МойОфис

Справочник макрокоманд на языке программирования Lua

2022.01

ООО «НОВЫЕ ОБЛАЧНЫЕ ТЕХНОЛОГИИ»

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

СПРАВОЧНИК МАКРОКОМАНД НА ЯЗЫКЕ LUA

2022.01

На 160 листах

Москва

2022

Все упомянутые в этом документе названия продуктов, логотипы, торговые марки и товарные знаки принадлежат их владельцам.

Товарные знаки «МойОфис» и «MyOffice» принадлежат ООО «НОВЫЕ ОБЛАЧНЫЕ ТЕХНОЛОГИИ».

Ни при каких обстоятельствах нельзя истолковывать любое содержимое настоящего документа как прямое или косвенное предоставление лицензии или права на использование товарных знаков, логотипов или знаков обслуживания, приведенных в нем. Любое несанкционированное использование этих товарных знаков, логотипов или знаков обслуживания без письменного разрешения их правообладателя строго запрещено.

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

В настоящем документе используются следующие сокращения (см. Таблица 1):

Таблица 1 - Сокращения и расшифровки

Сокращение	Расшифровка
ОС	Операционная система.
ПО МойОфис	Программное обеспечение МойОфис Plus. Документы.
Объектная модель	Совокупность структур данных для управления содержимым текстового или табличного документа.
EULA	End User License Agreement (пользовательское соглашение).
SDK	Software Development Kit (комплект для разработки программного обеспечения).

1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

1.1 Назначение программы

МойОфис Plus – комплексный продукт для организации рабочей среды в крупных государственных организациях и коммерческих предприятиях. Единая лицензия МойОфис Plus включает операционную систему, средства безопасного хранения файлов, работы с документами, ведения переписки по электронной почте и планирования рабочего времени.

«МойОфис Текст» обеспечивает удобное и быстрое создание документов с использованием шаблонов, стилей и средств форматирования текста. Функции совместного редактирования обеспечивают эффективную совместную работу сотрудников.

«МойОфис Таблица» – это приложение для быстрой и удобной работы с электронными таблицами и анализа данных. Продукт поддерживает расширенный набор формул и средств для обработки данных. Совместное редактирование на любой из поддерживаемых платформ обеспечивает быстрый анализ и подготовку документов группой сотрудников даже вне офиса.

«МойОфис Презентация» – приложение с полным набором инструментов для просмотра графических презентаций.

1.2 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 `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 `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.

1.3 Перечень эксплуатационной документации

Настоящий документ содержит описание объектной модели документа ПО МойОфис и примеры ее использования, а также является справочником по возможностям объектной модели редакторов ПО МойОфис.

Вся необходимая информация по использованию макрокоманд в ПО МойОфис приведена в настоящем документе.

2 РАБОТА С МАКРОКОМАНДАМИ

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

2.1 Редактор макрокоманд

2.1.1 Окно редактора макрокоманд

Для работы с макрокомандами используется редактор макрокоманд. Чтобы открыть окно редактора, в приложении «МойОфис Текст» или «МойОфис Таблица» выберите пункт командного меню **Инструменты** > **Редактирование макроса**.

Окно редактора макрокоманд содержит (см. Рисунок 1):

1. Перечень созданных в документе макрокоманд.
2. Кнопки для создания + и удаления − макрокоманд.
3. Область ввода текста макрокоманд.
4. Кнопки выполнения (см. раздел Выполнение макрокоманд) и отладки (см. раздел Отладка макрокоманд) макрокоманд. Кнопки становятся активными, изменяя цвет, после ввода текста макрокоманд в области 3 (см. раздел Редактирование макрокоманд).
5. Область вывода результата выполнения макрокоманд, а также отображения информации в процессе отладки макрокоманд.

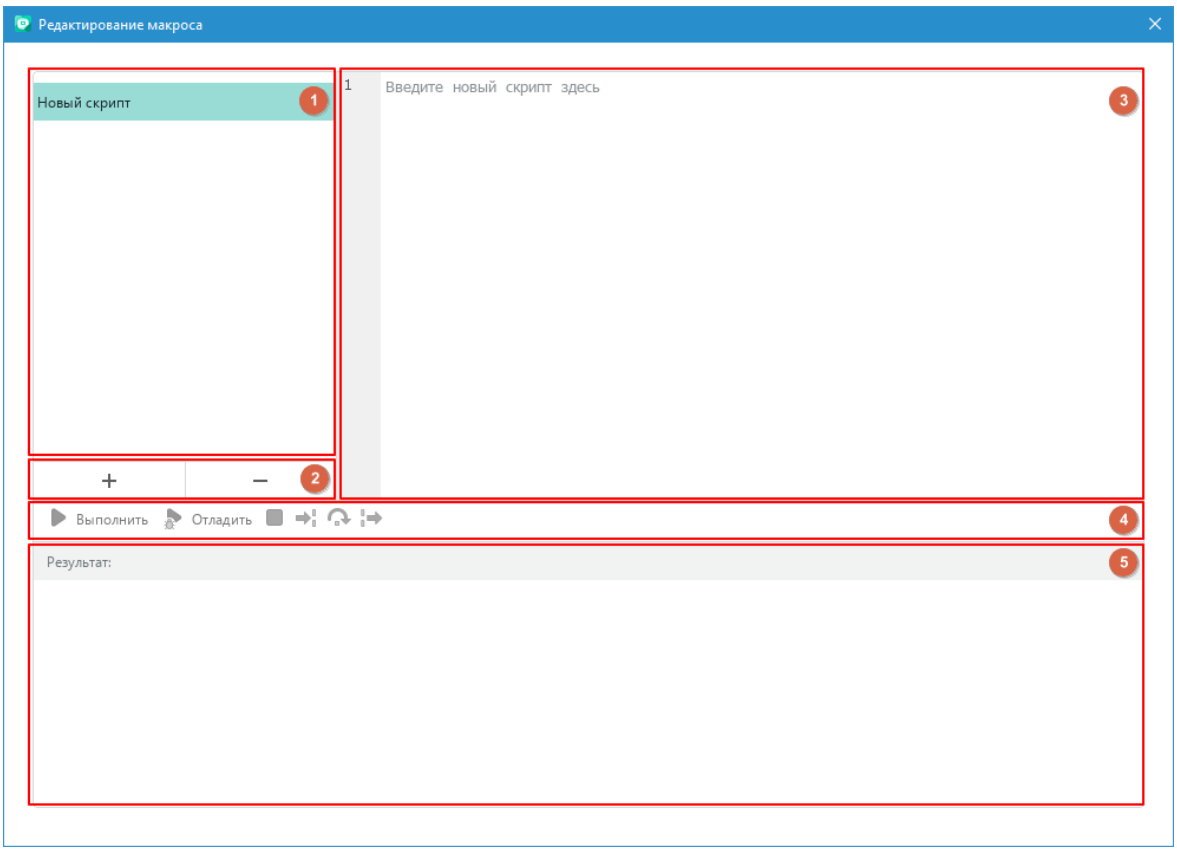


Рисунок 1 – Окно редактора макрокоманд

2.1.2 Создание макрокоманд

Для создания макрокоманды выполните следующие действия (см. Рисунок 1):

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 **Редактирование макроса.**
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 1).

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 1).

2.1.5 Удаление макрокоманд

Чтобы удалить макрокоманду, выберите ее в перечне макрокоманд и нажмите кнопку − в области 2 (см. Рисунок 1).

