения не удастся, то для ячейки устанавливается формат `DocumentAPI.CellFormat_Text`.

Список поддерживаемых форматов см. в разделе 4.8.2.1.

4.8.7.8 Метод Cell:unmerge

Разъединяет несколько ячеек, которые были объединены ранее.

Пример:

```
local tbl = document:getBlocks():getTable(0)
-- Ячейка A1 является результатом объединения диапазона A1:A2
tbl:getCell("A1"):unmerge()
```

4.8.7.9 Метод Cell:getHyperlink

Возвращает первый объект в ячейке типа `DocumentAPI.Hyperlink`.

```
local cell =
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))
local hyperlink = cell:getHyperlink()
```

4.8.7.10 Метод Cell:setContent

Определяет и устанавливает соответствующую формулу или значение, а затем форматирует ячейку. Устанавливает текст, если автоопределение не удалось.

Пример:

```
local cell =
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))
cell:setContent("=A2+A3")
```

4.8.7.11 Метод Cell:getBorders

Позволяет получить границы ячейки.

Пример:

```
local cell =
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))
local borders = cell:getBorders()
```

4.8.7.12 Метод Cell:getRawValue

Возвращает значение ячейки в формате «Общий» (без форматирования).

Пример:

```
local tbl = document:getBlocks():getTable(0)
local val = tbl:getCell("A6"):getRawValue()
```

4.8.7.13 Метод Cell:getCustomFormat

Возвращает строку формата ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cust_format = tbl:getCell("A6"):getCustomFormat()
```

4.8.7.14 Метод Cell:setCustomFormat

Устанавливает формат ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:getCell("A6"):setCustomFormat("0,00")
```

4.8.7.15 Метод Cell:setBool

Устанавливает для ячейки значение логического типа.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:getCell("A6"):setBool(true)
```

4.8.7.16 Метод Cell:setNumber

Устанавливает для ячейки значение числового типа.

Пример:

```
local sheet = document:getBlocks():getTable(0)
local cell = sheet:getCell(DocumentAPI.CellPosition(0, 0))
cell:setNumber(0.0001)
```

4.8.7.17 Метод Cell:setText

Устанавливает для ячейки значение строкового типа.

Пример:

```
local sheet = document:getBlocks():getTable(0)
local cell = sheet:getCell(DocumentAPI.CellPosition(0, 0))
local trackingChanges = "Disabled"
if document:isChangesTrackingEnabled() then
    trackingChanges = "Enabled"
end
cell:setText(trackingChanges)
```

4.8.7.18 Метод Cell:getFormulaAsString

Возвращает текст формулы ячейки. Формула – это любое выражение в ячейке, которое начинается со знака равенства (=).

Пример:

```
local tbl = document:getBlocks():getTable(0)
local formula = tbl:getCell("A6"):getFormulaAsString()
```

4.8.7.19 Метод Cell:getCellProperties

Позволяет получить свойства DocumentAPI.CellProperties ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell_props = tbl:getCell("A6"):getCellProperties()
```

4.8.7.20 Метод Cell:setCellProperties

Позволяет установить свойства ячейки DocumentAPI.CellProperties.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local props = tbl:getCell("A6"):getCellProperties()
props.verticalAlignment = DocumentAPI.VerticalAlignment_Center
tbl:getCell("A6"):setCellProperties(props)
```

4.8.7.21 Метод `Cell:getParagraphProperties`

Возвращает свойства абзаца `DocumentAPI.ParagraphProperties`, находящегося в ячейке.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A2") --(DocumentAPI.CellPosition(1,0))

local paraProps = cell:getParagraphProperties()
print(paraProps.alignment)
```

4.8.7.22 Метод `Cell:setParagraphProperties`

Устанавливает свойства абзаца `DocumentAPI.ParagraphProperties`, находящегося в ячейке.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A2") --(DocumentAPI.CellPosition(1,0))

local paraProps = cell:getParagraphProperties()
paraProps.alignment = DocumentAPI.Alignment_Center
cell:setParagraphProperties(paraProps)
```

4.8.7.23 Метод `Cell:getPivotTable`

Возвращает сводную таблицу `DocumentAPI.PivotTable`, относящуюся к ячейке.

Пример:

```
tbl = document:getBlocks():getTable(0)
cell = tbl:getCell("B4")
pivotTable = cell:getPivotTable()
```

4.8.8 Таблица `DocumentAPI.CellProperties`

Таблица `DocumentAPI.CellProperties` предназначена для форматирования содержимого в ячейках таблицы. Описание полей таблицы `DocumentAPI.CellProperties` представлено в таблице 64.

Для задания свойств ячейки используется метод `Cell.setCellProperties()`. Для получения свойств ячейки используется метод `Cell.getCellProperties()`. Иерархия таблиц и полей `DocumentAPI.CellProperties` отображена на рисунке 1214.

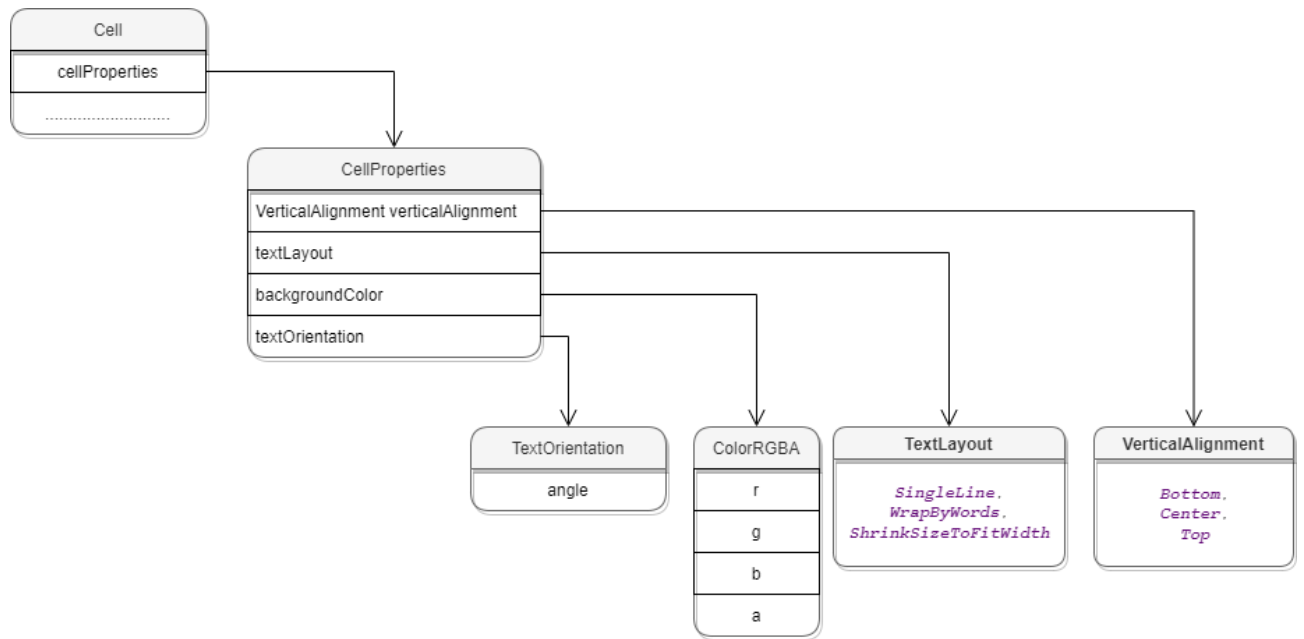


Рисунок 1214 – Объектная модель для работы со свойствами ячеек таблицы

Таблица 64 – Описание полей таблицы `DocumentAPI.CellProperties`

Поле	Тип	Значение
<code>CellProperties.verticalAlignment</code>	<code>VerticalAlignment</code>	Вертикальное выравнивание в ячейке.
<code>CellProperties.textLayout</code>	<code>TextLayout</code>	Способ отображения значения ячейки.
<code>CellProperties.backgroundColor</code>	<code>ColorRGBA</code>	Цвет фона ячейки.
<code>CellProperties.textOrientation</code>	<code>TextOrientation</code>	Ориентация текста в ячейке (угол поворота).

Пример:

```

local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("D2") --(DocumentAPI.CellPosition(3,1))
local props = cell:getCellProperties()

props.verticalAlignment = DocumentAPI.VerticalAlignment_Center
props.textLayout = DocumentAPI.TextLayout_ShrinkSizeToFitWidth
props.backgroundColor = DocumentAPI.ColorRGBA(255, 255, 0, 255)
    
```

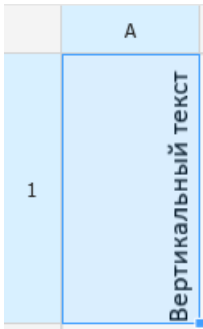
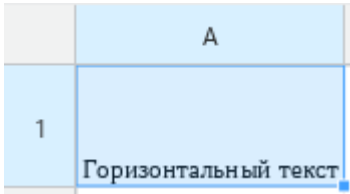

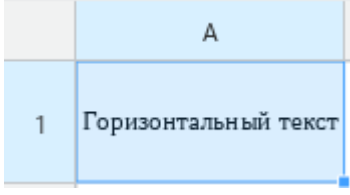
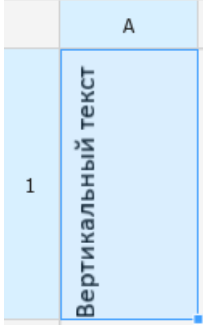
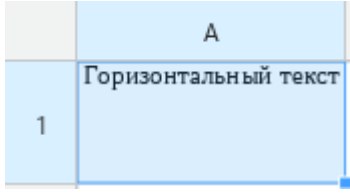
```
props.textOrientation = DocumentAPI.TextOrientation(45)

cell:setCellProperties(props)
```

4.8.9 Таблица DocumentAPI.VerticalAlignment

В таблице 65 представлены константы видов выравнивания текста по вертикали. Используется в DocumentAPI.CellProperties, DocumentAPI.ShapeProperties.

Таблица 65 – Виды выравнивания текста по вертикали

Наименование константы	Представление в интерфейсе	
DocumentAPI.VerticalAlignment_Bottom		
DocumentAPI.VerticalAlignment_Center		
DocumentAPI.VerticalAlignment_Top		

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A1")
```



```
local props = cell:getCellProperties()  
props.verticalAlignment = DocumentAPI.VerticalAlignment_Center  
cell:setCellProperties(props)
```

4.8.10 Таблица DocumentAPI.Hyperlink

Таблица DocumentAPI.Hyperlink описывает свойства ссылки. Может быть получена посредством вызова метода Cell.getHyperlink().

Таблица 66 – Описание полей таблицы DocumentAPI.Hyperlink

Поле	Тип	Описание
DocumentAPI.Hyperlink.url	Строка	Адрес ссылки
DocumentAPI.Hyperlink.tooltip	Строка	Текст подсказки
DocumentAPI.Hyperlink.label	Строка	Текст описания

Пример:

```
local cell =  
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))  
local hyperlink = cell:getHyperlink()  
print(hyperlink.url, hyperlink.tooltip, hyperlink.label)
```

4.8.10.1 Метод Hyperlink: __eq

Метод используется для определения эквивалентности двух объектов Hyperlink.

Пример:

```
table_0 = document:getBlocks():getTable(0)  
cell_00 = table_0:getCell(DocumentAPI.CellPosition(0, 0))  
cell_01 = table_0:getCell(DocumentAPI.CellPosition(0, 1))  
local hyperlink_00 = cell_00:getHyperlink()  
local hyperlink_01 = cell_01:getHyperlink()  
if (hyperlink_00 and hyperlink_01) then  
    print(hyperlink_00:__eq(hyperlink_01))  
end
```

4.8.11 Таблица DocumentAPI.TextLayout

В Таблице 67 приведены варианты размещения текста в ячейках таблицы. Используется в методах `Cell.getCellProperties()`, `Cell.setCellProperties()`.

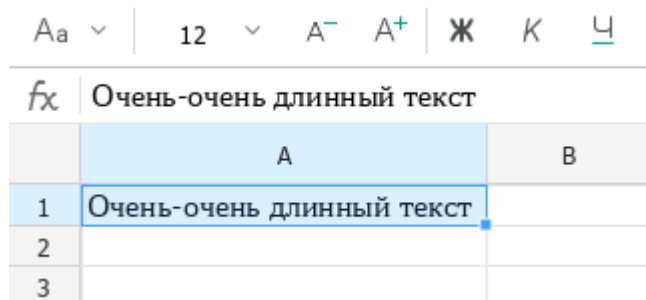
Таблица 67 – Варианты размещения текста в ячейках таблицы

Наименование константы	Описание
<code>DocumentAPI.TextLayout_SingleLine</code>	Расположение текста в одну строку с наложением на соседние ячейки.
<code>DocumentAPI.TextLayout_WrapByWords</code>	Перенос данных внутри ячейки по словам.
<code>DocumentAPI.TextLayout_ShrinkSizeToFitWidth</code>	Текст или число отображаются так, чтобы содержимое поместилось в ячейке полностью. Установленный размер шрифта не изменяется.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell_A1 = tbl:getCell("A1")
local props = cell_A1:getCellProperties()
props.textLayout = DocumentAPI.TextLayout_ShrinkSizeToFitWidth
cell_A1:setCellProperties(props)
```

Результат:



4.8.12 Таблица DocumentAPI.TextOrientation

Таблица `DocumentAPI.TextOrientation` предоставляет доступ к свойствам ориентации текста в ячейке, фигуре и т. д (см. раздел 4.8.8).

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("D2") --(DocumentAPI.CellPosition(3,1))
local props = cell:getCellProperties()
props.textOrientation = DocumentAPI.TextOrientation(45)
```

```
cell:setCellProperties(props)  
print(props.textOrientation:getAngle())
```

4.8.12.1 Метод TextOrientation:getAngle

Возвращает угол ориентации текста в ячейке. Значение угла указывается в градусах.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
local cell = tbl:getCell("D2") --(DocumentAPI.CellPosition(3,1))  
local cellProperties = cell:getCellProperties()  
print(cellProperties.textOrientation:getAngle())
```

4.8.13 Таблица DocumentAPI.CellPosition

Таблица DocumentAPI.CellPosition позволяет задать координаты ячейки электронной таблицы или таблицы в составе текстового документа.

Позиция ячейки A1 имеет координаты (0, 0).

Также для указания адреса ячейки в качестве параметра метода getCell можно использовать строку вида «A1».

Примеры:

```
local table = document:getBlocks():getTable(0) -- первый лист документа  
local cell = table:getCell(DocumentAPI.CellPosition(2, 0)) -- ячейка A3  
local table = document:getBlocks():getTable(0) -- первый лист документа  
local cell = table:getCell("A3") -- ячейка A3
```

4.8.13.1 Поле CellPosition.column

Номер столбца в значении ячейки. Нумерация столбцов начинается с нуля.

Пример:

```
cellPosition = DocumentAPI.CellPosition()  
cellPosition.column = 1
```

4.8.13.2 Поле CellPosition.row

Номер строки в позиции ячейки. Нумерация строк начинается с нуля.

Пример:

```
cellPosition = DocumentAPI.CellPosition()  
cellPosition.row = 1
```

4.8.13.3 Метод CellPosition:toString

Возвращает координаты ячейки в формате (row: R, column: C), где R и C – номер строки и столбца соответственно.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
local pos = DocumentAPI.CellPosition(0,0)  
print(pos:toString()) --(row: 0, column: 0)
```

4.8.13.4 Метод CellPosition:__eq

Метод используется для определения эквивалентности двух объектов CellPosition.

Пример:

```
local pos1 = DocumentAPI.CellPosition(0,0)  
local pos2 = DocumentAPI.CellPosition(0,0)  
print(pos1:__eq(pos2)) -- true
```

4.8.14 Таблица DocumentAPI.CellRange

Таблица DocumentAPI.CellRange описывает диапазон ячеек таблицы.

Пример:

```
local cellRange = table:getCellRange("B3:C4")
```

4.8.14.1 Метод CellRange:enumerate

Метод возвращает коллекцию ячеек в диапазоне.

Пример:

```
-- Печать значений ячеек в диапазоне B3:C4  
local tbl = document:getBlocks():getTable(0)  
local rng = tbl:getCellRange("B3:C4")  
for cell in rng:enumerate() do  
    print(cell:getFormattedValue())  
end
```

4.8.14.2 Метод `CellRange:getBeginRow`

Метод возвращает индекс строки первой ячейки диапазона. Нумерация строк начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getBeginRow()) -- 2
```

4.8.14.3 Метод `CellRange:getBeginColumn`

Метод возвращает индекс столбца первой ячейки диапазона. Нумерация столбцов начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getBeginColumn()) -- 1
```

4.8.14.4 Метод `CellRange:getLastRow`

Метод возвращает индекс строки последней ячейки диапазона. Нумерация строк начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getLastRow()) -- 3
```

4.8.14.5 Метод `CellRange:getLastColumn`

Метод возвращает индекс столбца последней ячейки диапазона. Нумерация столбцов начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getLastColumn()) -- 2
```

4.8.14.6 Метод `CellRange:setBorders`

Метод предназначен для установки границ диапазона ячеек. Отдельные границы устанавливаются с помощью методов таблицы `DocumentAPI.Borders`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("C3")
--
line_prop = DocumentAPI.LineProperties()
line_prop.style = DocumentAPI.LineStyle_Dash
line_prop.width = 1.5
line_prop.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0, 255))
--
newBorders = DocumentAPI.Borders()
newBorders = newBorders:setLeft(line_prop)
newBorders = newBorders:setRight(line_prop)
newBorders = newBorders:setTop(line_prop)
newBorders = newBorders:setBottom(line_prop)
--
local borders = cell:setBorders(newBorders)
```

4.8.14.7 Метод `CellRange:insertCurrentDateTime`

Метод служит для установки значения даты/времени `DocumentAPI.DateTimeFormat` диапазона ячеек.

4.8.14.8 Метод `CellRange:getCellProperties`

Метод возвращает набор свойств форматирования (`DocumentAPI.CellProperties`) для диапазона ячеек. Возвращаемая структура содержит свойства, общие для всех ячеек диапазона.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
local cellProperties = rng:getCellProperties()
print(cellProperties.backgroundColor.r)
```

4.8.14.9 Метод `CellRange:setCellProperties`

Метод предназначен для установки свойств `DocumentAPI.CellProperties` для всех ячеек диапазона.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
---
local props = DocumentAPI.CellProperties()
props.backgroundColor = DocumentAPI.ColorRGBA(55, 146, 179, 200)
rng:setCellProperties(props)
```

4.8.14.10 Метод `CellRange:merge`

Метод объединяет несколько ячеек таблицы в одну. Группа ячеек (диапазон) формируется с помощью таблицы `CellRange`. Содержимое крайней левой ячейки диапазона помещается в объединенной ячейке.

Пример:

```
-- Объединение ячеек A1 и A2 на первом листе табличного документа
local tbl = document:getBlocks():getTable(0)
tbl:getCellRange("A1:A2"):merge()
```

4.8.14.11 Метод `CellRange:unmerge`

Метод разъединяет ранее объединенные ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
-- Ячейка A1 является результатом объединения диапазона A1:A2
tbl:getCell("A1"):unmerge()
```

4.8.15 Таблица `DocumentApi.DateTimeFormat`

В таблице 68 представлены варианты масштабирования при печати табличных документов. Используется в качестве параметра метода `CellRange.insertCurrentDateTime()`.

Таблица 68 – Варианты масштабирования при печати табличных документов

Наименование константы	Описание
DocumentAPI.DateTimeFormat_DateTime	Дата/время.
DocumentAPI.DateTimeFormat_Date	Дата.
DocumentAPI.DateTimeFormat_Time	Время.

4.8.16 Таблица DocumentAPI.CellRangePosition

Таблица DocumentAPI.CellRangePosition представляет положение диапазона ячеек в таблице. Используется в качестве поля tableRange таблицы DocumentAPI.TableRangeInfo, а также в методах Table.getCellRange(), Chart.setRange() По умолчанию диапазон включает одну ячейку в позиции 0,0 что соответствует верхней левой ячейке таблицы для редактора текста, либо ячейке A1 для редактора таблиц.

Описание полей таблицы DocumentAPI.CellRangePosition представлено в таблице 69.

Таблица 69 – Поля таблицы DocumentAPI.CellRangePosition

Поле	Тип	Описание
topLeft	CellPosition	Позиция левой верхней ячейки таблицы прямоугольного диапазона. Значение 0,0 соответствует верхней левой ячейке таблицы для редактора текста, либо ячейке A1 для редактора таблиц.
bottomRight	CellPosition	Содержит позицию правой нижней ячейки таблицы прямоугольного диапазона.

Примеры:

```

local table = document:getBlocks():getTable(0)
local cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
local range = table:getCellRange(cellRangePosition)

local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)
local cellRangePosition = rangeInfo.tableRangeInfo.tableRange
print("topLeft=", cellRangePosition.topLeft.row,
cellRangePosition.topLeft.column)
print("bottomRight=", cellRangePosition.bottomRight.row,
cellRangePosition.bottomRight.column)

```


4.8.16.1 Метод `CellRangePosition.toString`

Возвращает информацию о диапазоне ячеек в виде строкового значения формата (topLeft: <value>, bottomRight: <value>).