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 2).

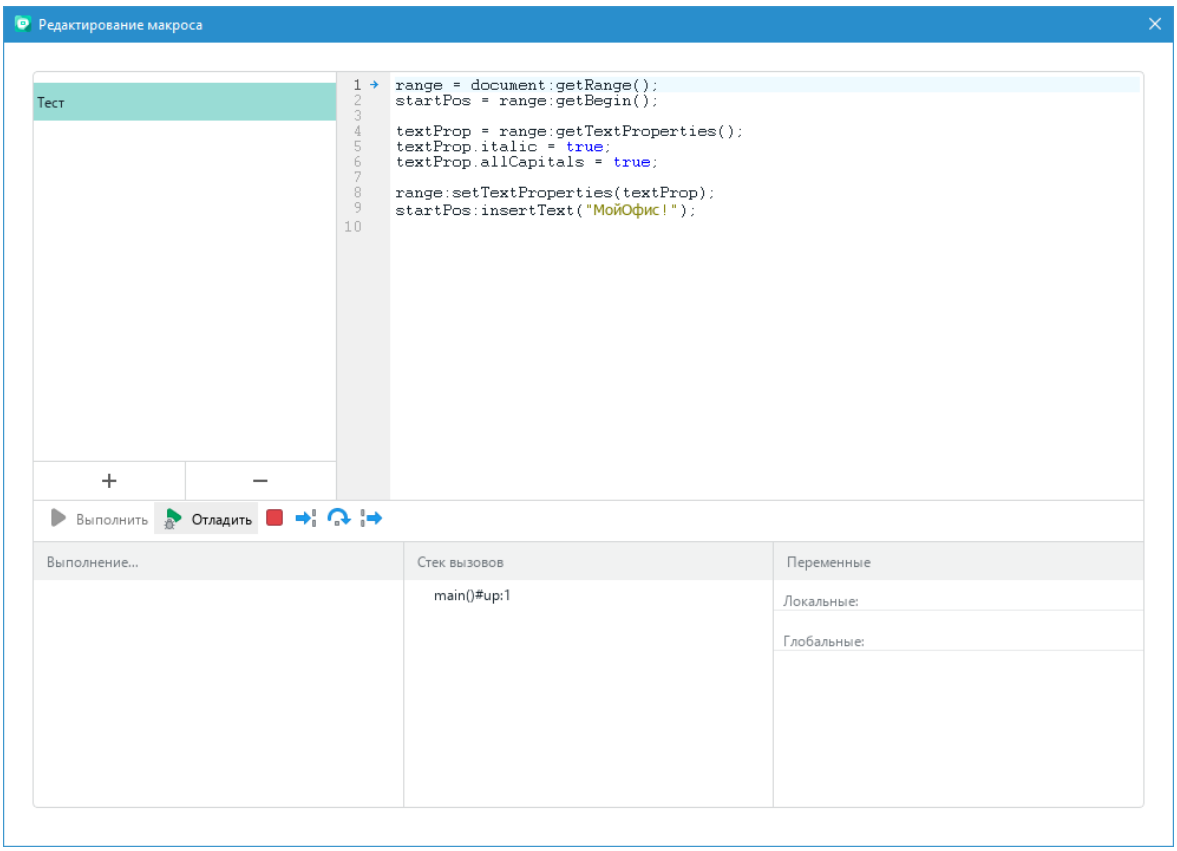


Рисунок 2 – Окно редактора макрокоманд в начале отладки

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 2):

– – 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 2), 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 3):

- **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.

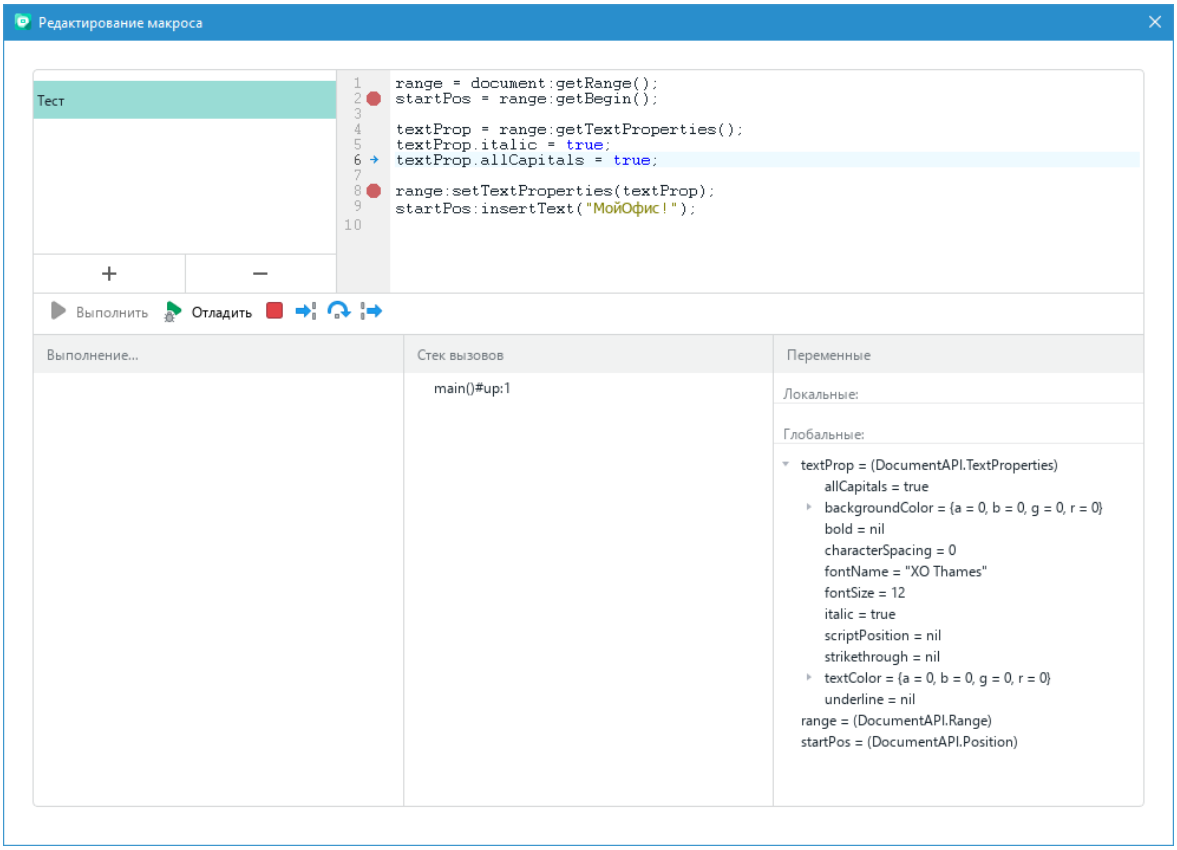


Рисунок 3 – Окно редактора макрокоманд в процессе отладки макрокоманд

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 > Редактирование макроса.
2. Нажать кнопку + для создания новой макрокоманды. Указать имя макрокоманды. По умолчанию новой макрокоманде присваивается имя «Новый скрипт».
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 Редактирование и запуск макрокоманды в текстовом документе.

С помощью последовательности вызовов устанавливается курсор в начало документа:

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

Таблица Document представляет текущий открытый текстовый документ.