Пример:

```
local cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
print(cellRangePosition.toString()) -- [topLeft: (row: 0, column: 0),
bottomRight: (row: 5, column: 5)]
```

4.8.16.2 Метод `CellRangePosition.__eq`

Метод используется для определения эквивалентности двух объектов `CellRangePosition`.

Пример:

```
local pos1 = DocumentAPI.CellRangePosition(0, 0, 5, 5)
local pos2 = DocumentAPI.CellRangePosition(0, 0, 5, 5)
print(pos1.__eq(pos2)) -- true
```

4.8.17 Таблица `DocumentAPI.FrozenRangePosition`

Таблица `DocumentAPI.FrozenRangePosition` представляет заблокированную область таблицы. Возвращается посредством метода `Table.getFrozenRange()`, устанавливается методом `Table.freeze()`.

4.8.17.1 Конструкторы

Конструктор с параметрами по умолчанию.

```
FrozenRangePosition()
```

Конструктор, создающий диапазон ячеек. В качестве параметров используются координаты левой верхней и правой нижней точек области.

```
FrozenRangePosition(top, left, bottom, right)
```

Примеры:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition()
print(frozenRangePosition.isRowsCols())

frozenRangePosition = DocumentAPI.FrozenRangePosition(0, 2, 5, 5)
print(frozenRangePosition.isRowsCols())
```

4.8.17.2 Метод FrozenRangePosition:create

Создает объект заблокированной области таблицы FrozenRangePosition. В качестве параметров используются координаты левой верхней и правой нижней точек области.

Вызов:

```
FrozenRangePosition create(top, left, bottom, right)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.create(0, 2, 5, 5)  
print(frozenRangePosition.isRowsCols())
```

4.8.17.3 Метод FrozenRangePosition:createFrozenArea

Создает объект заблокированной области таблицы FrozenRangePosition. Область содержит все ячейки прямоугольника {0, 0, bottom, right}.

Вызов:

```
FrozenRangePosition createFrozenArea(bottom, right)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenArea(0, 2)  
print(frozenRangePosition.isArea())
```

4.8.17.4 Метод FrozenRangePosition:createFrozenRows

Создает объект заблокированной области таблицы FrozenRangePosition. Область содержит все строки с first по last.

Вызов:

```
FrozenRangePosition createFrozenRows(first, last)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenRows(0, 2)  
print(frozenRangePosition.isRows())
```

4.8.17.5 Метод FrozenRangePosition:createFrozenCols

Создает объект заблокированной области таблицы FrozenRangePosition. Область содержит все колонки с first по last.

ВЫЗОВ:

```
FrozenRangePosition createFrozenCols(first, last)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)  
print(frozenRangePosition.isRowsCols())
```

4.8.17.6 Метод FrozenRangePosition:isRowsCols

Возвращает true, если диапазон содержит строки и колонки.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenArea(2, 2)  
print(frozenRangePosition.isRowsCols())
```

4.8.17.7 Метод FrozenRangePosition:isArea

Возвращает true, если диапазон является непрерывной областью.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenArea(2, 2)  
print(frozenRangePosition.isArea())
```

4.8.17.8 Метод FrozenRangePosition:isRows

Возвращает true, если диапазон состоит из строк.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenRows(0, 2)  
print(frozenRangePosition.isRows())
```

4.8.17.9 Метод FrozenRangePosition:isCols

Возвращает true, если диапазон состоит из колонок.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)  
print(frozenRangePosition.isCols())
```

4.8.17.10 Метод FrozenRangePosition: __eq

Метод используется для определения эквивалентности двух объектов FrozenRangePosition.

Пример:

```
local pos1 = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
local pos2 = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
print(pos1:__eq(pos2)) -- true
```

4.8.18 Таблица DocumentAPI.TableRangeInfo

Таблица DocumentAPI.TableRangeInfo описывает диапазон ячеек таблицы.

Описание полей таблицы DocumentAPI.TableRangeInfo представлено в таблице 70.

Таблица 70 – Поля таблицы DocumentAPI.TableRangeInfo

Поле	Тип	Описание
tableRange	DocumentAPI.CellRangePosition	Диапазон ячеек

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)
local tableRangeInfo = rangeInfo.tableRangeInfo
local tableRange = tableRangeInfo.tableRange
print("topLeft=", tableRange.topLeft.row, tableRange.topLeft.column)
print("topLeft=", tableRange.bottomRight.row, tableRange.bottomRight.column)
```

4.9 Сводные таблицы

Сводная таблица – инструмент обработки данных, служащий для их обобщения и удобства обработки. Схема взаимодействия объектов, связанных со сводными таблицами, приведена на рисунке 1215.

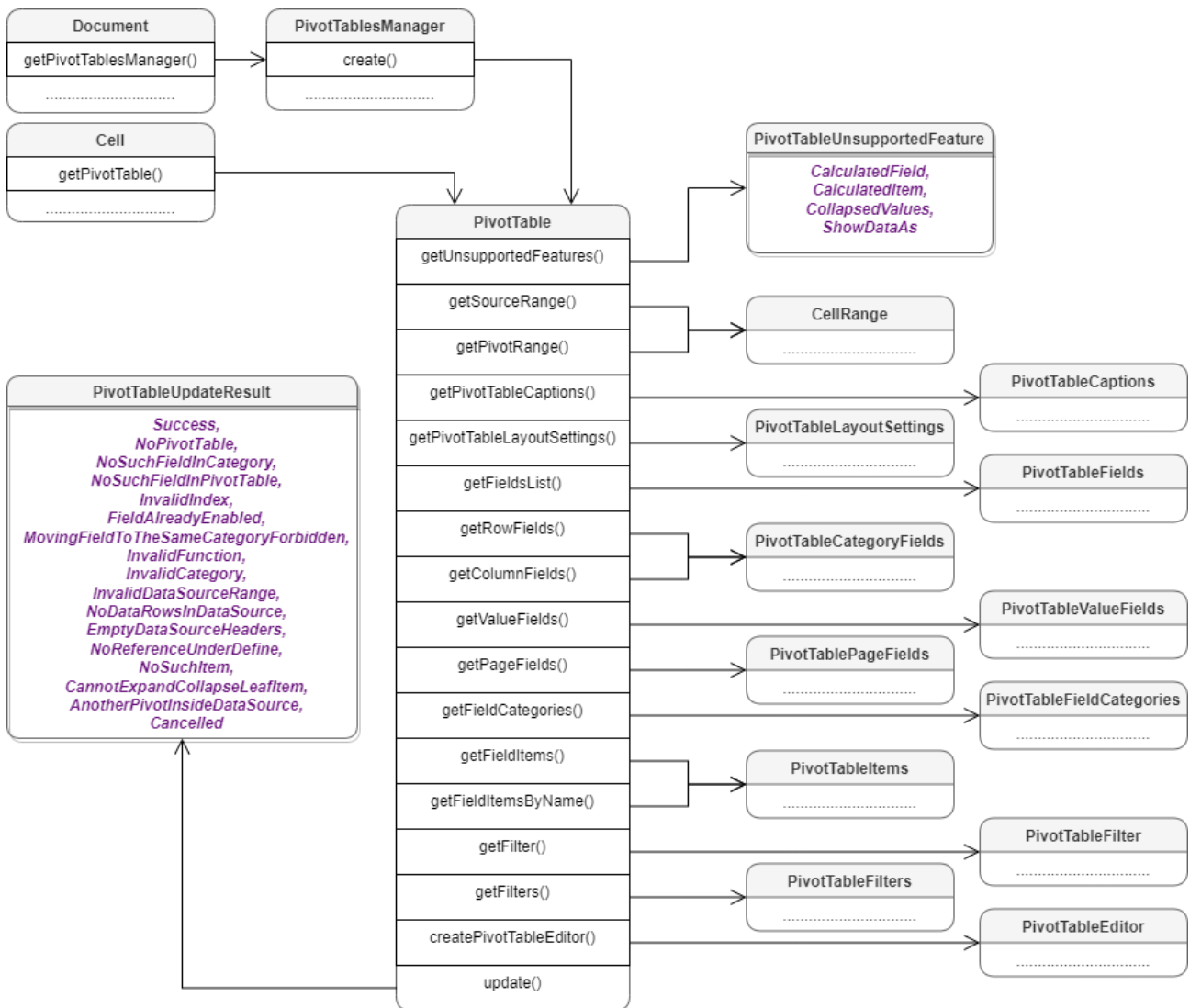


Рисунок 1215 – Сводные таблицы

4.9.1 Таблица DocumentAPI.PivotTablesManager

Таблица PivotTablesManager используется для создания сводных таблиц, содержит метод create(). Может быть получена вызовом Document.getPivotTablesManager().

Пример:

```
local pivotTablesManager = document:getPivotTablesManager()
```

4.9.1.1 Метод PivotTablesManager:create

Метод создает сводную таблицу PivotTable на основе диапазона исходных данных CellRange.

Если местоположение не задано, создается новый лист (таблица), и сводная таблица будет расположена по умолчанию.

Пример:

```
local pivotTablesManager = document:getPivotTablesManager()  
local tbl = document:getBlocks():getTable(0)  
local cellRange = tbl:getCellRange("I3:K7")  
local pivotTable = pivotTablesManager:create(cellRange, tbl:getCell("L8"))
```

4.9.2 Таблица DocumentAPI.PivotTable

Таблица для представления сводной таблицы. Может быть получена из ячейки `Cell.getPivotTable()` или при создании новой сводной таблицы `PivotTablesManager.create()`.

4.9.2.1 Метод PivotTable:remove

Метод удаляет сводную таблицу.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
local cell = tbl:getCell("L8")  
local pivotTable = cell:getPivotTable()  
pivotTable:remove()
```

4.9.2.2 Метод PivotTable:getSourceRangeAddress

Метод возвращает текстовое представление диапазона исходных данных сводной таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)  
local cell = tbl:getCell("L8")  
local pivotTable = cell:getPivotTable()  
print(pivotTable:getSourceRangeAddress()) -- 'Sheet1'!I3:K7
```

4.9.2.3 Метод PivotTable:getSourceRange

Метод возвращает диапазон `DocumentAPI.CellRange` исходных данных сводной таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
local cellRange = pivotTable:getSourceRange()
print(cellRange:getBeginRow(), cellRange:getLastRow()) -- 2 6
```

4.9.2.4 Метод PivotTable:getPivotRange

Метод возвращает диапазон ячеек DocumentAPI.CellRange, в котором размещена сводная таблица.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
local cellRange = pivotTable:getPivotRange()
print(cellRange:getBeginRow(), cellRange:getLastRow()) -- 7 10
```

4.9.2.5 Метод PivotTable:changeSourceRange

Метод позволяет задать новый диапазон исходных данных сводной таблицы без обновления самой таблицы. Параметр sourceRange – строка, представляющая новый диапазон таблицы.

Пример:

```
pivotTable:changeSourceRange("I3:K5")
local cellRange = pivotTable:getSourceRange()
print(cellRange:getBeginRow(), cellRange:getLastRow())
```

4.9.2.6 Метод PivotTable:isRowGrandTotalEnabled

Метод возвращает true, если разрешено показывать общие итоги для строк.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A3")
local pivotTable = cell:getPivotTable()
print(pivotTable:isRowGrandTotalEnabled())
```

4.9.2.7 Метод `PivotTable:isColumnGrandTotalEnabled`

Метод возвращает `true`, если разрешено показывать общие итоги для столбцов.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A3")
local pivotTable = cell:getPivotTable()
print(pivotTable:isColumnGrandTotalEnabled())
```

4.9.2.8 Метод `PivotTable:getPivotTableCaptions`

Метод возвращает информацию `DocumentAPI.PivotTableCaptions` о всех заголовках сводной таблицы.

Пример:

```
local pivotTableCaptions = pivotTable:getPivotTableCaptions()
print(pivotTableCaptions.grandTotalCaption)
print(pivotTableCaptions.valuesHeaderCaption)
print(pivotTableCaptions.rowHeaderCaption)
print(pivotTableCaptions.columnHeaderCaption)
print(pivotTableCaptions.errorCaption)
print(pivotTableCaptions.emptyCaption)
```

4.9.2.9 Метод `PivotTable:getPivotTableLayoutSettings`

Метод возвращает настройки отображения `DocumentAPI.PivotTableLayoutSettings` сводной таблицы.

Пример:

```
local settings = pivotTable:getPivotTableLayoutSettings()
print(settings.reportLayout)
print(settings.valueFieldsOrientation)
print(settings.pageFieldOrder)
print(settings.indentForCompactLayout)
print(settings.pageFieldWrapCount)
```

4.9.2.10 Метод `PivotTable:getUnsupportedFeatures`

Метод возвращает неподдерживаемые свойства `DocumentAPI.PivotTableUnsupportedFeature` сводной таблицы.

Пример:

```
local unsupportedFeatures = pivotTable:getUnsupportedFeatures()  
for featureIndex = 0, unsupportedFeatures:size() - 1 do  
    print(unsupportedFeatures[featureIndex])  
end
```

4.9.2.11 Метод PivotTable:getFieldsList

Метод возвращает список DocumentAPI.PivotTableField всех полей сводной таблицы.

Пример:

```
local fieldsList = pivotTable:getFieldsList()  
print(fieldsList:size())  
for fieldIdx = 0, fieldsList:size() - 1 do  
    print(fieldsList[fieldIdx].fieldProperties.fieldName)  
end
```

4.9.2.12 Метод PivotTable:getRowFields

Метод возвращает список полей DocumentAPI.PivotTableCategoryField из области строк.

Пример:

```
local rowFields = pivotTable:getRowFields()  
for fieldIdx = 0, rowFields:size() - 1 do  
    print(rowFields[fieldIdx].fieldProperties.fieldName)  
end
```

4.9.2.13 Метод PivotTable:getColumnFields

Метод возвращает список полей DocumentAPI.PivotTableCategoryField из области колонок.

Пример:

```
local columnFields = pivotTable:getColumnFields()  
for fieldIdx = 0, columnFields:size() - 1 do  
    print(columnFields[fieldIdx].fieldProperties.fieldName)  
end
```

4.9.2.14 Метод `PivotTable:getValueFields`

Метод возвращает список полей `DocumentAPI.PivotTableValueField` из области значений.

Пример:

```
local valueFields = pivotTable:getValueFields()
for fieldIdx = 0, valueFields:size() - 1 do
    print(valueFields[fieldIdx].baseFieldName)
    print(valueFields[fieldIdx].valueFieldName)
    print(valueFields[fieldIdx].cellNumberFormat)
    print(valueFields[fieldIdx].totalFunction)
end
```

4.9.2.15 Метод `PivotTable:getPageFields`

Метод возвращает список полей `DocumentAPI.PivotTablePageField` из области фильтров.

Пример:

```
local pageFields = pivotTable:getPageFields()
print(pageFields:size())
```

4.9.2.16 Метод `PivotTable:getFieldCategories`

Метод возвращает список категорий `DocumentAPI.PivotTableFieldCategories`, содержащих заданное поле `fieldName`.

Пример:

```
local fieldCategories = pivotTable:getFieldCategories("Age")
```

4.9.2.17 Метод `PivotTable:getFieldItems`

Метод возвращает все элементы `DocumentAPI.PivotTableItems` сводной таблицы по заданному имени поля `fieldName`.

Пример:

```
local pivotTableItems = pivotTable:getFieldItems("Age")
print(pivotTableItems)
```

4.9.2.18 Метод PivotTable:getFieldItemsByName

Метод возвращает все элементы `DocumentAPI.PivotTableItems` из заданного поля `fieldName` по имени `itemName`.

Пример:

```
local pivotTableItemsByName = pivotTable:getFieldItemsByName("Ultimate Question  
of Life", "42")  
print(pivotTableItemsByName)
```

4.9.2.19 Метод PivotTable:getFilter

Метод возвращает фильтр `DocumentAPI.PivotTableFilter` по заданному имени поля `fieldName`.

Пример:

```
local filter = pivotTable:getFilter("Age")  
print(filter:getFieldName())
```

4.9.2.20 Метод PivotTable:getFilters

Метод возвращает список фильтров `DocumentAPI.PivotTableFilter` сводной таблицы.

Пример:

```
local filters = pivotTable:getFilters()  
for filter in filters:enumerate() do  
    -- use filter  
end
```

4.9.2.21 Метод PivotTable:update

Метод обновляет и полностью пересчитывает сводную таблицу, возвращает `DocumentAPI.PivotTableUpdateResult`.

Пример:

```
local updateResult = pivotTable:update()  
if (updateResult ~= DocumentAPI.PivotTableUpdateResult_Success) then  
    print(updateResult)  
end
```

4.9.2.2 Метод `PivotTable.createPivotTableEditor`

Метод возвращает объект `DocumentAPI.PivotTableEditor`, который служит для обновления свойств и редактирования сводной таблицы.

Пример:

```
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
local pivotTableEditor = pivotTable:createPivotTableEditor()
```

4.9.3 Таблица `DocumentAPI.PivotTableCaptions`

Таблица `DocumentAPI.PivotTableCaptions` хранит все пользовательские заголовки сводной таблицы. Описание полей таблицы представлено в таблице 71.

Таблица 71 – Описание полей таблицы `DocumentAPI.PivotTableCaptions`

Поле	Описание
<code>PivotTableCaptions.errorCaption</code>	Алиас для значений, которые возвращают ошибку.
<code>PivotTableCaptions.emptyCaption</code>	Алиас для значений, которые возвращают пустое значение.
<code>PivotTableCaptions.grandTotalCaption</code>	Алиас общих итогов.
<code>PivotTableCaptions.valuesHeaderCaption</code>	Алиас поля из области значений; это поле отображается в отчете в случае, если в сводной таблице наличие более двух полей из области значений, и макет имеют тип 'outline' или 'tabular'.
<code>PivotTableCaptions.rowHeaderCaption</code>	Алиас заголовка строк (виден только при включенном компактном макете, это алиас по умолчанию).
<code>PivotTableCaptions.columnHeaderCaption</code>	Алиас заголовка колонок (виден только при включенном компактном макете, это алиас по умолчанию).

4.9.4 Таблица `DocumentAPI.PivotTableLayoutSettings`

Таблица `DocumentAPI.PivotTableLayoutSettings` содержит настройки отображения сводной таблицы. Данная таблица может быть получена в результате вызова `PivotTable.getPivotTableLayoutSettings()` и установлена методом `PivotTableEditor.setLayoutSettings()`. Описание полей таблицы представлено в таблице 72.

Таблица 72 – Описание полей таблицы DocumentAPI.PivotTableLayoutSettings

Поле	Описание
PivotTableLayoutSettings.reportLayout	Настройка вида макета сводной таблицы (PivotTableReportLayout: компактный, табличный, структурный).
PivotTableLayoutSettings.valueFieldsOrientation	Настраивает положение значений в случае, если в сводной таблице более двух полей значений. Тип – ValueFieldsOrientation.
PivotTableLayoutSettings.pageFieldOrder	Настройка порядка полей фильтров (PageFieldOrder: вниз, затем поперек или сначала поперек, потом вниз).
PivotTableLayoutSettings.indentForCompactLayout	Размер отступа для полей в области строк в компактном макете (режим иерархии в случае наличия более двух полей).
PivotTableLayoutSettings.pageFieldWrapCount	Настройка связана с pageFieldOrder, она показывает через сколько полей будет совершено указанное действие (перенос на следующую строку и т.д).
PivotTableLayoutSettings.isMergeAndCenterLabelsEnabled	Настройка позволяет объединить ячейки заголовков.
PivotTableLayoutSettings.useGridDropZones	Флаг, отвечающий за отображение классического вида (как в Excel 2003). Влияет только на расположение полей в отчете.
PivotTableLayoutSettings.displayFieldCaptions	Флаг, отвечающий за отображение заголовков полей.

4.9.5 Таблица DocumentAPI.PivotTableReportLayout

Таблица DocumentAPI.PivotTableReportLayout описывает внешний вид отчетов сводной таблицы. Является полем таблицы DocumentAPI.PivotTableLayoutSettings. Описание полей таблицы представлено в таблице 73.

Таблица 73 – Описание полей таблицы DocumentAPI.PivotTableReportLayout

Поле	Описание
DocumentAPI.PivotTableReportLayout_Compact	Компактный вид
DocumentAPI.PivotTableReportLayout_Tabular	Табличный вид
DocumentAPI.PivotTableReportLayout_Outline	Структурный вид

4.9.6 Таблица `DocumentAPI.ValueFieldsOrientation`

Таблица `DocumentAPI.ValueFieldsOrientation` описывает варианты ориентации в случае, когда в сводной таблице более, чем одно поле из области значений. Является полем таблицы `DocumentAPI.PivotTableLayoutSettings`. Описание полей таблицы представлено в таблице 74.