Таблица 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 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 K (**Курсив**) в пользовательском интерфейсе текстового редактора;
- свойство textProp.allCapitals принимает значение **true**, что равносильно нажатию кнопки AB (**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. 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.

ПО МойОфис 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 существует временное ограничение по работе в тексте макрокоманд с визуальными элементами, такими как выпадающие списки, переключатели и некоторыми другими.

3 СПРАВОЧНИК ТАБЛИЦ И МЕТОДОВ

3.1 Форматы документов DocumentFormat

В Таблице 2 приведены поддерживаемые форматы документов.

Таблица 2 – Форматы документов

Наименование константы	Описание
`DocumentAPI.DocumentFormat_PlainText`	Используется для работы с файлами TXT.
`DocumentAPI.DocumentFormat_DSV`	Используется для работы с табличными данными в текстовой форме (CSV, DSV). Строка текста содержит одно или несколько полей данных, разделенных запятыми или иным разделителем.
`DocumentAPI.DocumentFormat_OXML`	Используется для работы с текстовыми (DOCX) или табличными (XLSX) документами в формате Open Office XML.
`DocumentAPI.DocumentFormat_ODF`	Используется для работы с текстовыми (ODT) или табличными (ODS) документами формата Open Document Format (ГОСТ Р ИСО/МЭК 26300-2010).
`DocumentAPI.DocumentFormat_HTML`	Используется для работы с веб-документами в формате HTML. Работа с веб-документами в формате HTML средствами Document API в настоящий момент не поддерживается.
`DocumentAPI.DocumentFormat_PDF`	Используется для работы с документами в формате Portable Dcoument Format (PDF), версии 1.4. Средствами Document API поддерживается только операция экспорта документа в формат PDF/A-1b.
`DocumentAPI.DocumentFormat_PDFA`	Используется для работы с документами в формате Portable Document Format (PDF) для долгосрочного архивного хранения (PDF/A-1b). Средствами Document API поддерживается только операция экспорта документа в формат PDF/A-1b.

3.2 Типы документов DocumentType

В Таблице 3 приведены поддерживаемые типы документов.

Таблица 3 - Типы документов

Наименование константы	Описание
`DocumentAPI.DocumentType_Text`	Используется для работы с текстовыми документами – файлы DOCX, ODT, XODT, TXT.
`DocumentAPI.DocumentType_Workbook`	Используется для работы с табличными документами – файлы XLSX, ODS, XODS.
`DocumentAPI.DocumentType_Presentation`	Используется для работы с презентационными документами – файлы PPTX, ODP. Работа с презентационными документами

Наименование константы	Описание
	средствами Document API в настоящий момент не поддерживается.

3.3 Форматы экспорта документов ExportFormat

В Таблице 4 приведены поддерживаемые форматы экспорта документов.

Таблица 4. Форматы экспорта документов

Наименование константы	Описание
`DocumentAPI.ExportFormat_PDFA1`	Используется для работы с документами в формате Portable Document Format (PDF) для долгосрочного архивного хранения (PDF/A-1b).

3.4 Неподдерживаемые свойства документа SaveUnsupportedFeature

Имеются свойства документа, которые могут быть утеряны при сохранении документа. Данные свойства описаны в Таблице 5.

Таблица 5 - Форматы неподдерживаемых свойств документа

Наименование константы	Описание
`SaveUnsupportedFeature_CommentsInHeaderFooter`	Комментарии в колонтитулах.
`SaveUnsupportedFeature_CommentsInFootnote`	Комментарии в сносках.
`SaveUnsupportedFeature_CommentsInFootnote`	Параграфы с разным выравниванием в заметках.

3.5 Кодировки документов Encoding

В Таблице 6 приведены поддерживаемые кодировки документов.

Таблица 6 - Кодировки документов

Наименование константы	Кодировка
`DocumentAPI.Encoding_Unknown`	Невозможно определить кодировку.
`DocumentAPI.DocumentAPI.Encoding_UTF8`	UTF8
`DocumentAPI.Encoding_UTF16BE`	UTF16BE
`DocumentAPI.Encoding_UTF16LE`	UTF16LE
`DocumentAPI.Encoding_UTF32BE`	UTF32BE
`DocumentAPI.Encoding_UTF32LE`	UTF32LE

Наименование константы	Кодировка
DocumentAPI.Encoding_Windows1250	Windows1250
DocumentAPI.Encoding_Windows1251	Windows1251
DocumentAPI.Encoding_Windows1252	Windows1252
DocumentAPI.Encoding_ISO8859Part5	ISO8859Part5
DocumentAPI.Encoding_KOI8R	KOI8R
DocumentAPI.Encoding_KOI8U	KOI8U
DocumentAPI.Encoding_CP866	CP866

3.6 Системы адресации ячеек FormulaType

Поддерживаемые системы адресации ячеек (стили ссылок) в табличном документе представлены в Таблице 7.

Таблица 7 – Системы адресации ячеек в табличном документе

Наименование константы	Система адресации ячеек	Описание
DocumentAPI.FormulaType_A1	A1	Вариант A1 соответствует наиболее распространенной системе адресации ячеек, при которой столбцы задаются буквами, а строки – числами.
DocumentAPI.FormulaType_R1C1	R1C1	Вариант R1C1 соответствует альтернативной системе адресации ячеек, при которой столбцы и строки задаются числами.

3.7 Типы линии LineStyle

В Таблице 8 приведены типы линий.

Таблица 8 – Типы линий

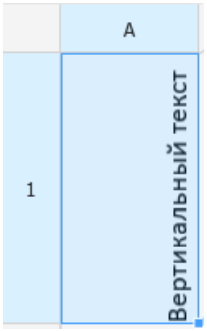
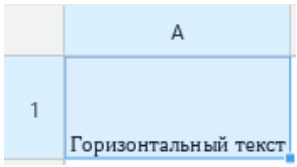
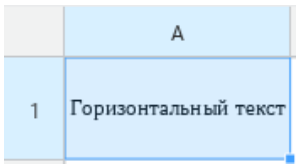
Наименование константы	Описание
DocumentAPI.LineStyle_NoLine	Линия отсутствует.
DocumentAPI.LineStyle_Solid	
DocumentAPI.LineStyle_Dot	
DocumentAPI.LineStyle_Dash	

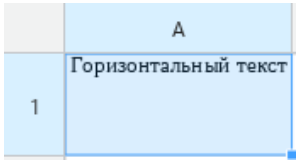
Наименование константы	Описание
DocumentAPI.LineStyle_LongDash	
DocumentAPI.LineStyle_DashDot	
DocumentAPI.LineStyle_DotDotDash	
DocumentAPI.LineStyle_Double	
DocumentAPI.LineStyle_Wave	

3.8 Виды выравнивания текста по вертикали VerticalAlignment

В Таблице 9 представлены константы видов выравнивания текста по вертикали.

Таблица 9 – Виды выравнивания текста по вертикали

Имя константы	Представление в интерфейсе	
DocumentAPI.VerticalAlignment_Bottom		
DocumentAPI.VerticalAlignment_Center		

Имя константы	Представление в интерфейсе	
DocumentAPI.VerticalAlignment_Top		

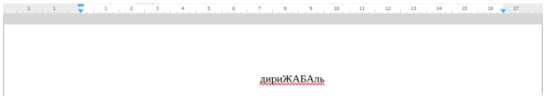
Пример:

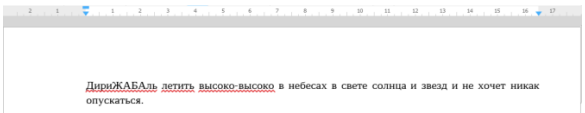
```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A1")
local props = cell:getCellProperties()
props.verticalAlignment = DocumentAPI.VerticalAlignment_Center
cell:setCellProperties(props)
```

3.9 Виды выравнивание текста по горизонтали Alignment

В Таблице 10 представлены константы видов выравнивания по горизонтали текстовых фрагментов в текстовом редакторе или содержимого ячеек в табличном редакторе.