Таблица 74 – Описание полей таблицы `DocumentAPI.ValueFieldsOrientation`

Поле	Описание
<code>DocumentAPI.ValueFieldsOrientation_ByRows</code>	По строкам
<code>DocumentAPI.ValueFieldsOrientation_ByColumns</code>	По столбцам

4.9.7 Таблица `DocumentAPI.PageFieldOrder`

Таблица `DocumentAPI.PageFieldOrder` описывает вид отображения полей из области фильтров. Является полем таблицы `DocumentAPI.PivotTableLayoutSettings`. Описание полей таблицы представлено в таблице 75.

Таблица 75 – Описание полей таблицы `DocumentAPI.PageFieldOrder`

Поле	Описание
<code>DocumentAPI.PageFieldOrder_DownThenOver</code>	Вниз, затем поперек
<code>DocumentAPI.PageFieldOrder_OverThenDown</code>	Поперек, затем вниз

4.9.8 Таблица `DocumentAPI.PivotTableUnsupportedFeature`

Таблица `DocumentAPI.PivotTableUnsupportedFeature` описывает неподдерживаемую функциональность сводных таблиц. Получение неподдерживаемой функциональности сводных таблиц описано в `PivotTable:getUnsupportedFeatures()`. Описание полей таблицы представлено в таблице 76.

Таблица 76 – Описание полей таблицы `DocumentAPI.PivotTableUnsupportedFeature`

Поле	Описание
<code>DocumentAPI.PivotTableUnsupportedFeature_CalculatedField</code>	Вычисляемые поля
<code>DocumentAPI.PivotTableUnsupportedFeature_CalculatedItem</code>	Вычисляемые элементы
<code>DocumentAPI.PivotTableUnsupportedFeature_CollapsedValues</code>	Свернутые поля

Поле	Описание
DocumentAPI.PivotTableUnsupportedFeature_ShowDataAs	Вычисления (Show data как в MS Excel)

4.9.9 Таблица DocumentAPI.PivotTableFieldCategories

Класс обеспечивает доступ к списку категорий поля сводной таблицы. Может быть получена посредством использования метода `PivotTable.getFieldCategories()`.

4.9.9.1 Метод PivotTableFieldCategories:enumerate

Метод для перечисления категорий поля `DocumentAPI.PivotTableFieldCategory`.

Пример:

```
local fieldCategories = pivotTable:getFieldCategories("Age")
for fieldCategory in fieldCategories:enumerate() do
    print(fieldCategory)
end
```

4.9.10 Таблица DocumentAPI.PivotTableFunction

Таблица `DocumentAPI.PivotTableFunction` описывает функции, которые могут быть использованы в сводных таблицах. Описание полей таблицы представлено в таблице 77. Таблица используется в качестве поля `subtotalFunctions` таблицы `DocumentAPI.PivotTableCategoryField`.

Таблица 77 – Описание полей таблицы `DocumentAPI.PivotTableFunction`

Поле	Описание
DocumentAPI.PivotTableFunction_Auto	Автозаполнение
DocumentAPI.PivotTableFunction_Sum	Суммирует все числовые данные
DocumentAPI.PivotTableFunction_Count	Количество всех ячеек
DocumentAPI.PivotTableFunction_CountNums	Количество числовых ячеек
DocumentAPI.PivotTableFunction_Average	Среднее значение
DocumentAPI.PivotTableFunction_Max	Наибольшее значение
DocumentAPI.PivotTableFunction_Min	Наименьшее значение
DocumentAPI.PivotTableFunction_Product	Произведение всех ячеек
DocumentAPI.PivotTableFunction_StdDeviation	Стандартное смещенное отклонение

Поле	Описание
DocumentAPI.PivotTableFunction_StdDeviationPopulation	Стандартное несмещенное отклонение
DocumentAPI.PivotTableFunction_Variance	Смещенная дисперсия
DocumentAPI.PivotTableFunction_VariancePopulation	Несмещенная дисперсия

4.9.11 Таблица DocumentAPI.PivotTableFilters

Таблица обеспечивает доступ к списку фильтров. Для получения DocumentAPI.PivotTableFilters используется метод PivotTable.getFilters().

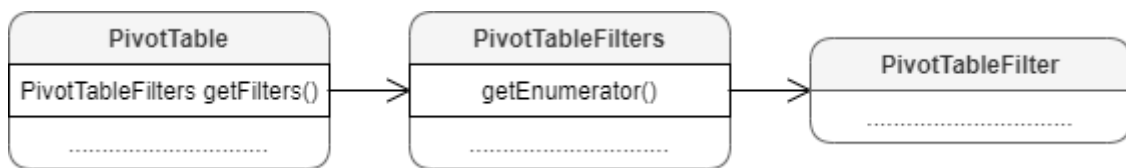


Рисунок 1216 – Объектная модель таблиц для работы с фильтрами

4.9.11.1 Метод PivotTableFilters:enumerate

Метод используется для доступа к коллекции фильтров (см. раздел 4.9.12).

Пример:

```

local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
    print(filter:getName(0))
    print(filter:getFieldName())
end
    
```

4.9.12 Таблица DocumentAPI.PivotTableFilter

Позволяет осуществить доступ к списку фильтров таблицы, каждый из которых обладает свойством видимости. При любом изменении фильтров они должны быть применены к сводной таблице посредством использования методов PivotTableEditor.setFilter(), PivotTableEditor.setFilters().

Пример:

```

local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
    for filterIdx = 0, filter:getCount() - 1 do
        filter:setHidden(filterIdx, false)
    end
end
    
```



```
end  
end  
pivotTableEditor:setFilters(filters)  
pivotTableEditor:apply()
```

4.9.12.1 Метод PivotTableFilter:getFieldName

Возвращает имя поля, с которым ассоциирован фильтр.

Пример:

```
local filters = pivotTable:getFilters()  
for filter in filters:enumerate() do  
  for filterIdx = 0, filter:getCount() - 1 do  
    print(filter:getFieldName())  
  end  
end  
end
```

4.9.12.2 Метод PivotTableFilter:getCount

Возвращает количество фильтруемых полей.

Пример:

```
local filters = pivotTable:getFilters()  
for filter in filters:enumerate() do  
  print(filter:getCount())  
end  
end
```

4.9.12.3 Метод PivotTableFilter:getName

Возвращает имя поля для заданного индекса.

Пример:

```
local filters = pivotTable:getFilters()  
for filter in filters:enumerate() do  
  for filterIdx = 0, filter:getCount() - 1 do  
    print(filter:getName(filterIdx))  
  end  
end  
end
```

4.9.12.4 Метод PivotTableFilter:isHidden

Возвращает видимость поля для заданного индекса `itemIndex`. Если `true`, то поле скрыто.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
    for filterIdx = 0, filter:getCount() - 1 do
        print(filter:isHidden(filterIdx))
    end
end
```

4.9.12.5 Метод PivotTableFilter:setHidden

Устанавливает видимость поля для заданного индекса. Параметры: `itemName` – индекс поля, `hidden` – видимость (`true` – поле скрыто).

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
    for filterIdx = 0, filter:getCount() - 1 do
        print(filter:setName(filterIdx, false))
    end
end
```

4.9.13 Таблица DocumentAPI.PivotTableField

Таблица `DocumentAPI.PivotTableField` содержит свойства полей сводной таблицы (см. Таблица 78). Таблица может быть получена посредством вызова `PivotTable:getFieldsList()`.

Таблица 78 – Описание полей таблицы `DocumentAPI.PivotTableField`

Поле	Описание
<code>PivotTableField.fieldProperties</code>	Свойства полей сводной таблицы <code>PivotTableFieldProperties</code> .
<code>PivotTableField.fieldCategories</code>	Категории полей сводной таблицы <code>PivotTableFieldCategories</code> .
<code>PivotTableField.customFormula</code>	Вычисляемая формула (строка).

4.9.14 Таблица `DocumentAPI.PivotTableFieldProperties`

`DocumentAPI.PivotTableFieldProperties` содержит свойства поля `DocumentAPI.PivotTableField` сводной таблицы (см. Таблица 79).

Таблица 79 – Описание полей таблицы `DocumentAPI.PivotTableFieldProperties`

Поле	Описание
<code>PivotTableFieldProperties.fieldName</code>	Имя поля
<code>PivotTableFieldProperties.fieldAlias</code>	Псевдоним поля (пользовательское имя)
<code>PivotTableFieldProperties.subtotalAlias</code>	Псевдоним подытогов конкретного поля

4.9.15 Таблица `DocumentAPI.PivotTableCategoryField`

`DocumentAPI.PivotTableCategoryField` содержит свойства поля сводной таблицы, используемого как строка / столбец (см. Таблица 80). Таблица может быть получена посредством вызовов `PivotTable.getRowFields()`, `PivotTable.getColumnFields()`.

Таблица 80 – Описание полей таблицы `DocumentAPI.PivotTableCategoryField`

Поле	Описание
<code>PivotTableCategoryField.fieldProperties</code>	Свойства поля <code>PivotTableFieldProperties</code> .
<code>PivotTableCategoryField.subtotalFunctions</code>	Список функций <code>PivotTableFunction</code> для вычисления подытога.

4.9.16 Таблица `DocumentAPI.PivotTableValueField`

`DocumentAPI.PivotTableValueField` содержит свойства поля сводной таблицы, используемого как значение столбец (см. Таблица 81). Таблица может быть получена посредством вызова `PivotTable.getValueFields()`.

Таблица 81 – Описание полей таблицы `DocumentAPI.PivotTableValueField`

Поле	Описание
<code>PivotTableValueField baseFieldName</code>	Оригинальное поле на основе которого было создано данное поле, тип – строка.
<code>PivotTableValueField valueFieldName</code>	Автоматический уникальный псевдоним такой как “Sum of %имя поля%“, тип –

Поле	Описание
	строка.
PivotTableValueField cellNumberFormat	Числовой формат типа CellFormat для конкретного поля значений.
PivotTableValueField totalFunction	Агрегирующая функция PivotTableFunction поля значений (SUM, COUNT, MAX и т.д).
PivotTableValueField customFormula	Вычисляемая формула для поля значений, тип – строка.

4.9.17 Таблица DocumentAPI.PivotTablePageField

Содержит свойства поля из области фильтров (см. Таблица 82). Таблица может быть получена посредством вызова PivotTable:getPageFields().

Таблица 82 – Описание полей таблицы DocumentAPI.PivotTablePageField

Поле	Описание
PivotTablePageField.fieldProperties	Свойства поля PivotTableFieldProperties

4.9.18 Таблица DocumentAPI.PivotTableItems

Таблица обеспечивает доступ к списку элементов сводной таблицы (см. Рисунок 1217).

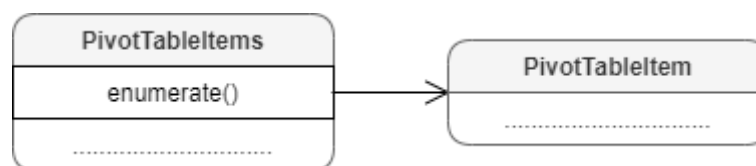


Рисунок 1217 – Объектная модель таблиц для работы с элементами сводных таблиц

4.9.18.1 Метод PivotTableItems:enumerate

Используется для перечисления элементов сводной таблицы.

Пример:

```

local fieldItems = pivotTable:getFieldItems("Age")
for fieldItem in fieldItems:enumerate() do
    print(fieldItem:getName())
end
    
```

```
print(fieldItem:getAlias())  
print(fieldItem:getItemType())  
print(fieldItem:isCollapsed())  
end
```

4.9.19 Таблица DocumentAPI.PivotTableItem

DocumentAPI.PivotTableItem описывает элемент сводной таблицы (см. Рисунок 1218). См. пример в разделе 4.9.18.1.

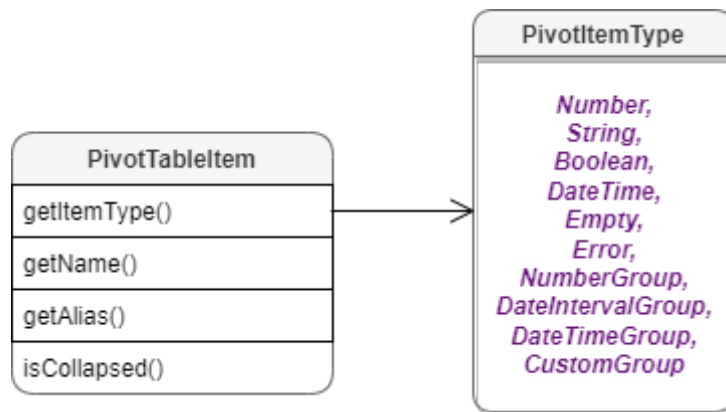


Рисунок 1218 – Таблица DocumentAPI.PivotTableItem

4.9.19.1 Метод PivotTableItem:getName

Метод возвращает имя элемента сводной таблицы, тип - строка. См. пример в разделе 4.9.18.1.

4.9.19.2 Метод PivotTableItem:getAlias

Метод возвращает псевдоним элемента (идентификатор, созданный пользователем), тип – строка. См. пример в разделе 4.9.18.1.

4.9.19.3 Метод PivotTableItem:getItemType

Метод возвращает тип `DocumentAPI.PivotTableItemType` элемента сводной таблицы. См. пример в разделе 4.9.18.1.

4.9.19.4 Метод PivotTableItem:isCollapsed

Метод возвращает true, если элемент сводной таблицы свернут. См. пример в разделе 4.9.18.1.

4.9.20 Таблица DocumentAPI.PivotTableItemType

Таблица DocumentAPI.PivotTableItemType содержит возможные типы элементов сводной таблицы. Описание полей таблицы представлено в таблице 83.

Таблица 83 – Описание полей таблицы DocumentAPI.PivotTableItemType

Поле	Описание
DocumentAPI.PivotTableItemType_Number	Числовой.
DocumentAPI.PivotTableItemType_String	Строковый.
DocumentAPI.PivotTableItemType_Boolean	Логический.
DocumentAPI.PivotTableItemType_DateTime	Дата / время.
DocumentAPI.PivotTableItemType_Empty	Пустой тип.
DocumentAPI.PivotTableItemType_Error	Ошибка.
DocumentAPI.PivotTableItemType_NumberGroup	Интервальная группировка.
DocumentAPI.PivotTableItemType_DateIntervalGroup	Интервальная группировка по датам.
DocumentAPI.PivotTableItemType_DateTimeGroup	Группировка по дате / времени.
DocumentAPI.PivotTableItemType_CustomGroup	Пользовательская (произвольная) группировка.

Пример:

```
local fieldItems = pivotTable:getFieldItems("Age")
for fieldItem in fieldItems:enumerate() do
    if (fieldItem:getItemType() == DocumentAPI.PivotTableItemType_Number) then
        print("Numeric type")
    end
end
```

4.9.21 Таблица DocumentAPI.PivotTableEditor

Предназначена для редактирования сводных таблиц. Возвращается посредством метода PivotTable:createPivotTableEditor().

4.9.21.1 Метод PivotTableEditor:addField

Метод добавляет новое поле в сводную таблицу, используя параметры:

- `fieldName` – имя поля;
- `toCategory` – категория поля (тип – `DocumentAPI.PivotTableFieldCategory`);
- `index` – позиция в категории.

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor.addField("CC",  
DocumentAPI.PivotTableFieldCategory_Values)  
pivotTableEditor.apply()
```

4.9.21.2 Метод PivotTableEditor:moveField

Метод перемещает поле между категориями.

Параметры:

- `fieldName` – имя поля;
- `toCategory` – область, в которую перемещается поле (тип – `DocumentAPI.PivotTableFieldCategory`);
- `index` – позиция в новой категории.

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor.moveField("BB",  
DocumentAPI.PivotTableFieldCategory_Values, 0)  
pivotTableEditor.apply()
```

4.9.21.3 Метод PivotTableEditor:removeField

Метод удаляет поле из категории.

Параметры:

- `fieldName` – имя поля,
- `fromCategory` – область, из которой удаляется поле (тип – `DocumentAPI.PivotTableFieldCategory`).

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor.removeField("Age",  
DocumentAPI.PivotTableFieldCategory_Values)  
pivotTableEditor.apply()
```

4.9.21.4 Метод `PivotTableEditor:reorderField`

Метод изменяет позицию поля в пределах категории.

Параметры:

- `fieldName` – имя поля;
- `category` – область (тип – `DocumentAPI.PivotTableFieldCategory`);
- `toIndex` – новая позиция поля.

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor.reorderField("Age",  
DocumentAPI.PivotTableFieldCategory_Values, 0)  
pivotTableEditor.apply()
```

4.9.21.5 Метод `PivotTableEditor:enableField`

Метод добавляет поле в область, зависящую от типа поля. Параметр `fieldName` – имя поля. Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
local pivotTableEditor = pivotTable.createPivotTableEditor()  
pivotTableEditor.enableField("Age")  
pivotTableEditor.apply()
```

4.9.21.6 Метод `PivotTableEditor:disableField`

Метод удаляет поле из всех областей. Параметр `fieldName` – имя поля (тип – строка). Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
local pivotTableEditor = pivotTable:createPivotTableEditor()  
pivotTableEditor:disableField("Age")  
pivotTableEditor:apply()
```

4.9.21.7 Метод PivotTableEditor:setSummarizeFunction

Метод задает суммирующую функцию для поля из области значений.

Параметры:

- valueFieldName – имя поля (тип – строка);
- summarizeFunction – суммирующая функция, тип – DocumentAPI.PivotTableFunction.

Метод возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
pivotTableEditor = pivotTableEditor:setSummarizeFunction("Age",  
PivotTableFunction_Sum)  
pivotTableEditor:apply()
```

4.9.21.8 Метод PivotTableEditor:setFilter

Метод задает фильтр DocumentAPI.PivotTableFilter сводной таблицы. Если фильтр не может быть применен, вызывается исключение PivotTableError. Метод возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
local filters = pivotTable:getFilters()  
for filter in filters:enumerate() do  
  for filterIdx = 0, filter:getCount() - 1 do  
    filter:setHidden(filterIdx, false)  
    pivotTableEditor:setFilter(filter)  
  end  
end  
pivotTableEditor:apply()
```

4.9.21.9 Метод PivotTableEditor:setFilters

Метод задает фильтры DocumentAPI.PivotTableFilters сводной таблицы. Если какой-то из фильтров не может быть применен, он пропускается. Метод возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
    for filterIdx = 0, filter:getCount() - 1 do
        filter:setHidden(filterIdx, false)
    end
end
pivotTableEditor:setFilters(filters)
pivotTableEditor:apply()
```

4.9.21.10 Метод PivotTableEditor:setCaptions

Метод задает заголовки сводной таблицы DocumentAPI.PivotTableCaptions, возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
local pivotTableCaptions = pivotTable:getPivotTableCaptions()
pivotTableCaptions.grandTotalCaption = "Общий итог за год"

local pivotTableEditor = pivotTable:createPivotTableEditor()
pivotTableEditor = pivotTableEditor:setCaptions(pivotTableCaptions)
pivotTableEditor:apply()
```

4.9.21.11 Метод PivotTableEditor:setLayoutSettings

Метод	устанавливает	настройки	отображения
DocumentAPI.PivotTableLayoutSettings	сводной	таблицы,	возвращает объект

DocumentAPI.PivotTableEditor.

Пример:

```
local layoutSettings = pivotTable:getPivotTableLayoutSettings()
layoutSettings.reportLayout = DocumentAPI.PivotTableReportLayout_Tabular

local pivotTableEditor = pivotTable:createPivotTableEditor()
```

```
pivotTableEditor = pivotTableEditor:setLayoutSettings(layoutSettings)  
pivotTableEditor:apply()
```

4.9.21.12 Метод PivotTableEditor:setGrandTotalSettings

Метод задает настройки отображения общего итога.

Параметры:

- isRowGrandTotalEnabled – показывать общие итоги для строк;
- isColGrandTotalEnabled – показывать общие итоги для столбцов.

Пример:

```
local pivotTableEditor = pivotTable:createPivotTableEditor()  
pivotTableEditor:setGrandTotalSettings(true, true)
```

4.9.21.13 Метод PivotTableEditor:apply

Метод обновляет сводную таблицу с заданными свойствами и возвращает результат DocumentAPI.PivotTableUpdateResult.

Пример:

```
local pivotTableEditor = pivotTable:createPivotTableEditor()  
if DocumentAPI.PivotTableUpdateResult_Success == pivotTableEditor:apply() then  
    print("Successfully applied");  
end
```

4.9.22 Таблица DocumentAPI.PivotTableUpdateResult

В таблице 84 приведены константы, которые соответствуют возможным результатам обновления сводной таблицы (см. разделы 4.9.2.21, 4.9.21.13).