Таблица 10 – Виды выравнивания по горизонтали текстовых фрагментов

Имя константы	Описание
DocumentAPI.Alignment_Default	Выравнивание по умолчанию.
DocumentAPI.Alignment_Left	По левому краю 
DocumentAPI.Alignment_Center	По центру 
DocumentAPI.Alignment_Right	По правому краю 
DocumentAPI.Alignment_Justify	По ширине

Имя константы	Описание
	

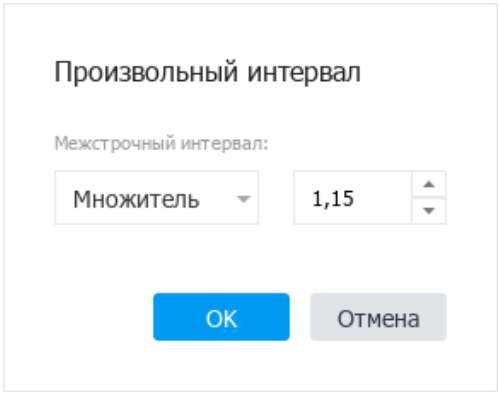
Пример:

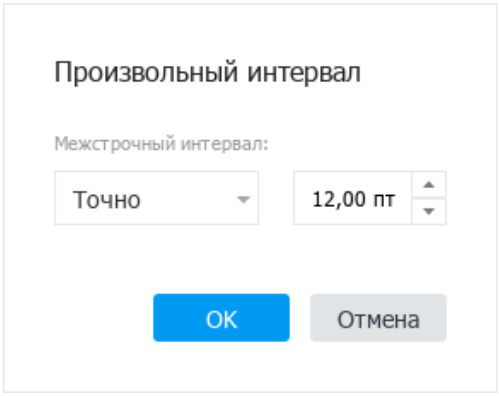
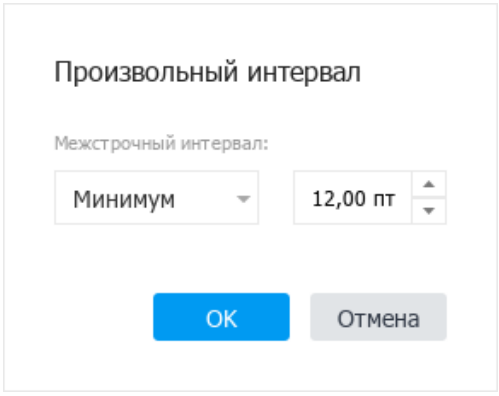
```
local para = document:getBlocks():getParagraph(0)
local props = para:getParagraphProperties()
props.alignment = DocumentAPI.Alignment_Center
para:setParagraphProperties(props)
```

3.10 Виды межстрочного интервала LineSpacingRule

В Таблице 11 представлены константы видов межстрочного интервала для абзаца текста.

Таблица 11 – Виды межстрочного интервала

Наименование константы	Описание
`DocumentAPI.LineSpacingRule_Multiple`	Установка произвольного межстрочного интервала с использованием множителя. При вызове необходимо указать значение множителя, например: `pPr.lineSpacing = DocumentAPI.LineSpacing(1.15, DocumentAPI.LineSpacingRule_Multiple)` В данном примере используется значение множителя **1.15**. Действие команды аналогично настройке межстрочного интервала вручную с помощью окна **Произвольный интервал** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»). 
`DocumentAPI.LineSpacingRule_Exact`	Установка произвольного межстрочного интервала с использованием точного значения. При вызове необходимо указать точное значение, например: `pPr.lineSpacing = DocumentAPI.LineSpacing(12.0, DocumentAPI.LineSpacingRule_Exact)`

Наименование константы	Описание
	В данном примере используется точное значение **12.0**. Действие команды аналогично настройке межстрочного интервала вручную с помощью окна **Произвольный интервал** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»). 
`DocumentAPI.LineSpacingRule_AtLeast`	Установка произвольного междустрочного интервала с использованием минимального значения. При вызове необходимо указать минимальное значение, например: `pPr.lineSpacing = DocumentAPI.LineSpacing(12.0, DocumentAPI.LineSpacingRule_AtLeast)` В данном примере используется точное значение **12.0**. Действие команды аналогично настройке межстрочного интервала вручную с помощью окна **Произвольный интервал** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»). 

Пример:

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

```
DocumentAPI.LineSpacingRule_Multiple)
p:setParagraphProperties(pPr)
```

3.11 Виды размещения текста в ячейках таблиц

В Таблице 12 приведены константы видов размещения текста в ячейках таблицы.

Таблица 12 – Виды размещения текста в ячейках таблиц

Наименование константы	Описание
DocumentAPI.TextLayout_SingleLine	Отображение текста в одну строку с наложением на соседние ячейки.
DocumentAPI.TextLayout_WrapByWords	Перенос данных внутри ячейки по словам.
DocumentAPI.TextLayout_ShrinkSizeToFitWidth	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)
```

3.12 Форматы ячеек таблицы CellFormat

По умолчанию при создании документа всем ячейкам присваивается формат «Общий».

Поддерживаемые форматы представлены в Таблице 13.

Таблица 13 – Поддерживаемые форматы ячеек таблицы

Наименование константы	Описание
DocumentAPI.CellFormat_General	Формат ячейки «Общий». 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.

Наименование константы	Описание
	Числа, состоящие более чем из 12 символов, переводятся в экспоненциальную форму после завершения ввода в ячейку.
`DocumentAPI.CellFormat_Percentage`	Формат ячейки «Процентный». Этот формат используется для представления чисел как процентов. При применении формата «Процентный» введенное число умножается на 100 и обозначается знаком «%».
`DocumentAPI.CellFormat_Number`	Формат ячейки «Числовой». Если в ячейке с форматом «Числовой» содержится дробное число, то можно указать количество знаков, отображаемых в данном числе после разделителя.
`DocumentAPI.CellFormat_Text`	Формат ячейки «Текстовый».
`DocumentAPI.CellFormat_Currency`	Формат ячейки «Денежный». Этот формат используется для представления чисел со знаком или кодом валюты.
`DocumentAPI.CellFormat_Accounting`	Формат ячейки «Финансовый». Этот формат применяется для чисел, используемых в бухгалтерских документах. В формате «Финансовый» введенное число автоматически дополняется названием валюты, которая соответствует настройкам системы компьютера. Отрицательные числа в формате «Финансовый» заключаются в круглые скобки в ячейке, а в строке формул остаются в том виде, в котором они были введены.
`DocumentAPI.CellFormat_Date`	Формат ячейки «Дата». Этот формат автоматически присваивается числам, введенным в определенном виде, например, ДД.ММ.ГГГГ.
`DocumentAPI.CellFormat_Time`	Формат ячейки «Время». Этот формат автоматически присваивается числам, введенным в определенном виде, например, ЧЧ:ММ.

Наименование константы	Описание
`DocumentAPI.CellFormat_Fraction`	Формат ячейки «Дробный». Этот формат используется для представления дробных чисел в виде обыкновенных дробей (то есть дробная часть заменяется на числитель и знаменатель).
`DocumentAPI.CellFormat_Scientific`	Формат ячейки «Экспоненциальный». Экспоненциальный (или научный) формат используется для представления больших чисел в короткой форме. Все введенные числа длиной более 12 символов автоматически переводятся в этот формат. В ячейке число в формате «Экспоненциальный» представлено следующим образом: • 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.

Пример:

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

Результат:

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

3.13 Типы форматов даты DatePatterns

Перечисления форматов даты представлены в Таблице 14.

Таблица 14 – Форматы даты

Наименование константы	Описание
DocumentAPI.DatePatterns_DayMonthTextLongYearLong	'mmmm dd, yyyy' для языка en_US.
DocumentAPI.DatePatterns_FullDate	'день недели, mmmm dd, yyyy' 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-yyyy'.
DocumentAPI.DatePatterns_DayMonthYearLongNonLocalizableSlash	Нелокализуемый шаблон 'dd/mm/yyyy'.

3.14 Типы форматов времени TimePatterns

Перечисления форматов времени представлены в Таблице 15.

Таблица 15 – 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.

3.15 Типы надстрочного и подстрочного форматирования ScriptPosition

Варианты представления текста в виде надстрочных или подстрочных знаков при работе в текстовом редакторе представлены в Таблице 16.

Таблица 16 – Типы надстрочного и подстрочного форматирования

Наименование константы	Описание
DocumentAPI.ScriptPosition_SuperScript	Надстрочный знак (верхний индекс).
DocumentAPI.ScriptPosition_SubScript	Подстрочный знак (нижний индекс).
DocumentAPI.ScriptPosition_NoramlScript	Без указания индекса.

3.16 Типы схем форматирования списков ListSchema

Типы схем форматирования списков, которые могут быть применены к абзацам текста представлены в Таблице 17.

Таблица 17 – Типы схем форматирования списков

Наименование константы	Описание
DocumentAPI.ListSchema.Unknown	Неизвестно.
DocumentAPI.ListSchema.UnknownBullet	Список без маркера.
DocumentAPI.ListSchema.UnknownNumbering	Нумерация без номера.
DocumentAPI.ListSchema.BulletCircleSolid	Список с маркерами в виде 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.
DocumentAPI.ListSchema.EnumeratorLatinLowercaseDot	Нумерация латинскими строчными буквами с точкой.

Наименование константы	Описание
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)
```

3.17 Типы ориентации страницы PageOrientation

Типы ориентации страницы представлены в Таблице 18.

Таблица 18 – Типы ориентации страницы

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

3.18 Типы отслеживаемых изменений TrackedChangeType

Типы отслеживаемых изменений представлены в Таблице 19.

Таблица 19 – Типы отслеживаемых изменений

Наименование константы	Описание
DocumentAPI.TrackedChangeType_Added	Добавленные изменения.
DocumentAPI.TrackedChangeType_Removed	Удаленные изменения.

3.19 Типы колонтитулов HeaderFooterType

Типы колонтитулов представлены в Таблице 20.

Таблица 20 – Типы колонтитулов

Наименование константы	Описание
DocumentAPI.HeaderFooterType_Header	Верхний колонтитул.

Наименование константы	Описание
DocumentAPI.HeaderFooterType_Footer	Нижний колонтитул.

3.20 Варианты кодов, возвращаемых после печати PrintDocumentResult

В Таблице 21 представлены коды, возвращаемые после печати.

Таблица 21 – Коды, возвращаемые после печати

Наименование константы	Описание
DocumentAPI.PrintDocumentResult_Success	Печать прошла успешна.
DocumentAPI.PrintDocumentResult_OneCopyPrinted	Напечатана только одна копия из заданных.
DocumentAPI.PrintDocumentResult_CancelPrinting	Печать была отменена.
DocumentAPI.PrintDocumentResult_NoPrinter	Принтер не найден.
DocumentAPI.PrintDocumentResult_BlankDocument	На печать отправлен пустой документ.

3.21 Типы окончания линии LineEndingStyle

В Таблице 22 приведены типы окончания линий.

Таблица 22 – Типы окончания линии

Наименование константы	Описание
DocumentAPI.LineEndingStyle_Arrow	
DocumentAPI.LineEndingStyle_Diamond	
DocumentAPI.LineEndingStyle_Oval	
DocumentAPI.LineEndingStyle_Stealth	
DocumentAPI.LineEndingStyle_Triangle	
DocumentAPI.LineEndingStyle_None	

3.22 Типы идентификаторов цветов тем ThemeColorID

В Таблице 23 представлены типы идентификаторов цветов тем.

Таблица 23 – Типы идентификаторов цветов тем

Наименование константы	Описание
DocumentAPI.ThemeColorID_Background1	Фон1

Наименование константы	Описание
DocumentAPI.ThemeColorID_Text1	Текст1
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.

3.23 Варианты обтекания текстом встроенного объекта TextWrapType

В Таблице 24 представлены варианты обтекания текстом встроенного объекта.

Таблица 24 – Варианты обтекания текстом встроенного объекта

Наименование константы	Описание
DocumentAPI.TextWrapType_Inline	Встроенный объект располагается в тексте.
DocumentAPI.TextWrapType_InFrontOfText	Встроенный объект располагается перед текстом.
DocumentAPI.TextWrapType_BehindText	Встроенный объект располагается за текстом.
DocumentAPI.TextWrapType_TopAndBottom	Текст располагается сверху и снизу встроенного объекта.
DocumentAPI.TextWrapType_Square	Текст располагается вокруг прямоугольной рамки встроенного объекта.
DocumentAPI.TextWrapType_Through	Текст обтекает встроенный объект по сторонам и внутри.

3.24 Типы размещения объекта по вертикали VerticalRelativeTo

В Таблице 25 представлены типы размещения объекта относительно закрепленной позиции по вертикали.

Таблица 25 – Типы размещения относительно закрепленной позиции по вертикали

Наименование константы	Описание
DocumentAPI.VerticalRelativeTo_Character	Символ.
DocumentAPI.VerticalRelativeTo_BaseLine	Базовая линия.
DocumentAPI.VerticalRelativeTo_Paragraph	Абзац.
DocumentAPI.VerticalRelativeTo_Page	Страница.
DocumentAPI.VerticalRelativeTo_PageContent	Содержимое страницы.
DocumentAPI.VerticalRelativeTo_PageTopMargin	Верхнее поле страницы.
DocumentAPI.VerticalRelativeTo_PageBottomMargin	Нижнее поле страницы.
DocumentAPI.VerticalRelativeTo_PageInsideMargin	Внутреннее поле страницы.
DocumentAPI.VerticalRelativeTo_PageOutsideMargin	Внешнее поле страницы.

3.25 Типы размещения объекта по горизонтали HorizontalRelativeTo

В Таблице 26 представлены типы размещения объекта относительно закрепленной позиции по горизонтали.

Таблица 26 – Типы размещения 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.

Наименование константы	Описание
DocumentAPI.HorizontalRelativeTo_PageInsideMargin	Внутреннее поле страницы.
DocumentAPI.HorizontalRelativeTo_PageOutsideMargin	Внешнее поле страницы.