Таблица 84 – Результаты обновления сводной таблицы

Наименование константы	Описание
DocumentAPI.PivotTableUpdateResult_Success	Успешное обновление таблицы
DocumentAPI.PivotTableUpdateResult_NoPivotTable	Сводная таблица не найдена
DocumentAPI.PivotTableUpdateResult_NoSuchFieldInCategory	Не найдено поле в категории
DocumentAPI.PivotTableUpdateResult_	Не найдено поле в сводной таблице

Наименование константы	Описание
NoSuchFieldInPivotTable	
DocumentAPI.PivotTableUpdateResult_InvalidIndex	Ошибка в индексе
DocumentAPI.PivotTableUpdateResult_FieldAlreadyEnabled	Поле уже существует
DocumentAPI.PivotTableUpdateResult_MovingFieldToTheSameCategoryForbidden	Попытка перемещения поля в рамках текущей категории
DocumentAPI.PivotTableUpdateResult_InvalidFunction	Неправильная функция
DocumentAPI.PivotTableUpdateResult_InvalidCategory	Неправильная область
DocumentAPI.PivotTableUpdateResult_InvalidDataSourceRange	Ошибка диапазона исходных данных
DocumentAPI.PivotTableUpdateResult_NoDataRowsInDataSource	В исходных данных нет строк с данными
DocumentAPI.PivotTableUpdateResult_EmptyDataSourceHeaders	Пустые заголовки исходных данных
DocumentAPI.PivotTableUpdateResult_NoReferenceUnderDefine	Попытка обновить или создать сводную таблицу на именованном диапазоне который не содержит ссылку, а содержит константу
DocumentAPI.PivotTableUpdateResult_NoSuchItem	Элемент не найден
DocumentAPI.PivotTableUpdateResult_CannotExpandCollapseLeafItem	Не удастся раскрыть свернутый элемент
DocumentAPI.PivotTableUpdateResult_AnotherPivotInsideDataSource	Найдена другая сводная таблица в этом же диапазоне
DocumentAPI.PivotTableUpdateResult_Canceled	Обновление сводной таблицы отменено

4.9.23 Таблица DocumentAPI.PivotTableFieldCategory

Таблица DocumentAPI.PivotTableFieldCategory описывает флаги, которые задают категорию области полей. Описание полей таблицы представлено в таблице 85.

Таблица 85 – Описание полей таблицы DocumentAPI.PivotTableFieldCategory

Поле	Описание
DocumentAPI.PivotTableFieldCategory_Pages	Область фильтров

Поле	Описание
DocumentAPI.PivotTableFieldCategory_Rows	Область строк
DocumentAPI.PivotTableFieldCategory_Columns	Область колонок
DocumentAPI.PivotTableFieldCategory_Values	Область значений

4.10 Графические объекты

Редакторы текста и таблиц МойОфис поддерживают несколько типов графических объектов со схожим поведением: изображения (DocumentAPI.Image) и фигуры (DocumentAPI.Shape).

Объектная модель документа в части управления изображениями развивается и дополняется возможностями. Доступны следующие операции:

- Перечисление графических объектов, находящихся в документе, определение их типа и геометрических размеров.
- Вставка изображений в текстовый документ. Место вставки определяется типом Position.
- Перемещение графических объектов, изменение их размеров и масштаба.

Перечисление графических объектов в текстовом документе.

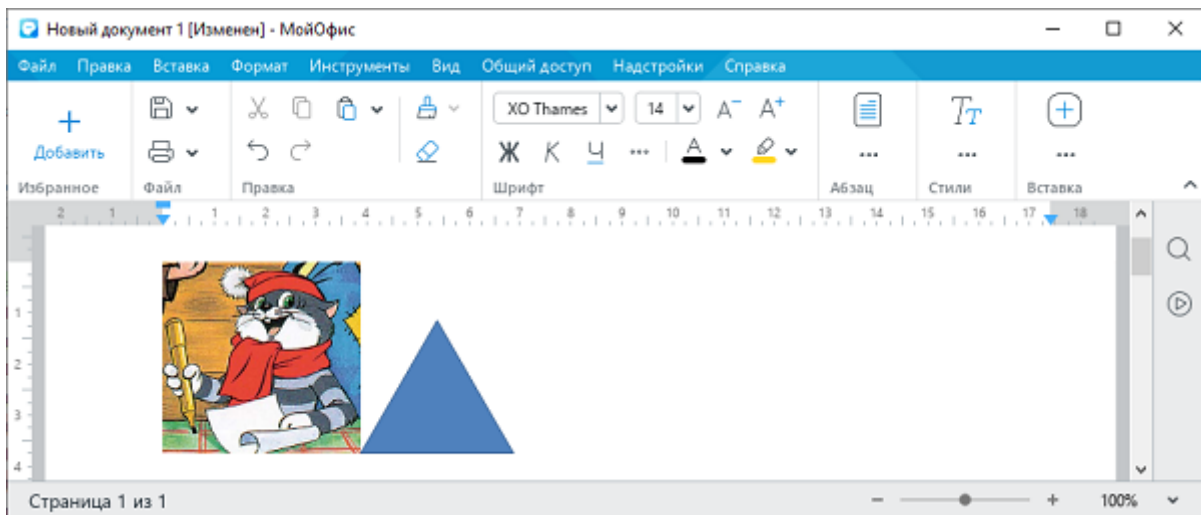


Рисунок 1219 – Графические объекты в текстовом документе

Вариант 1: перечисление графических объектов в текстовом документе

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
    image = mediaObject:toImage()
    if image then
        print("Image: ", image)
```

```
else
    print("Shape: ", mediaObject)
end
end

> Image: <userdata of type 'CO::API::Document::Image *' ..... >
> Shape: <userdata of type 'CO::API::Document::MediaObject *' ..... >
```

Вариант 2: перечисление изображений в текстовом документе

```
local images = document:getRange():getImages()
for image in images:enumerate() do
    print("Image: ", image)
end

> Image: <userdata of type 'CO::API::Document::Image *' ..... >
```

Перечисление графических объектов в табличном документе

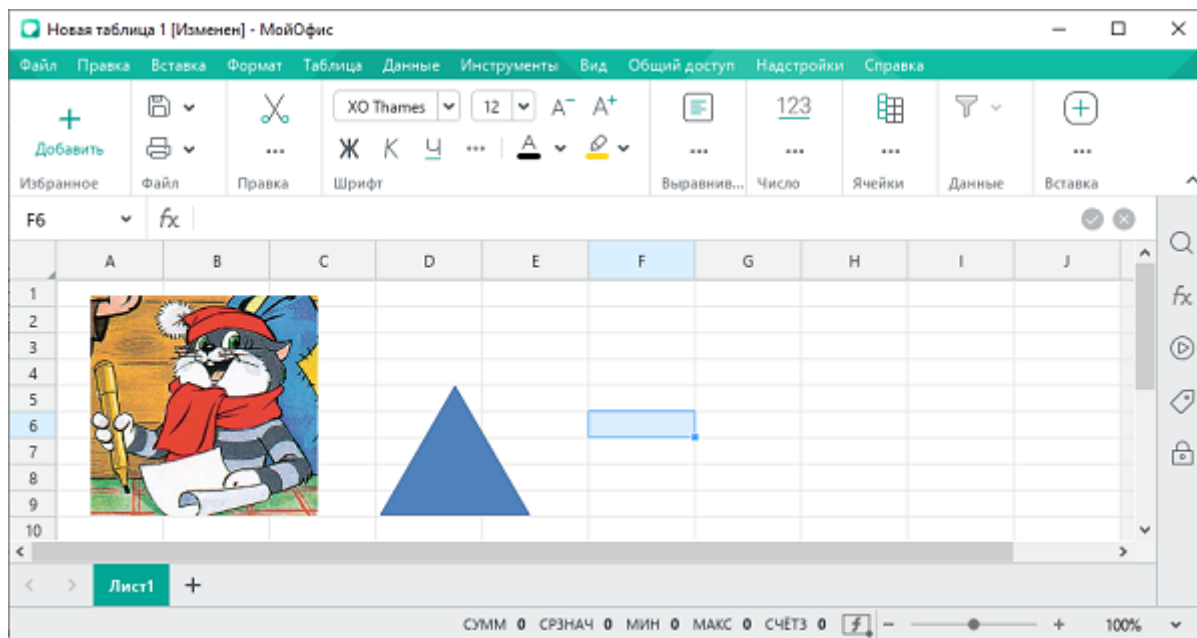


Рисунок 1220 – Графические объекты в табличном документе

Вариант 1: перечисление графических объектов в табличном документе

```
local tbl = document:getBlocks():getTable(0)
local mediaObjects = tbl:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
    image = mediaObject:toImage()
    if image then
        print("Image: ", image)
    else
        print("Shape", mediaObject)
    end
end
```

```
end  
end  
  
> Image: <userdata of type 'CO::API::Document::Image *' ..... >  
> Shape: <userdata of type 'CO::API::Document::MediaObject *' ..... >
```

Вариант 2: перечисление изображений в табличном документе

```
images = document:getBlocks():getTable(0):getImages()  
for image in images:enumerate() do  
    print("Image: ", image)  
end  
  
> Image: <userdata of type 'CO::API::Document::Image *' ..... >
```

Перечисление графических объектов в таблицах текстового документа

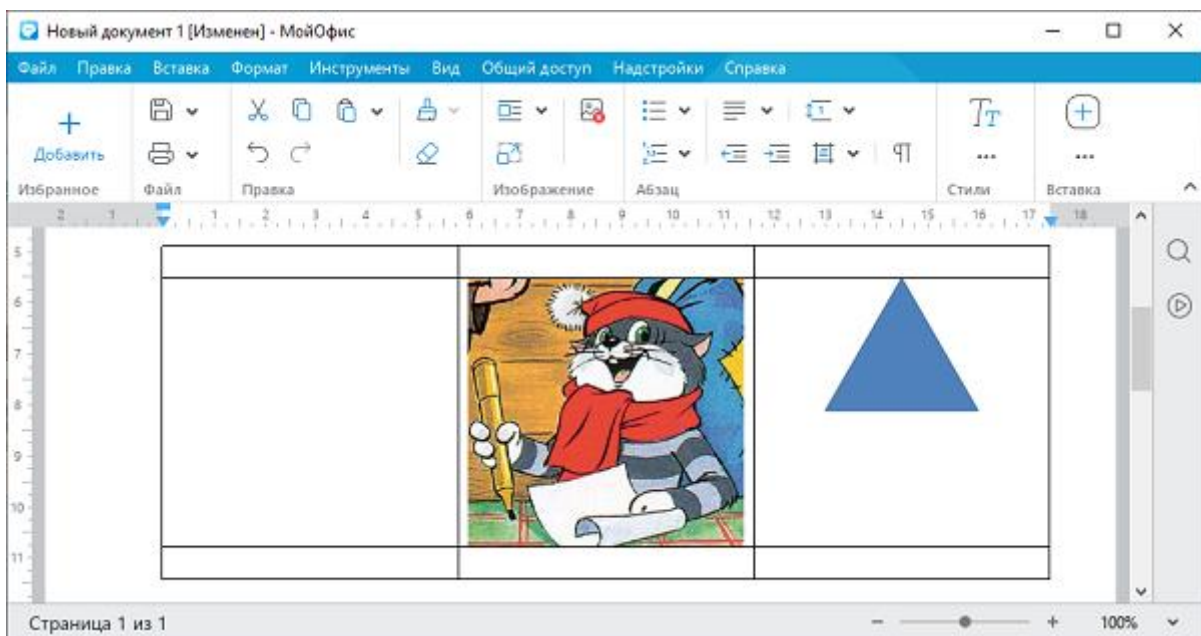


Рисунок 1221 – Графические объекты в таблице текстового документа

Вариант 1: перечисление графических объектов в таблице текстового документа

```
local table = document:getBlocks():getTable(0)  
local mediaObjects = table:getMediaObjects()  
for mediaObject in mediaObjects:enumerate() do  
    local image = mediaObject:toImage()  
    if image then  
        print("Image: ", image)  
    else  
        print("Shape", mediaObject)  
    end  
end
```

```
> Image: <userdata of type 'CO::API::Document::Image *' ..... >  
> Shape: <userdata of type 'CO::API::Document::MediaObject *' ..... >
```

Вариант 2: перечисление изображений в таблице текстового документа

```
images = document:getBlocks():getTable(0):getImages()  
for image in images:enumerate() do  
    print("Image: ", inlineFrame)  
end  
  
> Image: <userdata of type 'CO::API::Document::Image *' ..... >
```

Стоит обратить внимание на то, что графический объект обладает свойством `frame`, описывающим позицию, размеры и выравнивание. Данное свойство возвращается посредством методов `MediaObject:getFrame()` или `Image:getFrame()`. В текстовом документе данный метод возвращает тип `DocumentAPI.InlineFrame`, в табличном документе возвращается `DocumentAPI.AbsoluteFrame`.

Вставка изображения в текстовый документ

Вариант 1: вставка изображения в позицию диапазона текстового документа

```
local range = document:getRange()  
local imageSize = DocumentAPI.SizeU(50, 50)  
range:getBegin():insertImage("C://Tmp/123.jpg", imageSize)
```

Вариант 2: вставка изображения в ячейку таблицы текстового документа

```
local table = document:getBlocks():getTable(0)  
local cell = table:getCell("A1")  
local range = cell:getRange()  
local imageSize = DocumentAPI.SizeU(50, 50)  
range:getBegin():insertImage("C://Tmp/123.jpg", imageSize)
```

Вставка изображения в табличный документ

В текущей версии не поддерживается.

4.10.1 Таблица `DocumentAPI.MediaObjects`

Таблица `DocumentAPI.MediaObjects` предназначен для доступа к коллекции графических объектов. Может быть получена вызовом методов `Table.getMediaObjects()` или `Range.getInlineObjects()` (см. Рисунок 1222).

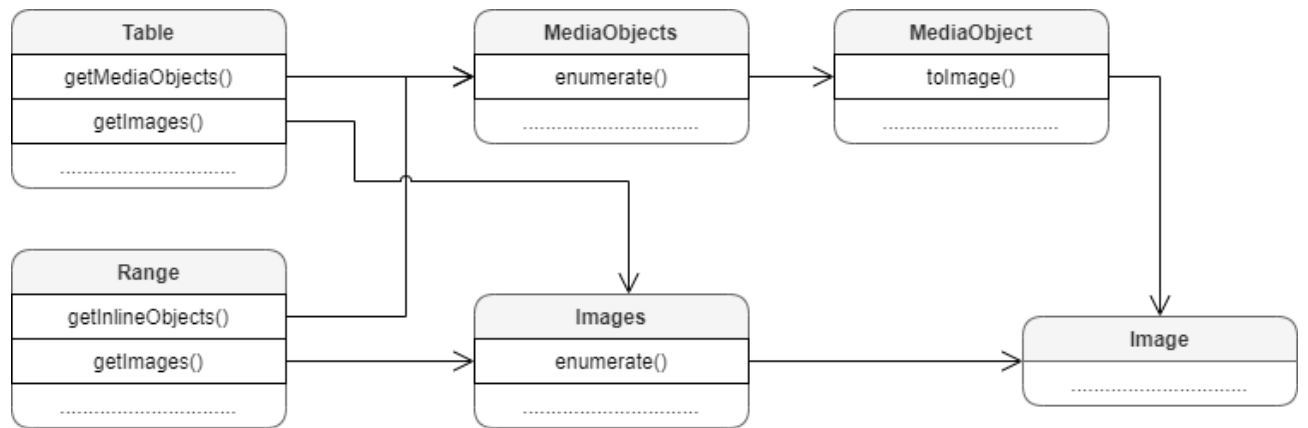


Рисунок 1222 – Графические объекты

4.10.1.1 Метод MediaObjects:enumerate

Метод позволяет перечислить коллекцию встроенных объектов.

Примеры для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
    print("Встроенный объект:", mediaObject)
end

local mediaObjects = EditorAPI.getSelection():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
    print("Встроенный объект:", mediaObject)
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
    local image = mediaObject:toImage()
    if image ~= nil then
        print("Объект является изображением")
    else
        print("Объект является фигурой")
    end
end
end
```

4.10.2 Таблица DocumentAPI.MediaObject

Таблица `DocumentAPI.MediaObject` представляет собой встроенный объект документа.

4.10.2.1 Метод `MediaObject:toImage`

Метод возвращает изображение `DocumentAPI.Image`, связанное со встроенным объектом. Если объект не является изображением, метод возвращает `nil`.

Пример для текстового документа:

```
for mediaObject in document:getRange():getInlineObjects():enumerate() do
  local image = mediaObject:toImage()
  if image then
    print("Текущий объект является изображением")
  else
    print("Текущий объект является фигурой")
  end
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
  local image = mediaObject:toImage()
  if image ~= nil then
    print("Текущий объект является изображением")
  else
    print("Текущий объект является фигурой")
  end
end
```

4.10.2.2 Метод `MediaObject:getFrame`

Метод возвращает свойства позиции встроенного объекта. В зависимости от текущего редактора метод возвращает разные типы таблиц. Графические объекты текстового редактора привязаны к позиции в документе, поэтому для описания местоположения и размеров используют таблицу `DocumentAPI.InlineFrame`, табличные документы работают с абсолютной позицией и используют таблицу `DocumentAPI.AbsoluteFrame`.

Пример для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
    print(mediaObject:getFrame()) -- <userdata of type
'CO::API::Document::InlineFrame'>
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
    print(mediaObject:getFrame()) -- <userdata of type
'CO::API::Document::AbsoluteFrame'>
end
```

4.10.3 Таблица DocumentAPI.Image

Таблица DocumentAPI.Image представляет собой изображение, находящееся в текстовом или табличном документе.

4.10.3.1 Метод Image:getFrame

Метод аналогичен методу MediaObject:getFrame(), он возвращает свойства позиции изображения. В зависимости от текущего редактора метод возвращает разные типы рамок. Графические объекты текстового редактора привязаны к позиции в документе, поэтому для описания местоположения и размеров используют тип DocumentAPI.InlineFrame, табличные документы работают с абсолютной позицией и используют тип DocumentAPI.AbsoluteFrame.

Пример для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
    local image = mediaObject:toImage()
    if (image) then
        print(image:getFrame()) -- <userdata of type
'CO::API::Document::InlineFrame'>
    end
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
    local image = mediaObject:toImage()
    if (image) then
        print(image:getFrame()) -- <userdata of type
'CO::API::Document::AbsoluteFrame'>
    end
end
```

4.10.3.2 Метод Image:remove

Метод удаляет изображение из документа.