3.26 Типы выравнивания объекта по вертикали VerticalAnchorAlignment

В Таблице 27 представлены типы выравнивания объекта относительно закрепленной позиции по вертикали.

Таблица 27 – Типы выравнивания 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.

3.27 Типы выравнивания объекта по горизонтали HorizontalAnchorAlignment

В Таблице 28 представлены типы выравнивания объекта относительно закрепленной позиции по горизонтали.

Таблица 28 – Типы выравнивания 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.

3.28 Выбор страниц для экспорта и печати PageParity

Варианты отбора страниц для экспорта и печати представлены в Таблице 29.

Таблица 29 – Варианты отбора страниц для экспорта и печати

Наименование константы	Описание
DocumentAPI.PageParity_Odd	Печать только нечетных страниц.
DocumentAPI.PageParity_Even	Печать только четных страниц.

Наименование константы	Описание
DocumentAPI.PageParity_All	Печать всех страниц.

3.29 Диапазон страниц для экспорта и печати PrintingScope

Варианты выбора печатаемого диапазона страниц представлены в Таблице 30.

Таблица 30 – Диапазон страниц для экспорта и печати

Наименование константы	Описание
DocumentAPI.PrintingScope_PrintArea	Выбранная область печати (по умолчанию).
DocumentAPI.PrintingScope_WholeSheet	Печать всего документа.

3.30 Масштабирование при печати табличных документов

В Таблице 31 представлены варианты масштабирования при печати табличных документов.

Таблица 31 – Варианты масштабирования при печати табличных документов

Наименование константы	Описание
DocumentAPI.WorksheetPrinterFitType_ActualSize	Фактический размер.
DocumentAPI.WorksheetPrinterFitType_ByPageScale	По масштабу страницы.
DocumentAPI.WorksheetPrinterFitType_ByPageBreaksOnly	По разрыву страниц.
DocumentAPI.WorksheetPrinterFitType_FitToPage	Вписать в страницу.
DocumentAPI.WorksheetPrinterFitType_FitToWidth	Вписать по ширине.
DocumentAPI.WorksheetPrinterFitType_FitToHeight	Вписать по высоте.

3.31 Таблица DocumentAPI.Document

Таблица `DocumentAPI.Document` предоставляет доступ к содержимому открытого текстового или табличного документа.

Доступ к содержимому документа реализуется через таблицу `document`. Например, данный пример

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

предоставляет доступ к первому абзацу текстового документа.

3.31.1 Метод Document:saveAs

Метод `Document.saveAs` сохраняет документ в файл по указанному пути. Формат и тип документа определяются расширением файла, если не указаны в явном виде.

При необходимости, в качестве второго аргумента можно использовать таблицу DocumentAPI.SaveDocumentSettings, которая содержит формат документа DocumentFormat, тип документа DocumentType и пароль для защиты документа от несанкционированного доступа.

Пример:

```
function CheckCtx.saveAs(context)
    context.doWithDocument(function(document)
        -- Сохраняет открытый документ как файл на диске.
        document:saveAs( filePath )
    end)
end
```

3.31.2 Метод Document:exportAs

Метод `Document.exportAs` экспортирует документ в файл по указанному пути и с указанным форматом.

Расширенные версии метода позволяют указать дополнительные настройки экспорта документа:

- для текстовых документов – таблица DocumentAPI.TextExportSettings;
- для табличных документов – DocumentAPI.WorkbookExportSettings.

В настоящее время поддерживается только операция экспорта документа в формат PDF/A-1b.

Пример:

```
function CheckCtx.exportAs(context)
    context.doWithDocument(function(document)
        -- Сохраняет открытый документ как файл формата PDF на диске
        document:exportAs( filePath, DocumentAPI.ExportFormat_PDFA1 )
    end)
end
```

3.31.3 Метод Document:merge

Метод `Document:merge` возвращает документ, в котором различия отмечены отслеживаемыми изменениями. В качестве параметра метод принимает объект DocumentAPI.Document, который сравнивается с текущим.

Метод возвращает объект DocumentAPI.Document, содержащий результат сравнения.

Пример:

```
function CheckCtx.exportAs(context)
    context.doWithDocument(function(document)
        -- Сохраняет открытый документ как файл формата PDF на диске
        local docResult = document:merge(anotherDoc)
    end)
end
```

3.31.4 Метод document:getBlocks

Метод предоставляет доступ к метатаблице `Blocks` и далее к отдельным фрагментам (абзацам, таблицам и т. д.), из которых составлен документ.

Пример:

```
local blocks = document:getBlocks()
```

3.31.5 Метод document:getBookmarks

Метод предоставляет доступ к таблице закладок `Bookmarks`.

Пример:

```
local bookmarks = document:getBookmarks()
```

3.31.6 Метод document:getScripts

Метод предоставляет доступ к таблице макрокоманд `Scripts`, содержащихся в документе.

Пример:

```
local scripts = document:getScripts()
```

3.31.7 Метод document:getRange

Метод предоставляет доступ ко всему документу как области данных.

Пример:

```
local range = document:getRange()
print(range:extractText())
```

3.31.8 Метод document:isChangesTrackingEnabled

Метод возвращает состояние включения отслеживания изменений в документе.

Пример:

```
local trackingChanges = "Disabled"
if document:isChangesTrackingEnabled() then
```

```
   trackingChanges = "Enabled"
end
```

3.31.9 Метод document:setChangesTrackingEnabled

Метод управляет состоянием включения отслеживания изменений в документе.

Пример:

```
if trackingChanges == "Disabled" then
  document:setChangesTrackingEnabled(true)
  if document:isChangesTrackingEnabled() then
     trackingChanges = "Enabled"
  end
end
```

3.31.10 Метод document:getComments

Метод обеспечивает доступ к комментариям, которые хранятся в документе.

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
```

3.31.11 Метод document:setPageProperties

Метод устанавливает свойство `PageProperties` (ширину и высоту страниц) в документе.

Пример:

```
local properties = DocumentAPI.PageProperties()
properties.width = 100
properties.height = 200
document:setPageProperties(properties)
```

3.31.12 Метод document:setPageOrientation

Метод устанавливает альбомную, либо книжную ориентацию страниц в документе (см. PageOrientation).

Пример:

```
document:setPageOrientation(DocumentAPI.PageOrientation_Landscape)
```

3.31.13 Метод document:enumerateSections

Возвращает таблицу объектов типа Sections.

Пример:

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

3.31.14 Метод document:setMirroredMarginsEnabled

Метод позволяет включать/отключать зеркальные поля в документе.

Пример:

```
document:setMirroredMarginsEnabled(true)
print(document:areMirroredMarginsEnabled())
```

3.31.15 Метод document:areMirroredMarginsEnabled

Возвращает состояние режима зеркальных полей в документе.

Пример:

```
document:setMirroredMarginsEnabled(true)
print(document:areMirroredMarginsEnabled())
```

3.31.16 Метод Document:getPivotTablesManager

Возвращает объект PivotTablesManager, который используется для создания сводных таблиц. Метод может быть использован только в табличном редакторе.

Пример:

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

3.31.17 Метод Document:getNamedExpressions

Используется для получения списка именованных выражений NamedExpressions.

3.32 Таблица DocumentAPI.Block

Таблица `DocumentAPI.Block` является базовой для всех блоков документа.

3.32.1 Методы toParagraph, toTable, toShape, toField

Преобразует объект `Block` в объект соответствующего типа.

Пример:

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

3.32.2 Метод Block.getRange

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

Пример:

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

3.32.3 Метод Block.remove

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

Пример:

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

3.32.4 Метод Block.getSection

Метод возвращает раздел `Section`, содержащий блок.

Пример:

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

3.33 Таблица DocumentAPI.Blocks

3.33.1 Метод Blocks:getBlock

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

Пример:

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

3.33.2 Метод Blocks:getParagraph

Возвращает `Paragraph` с указанным индексом. Нумерация индексов абзаца начинается с нуля.

Пример:

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

3.33.3 Метод Blocks:getTable

Для табличного документа возвращает лист (`worksheet`) с указанным номером. Нумерация листов начинается с нуля.

Для текстового документа возвращает таблицу с указанным порядковым номером. Нумерация таблиц 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")
```

3.33.4 Метод Blocks:getShape

Возвращает фигуру `Shape` с указанным индексом.