Пример для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
    local image = mediaObject:toImage()
    if image then
        image:remove()
        break
    end
end
```

4.10.4 Таблица DocumentAPI.Images

Таблица `DocumentAPI.Images` используется для доступа к коллекции изображений. Может быть получена вызовом методов `Table.getImages()`, `Range.getImages()`.

4.10.4.1 Метод Images:enumerate

Метод позволяет перечислить коллекцию изображений.

Пример для текстового документа:

```
for image in EditorAPI.getSelection():getImages():enumerate() do
    print(image:getFrame():getWrapType())
end
```

Пример для табличного документа:

```
local sheet = document:getBlocks():getTable(0)
local images = sheet:getImages()
for image in images:enumerate() do
    print(image:getFrame():getTopLeft().x)
end
```

4.10.5 Таблица DocumentAPI.AbsoluteFrame

Таблица `DocumentAPI.AbsoluteFrame` описывает прямоугольную область медиаобъекта, находящегося в абсолютной позиции документа (см. Рисунок 1223). Предназначена для получения и изменения свойств позиции медиаобъектов. Используется в табличном документе.

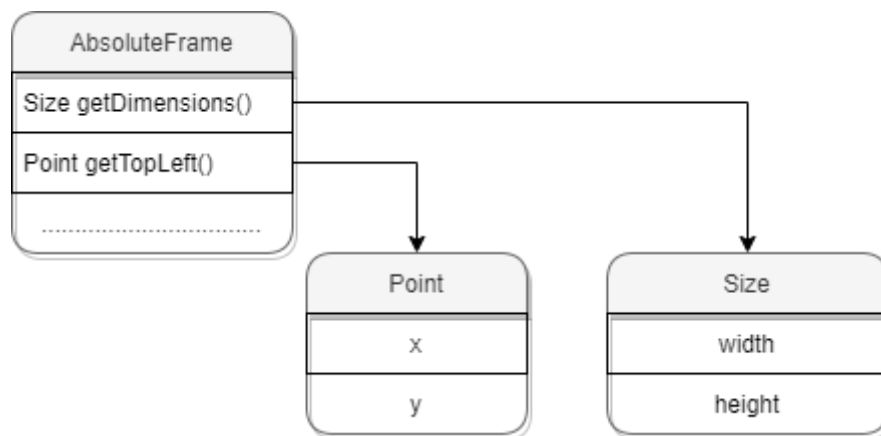


Рисунок 1223 – Объектная модель таблицы `DocumentAPI.AbsoluteFrame`

Пример для табличного документа:

```
local sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
    local absoluteFrame = mediaObject:getFrame()
    print(absoluteFrame:getDimensions())
    print(absoluteFrame:getTopLeft())
end
```

4.10.5.1 Метод AbsoluteFrame:moveTo

Метод перемещает объект в заданную позицию, тип аргумента – `DocumentAPI.PointU`.

Пример:

```
local sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
    local absoluteFrame = mediaObject:getFrame()
    newFramePosition = DocumentAPI.PointU(20, 20)
    absoluteFrame:moveTo(newFramePosition)
end
```

4.10.5.2 Метод AbsoluteFrame:getTopLeft

Метод возвращает позицию верхней левой точки медиаобъекта, тип – DocumentAPI.PointU.

Пример:

```
sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
    topLeftPosition = mediaObject:getFrame():getTopLeft()
    print("x=", topLeftPosition.x, "y=", topLeftPosition.y)
end
```

4.10.5.3 Метод AbsoluteFrame:scale

Метод scale изменяет размер объекта, масштабируя его по горизонтали и вертикали. Возможно изменение позиции объекта в соответствии со значением аргумента scaleFrom.

Вызов:

```
scale(widthScale, heightScale, scaleFrom)
```

Параметры:

- widthScale – коэффициент масштабирования по горизонтали, тип – числовой;
- heightScale – коэффициент масштабирования по вертикали, тип – числовой;
- scaleFrom – точка, сохраняющая позицию при масштабировании, тип – DocumentAPI.ScaleFrom.

Пример:

```
-- Уменьшение масштаба всех медиаобъектов на 50%
sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
    mediaObject:getFrame():scale(0.5, 0.5, DocumentAPI.ScaleFrom_TopLeft)
end
```

4.10.5.4 Метод `AbsoluteFrame:setDimensions`

Метод задает размеры (изменяет размер) медиаобъекта.

Вызов:

```
setDimensions(size)
```

Параметры:

size – размеры встроенного объекта, тип – `DocumentAPI.SizeU`.

Пример:

```
-- Изменение размера всех медиаобъектов  
sheet = document:getBlocks():getTable(0)  
for mediaObject in sheet:getMediaObjects():enumerate() do  
    mediaObject:getFrame():setDimensions(100, 100)  
end
```

4.10.5.5 Метод `AbsoluteFrame:getDimensions`

Возвращает размеры медиаобъекта, тип – `DocumentAPI.SizeU`.

4.10.6 Таблица `DocumentAPI.ScaleFrom`

В таблице 86 представлены позиции объекта, остающиеся неизменными при масштабировании объекта. Используется в `AbsoluteFrame.scale()`.

Таблица 86 – Неизменные позиции объекта при масштабировании

Наименование константы	Позиция
<code>DocumentAPI.ScaleFrom_BottomRight</code>	Правый нижний угол
<code>DocumentAPI.ScaleFrom_BottomLeft</code>	Левый нижний угол
<code>DocumentAPI.ScaleFrom_TopLeft</code>	Левый верхний угол
<code>DocumentAPI.ScaleFrom_TopRight</code>	Правый верхний угол

4.10.7 Таблица DocumentAPI.InlineFrame

Таблица `DocumentAPI.InlineFrame` описывает прямоугольную область графического объекта, находящегося в текстовой позиции документа (см. Рисунок 1224). Предназначена для получения и изменения свойств позиции графических объектов. Используется в текстовом документе.

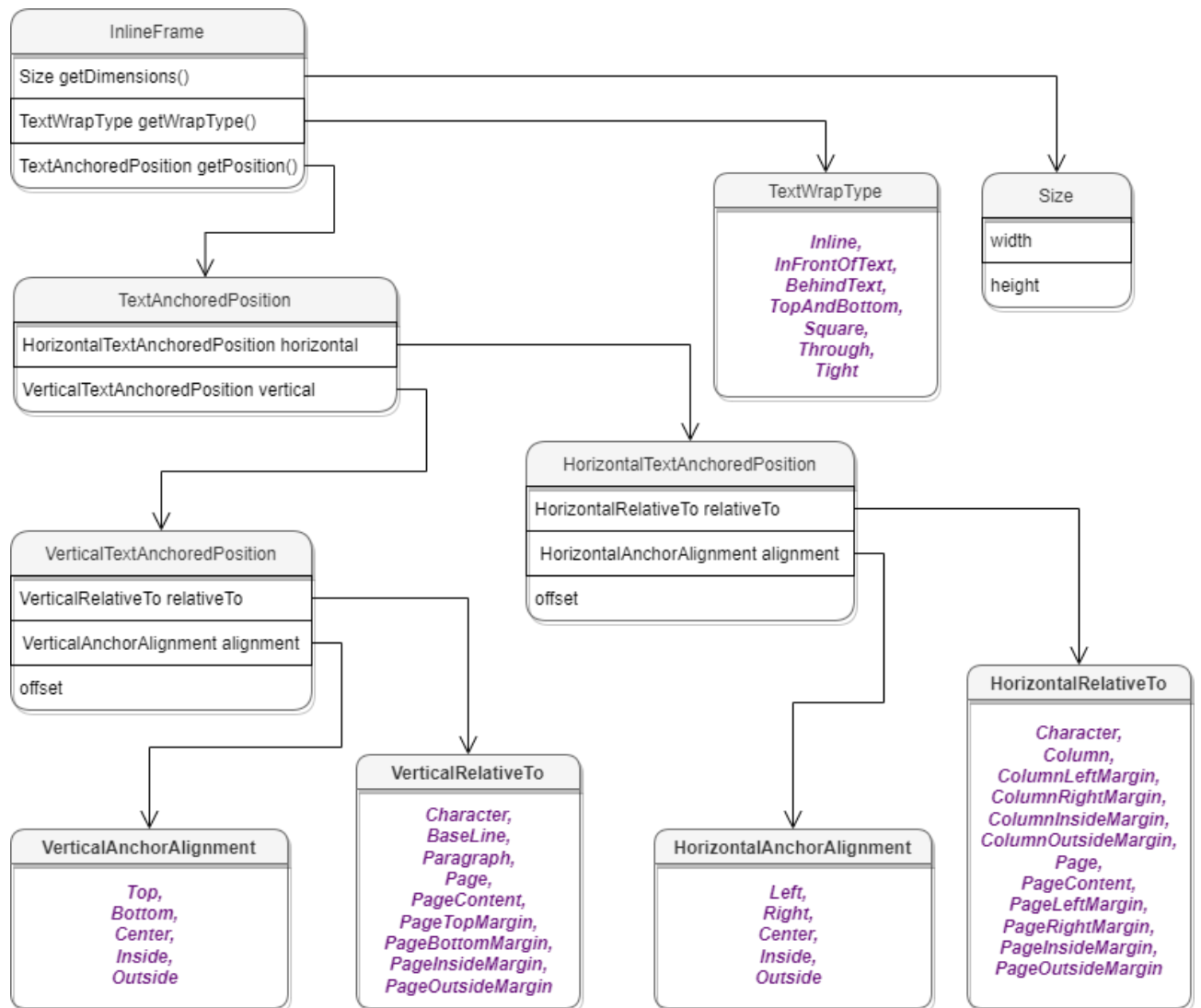


Рисунок 1224 – Объектная модель таблицы `DocumentAPI.InlineFrame`

Пример для текстового документа:

```

local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
    local inlineFrame = mediaObject:getFrame()
    print(inlineFrame:getDimensions())
    print(inlineFrame:getWrapType())
end
  
```



```
print(inlineFrame:getPosition())  
end
```

4.10.7.1 Метод `InlineFrame:setPosition`

Метод задает положение встроенного объекта, тип аргумента `DocumentAPI.TextAnchoredPosition`. Новая позиция может быть установлена только для встроенных объектов, тип переноса текста которых не является типом `DocumentAPI.TextWrapType_Inline`.

Пример:

```
local pos = DocumentAPI.TextAnchoredPosition()  
  
-- Установка смещения по горизонтали относительно края колонки  
pos.horizontal =  
DocumentAPI.HorizontalTextAnchoredPosition(DocumentAPI.HorizontalRelativeTo_Col  
umn)  
pos.horizontal.offset = x  
  
-- Установка смещения по вертикали относительно края страницы  
pos.vertical =  
DocumentAPI.VerticalTextAnchoredPosition(DocumentAPI.VerticalRelativeTo_Page)  
pos.vertical.offset = y  
  
-- Установка позиции рамки графического объекта  
inlineFrame:setPosition(pos)
```

4.10.7.2 Метод `InlineFrame:getPosition`

Метод возвращает позицию встроенного объекта на странице в виде таблицы `DocumentAPI.TextAnchoredPosition`.

Пример:

```
local mediaObjects = document:getRange():getImages()  
for mediaObject in mediaObjects:enumerate() do  
    local inlineFrame = mediaObject:getFrame()  
    local textAnchoredPosition = inlineFrame:getPosition()  
    if (textAnchoredPosition) then  
        print(textAnchoredPosition.horizontal, textAnchoredPosition.vertical)    end  
end
```

```
end  
end
```

4.10.7.3 Метод `InlineFrame:setDimensions`

Метод задает размер `DocumentAPI.SizeU` встроенного объекта.

Пример:

```
inlineFrame:setDimensions(DocumentAPI.SizeU(100, 100))
```

4.10.7.4 Метод `InlineFrame:getDimensions`

Метод возвращает задает размеры встроенного объекта, тип – `DocumentAPI.Size`.

Пример:

```
local mediaObjects = document:getRange():getImages()  
for mediaObject in mediaObjects:enumerate() do  
    local inlineFrame = mediaObject:getFrame()  
    local dimensions = inlineFrame:getDimensions()  
    if (dimensions) then  
        print(dimensions.width, dimensions.height)  
    end  
end
```

4.10.7.5 Метод `InlineFrame:setWrapType`

Метод устанавливает вариант обтекания текстом встроенного объекта (см. `DocumentAPI.TextWrapType`).

Пример:

```
local mediaObjects = document:getRange():getImages()  
for mediaObject in mediaObjects:enumerate() do  
    local inlineFrame = mediaObject:getFrame()  
    inlineFrame:setWrapType(DocumentAPI.TextWrapType_InFrontOfText)  
end
```

4.10.7.6 Метод `InlineFrame:getWrapType`

Метод возвращает вариант обтекания текстом встроенного объекта (см. `DocumentAPI.TextWrapType`).

Пример:

```
local mediaObjects = document:getRange():getImages()
for mediaObject in mediaObjects:enumerate() do
    print(mediaObject:getFrame():getWrapType())
end
```

4.10.8 Таблица DocumentAPI.TextWrapType

В таблице 87 представлены варианты обтекания текстом встроенного объекта. Используется в `InlineFrame.setWrapType()`.

Таблица 87 – Варианты обтекания текстом встроенного объекта

Наименование константы	Описание
<code>DocumentAPI.TextWrapType_Inline</code>	Встроенный объект располагается в тексте.
<code>DocumentAPI.TextWrapType_InFrontOfText</code>	Встроенный объект располагается перед текстом.
<code>DocumentAPI.TextWrapType_BehindText</code>	Встроенный объект располагается за текстом.
<code>DocumentAPI.TextWrapType_TopAndBottom</code>	Текст располагается сверху и снизу от встроенного объекта.
<code>DocumentAPI.TextWrapType_Square</code>	Текст располагается вокруг прямоугольной рамки встроенного объекта.
<code>DocumentAPI.TextWrapType_Through</code>	Текст обтекает встроенный объект по сторонам и внутри.

4.10.9 Таблица DocumentAPI.TextAnchoredPosition

Таблица `DocumentAPI.TextAnchoredPosition` представляет позицию объекта на странице текстового документа. Пример использования см. в разделе 4.10.7.1. Описание полей таблицы представлено в таблице 88.

Таблица 88 – Описание полей таблицы `DocumentAPI.TextAnchoredPosition`

Поле	Описание
<code>DocumentAPI.TextAnchoredPosition.horizontal</code>	Позиция по горизонтали <code>HorizontalTextAnchoredPosition</code>
<code>DocumentAPI.TextAnchoredPosition.vertical</code>	Позиция по вертикали <code>VerticalTextAnchoredPosition</code>

4.10.9.1 Метод `TextAnchoredPosition: __eq`

Метод используется для определения эквивалентности значений двух позиций объектов.

Пример:

```
local pos1 = DocumentAPI.TextAnchoredPosition()
pos1.horizontal =
DocumentAPI.HorizontalTextAnchoredPosition(DocumentAPI.HorizontalRelativeTo_Col
umn)
pos1.horizontal.offset = 1

local pos2 = DocumentAPI.TextAnchoredPosition()
pos2.horizontal =
DocumentAPI.HorizontalTextAnchoredPosition(DocumentAPI.HorizontalRelativeTo_Col
umn)
pos2.horizontal.offset = 1

print(pos1: __eq(pos2))
```

4.10.10 Таблица `DocumentAPI.HorizontalTextAnchoredPosition`

Таблица `DocumentAPI.HorizontalTextAnchoredPosition` предназначена для управления относительным положением объекта со смещением или выравниванием по горизонтали. Пример использования см. в разделе 4.10.7.1. Описание полей таблицы `DocumentAPI.HorizontalTextAnchoredPosition` представлено в таблице 89.

Таблица 89 – Описание полей таблицы `DocumentAPI.HorizontalTextAnchoredPosition`

Поле	Описание
<code>DocumentAPI.HorizontalTextAnchoredPosition.relativeTo</code>	Тип размещения объекта относительно закрепленной позиции по горизонтали <code>HorizontalRelativeTo</code> .
<code>DocumentAPI.HorizontalTextAnchoredPosition.offset</code>	Смещение объекта.
<code>DocumentAPI.HorizontalTextAnchoredPosition.alignment</code>	Тип выравнивания объекта относительно закрепленной позиции по горизонтали <code>HorizontalAnchorAlignment</code> .

4.10.10.1 Метод `HorizontalTextAnchoredPosition: __eq`

Метод используется для определения эквивалентности двух положений объекта по горизонтали.

Пример:

```
local pos1 = DocumentAPI.TextAnchoredPosition()
pos1.horizontal =
DocumentAPI.HorizontalTextAnchoredPosition(DocumentAPI.HorizontalRelativeTo_Col
umn)
pos1.horizontal.offset = 1

local pos2 = DocumentAPI.TextAnchoredPosition()
pos2.horizontal =
DocumentAPI.HorizontalTextAnchoredPosition(DocumentAPI.HorizontalRelativeTo_Col
umn)
pos2.horizontal.offset = 1

print(pos1.horizontal: __eq(pos2.horizontal))
```

4.10.11 Таблица `DocumentAPI.VerticalTextAnchoredPosition`

Таблица `DocumentAPI.VerticalTextAnchoredPosition` предназначена для управления относительным положением объекта со смещением или выравниванием по вертикали. Пример использования см. в разделе 4.10.7.1. Описание полей таблицы `DocumentAPI.VerticalTextAnchoredPosition` представлено в таблице 90.

Таблица 90 – Описание полей таблицы `DocumentAPI.VerticalTextAnchoredPosition`

Поле	Описание
<code>DocumentAPI.VerticalTextAnchoredPosition.relativeTo</code>	Тип размещения объекта относительно закрепленной позиции по вертикали <code>VerticalRelativeTo</code> .
<code>DocumentAPI.VerticalTextAnchoredPosition.offset</code>	Смещение объекта.
<code>DocumentAPI.VerticalTextAnchoredPosition.alignment</code>	Тип выравнивания объекта относительно закрепленной позиции по вертикали <code>VerticalAnchorAlignment</code> .

4.10.11.1 Метод `VerticalTextAnchoredPosition: __eq`

Метод используется для определения эквивалентности двух положений объекта по вертикали.

Пример:

```
local pos1 = DocumentAPI.TextAnchoredPosition()
pos1.vertical =
DocumentAPI.VerticalTextAnchoredPosition(DocumentAPI.VerticalRelativeTo_Page)
pos1.vertical.offset = 1

local pos2 = DocumentAPI.TextAnchoredPosition()
pos2.vertical =
DocumentAPI.VerticalTextAnchoredPosition(DocumentAPI.VerticalRelativeTo_Page)
pos2.vertical.offset = 1

print(pos1.vertical:__eq(pos2.vertical))
```

4.10.12 Таблица `DocumentAPI.VerticalRelativeTo`

В таблице 91 представлены типы размещения объекта относительно закрепленной позиции по вертикали.

Таблица 91 – Типы размещения объекта относительно закрепленной позиции по вертикали

Наименование константы	Описание
<code>DocumentAPI.VerticalRelativeTo_Character</code>	Символ
<code>DocumentAPI.VerticalRelativeTo_BaseLine</code>	Базовая линия
<code>DocumentAPI.VerticalRelativeTo_Paragraph</code>	Абзац
<code>DocumentAPI.VerticalRelativeTo_Page</code>	Страница
<code>DocumentAPI.VerticalRelativeTo_PageContent</code>	Содержимое страницы
<code>DocumentAPI.VerticalRelativeTo_PageTopMargin</code>	Верхнее поле страницы
<code>DocumentAPI.VerticalRelativeTo_PageBottomMargin</code>	Нижнее поле страницы
<code>DocumentAPI.VerticalRelativeTo_PageInsideMargin</code>	Внутреннее поле страницы
<code>DocumentAPI.VerticalRelativeTo_PageOutsideMargin</code>	Внешнее поле страницы

4.10.13 Таблица DocumentAPI.HorizontalRelativeTo

В таблице 92 представлены типы размещения объекта относительно закрепленной позиции по горизонтали.

Таблица 92 – Типы размещения объекта относительно закрепленной позиции по горизонтали

Наименование константы	Описание
DocumentAPI.HorizontalRelativeTo_Character	Символ
DocumentAPI.HorizontalRelativeTo_Column	Столбец
DocumentAPI.HorizontalRelativeTo_ColumnLeftMargin	Левое поле столбца
DocumentAPI.HorizontalRelativeTo_ColumnRightMargin	Правое поле столбца
DocumentAPI.HorizontalRelativeTo_ColumnInsideMargin	Внутреннее поле столбца
DocumentAPI.HorizontalRelativeTo_ColumnOutsideMargin	Внешнее поле столбца
DocumentAPI.HorizontalRelativeTo_Page	Страница
DocumentAPI.HorizontalRelativeTo_PageContent	Содержимое страницы
DocumentAPI.HorizontalRelativeTo_PageLeftMargin	Левое поле страницы
DocumentAPI.HorizontalRelativeTo_PageRightMargin	Правое поле страницы
DocumentAPI.HorizontalRelativeTo_PageInsideMargin	Внутреннее поле страницы
DocumentAPI.HorizontalRelativeTo_PageOutsideMargin	Внешнее поле страницы

4.10.14 Таблица DocumentAPI.VerticalAnchorAlignment

В таблице 93 представлены типы выравнивания объекта относительно закрепленной позиции по вертикали.

Таблица 93 – Типы выравнивания объекта относительно закрепленной позиции по вертикали

Наименование константы	Описание
DocumentAPI.VerticalAnchorAlignment_Top	По верхнему краю
DocumentAPI.VerticalAnchorAlignment_Bottom	По нижнему краю
DocumentAPI.VerticalAnchorAlignment_Center	По центру
DocumentAPI.VerticalAnchorAlignment_Inside, DocumentAPI.VerticalAnchorAlignment_Outside	По границам

4.10.15 Таблица DocumentAPI.HorizontalAnchorAlignment

В таблице 94 представлены типы выравнивания объекта относительно закрепленной позиции по горизонтали.

Таблица 94 – Типы выравнивания объекта относительно закрепленной позиции по горизонтали

Наименование константы	Описание
DocumentAPI.HorizontalAnchorAlignment_Left	По верхнему краю.
DocumentAPI.HorizontalAnchorAlignment_Right	По нижнему краю.
DocumentAPI.HorizontalAnchorAlignment_Center	По центру.
DocumentAPI.HorizontalAnchorAlignment_Inside , DocumentAPI.HorizontalAnchorAlignment_Outside	По границам.

4.10.16 Таблица DocumentAPI.Shape

Таблица Shape представляет собой фигуру, содержит методы для установки и получения свойств DocumentAPI.ShapeProperties.

4.10.16.1 Метод Shape:getShapeProperties

Метод возвращает свойства фигуры DocumentAPI.ShapeProperties.

Пример:

```
local shape = document:getBlocks():getShape(0)
local shape_properties = shape:getShapeProperties()
```

4.10.16.2 Метод Shape:setShapeProperties

Метод устанавливает свойства фигуры DocumentAPI.ShapeProperties.

Пример:

```
local shape = document:getBlocks():getShape(0)
local shape_properties = shape:getShapeProperties()
shape_properties.verticalAlignment = DocumentAPI.VerticalAlignment_Center
shape:setShapeProperties(shape_properties)
```


4.10.17 Таблица `DocumentAPI.ShapeProperties`

Таблица описывает свойства фигуры и содержит следующие поля:

- `verticalAlignment` – вертикальное выравнивание, тип `DocumentAPI.VerticalAlignment`;
- `borderProperties` – свойства границ фигуры, тип `DocumentAPI.LineProperties`;
- `fill` – свойства заполнения фигуры, тип `DocumentAPI.Fill`;
- `shapeTextLayout` – свойства текста внутри фигуры, тип `DocumentAPI.ShapeTextLayout`.

4.10.17.1 Поле `ShapeProperties:borderProperties`

Поле предназначено для установки свойств границ фигуры `DocumentAPI.LineProperties`.

4.10.17.2 Поле `ShapeProperties:verticalAlignment`

Поле предназначено для установки типа вертикального выравнивания `DocumentAPI.VerticalAlignment`.

4.10.17.3 Поле `ShapeProperties:fill`

Поле предназначено для установки свойств заполнения фигуры `DocumentAPI.Fill`.

4.10.17.4 Поле `ShapeProperties:shapeTextLayout`

Поле предназначено для установки свойств текста внутри фигуры `DocumentAPI.ShapeTextLayout`.

4.10.18 Таблица `DocumentAPI.ShapeTextLayout`

Таблица `DocumentAPI.ShapeTextLayout` описывает свойства текста, находящегося внутри фигуры. Описание полей представлено в таблице 95. Используется в таблице `DocumentAPI.ShapeProperties`.

Таблица 95 – Описание полей таблицы DocumentAPI.ShapeTextLayout

Поле	Описание
ShapeTextLayout_DoNotAutoFit	Размещение текста в фигуре по умолчанию.
ShapeTextLayout_FitShapeExtentToText	Расширение фигуры под текст.
ShapeTextLayout_FitTextToShape	Заполнение фигуры текстом.

4.10.19 Таблица DocumentAPI.Fill

Таблица описывает свойства заполнения фигуры: цвет заполнения, путь к изображению фона.

4.10.19.1 Метод Fill:getColor

Метод возвращает цвет заполнения DocumentAPI.Color.

4.10.19.2 Метод Fill:getUrl

Метод возвращает путь к изображению, которое используется в качестве заполнения, тип – строка.

4.10.19.3 Метод Fill:isNoFill

Метод возвращает true, если заполнения нет.

4.11 Поиск в документе

Для поиска в документе необходимо создать экземпляр таблицы DocumentAPI.Search посредством вызова DocumentAPI.createSearch, затем использовать метод Search:findText (см. Рисунок 1225).

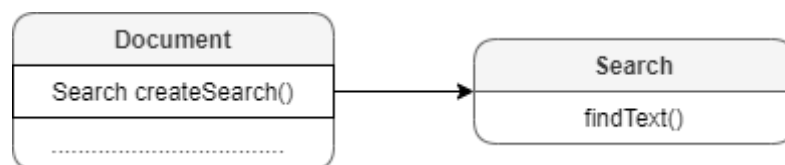


Рисунок 1225 – Объектная модель для поиска в документе

4.11.1 Метод `DocumentAPI:createSearch`

Метод инициализирует механизм поиска для текущего документа. Возвращает ссылку на таблицу `DocumentAPI.Search`, с помощью методов которой выполняются поисковые запросы.

Пример:

```
search = DocumentAPI.createSearch(document)
ranges = search.findText("English")
```

4.11.2 Таблица `DocumentAPI.Search`

Таблица `DocumentAPI.Search` предоставляет доступ к механизму поиска и замены фрагментов документа, открытого в редакторе текста или таблиц.

4.11.2.1 Метод `Search:findText`

Метод выполняет поиск строки без учета регистра во всем документе или выбранном диапазоне документа. Результат возвращается в виде диапазона `DocumentAPI.Range`, содержащего искомым фрагмент.

Если строка не обнаружена, возвращается пустая таблица.

Примеры:

```
search = DocumentAPI.createSearch(document)
-- Поиск по всему документу
ranges = search.findText("English")
for occurrence in ranges do
    print(occurrence.extractText())
end
-- Поиск только в диапазоне первого блока
local firstBlockRange = document:getBlocks():getBlock(0):getRange()
ranges = search.findText("English", firstBlockRange)
for occurrence in ranges do
    print(occurrence.extractText())
end
```

4.12 Таблица `DocumentAPI.document`

Таблица `DocumentAPI.Document` осуществляет доступ к содержимому открытого текстового или табличного документа.

Например, данный пример предоставляет доступ к первому абзацу текстового документа:

```
local para = document:getBlocks():getParagraph(0)
```

4.12.1 Метод `document:getBlocks`

Метод предоставляет доступ к метатаблице `DocumentAPI.Blocks` и далее к отдельным фрагментам (абзацам, таблицам и т. д.), из которых состоит документ.

Пример:

```
local blocks = document:getBlocks()
```

4.12.2 Метод `document:getBookmarks`

Метод предоставляет доступ к таблице закладок `DocumentAPI.Bookmarks`. Используется только в текстовых документах.

Пример:

```
local bookmarks = document:getBookmarks()
```

4.12.3 Метод `document:getScripts`

Метод предоставляет доступ к таблице макрокоманд `DocumentAPI.Scripts`, содержащихся в документе.

Пример:

```
local scripts = document:getScripts()
```

4.12.4 Метод `document:getRange`

Метод предоставляет доступ ко всему диапазону `DocumentAPI.Range` документа.

Пример:

```
local range = document:getRange()  
print(range:extractText())
```

4.12.5 Метод `document:isChangesTrackingEnabled`

Метод возвращает текущее состояние отслеживания изменений в документе (`true` – включены). Используется только в текстовых документах.

Пример:

```
local trackingChanges = "Disabled"
if document:isChangesTrackingEnabled() then
    trackingChanges = "Enabled"
end
print(trackingChanges)
```

4.12.6 Метод `document:setChangesTrackingEnabled`

Метод управляет состоянием отслеживания изменений в документе (включены или выключены). Используется только в текстовых документах.

Пример:

```
if trackingChanges == "Disabled" then
    document:setChangesTrackingEnabled(true)
    if document:isChangesTrackingEnabled() then
        trackingChanges = "Enabled"
    end
end
```

4.12.7 Метод `document:getComments`

Метод обеспечивает доступ к комментариям документа, возвращает таблицу `DocumentAPI.Comments`. Используется только в текстовых документах.

Пример:

```
local comments = document:getComments()
for comment in comments:enumerate() do
    print(comment:getRange())
    print(comment:getText())
    print(comment:getInfo().author)
    print(comment:getInfo().timeStamp)
    print(comment:isResolved())
    print(comment:getReplies())
end
```

4.12.8 Метод `document:setPageProperties`

Метод устанавливает свойство `DocumentAPI.PageProperties` в документе.

Пример:

```
local properties = DocumentAPI.PageProperties()  
properties.width = 100  
properties.height = 200  
document:setPageProperties(properties)
```

4.12.9 Метод `document:setFormulaType`

Метод устанавливает поддерживаемую адресацию ячеек `DocumentAPI.FormulaType` документа.

Пример:

```
document:setFormulaType(DocumentAPI.FormulaType_A1)
```

4.12.10 Метод `document:getFormulaType`

Метод возвращает поддерживаемую адресацию ячеек `DocumentAPI.FormulaType` документа.

4.12.11 Метод `document:setPageOrientation`

Метод устанавливает альбомную, либо книжную ориентацию страниц в документе (см. раздел 4.4.6).

Пример:

```
document:setPageOrientation(DocumentAPI.PageOrientation_Landscape)
```

4.12.12 Метод `document:enumerateSections`

Возвращает таблицу объектов типа `DocumentAPI.Section`.

Пример:

```
local sections = document:enumerateSections()  
for section in sections do  
    print(section:getPageProperties().width)  
end
```

4.12.13 Метод `document:getSections`

Возвращает таблицу объектов типа `DocumentAPI.Sections`.

Пример:

```
local sections = document:getSections()  
for section in sections:enumerate() do  
    local properties = section:getPageProperties()  
    print(properties.width)  
    print(properties.height)  
end
```

4.12.14 Метод `document:setMirroredMarginsEnabled`

Метод позволяет включать и выключать зеркальные поля в документе.

Пример:

```
document:setMirroredMarginsEnabled(true)  
print(document:areMirroredMarginsEnabled())
```

4.12.15 Метод `document:areMirroredMarginsEnabled`

Возвращает состояние режима зеркальных полей в документе (разрешены или запрещены).

Пример:

```
document:setMirroredMarginsEnabled(true)  
print(document:areMirroredMarginsEnabled())
```

4.12.16 Метод `document:getPivotTablesManager`

Возвращает объект `DocumentAPI.PivotTablesManager`, который используется для создания сводных таблиц. Метод может быть использован только в табличном редакторе.

Пример:

```
local pivotTablesManager = document:getPivotTablesManager()  
print(pivotTablesManager)
```

4.12.17 Метод `document:getNamedExpressions`

Используется для получения списка именованных диапазонов `DocumentAPI.NamedExpressions`.

Пример:

```
namedExpressions = document.getNamedExpressions()  
print(namedExpressions)
```

4.13 Таблица DocumentAPI.TextProperties

Таблица `DocumentAPI.TextProperties` предназначена для форматирования текста. Описание полей таблицы `DocumentAPI.TextProperties` представлено в таблице 96. Свойства `DocumentAPI.TextProperties` применяются к диапазону текста `DocumentAPI.Range` (методы `Range.getTextProperties()`, `Range.setTextProperties()`).

Таблица 96 – Описание полей таблицы `DocumentAPI.TextProperties`

Поле	Тип	Описание
<code>TextProperties.fontName</code>	Строковое	Наименование шрифта, использованного для оформления фрагмента документа.
<code>TextProperties.fontSize</code>	Числовое	Размер шрифта, использованного для оформления фрагмента документа.
<code>TextProperties.bold</code>	Логическое	Значение <code>true</code> устанавливает жирное начертание для указанного фрагмента текста.
<code>TextProperties.italic</code>	Логическое	Значение <code>true</code> устанавливает начертание курсивом для указанного фрагмента текста.
<code>TextProperties.underline</code>	Логическое	Значение <code>true</code> устанавливает подчеркивание для указанного фрагмента текста.
<code>TextProperties.strikethrough</code>	Логическое	Значение <code>true</code> устанавливает начертание «зачеркнутый» для указанного фрагмента текста.
<code>TextProperties.allCapitals</code>	Логическое	Значение <code>true</code> устанавливает все буквы указанного фрагмента текста как прописные. Значение <code>false</code> устанавливает все буквы указанного фрагмента текста как строчные.
<code>TextProperties.scriptPosition</code>	<code>ScriptPosition</code>	Устанавливает отображение символа в виде надстрочного, подстрочного знака или в

Поле	Тип	Описание
		нормальном режиме.
TextProperties.textColor	Color	Цвет указанного фрагмента документа.
TextProperties.backgroundColor	ColorRGBA	Цвет фона указанного фрагмента документа.
TextProperties.characterSpacing	Числовое	Размер межсимвольного интервала.

Пример:

```

local props = DocumentAPI.TextProperties()
props.fontName = "XO Oriel"
props.fontSize = 20

-- доступ к тексту третьего абзаца
local range = document:getBlocks():getParagraph(2):getRange()

-- установить свойства фрагмента текста
range:setTextProperties(props)

```

4.14 Таблица DocumentAPI.ScriptPosition

Варианты представления текста в виде надстрочных или подстрочных знаков при работе в текстовом редакторе представлены в таблице 97. Используется в качестве поля scriptPosition таблицы DocumentAPI.TextProperties.

Таблица 97 – Типы надстрочного и подстрочного форматирования

Наименование константы	Описание
DocumentAPI.ScriptPosition_SuperScript	Надстрочный знак (верхний индекс)
DocumentAPI.ScriptPosition_SubScript	Подстрочный знак (нижний индекс)
DocumentAPI.ScriptPosition_NormalScript	Без указания индекса

Пример:

```

local props = DocumentAPI.TextProperties()
props.scriptPosition = DocumentAPI.ScriptPosition_SuperScript
range:setTextProperties(props)

```

4.15 Таблица DocumentAPI.Range

Таблица DocumentAPI.Range предоставляет доступ к диапазону документа. На рисунке 1226 изображена объектная модель таблиц, относящихся к работе с диапазонами.

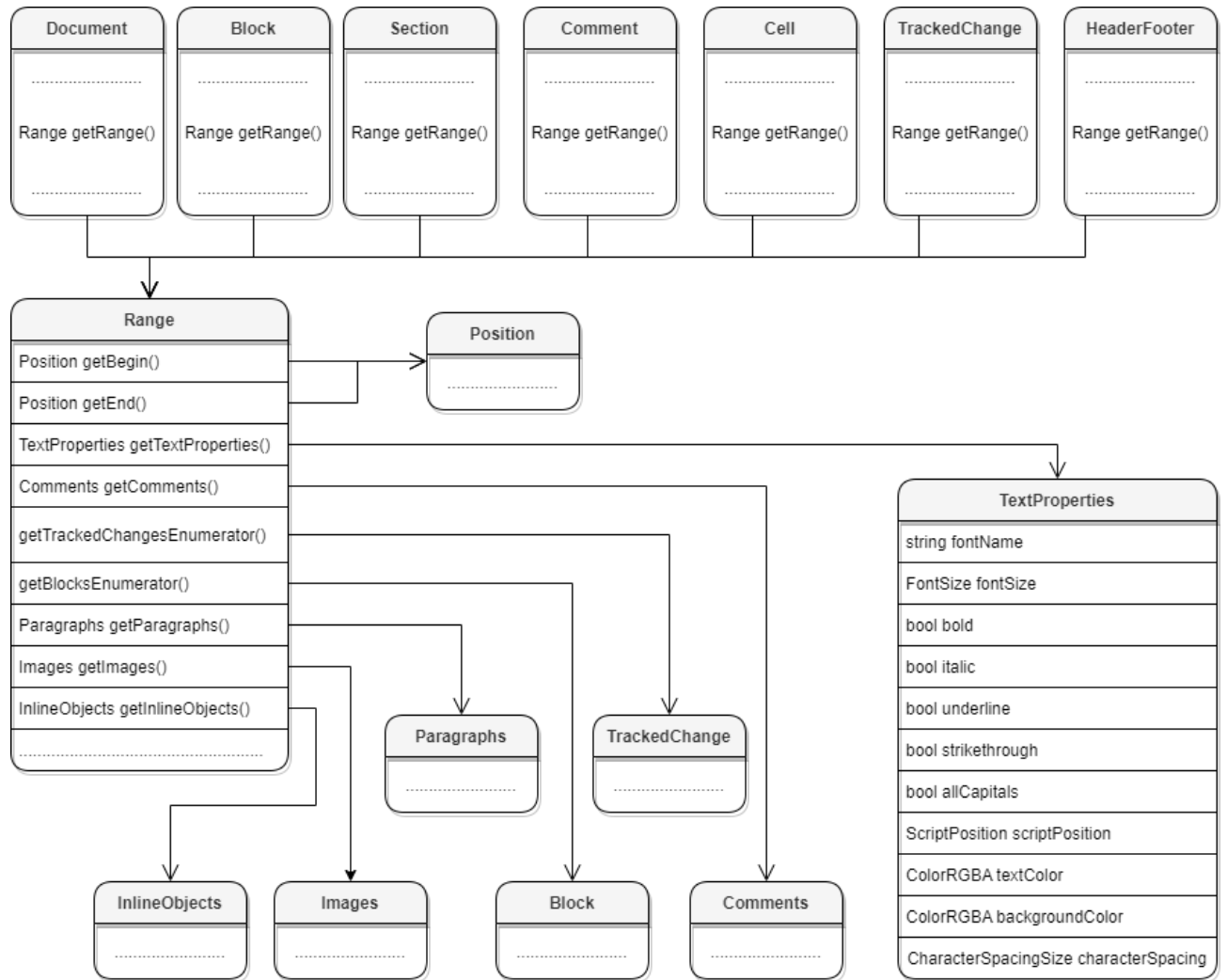


Рисунок 1226 – Объектная модель для работы с таблицей DocumentAPI . Range

Варианты получения диапазона для текстового документа:

```
-- диапазон всего документа
documentRange = document:getRange()

-- диапазон блока
block = document:getBlocks():getBlock(0)
blockRange = block:getRange()

-- диапазон секций
sections = document:getSections()
for section in sections:enumerate() do
    sectionRange = section:getRange()
```

```
end
-- диапазон комментариев
commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    commentRange = comment:getRange()
end
-- диапазон ячейки
table = document:getBlocks():getTable(0)
cell = table:getCell("B2")
cellRange = cell:getRange()
-- диапазон верхних колонтитулов
section = document:getBlocks():getBlock(0):getSection()
headers = section:getHeaders()
for header in headers:enumerate() do
    headerRange = header:getRange()
end
-- диапазон отслеживаемых изменений
local trackedChangesList = document:getRange():enumerateTrackedChanges()
for trackedChange in trackedChangesList do
    trackedChangeRange = trackedChange:getRange()
end
```

4.15.1 Метод Range:getBegin

Метод возвращает позицию в начале диапазона.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
local pos = range:getBegin() -- в начало документа
pos:insertText("Привет")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки B2
local pos = range:getBegin() -- в начало ячейки
pos:insertText("Привет")
```

4.15.2 Метод Range:getEnd

Метод возвращает позицию в конце диапазона, не включая последний символ paragraph mark.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
local pos = range:getEnd() -- в конец документа
pos:insertText("Привет")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки B2
local pos = range:getEnd() -- в конец ячейки
pos:insertText("Привет")
```

4.15.3 Метод Range:extractText

Метод возвращает содержимое фрагмента в виде строки текста. Находящиеся внутри области изображения, таблицы и другие объекты игнорируются.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
local text = range:extractText()
print (text)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки B2
print (range:extractText())
```

4.15.4 Метод Range:removeContent

Метод полностью удаляет содержимое диапазона.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:removeContent()
print (range:extractText())
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:removeContent()
print (range:extractText())
```

4.15.5 Метод Range:lockContent

Метод запрещает изменения содержимого диапазона.

Важно – Метод может быть использован только в текстовых документах.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:lockContent()
```

Пример для таблицы внутри текстового документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:lockContent()
```

4.15.6 Метод Range:unlockContent

Метод разрешает изменения содержимого диапазона.

Важно – Метод может быть использован только в текстовых документах.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:unlockContent()
```

Пример для таблицы внутри текстового документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:unlockContent()
```

4.15.7 Метод Range:isContentLocked

Метод возвращает значение true, если изменения содержимого диапазона запрещены.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
if range:isContentLocked() then
  print("Документ содержит заблокированное содержимое")
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
if range:isContentLocked() then
  print("Ячейка содержит заблокированное содержимое")
end
```

4.15.8 Метод Range:replaceText

Метод заменяет содержимое фрагмента на указанный текст.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:replaceText("Новый текст")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки таблицы
range:replaceText("Новый текст")
```

4.15.9 Метод Range:setHyperlink

Метод setHyperlink вставляет ссылку в содержимое диапазона и заменяет его текст текстом ссылки.

Вызов:

```
setHyperlink( url, label )
```

Параметры:

- url – адрес ссылки;
- label – текст ссылки.

Пример для текстового документа:

```
local range = document:getRange()  
range:setHyperlink("https://testhyperlink.com", "Hyperlink")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)  
local cell = tbl:getCell("B2")  
local range = cell:getRange()  
range:setHyperlink("https://testhyperlink.com", "Hyperlink")  
print(cell:getFormattedValue())
```

4.15.10 Метод Range:getTextProperties

Метод возвращает таблицу с текущими настройками форматирования для фрагмента текстового документа. Описание настроек форматирования осуществляется с помощью таблицы `DocumentAPI.TextProperties`.

Пример для текстового документа:

```
local range = document:getRange()  
local props = range:getTextProperties()  
print(props.italic)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)  
local cell = tbl:getCell("B2")  
local range = cell:getRange()  
local props = range:getTextProperties()  
print(props.italic)
```

4.15.11 Метод Range:setTextProperties

Метод применяет настройки форматирования `DocumentAPI.TextProperties` для диапазона.

Пример для текстового документа:

```
local range = document:getRange()  
local props = range:getTextProperties()  
props.italic = true  
range:setTextProperties(props) -- текстовый фрагмент оформлен курсивом
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local props = range:getTextProperties()
props.italic = true
range:setTextProperties(props)
```

4.15.12 Метод Range:enumerateBlocks

Предоставляет возможность итерации по блокам.

Пример для текстового документа:

```
local range = document:getRange()
for block in range:enumerateBlocks() do
    print(block:getRange():extractText())
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
for block in range:enumerateBlocks() do
    print(block:getRange():extractText())
end
```

4.15.13 Метод Range:enumerateTrackedChanges

Предоставляет возможность итерации по отслеживаемым изменениям DocumentAPI.TrackedChange. Метод может быть использован только в текстовых документах.

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()
for change in changesList do
    print(change:getRange():extractText())
end
```

4.15.14 Метод Range:getComments

Обеспечивает доступ к комментариям в диапазоне.

Комментарии, примененные к одному и тому же диапазону, упорядочиваются по датам. Если дат нет, то порядок комментариев не определен.

Пример:

```
local comments = document:getRange():getComments()
for comment in comments:enumerate() do
    print(comment:getRange())
    print(comment:getText())
    print(comment:getInfo().author)
    print(comment:getInfo().timeStamp)
    print(comment:isResolved())
    print(comment:getReplies())
end
```

4.15.15 Метод Range:getParagraphs

Обеспечивает доступ к абзацам DocumentAPI.Paragraphs в диапазоне.

Пример для текстового документа:

```
local paragraphs = document:getRange():getParagraphs()
for para in paragraphs:enumerate() do
    print(para:getRange():extractText())
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    print(para:getRange():extractText())
end
```

4.15.16 Метод Range:getImages

Обеспечивает доступ к изображениям (DocumentAPI.Image) в диапазоне.

Примеры:

```
local images = document:getRange():getImages()
for image in images:enumerate() do
```

```
print(image:getFrame():getWrapType())  
end  
for image in EditorAPI.getSelection():getImages():enumerate() do  
    print(image:getFrame():getWrapType())  
end
```

4.15.17 Метод Range:getInlineObjects

Обеспечивает доступ к перечислению DocumentAPI.MediaObjects графических объектов диапазона.

Пример:

```
local mediaObjects = document:getRange():getInlineObjects()  
for mediaObject in mediaObjects:enumerate() do  
    print(mediaObject)  
end
```

4.16 Таблица DocumentAPI.Position

Таблица DocumentAPI.Position представляет местоположение в текстовом документе. Используется для обозначения начала и конца диапазона DocumentAPI.Range.

4.16.1 Метод Position:insertText

Метод предназначен для вставки текстовой строки в заданное местоположение в документе.

Пример:

```
local rng = document:getRange()  
local begin_pos = rng:getBegin()  
begin_pos:insertText("Текст в начале строки")
```

4.16.2 Метод Position:insertTable

Метод предназначен для вставки таблицы с заданным числом строк и столбцов в заданное местоположение в документе. Возвращает объект таблицы.

Следует учитывать, что при вставке таблицы к ее имени автоматически добавляется порядковый номер, начинающийся с единицы. Таким образом, вызов

```
t = position:insertTable(3, 3, "Table")
```

приведет к созданию в текстовом документе таблицы с именем «Table1».

Пример вставки таблицы в начало текстового документа:

```
local rng = document:getRange()  
local begin_pos = rng:getBegin()  
t = begin_pos:insertTable(3, 3, "Table")
```

Пример вставки таблицы в конец текстового документа:

```
local rng = document:getRange()  
local begin_pos = rng:getEnd()  
t = begin_pos:insertTable(3, 3, "Table")
```

В табличном документе данный метод используется для вставки нового рабочего листа.

Пример вставки нового листа в табличный документ:

```
local rng = document:getRange()  
local end_pos = rng:getEnd()  
t = end_pos:insertTable(3, 3, "Table")
```

4.16.3 Метод Position:insertPageBreak

Метод предназначен для вставки разрыва страницы в указанную позицию текстового документа.

Важно – Метод может быть использован только в текстовом редакторе.

Пример:

```
local rng = document:getRange()  
local end_pos = rng:getEnd()  
end_pos:insertPageBreak()
```

4.16.4 Метод Position:insertLineBreak

Метод предназначен для вставки перевода строки в указанную позицию текстового документа.

Важно – Метод может быть использован только в текстовом редакторе.

Пример:

```
local rng = document:getRange()  
local end_pos = rng:getEnd()  
end_pos:insertLineBreak()
```

4.16.5 Метод Position:insertBookmark

Вставляет закладку с наименованием в заданную позицию текстового документа.

Важно – Метод может быть использован только в текстовом редакторе.

Пример:

```
document:getRange():getBegin():insertBookmark("Bookmark example")
```

4.16.6 Метод Position:insertSectionBreak

Вставляет разрыв раздела в текущую позицию текстового документа.

Важно – Метод может быть использован только в текстовом редакторе.

Пример:

```
document:getRange():getBegin():insertSectionBreak()
```

4.16.7 Метод Position:insertHyperlink

Метод `insertHyperlink` вставляет ссылку в текущую позицию. В качестве параметров передаются адрес ссылки и текст ссылки.

Вызов:

```
insertHyperlink( url, size )
```

Параметры:

- url – адрес ссылки;
- label – текст ссылки.

Пример:

```
document:getRange():getBegin():insertHyperlink("https://testhyperlink.com",  
"Hyperlink")
```

4.16.8 Метод Position:insertImage

Вставляет изображение из файла в заданную позицию текстового документа.

Важно – В данный момент метод может быть использован только в текстовом редакторе.

Вызов:

```
insertImage( url, size )
```

Параметры:

- url – полный путь к файлу;
- size – геометрические размеры изображения для вставки.

Пример:

```
document.getRange().getBegin().insertImage("C://Tmp//123.jpg",  
DocumentAPI.SizeU(100, 100))
```

4.16.9 Метод Position:removeBackward

Метод удаляет count объектов (символов, картинок и т.д.) до текущей позиции.

Пример:

```
document.getRange().getEnd().removeBackward(3)
```

4.16.10 Метод Position:removeForward

Метод удаляет count объектов (символов, картинок и т.д.) после текущей позиции.

Пример:

```
document.getRange().getBegin().removeForward(3)
```

4.16.11 Метод Position:__eq

Метод используется для определения эквивалентности значений двух местоположений в документе.

Пример:

```
print(document.getRange().getBegin().__eq(document.getRange().getEnd()))
```

4.17 Таблица DocumentAPI.Color

Таблица DocumentAPI.Color представляет либо цветовой объект RGBA, либо заданные цвета идентификатора темы. В качестве параметров конструктора используются таблицы DocumentAPI.ColorRGBA, DocumentAPI.ThemeColorID.

Пример:

```
local rgbaColor = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0, 255))
local themeColor = DocumentAPI.Color(DocumentAPI.ThemeColorID_Text1)
```

4.17.1 Метод Color:getRGBAColor

Метод возвращает цвет DocumentAPI.ColorRGBA.

Пример:

```
local color = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0, 255))
local rgbaColor = color:getRGBAColor()
if rgbaColor then
    print(rgbaColor.r, rgbaColor.g, rgbaColor.b, rgbaColor.a)
end
```

4.17.2 Метод Color:getThemeColorID

Метод возвращает цвет идентификатора темы DocumentAPI.ThemeColorID.

Пример:

```
local color = DocumentAPI.Color(DocumentAPI.ThemeColorID_Text1)
local themeColorID = color:getThemeColorID()
if themeColorID then
    print(themeColorID)
end
```

4.17.3 Метод Color:__eq

Метод используется для определения эквивалентности двух значений цвета.

Пример:

```
local color = DocumentAPI.Color(DocumentAPI.ThemeColorID_Text1)
if color:__eq(color) then
    print("Equals")
end
```

4.18 Таблица DocumentAPI.ColorRGBA

Таблица DocumentAPI.ColorRGBA предназначена для настройки цвета текста и линий. Используется четырехканальный формат, содержащий данные для красного (r), голубого (b), зеленого (g) цветов и альфа-канала (a).

Описание таблицы DocumentAPI.ColorRGBA представлено в таблице 98.

Таблица 98 – Описание таблицы DocumentAPI.ColorRGBA

Цвет	Описание
r	Значение от 0 до 255 для установки интенсивности красного цвета.
g	Значение от 0 до 255 для установки интенсивности зеленого цвета.
b	Значение от 0 до 255 для установки интенсивности голубого цвета.
a	Значение от 0 до 255 для регулировки прозрачности. Значение 255 соответствует полностью непрозрачному цвету.

Пример:

```
local line_prop = DocumentAPI.LineProperties()  
line_prop.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(55, 146, 179, 200))
```

4.18.1 Метод ColorRGBA: __eq

Метод используется для определения эквивалентности двух значений цвета DocumentAPI.ColorRGBA.

Пример:

```
colorRGBA = DocumentAPI.Color(DocumentAPI.ColorRGBA(55, 146, 179, 200))  
if colorRGBA:__eq(colorRGBA) then  
    print("Equals")  
end
```

4.19 Таблица DocumentAPI.ThemeColorID

В таблице 99 представлены типы идентификаторов цветов тем. Используется в DocumentAPI.Color.

Таблица 99 – Типы идентификаторов цветов тем

Наименование константы	Описание
DocumentAPI.ThemeColorID_Background1	Фон1
DocumentAPI.ThemeColorID_Text1	Текст1

Наименование константы	Описание
DocumentAPI.ThemeColorID_Background2	Фон2
DocumentAPI.ThemeColorID_Text2	Текст2
DocumentAPI.ThemeColorID_Dark1	Темная1
DocumentAPI.ThemeColorID_Dark2	Темная2
DocumentAPI.ThemeColorID_Light1	Светлая1
DocumentAPI.ThemeColorID_Light2	Светлая2
DocumentAPI.ThemeColorID_Accent1	Акцент1
DocumentAPI.ThemeColorID_Accent2	Акцент2
DocumentAPI.ThemeColorID_Accent3	Акцент3
DocumentAPI.ThemeColorID_Accent4	Акцент4
DocumentAPI.ThemeColorID_Accent5	Акцент5
DocumentAPI.ThemeColorID_Accent6	Акцент6
DocumentAPI.ThemeColorID_Hyperlink	Гиперссылка
DocumentAPI.ThemeColorID_FollowedHyperlink	Следующая гиперссылка

4.20 Таблица DocumentAPI.PrintSettings

Таблица DocumentAPI.PrintSettings представляет установки, используемые при печати документов. Описание полей таблицы DocumentAPI.PrintSettings представлено в таблице 100. Используется в EditorAPI.printDocument.

Таблица 100 – Описание полей таблицы DocumentAPI.PrintSettings

Поле	Тип	Описание
DocumentAPI.PrintSettings.printerName	string	Имя используемого принтера. Если не указано, то используется принтер по умолчанию. Если принтер с указанным именем недоступен, то возникает ошибка.
DocumentAPI.PrintSettings.landscapeOrientation	bool	Если значение равно true, то размер страницы поворачивается на 90 градусов. В настоящее время используется только для рабочих таблиц.
DocumentAPI.PrintSettings.leftMargin	number	Ширина левого поля. Размер указывается в типографских точках. В настоящее время

Поле	Тип	Описание
		используется только для рабочих таблиц.
DocumentAPI.PrintSettings.topMargin	number	Ширина верхнего поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
DocumentAPI.PrintSettings.rightMargin	number	Ширина правого поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
DocumentAPI.PrintSettings.bottomMargin	number	Ширина нижнего поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
DocumentAPI.PrintSettings.parity	PageParity	Выбор страниц для печати.
DocumentAPI.PrintSettings.firstPage	number	Номер первой страницы для печати.
DocumentAPI.PrintSettings.lastPage	number	Номер последней страницы для печати.
DocumentAPI.PrintSettings.printSelection	bool	Область печати. Значение по умолчанию false.
DocumentAPI.PrintSettings.worksheetPrinterFitType	WorksheetPrinterFitType	Вариант масштабирования при печати табличных документов.
DocumentAPI.PrintSettings.copies	number	Количество копий при печати.
DocumentAPI.PrintSettings.collateCopies	bool	Если параметр имеет значение false, то печать каждой отдельной страницы будет повторена заданное количество копий раз до начала печати следующей страницы. Если параметр имеет значение true, то все страницы печатаются до запуска печати следующей копии этих страниц. Значение по умолчанию false.

4.21 Таблица DocumentAPI.WorksheetPrinterFitType

В таблице 101 представлены варианты масштабирования при печати табличных документов. Используется в качестве поля `worksheetPrinterFitType` таблицы `DocumentAPI.PrintSettings`.

Таблица 101 – Варианты масштабирования при печати табличных документов

Наименование константы	Описание
<code>DocumentAPI.WorksheetPrinterFitType_ActualSize</code>	Фактический размер
<code>DocumentAPI.WorksheetPrinterFitType_ByPageScale</code>	По масштабу страницы
<code>DocumentAPI.WorksheetPrinterFitType_ByPageBreaksOnly</code>	По разрыву страниц
<code>DocumentAPI.WorksheetPrinterFitType_FitToPage</code>	Вписать в страницу
<code>DocumentAPI.WorksheetPrinterFitType_FitToWidth</code>	Вписать по ширине
<code>DocumentAPI.WorksheetPrinterFitType_FitToHeight</code>	Вписать по высоте

4.22 Таблица DocumentAPI.PrintDocumentResult

В таблице 102 представлены коды, возвращаемые после печати (см. раздел 5.4).

Таблица 102 – Коды, возвращаемые после печати

Наименование константы	Описание
<code>DocumentAPI.PrintDocumentResult_Success</code>	Печать прошла успешно
<code>DocumentAPI.PrintDocumentResult_OneCopyPrinted</code>	Напечатана только одна копия из заданных
<code>DocumentAPI.PrintDocumentResult_CancelPrinting</code>	Печать была отменена
<code>DocumentAPI.PrintDocumentResult_NoPrinter</code>	Принтер не найден
<code>DocumentAPI.PrintDocumentResult_BlankDocument</code>	На печать отправлен пустой документ

4.23 Таблица DocumentAPI.SizeU

Таблица `DocumentAPI.SizeU` представляет размер объекта в двухмерном пространстве. Описание полей таблицы `DocumentAPI.SizeU` представлено в таблице 103.

Таблица 103 – Описание полей таблицы `DocumentAPI.SizeU`

Поле	Тип	Описание
<code>DocumentAPI.SizeU.width</code>	number	Ширина
<code>DocumentAPI.SizeU.height</code>	number	Высота

Пример:

```
local size = DocumentAPI.SizeU(2, 3)
print("width=", size.width, ", height=", size.height)  --(width = 2,0, height = 3,0)
```

4.23.1 Метод SizeU:toString

Возвращает информацию о размерах в виде строкового значения формата (width: <value>, height: <value>).

Пример:

```
local size = DocumentAPI.SizeU(2, 3)
print(size:toString())  --(width: 2.0, height: 3.0)
```

4.24 Таблица DocumentAPI.PointU

Таблица DocumentAPI.PointU представляет позицию объекта в двухмерном пространстве. Описание полей таблицы DocumentAPI.PointU представлено в таблице 104.

Таблица 104 – Описание полей таблицы DocumentAPI.PointU

Поле	Описание
DocumentAPI.PointU.x	Позиция x
DocumentAPI.PointU.y	Позиция y

Пример:

```
local point = DocumentAPI.PointU(2, 3)
print("x=", point.x, ", y=", point.y)  --(x = 2.0, y = 3.0)
```

4.24.1 Метод PointU:toString

Возвращает информацию о позиции в виде строкового значения формата (width: <value>, height: <value>).

Пример:

```
local point = DocumentAPI.PointU(2, 3)
print(point:toString())  --(x: 2.0, y: 3.0)
```

4.25 Таблица DocumentAPI.RectU

Таблица DocumentAPI.RectU представляет описание прямоугольной области. Описание полей таблицы DocumentAPI.RectU представлено в таблице 105.

Таблица 105 – Описание полей таблицы DocumentAPI.RectU

Поле	Описание
DocumentAPI.RectU.topLeft	Координаты левого верхнего угла области, тип CellPosition.
DocumentAPI.RectU.rightBottom	Координаты правого нижнего угла области, тип CellPosition.

Пример:

```
local rect = DocumentAPI.RectU(2, 3, 4, 5)
print("tlx=", rect.topLeft.x, ", tly=", rect.topLeft.y, ", rbx=",
rect.bottomRight.x, ", rby=", rect.bottomRight.y) --(tlx = 2.0, tly = 3.0, brx
= 4.0, bry = 5.0)
```

4.25.1 Метод RectU:toString

Возвращает информацию о прямоугольной области в виде строкового описания координат [topLeft: (x: <value>, y: <value>), bottomRight: (x: <value>, y: <value>)].

Пример:

```
local point = DocumentAPI.RectU(2, 3, 4, 5)
print(point:toString()) --[topLeft: (x: 2.0, y: 3.0), bottomRight: (x: 4.0, y:
5.0)]
```

4.26 Таблица DocumentAPI.VectorUInt

Таблица DocumentAPI.VectorUInt предназначена для реализации массива данных.

Пример:

```
vector = DocumentAPI.VectorUInt(3)
vector[0] = 1
vector[1] = 13
vector[2] = 25
print(vector:size()) -- 3
```

4.26.1 Метод `VectorUInt:size`

Метод возвращает размер вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
vector:push_back(13)  
vector:push_back(14)  
print(vector:size()) -- 3
```

4.26.2 Метод `VectorUInt:max_size`

Метод возвращает максимальный размер вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
print(vector:max_size())
```

4.26.3 Метод `VectorUInt:empty`

Метод возвращает `true`, если вектор не содержит элементов.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
print(vector:empty()) -- false
```

4.26.4 Метод `VectorUInt:clear`

Метод очищает содержимое вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
vector:clear()  
print(vector:empty()) -- true
```

4.26.5 Метод `VectorUInt:push_back`

Метод добавляет элемент в конец вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
print(vector:size()) -- 1
```

4.26.6 Метод VectorUInt:pop_back

Метод удаляет последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
vector:pop_back()  
print(vector:size()) -- 0
```

4.26.7 Метод VectorUInt:front

Метод возвращает первый элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
vector:push_back(13)  
print(vector:front()) -- 12
```

4.26.8 Метод VectorUInt:back

Метод возвращает последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
vector:push_back(13)  
print(vector:back()) -- 13
```

4.26.9 Метод VectorUInt:__getitem

Метод возвращает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorUInt()  
vector:push_back(12)  
vector:push_back(13)  
print(vector:__getitem(0)) -- 12  
print(vector:__getitem(1)) -- 13
```

4.26.10 Метод VectorUInt:__setitem

Метод устанавливает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorUInt(2)  
vector:__setitem(0, 12)  
vector:__setitem(1, 13)  
print(vector:__getitem(0)) -- 12  
print(vector:__getitem(1)) - 13
```

4.27 Таблица DocumentAPI.VectorString

Таблица DocumentAPI.VectorString предназначена для реализации массива строк.

Пример:

```
vector = DocumentAPI.VectorString(3)
vector[0] = "1"
vector[1] = "2"
vector[2] = "3"
print(vector:size()) -- 3
```

4.27.1 Метод VectorString:size

Метод возвращает размер вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
vector:push_back("13")
vector:push_back("14")
print(vector:size()) -- 3
```

4.27.2 Метод VectorString:max_size

Метод возвращает максимальный размер вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
print(vector:max_size())
```

4.27.3 Метод VectorString:empty

Метод возвращает true, если вектор не содержит элементов.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
print(vector:empty()) -- false
```


4.27.4 Метод `VectorString:clear`

Метод очищает содержимое вектора.

Пример:

```
local vector = DocumentAPI.VectorString()  
vector:push_back("12")  
vector:clear()  
print(vector:empty()) -- true
```

4.27.5 Метод `VectorString:push_back`

Метод добавляет элемент в конец вектора.

Пример:

```
local vector = DocumentAPI.VectorString()  
vector:push_back("12")  
print(vector:size()) -- 1
```

4.27.6 Метод `VectorString:pop_back`

Метод удаляет последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorString()  
vector:push_back("12")  
vector:pop_back()  
print(vector:size()) -- 0
```

4.27.7 Метод `VectorString:front`

Метод возвращает первый элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorString()  
vector:push_back("12")  
vector:push_back("13")  
print(vector:front()) -- 12
```

4.27.8 Метод VectorString:back

Метод возвращает последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorString()  
vector:push_back("12")  
vector:push_back("13")  
print(vector:front()) -- 13
```

4.27.9 Метод VectorString:__getitem

Метод возвращает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorString()  
vector:push_back("12")  
vector:push_back("13")  
print(vector:__getitem(0)) -- 12  
print(vector:__getitem(1)) -- 13
```

4.27.10 Метод VectorString:__setitem

Метод устанавливает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorString(2)  
vector:__setitem(0, "12")  
vector:__setitem(1, "13")  
print(vector:__getitem(0)) -- 12  
print(vector:__getitem(1)) -- 13
```

5 СПРАВОЧНИК ФУНКЦИЙ EDITORAPI

Глобальная таблица EditorAPI содержит функции доступа к внешней функциональности редактора.

5.1 Функция EditorAPI.getSelection

Функция EditorAPI.getSelection предоставляет доступ к выделенному фрагменту документа.

В открытом документе может быть выделен только один фрагмент.

При использовании в редакторе текста функция EditorAPI.getSelection возвращает Range, а при использовании в редакторе таблиц – CellRange.

Пример для текстового редактора:

Использование функции EditorAPI.getSelection в редакторе текста для печати выделенного фрагмента текста.

```
range = EditorAPI.getSelection()  
text = range.extractText()  
print(text)
```

Пример для табличного редактора:

Использование функции EditorAPI.getSelection в редакторе таблиц для печати значений ячеек в выделенном фрагменте таблицы.

```
cellRange = EditorAPI.getSelection()  
for cell in cellRange.enumerate() do  
    print(cell.getFormattedValue())  
end
```

5.2 Функция EditorAPI.setSelection

Функция EditorAPI.setSelection позволяет выделить фрагмент документа.