Пример:

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

3.33.5 Метод Blocks:getField

Возвращает объект типа `Field` с указанным индексом.

Пример:

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

3.33.6 Метод Blocks:enumerate

Возвращает таблицу объектов типа `Block`.

Пример:

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

3.33.7 Метод Blocks:enumerateParagraphs

Возвращает таблицу объектов типа Paragraph (абзац).

Пример:

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

3.33.8 Метод Blocks:enumerateTables

Возвращает таблицу объектов типа Table.

Пример:

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

3.33.9 Метод Blocks:enumerateShapes

Возвращает таблицу объектов типа Shape.

Пример:

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

3.33.10 Метод Blocks:enumerateFields

Возвращает таблицу объектов типа Field.

Пример:

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

3.34 Таблица DocumentAPI.Paragraph

Таблица `DocumentAPI.Paragraph` предоставляет доступ к свойствам абзаца.

3.34.1 Метод Paragraph:getParagraphProperties

Метод предоставляет доступ к таблице свойств форматирования абзаца 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
```

3.34.2 Метод Paragraph:setParagraphProperties

Метод предназначен для обновления таблицы свойств форматирования абзаца 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
```

3.34.3 Метод Paragraph:getListSchema

Метод возвращает схему форматирования абзаца ListSchema либо значение nil,

если схема нумерации не установлена для абзаца.

Пример:

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

3.34.4 Метод Paragraph:setListSchema

Метод позволяет установить тип маркированного или нумерованного списка ListSchema.

Пример:

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

3.34.5 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()
```

3.34.6 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)
```

3.34.7 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()
```

3.34.8 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()
```

3.35 Таблица DocumentAPI.Paragraphs

Таблица `DocumentAPI.Paragraphs` представляет предоставляет доступ к коллекции абзацев.

3.35.1 Метод Paragraphs:setListSchema

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

Пример:

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

3.35.2 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)
```

3.35.3 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()
```

3.35.4 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()
```

3.35.5 Метод Paragraphs:enumerate

Метод возвращает 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
```

3.36 Таблица DocumentAPI.Field

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

3.37 Таблица DocumentAPI.Fill

Таблица описывает свойства заполнения фигуры, ячейки и т.д.

3.37.1 Метод Fill:getColor

Метод возвращает цвет заполнения `Color`.

3.37.2 Метод Fill:getUrl

Метод возвращает путь к изображению, которое используется в качестве заполнения.

3.37.3 Метод Fill:isNoFill

Метод возвращает свойство заполнения.

3.38 Таблица DocumentAPI.Shape

Таблица `Shape` представляет собой фигуру, содержит методы для установки и получения свойств `ShapeProperties`.

3.38.1 Метод Shape:getShapeProperties

Метод возвращает свойства фигуры `ShapeProperties`.

Пример:

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

3.38.2 Метод Shape:setShapeProperties

Метод устанавливает свойства фигуры `ShapeProperties`.

Пример:

```
local shape = document:getBlocks():getShape(0)
local shape_properties = shape:getShapeProperties()
shape_properties.verticalAlignment = DocumentAPI.VerticalAlignment_Center
shape:setShapeProperties(shape_properties)
```

3.39 Таблица DocumentAPI.ShapeProperties

Таблица описывает свойства фигуры и содержит следующие поля: `verticalAlignment` - вертикальное выравнивание, `borderProperties` – свойства границ фигуры, `fill` – свойства заполнения фигуры, `shapeTextLayout` – свойства текста внутри фигуры.

3.39.1 Поле ShapeProperties:verticalAlignment

Поле предназначено для установки типа вертикального выравнивания `VerticalAlignment`.

3.39.2 Поле ShapeProperties:borderProperties

Поле предназначено для установки свойств границ фигуры `LineProperties`.

3.39.3 Поле ShapeProperties:fill

Поле предназначено для установки свойств заполнения фигуры `Fill`.

3.39.4 Поле ShapeProperties:shapeTextLayout

Поле предназначено для установки свойств текста внутри фигуры `ShapeTextLayout`.

3.40 Таблица DocumentAPI.ShapeTextLayout

Таблица `DocumentAPI.ShapeTextLayout` описывает свойства текста, находящегося внутри фигуры.

Описание полей представлено в Таблице 32.

Таблица 32 – 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.

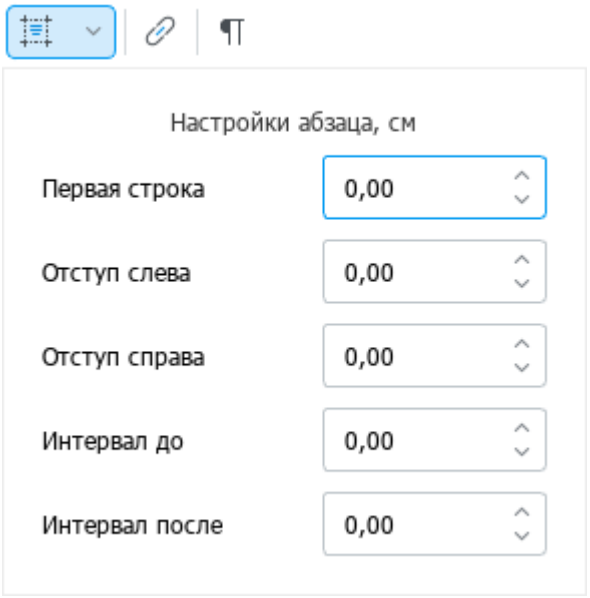
3.41 Таблица DocumentAPI.ParagraphProperties

Таблица `DocumentAPI.ParagraphProperties` предназначена для управления свойствами форматирования абзаца.

Описание полей таблицы `DocumentAPI.ParagraphProperties` представлено в Таблице 33.

Таблица 33 – Описание полей таблицы DocumentAPI.ParagraphProperties

Поле	Тип	Описание
`ParagraphProperties.afterSpacing`	Number	Установка величины расстояния после абзаца. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца**, в поле **Интервал после** (подробнее см. в документе

Поле	Тип	Описание
		«МойОфис Стандартный. МойОфис Текст. Руководство пользователя»). 
`ParagraphProperties.beforeSpacing`	Number	Установка величины расстояния до абзаца. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца,** (см. рисунок выше), в поле **Интервал до** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).
`ParagraphProperties.alignment`	`Alignment`	Выравнивание текстового фрагмента по горизонтали.
`ParagraphProperties.firstLineIndent`	Number	Расстояние от левого поля документа до первой строки в абзаце с учетом отступа слева. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца**, (см. рисунок выше), в поле **Первая строка** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).
`ParagraphProperties.leftIndent`	Number	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 документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).