В открытом документе может быть выделен только один фрагмент.

Вызов функции для текстового документа:

```
EditorAPI.setSelection(range)
```

Где:

- range – выделяемый в текстовом документе фрагмент текста типа DocumentAPI.Range.

Пример для текстового документа:

```
EditorAPI.setSelection(document:getBlocks():getParagraph(0):getRange())
```

Вызов функции для табличного документа:

```
EditorAPI.setSelection(cellRange)
```

Где:

- cellRange – выделяемый в табличном документе фрагмент таблицы типа DocumentAPI.CellRange.

Пример для табличного документа:

```
cellRange = document:getBlocks():getTable(0):getCellRange("A1:E5")  
EditorAPI.setSelection(cellRange)
```

5.3 Функция EditorAPI.messageBox

Функция EditorAPI.messageBox() выводит на экран сообщение с заданным текстом и отображением кнопки **ОК**, при этом исполнение макрокоманды приостанавливается до нажатия кнопки **ОК**.

Вызов:

```
messageBox(prompt : string)  
messageBox(prompt : string)  
messageBox(prompt : string, title : string)
```

Параметры:

- prompt – текст сообщения;
- title – заголовок окна сообщения.

Пример:

```
EditorAPI.messageBox(cell:getFormattedValue())
```

5.4 Функция EditorAPI.showPrintDialog

Функция EditorAPI.showPrintDialog() показывает стандартное окно печати редактора и распечатывает документ, если пользователь подтверждает необходимость печати. Значения, возвращаемые функцией EditorAPI.showPrintDialog(), перечислены в разделе 4.22.

Пример:

```
printDocumentResult = EditorAPI.showPrintDialog()  
print(printDocumentResult)
```

5.5 Функция EditorAPI.printDocument

Функция `EditorAPI.printDocument()` предоставляет возможность печати документа с заданными параметрами печати. Описание параметров печати представлено в разделе 4.20.

Значения, возвращаемые функцией `EditorAPI.printDocument()`, перечислены в разделе 4.22.

Пример:

```
local printSettings = {}  
printSettings.printSelection = true  
EditorAPI.printDocument(printSettings)
```

5.6 Функция EditorAPI.isPrinterAvailable

Функция `EditorAPI.isPrinterAvailable` позволяет проверить доступность последнего использованного принтера. Возвращает `false`, если принтер недоступен.

Пример:

```
if EditorAPI.isPrinterAvailable() then  
    EditorAPI.messageBox("Printer is available")  
else  
    EditorAPI.messageBox("Printer is not available")  
end
```

5.7 Функция EditorAPI.getActiveWorksheet

Функция `EditorAPI.getActiveWorksheet()` возвращает активный лист в табличном документе (класс `Table`). Метод не предназначен для текстовых документов.

Пример:

```
activeWorksheet = EditorAPI.getActiveWorksheet()  
print(activeWorksheet.getName()) -- Лист1
```

5.8 Функция `EditorAPI.setActiveWorksheet`

Функция `EditorAPI.setActiveWorksheet()` устанавливает активный лист в табличном документе. В качестве параметра используется имя листа. Возвращает `true`, если лист найден по имени и активирован. Метод не предназначен для текстовых документов.

Пример:

```
print(EditorAPI.setActiveWorksheet("Лист1")) -- true or false
```

6 ФУНКЦИИ ДЛЯ РАБОТЫ СО СТРОКАМИ В ФОРМАТЕ ЮНИКОД (UTF-8)

Для работы со строками, содержащими русские символы, можно использовать методы таблицы `utf8`. Предполагается, что аргументы методов являются допустимыми строками UTF-8.

6.1 Функция `utf8.char`

Функция `utf8.char` возвращает строку в формате UTF-8, соответствующую коду символа.

Вызов:

```
utf8.char(code)
```

Параметры:

– `code`: код символа UTF-8, тип `number`.

Возвращает:

– `string`: символ UTF-8, полученный по коду.

Пример:

```
print(utf8.char(244)) -- ô
```

6.2 Функция `utf8.codes`

Функция `utf8.codes` возвращает последовательность кодов символов, из которых состоит строка UTF-8.

Вызов:

```
utf8.codes(str)
```

Параметры:

– `str`: строка в формате UTF-8

Возвращает:

– итератор, с помощью которого можно получить коды символов исходной строки.

Пример:

```
str = "МойОфис"  
for p, c in utf8.codes(str) do  
  print(c)  
end
```

```
-- 1052, 1086, 1081, 1054, 1092, 1080, 1089
```

6.3 Функция `utf8.codepoint`

Функция `utf8.codepoint` возвращает код заданного символа.

Вызов:

```
utf8.codepoint(char)
```

Параметры:

– `char`: символ UTF-8, тип `utf-char`.

Возвращает:

– `number`: код символа UTF-8.

Примеры:

```
print(utf8.codepoint("瑞")) -- 29790  
print(utf8.codepoint("A")) -- 1040
```

6.4 Функция `utf8.charpattern`

Функция `utf8.charpattern` возвращает шаблон `"[\0-\x7F\xC2-\xF4][\x80-\xBF]*"` для определения последовательности символов формата UTF-8.

Пример:

```
function len(s)  
  local n = 0  
  for match in s:gmatch(utf8.charpattern) do  
    n = n + 1  
  end  
  return n  
end  
  
str = "МойОфис"  
print(len(str))
```

6.5 Функция `utf8.upper`

Функция `utf8.upper` возвращает строку в формате UTF-8, полученную из исходной строки путем преобразования в верхний регистр.

Вызов:

```
utf8.upper(str)
```

Параметры:

– str: строка в формате UTF-8

Возвращает:

– string: строка в верхнем регистре.

Пример:

```
print(utf8.upper("a")) -- A  
print(utf8.upper("Abc")) -- ABC
```

6.6 Функция utf8.lower

Функция `utf8.lower` возвращает строку в формате UTF-8, полученную из исходной строки путем преобразования в нижний регистр.

Вызов:

```
utf8.lower(str)
```

Параметры:

– str: строка в формате UTF-8

Возвращает:

– string: строка в нижнем регистре.

Пример:

```
print(utf8.lower("A")) -- a  
print(utf8.lower("Abc")) -- abc
```

6.7 Функция utf8.substr

Функция `utf8.substr` возвращает подстроку в формате UTF-8, начиная с индекса `first` и заканчивая индексом `last`.

```
utf8.substr(str, first[, last])
```

Параметры:

- str: string – исходная строка в формате UTF-8;
- first: number – позиция первого символа подстроки;

– `last`: `number` – позиция последнего символа подстроки (по умолчанию равна позиции последнего символа в строке).

Возвращает:

– `string`: подстрока в формате UTF-8

- если позиция первого или последнего символа находится вне строки, то диапазон усекается до корректного;
- если диапазон задан некорректно, то возвращается пустая строка.

Пример:

```
print(utf8.substr("регистр", 5))    -- стр
print(utf8.substr("регистр", 0, 2)) -- ре
print(utf8.substr("регистр", 2, 1)) --
print(utf8.substr("регистр", 2, 100)) -- егистр
```

6.8 Функция `utf8.compare`

Функция `utf8.compare` возвращает результат сравнения двух строк согласно алгоритму сортировки по Юникоду.

Вызов:

```
utf8.compare(str1, str2, opt)
```

Параметры:

- `str1` – первая строка (`string`) в формате UTF-8;
- `str2` – вторая строка (`string`) в формате UTF-8;
- `opt` – параметр (`number`) учета регистра при сравнении:
 - 0 – без учета регистра;
 - 1 – с учетом регистра.

Возвращает:

- `number`: результат сравнения аргументов:
 - -1 – если `str1 < str2`;
 - 0 – если `str1 = str2`;
 - 1 – если `str1 > str2`.

Пример:

```
print(utf8.compare("A", "a", 0)) -- 0, arg1 = arg2 с учетом регистра  
print(utf8.compare("A", "a", 1)) -- -1, arg1 < arg2 без учета регистра
```

6.9 Функция utf8.islower

Функция `utf8.islower` проверяет, находится ли в нижнем регистре переданный символ или строка.

```
utf8.islower(str)
```

Параметр:

– `str`: строка, символ или число, представляющее код UTF-8.

Возвращает:

– `boolean`: `true`, если передан код символа в нижнем регистре.

Пример:

```
print(utf8.islower("a")) -- false  
print(utf8.islower("A")) -- true
```

6.10 Функция utf8.isupper

Функция `utf8.isupper` проверяет, находится ли в верхнем регистре переданный символ или строка.

```
utf8.isupper(str)
```

Параметр:

– `str`: строка, символ или число, представляющее код UTF-8.

Возвращает:

– `boolean`: `true`, если передан код символа в верхнем регистре.

Пример:

```
print(utf8.islower("a")) -- false  
print(utf8.islower("A")) -- true
```

6.11 Функция `utf8.isdigit`

Функция `utf8.isdigit` проверяет, является ли цифровым символом переданный символ или число.

```
utf8.isdigit(char)
```

Параметр:

– `char`: UTF-8-character – символ в кодировке UTF-8.

Возвращает:

– `boolean`: значение `true`, если передан код цифрового символа.

Пример:

```
print(utf8.isdigit("a")) -- false  
print(utf8.isdigit("1")) -- true
```

6.12 Функция `utf8.isalpha`

Функция `utf8.isalpha` проверяет, является ли буквенным символом переданный символ или число.

```
utf8.isalpha(char)
```

Параметр:

– `char`: символ в кодировке UTF-8.

Возвращает:

– `boolean`: `true`, если передан код буквенного символа.

Пример:

```
print(utf8.isalpha('A')) -- true  
print(utf8.isalpha('1')) -- false
```

6.13 Функция `utf8.len`

Функция `utf8.len` позволяет определить длину заданной строки в символах.

```
utf8.len(str)
```

Параметр:

– `str`: `string` – строка в формате UTF-8.

Возвращает:

– number: длина заданной строки в символах.

Пример:

```
print(utf8.len("МойОфис")) -- 7
```

6.14 Функция utf8.offset

Функция `utf8.offset` возвращает позицию (в байтах), с которой начинается кодирование символа с заданной позицией.

```
utf8.offset(str, charPos)
```

Параметр:

- str: string – строка в формате UTF-8;
- charPos: number – позиция символа.

Возвращает:

– number: позиция (в байтах), с которой начинается кодирование символа с заданной позицией.

Пример:

```
print(utf8.offset("АБВГДЕЖЗ", 5)) -- 9
```

6.15 Функция utf8.next

Функция `utf8.next` позволяет получить байтовое смещение символа, следующего за указанным.

Вызов:

```
utf8.next(str, offset)
```

Параметры:

- str – строка (string) в формате UTF-8;
- offset – байтовое смещение внутри UTF-8 строки (по умолчанию равно 1).

Возвращает:

– number – байтовое смещение следующего символа.

Пример:

```
next_idx = utf8.next("АБВГДЕЖЗ", 5)  
print(next_idx)
```

7 ФУНКЦИИ ДЛЯ РАБОТЫ С РЕГУЛЯРНЫМИ ВЫРАЖЕНИЯМИ

7.1 Функция `Re.create`

Функция `Re.create` компилирует регулярное выражение и возвращает его в виде объекта. По умолчанию используется Perl – совместимый формат регулярных выражений.

Вызов:

```
Re.create(pattern)
```

Параметры:

- `pattern (string)` – строка шаблона.

Возвращает:

- `regex (object)` – объект `Regex`, который содержит скомпилированное регулярное выражение для дальнейшего использования;
- `err (string)` – сообщение об ошибке или `nil`.

7.2 Функция `Re.match`

Сопоставляет скомпилированное регулярное выражение с заданной исходной строкой. Возвращает найденные подстроки.

Вызов:

```
Re.match(subject, matchFlags, pattern)
```

Параметры:

- `subject (string)` – исходная строка;
- `matchFlags (int)` – флаги, задающие правила применения регулярного выражения;
- `pattern (string, Regex)` – строка шаблона или скомпилированный шаблон.

Возвращает:

- `matches (object)` – подстроки, найденные в соответствии с шаблоном;
- `err (string)` – сообщение об ошибке или `nil`.

7.2.1 Флаги, используемые в `Re.match`

Эти флаги определены в пространстве имен `Re.Match`. Они используются во всех алгоритмах. Когда регулярное выражение применяется к последовательности символов, применяются правила, описанные в таблице 106.

Таблица 106 – Описание флагов `Re.Match`

Флаг	Описание
Default	Указывает, что работа с регулярными выражениями происходит в соответствии с обычными правилами: ECMA-262, спецификация языка ECMAScript, глава 15, часть 10, Регулярные Выражения.
NotBOB	Указывает, что выражения <code>"\A"</code> и <code>"\"</code> не должны совпадать с подмножеством <code>[first, first)</code> .
NotEOB	Указывает, что выражения <code>"\"</code> , <code>"\ z"</code> и <code>"\Z"</code> не должны совпадать с подмножеством <code>[last, last)</code> .
NotBOL	Указывает, что выражение <code>"^"</code> не должны совпадать с подмножеством <code>[first, first)</code> .
NotEOL	Указывает, что выражение <code>"\$"</code> не должно совпадать с подмножеством <code>[last, last)</code> .
NotBOW	Указывает, что выражения <code>"\<"</code> и <code>"\b"</code> не должны совпадать с подмножеством <code>[first, first)</code> .
NotEOW	Указывает, что выражения <code>"\>"</code> и <code>"\b"</code> не должны совпадать с подмножеством <code>[last, last)</code> .
Any	Указывает, что если существует более одного совпадения, то любое из совпадений является приемлемым результатом. Будет применено самое первое встретившееся совпадение, при этом, не всегда самое точное. Используйте этот флаг, если для вас важна скорость обработки, но не особо важно качество результата.
NotNull	Указывает, что выражение не может быть использовано для пустой последовательности.
Continious	Указывает, что выражение должно применяться к подмножеству, которое начинается с начала.
Partial	Указывает, что если не найдено ни одного совпадения, то допустимо вернуть совпадение <code>[from, last)</code> , при этом <code>from! = last</code> , если может существовать более длинная последовательность символов <code>[from, to)</code> , из которых <code>[from, last)</code> – это префикс, который приведет к полному совпадению. Этот флаг используется при сопоставлении неполных или очень длинных текстов; дополнительную информацию см. в документации по частичным сравнениям.
Extra	Дает указание механизму поиска сохранять всю доступную информацию о наличии совпадений; если совпадение повторяется снова, то информация о каждом из них будет доступна через методы <code>match_results::captures()</code> или <code>sub_match_captures()</code> .
SingleLine	Эквивалентно обратному модификатору <code>"m/"</code> языка Perl; предотвращает поиск <code>^</code> после встроенного символа новой строки (для того, чтобы он совпадал только в начале исходного текста) и <code>\$</code> от поиска перед встроенным символом новой строки (чтобы он совпадал только в конце исходного текста).
PrevAvail	Указывает, что <code>--first</code> является допустимой позицией итератора, когда этот флаг установлен, тогда флаги <code>match_not_bol</code> и <code>match_not_bow</code> игнорируются алгоритмами регулярных выражений (RE.7) и итераторов (RE.8).

Флаг	Описание
DotNewLine	Указывает, что выражение "." не распознается как символ новой строки. Это инверсия модификатора "s/" языка Perl.
NotDotNull	Указывает, что выражение "." не распознается как символ null ('\0').
Posix	Указывает, что выражение должно быть обработано в соответствии с правилом POSIX <i>"leftmost-longest"</i> , независимо от того, какое выражение было скомпилировано. Эти правила плохо работают со многими специфичными для Perl моментами, например, такими как ленивые (<i>"non-greedy"</i>) повторы.
NoSubs	Заставляет выражение вести себя так, как будто оно не имеет найденных подмножеств, независимо от того, сколько их присутствует на самом деле. Класс Matches будет содержать только информацию об общем совпадении, а не о совпадении в подмножествах.

7.3 Функция Re.search

Ищет скомпилированное регулярное выражение по заданной строке. Метод возвращает найденные подстроки.

Вызов:

```
Re.search(subject, matchFlags, pattern)
```

Параметры:

- `subject (string)` – исходная строка;
- `matchFlags (int)` – флаги, задающие правила применения регулярного выражения;
- `pattern (string, Regex)` – строка шаблона или скомпилированный шаблон.

Возвращает:

- `matches (object)` – подстроки, найденные в соответствии с шаблоном;
- `err (string)` – сообщение об ошибке или `nil`.

7.4 Функция Re.replace

Находит в заданной строке все фрагменты, удовлетворяющие регулярному выражению. Каждый найденный фрагмент форматируется в соответствии с форматтером и заменяет собой исходный текст.

Вызов:

```
Re.replace(subject, formatter, matchFlags, pattern)
```

Параметры:

- `subject (string)` – исходная строка для поиска;

- `formatter (string)` – строка, задающая форматирование найденных фрагментов;
- `matchFlags (int)` – флаги, задающие правила применения регулярного выражения, а также флаги, специфичные для замены;
- `pattern (string, Regex)` – строка шаблона или скомпилированный шаблон.

Возвращает:

- `newString (string)` – новая строка с замененными подстроками;
- `err (string)` – сообщение об ошибке или `nil`.

7.5 Флаги, используемые для замены

Эти флаги определены в пространстве имен `Re.Replace`. Они используются в алгоритме, используемом методом `Re.replace()` и находятся в таблице 107.

Таблица 107 – Описание флагов для функции `Re.replace()`

Флаг	Описание
FormatDefault	<p>Когда к исходному тексту применяется регулярное выражение, и находится очередной фрагмент для замены, новая строка формируется с использованием функции замены <i>ECMAScript</i>, ECMA-262, Спецификация языка ECMAScript, глава 15, часть 5.4.11 <code>String.prototype.replace</code>.</p> <p>Эта функциональность идентична описанной в руководстве Perl <code>Format String Syntax</code>.</p> <p>После того, как все неперекрывающиеся вхождения регулярного выражения найдены и заменены, фрагменты исходного текста, не соответствующие регулярному выражению, копируются в результирующую строку без изменений.</p>
FormatSed	<p>В случае, когда регулярное выражение применяется для замены в строке, новая строка формируется с использованием правил, описанных в стандарте IEEE Std 1003.1-2001, Portable Operating System Interface (POSIX™), Shells and Utilities. См. также <code>Sed Format String Syntax</code>.</p>
FormatPerl	<p>В случае, когда регулярное выражение применяется для замены в строке, новая строка формируется по правилам Perl 5.</p>
FormatLiteral	<p>В случае, когда регулярное выражение применяется для замены в строке, новая строка будет являться строковой копией заменяемого текста.</p>
FormatNoCopy	<p>В случае, когда регулярное выражение применяется для замены в строке, фрагменты, не соответствующие регулярному выражению, не будут копироваться в результирующую строку.</p>
FormatFirstOnly	<p>Когда данный флаг установлен при операции поиска или замены, то заменяется только первое вхождение регулярного выражения.</p>

Флаг	Описание
FormatAll	Все синтаксические расширения включены, включая условные замены (<i>? ddexpression1:expression2</i>). Для дополнительных деталей см. Руководство по форматированию строк Boost.

8 КЛАСС MATCHES

Класс `Matches` содержит результат функций `Re.match()` и `Re.search()`.

8.1 Метод `getFirst`

Вызов:

```
position, err = matches.getFirst(group)
```

Параметры:

- `group (int, string)` – позиция (или имя группы) найденных результатов, начинающаяся с 1.

Возвращает:

- `position (int)` – первая позиция (в байтах) исходной строки;
- `err (string)` – сообщение об ошибке или `nil`.

8.2 Метод `getLength`

Вызов:

```
position, err = matches.getLength(group)
```

Параметры:

- `group (int, string)` – позиция (или имя группы) найденных результатов, начинающаяся с 1.

Возвращает:

- `length (int)` – длина исходной строки в байтах;
- `err (string)` – сообщение об ошибке или `nil`.

8.3 Метод `getSize`

Вызов:

```
size, err = matches.getSize()
```

Возвращает:

- `size (int)` – количество найденных групп;
- `err (string)` – сообщение об ошибке или `nil`.

8.4 Метод getString

Вызов:

```
substr, err = matches.getString(group, subject)
```

Параметры:

- group (int, string) – позиция (или имя группы) найденных результатов, начинающаяся с 1;
- subject (string) – исходная строка. **Внимание:** объект Matches сохраняет только смещения и не хранит исходную строку. Таким образом, необходимо передать ту же строку, которая использовалась для поиска.

Возвращает:

- substr (string) – найденная подстрока;
- err (string) – сообщение об ошибке или nil.

8.5 Метод _toString

Стандартная метафункция.

```
string = matches.__toString()
```

Пример:

```
local str = "-Номер:1234"
local regex = Re.create("-(\\w+):(\\d{4})")

local matches, err = Re.match(str, Re.Match.Default, regex)
print(tostring(regex), tostring(matches))

local number = matches.getString(3, str)
print(number)
```