Поле	Тип	Описание
ParagraphProperties.lineSpacing	LineSpacingRule	Расстояние между строк одного абзаца (межстрочный интервал).
ParagraphProperties.rightIndent	Number	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 документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).

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
```

3.42 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 можно использовать строку вида «A1» в качестве параметра метода `getCell`.

Примеры:

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

```
local table = document:getBlocks():getTable( 0 ) -- первый лист книги
local cell = table:getCell ( "A3" ) -- ячейка A3
```

3.42.1 Поле CellPosition.column

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

3.42.2 Поле CellPosition.row

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

3.42.3 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)
```

3.43 Таблица DocumentAPI.ColorRGBA

Таблица `DocumentAPI.ColorRGBA` предназначена для настройки цвета отображения 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 34.

Таблица 34 – 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))
```

3.43.1 Метод ColorRGBA:__eq

Метод используется для определения эквивалентности двух объектов `ColorRGBA`.

3.43.2 Метод ColorRGBA:__ne

Метод используется для определения неэквивалентности двух объектов `ColorRGBA`.

3.44 Таблица DocumentAPI.TextOrientation

Таблица `DocumentAPI.TextOrientation` предоставляет доступ к свойствам ориентации текста в ячейке, фигуре и т. д.

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())
```

3.44.1 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 props = cell:getCellProperties()
print(props.textOrientation:getAngle())
```

3.45 Таблица DocumentAPI.CellProperties

Таблица `DocumentAPI.CellProperties` предназначена для выравнивания содержимого в ячейках таблицы. Описание полей таблицы `DocumentAPI.CellProperties` представлено в Таблице 35.

Таблица 35 – Описание полей таблицы DocumentAPI.CellProperties

Поле	Тип	Значение
`CellProperties.verticalAlignment`	`VerticalAlignment`	Настройка вертикального выравнивания значения ячейки.
`CellProperties.backgroundColor`	`ColorRGBA`	Настройка цвета фона ячейки.
`CellProperties.textLayout`	`TextLayout`	Настройка вариантов отображения значения ячейки.
`CellProperties.textOrientation`	`TextOrientation`	Настройка свойств ориентации текста в ячейке.

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(255,255,0,1)
props.textOrientation = DocumentAPI.TextOrientation(45)
--
cell:setCellProperties(props)
```

3.46 Таблица DocumentAPI.LineProperties

Таблица `DocumentAPI.LineProperties` предназначена для установки параметров линии, таких как тип линии, ее ширина или цвет.

Пример:

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

```
line_prop = DocumentAPI.LineProperties()
line_prop.style = DocumentAPI.LineStyle_Solid
line_prop.width = 1.5
line_prop.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(55, 146, 179, 200))
--
rb = DocumentAPI.Borders()
rb = rb:setTop(line_prop)
local brds = cell:setBorders(rb)
```

3.46.1 Поле LineProperties.style

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

3.46.2 Поле LineProperties.width

Поле предназначено для установки ширины линии.

3.46.3 Поле LineProperties.color

Поле предназначено для установки цвета линии.

3.46.4 Поле LineProperties.headLineEndingProperties

Поле предназначено для оформления начала линии `LineEndingProperties`.

3.46.5 Поле LineProperties.tailLineEndingProperties

Поле предназначено для оформления конца линии `LineEndingProperties`.

3.47 Таблица DocumentAPI.Borders

Таблица `DocumentAPI.Borders` предназначена для оформления границ отдельной ячейки таблицы (см. Таблица 36). Параметры линии, такие как тип линии, ее ширина и цвет, задаются с помощью таблицы `DocumentAPI.LineProperties`.

Таблица 36 – 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.

Метод	Описание
Borders:setDiagonalUp	Установка диагональной линии.
Borders:setOuter	Установка внешних границ ячейки.
Borders:setDiagonals	Установка обеих 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.
Borders:getInnerVertical	Получение внутренних вертикальных границ ячейки.

Пример для табличного документа:

```
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))
--
rb = DocumentAPI.Borders()
rb = rb:setLeft(line_prop)
rb = rb:setRight(line_prop)
rb = rb:setTop(line_prop)
rb = rb:setBottom(line_prop)
--
local brds = cell:setBorders(rb)
```

3.48 Таблица DocumentAPI.RangeBorders

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

3.49 Таблица DocumentAPI.CellRange

Таблица `DocumentAPI.CellRange` предоставляет доступ к указанному диапазону ячеек таблицы.

3.49.1 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
```

3.49.2 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
```

3.49.3 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
```

3.49.4 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
```

3.49.5 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
```

3.49.6 Метод CellRange:setBorders

Метод предназначен для установки границ диапазона ячеек. Отдельные границы устанавливаются с помощью методов таблицы 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))
--
rb = DocumentAPI.Borders()
rb = rb:setLeft(line_prop)
rb = rb:setRight(line_prop)
rb = rb:setTop(line_prop)
rb = rb:setBottom(line_prop)
--
local brds = cell:setBorders(rb)
```

3.49.7 Метод CellRange:getCellProperties

Метод возвращает набор свойств форматирования (CellProperties) для диапазона. 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)
```

3.49.8 Метод CellRange:setCellProperties

Метод предназначен для установки свойств 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)
```

3.49.9 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()
```

3.50 Таблица DocumentAPI.CellRangePosition

Таблица `DocumentAPI.CellRangePosition` представляет положение диапазона ячеек в таблице. 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 37.

Таблица 37 – Поля таблицы DocumentAPI.CellRangePosition

Поле	Тип	Описание
DocumentAPI.CellRangePosition_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.
DocumentAPI.CellRangePosition_bottomRight	CellPosition	Содержит позицию правой нижней ячейки таблицы прямоугольного диапазона.

Пример:

```
local cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
```

3.50.1 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)]
```

3.50.2 Метод CellRangePosition:__eq

Метод используется для определения эквивалентности двух объектов `CellRangePosition`.

3.50.3 Метод CellRangePosition:__ne

Метод используется для определения неэквивалентности двух объектов `CellRangePosition`.

3.51 Таблица DocumentAPI.TextProperties

Таблица `DocumentAPI.TextProperties` предназначена для форматирования текста. Описание полей таблицы `DocumentAPI.TextProperties` представлено в Таблице 38.

Таблица 38 – Описание полей таблицы DocumentAPI.TextProperties

Поле	Значение	Описание
`TextProperties.fontName`	String	Наименование шрифта, использованного для оформления фрагмента документа.
`TextProperties.fontSize`	Number	Размер шрифта, использованного для оформления фрагмента документа.
`TextProperties.bold`	Boolean	Значение `true` устанавливает жирное начертание для указанного фрагмента текста.
`TextProperties.italic`	Boolean	Значение `true` устанавливает начертание курсивом для указанного фрагмента текста.
`TextProperties.underline`	Boolean	Значение `true` устанавливает подчеркивание для указанного фрагмента текста.
`TextProperties.strikethrough`	Boolean	Значение `true` устанавливает начертание «зачеркнутый» для указанного фрагмента текста.
`TextProperties.allCapitals`	Boolean	Значение `true` устанавливает все буквы указанного фрагмента текста как прописные. Значение `false` устанавливает все буквы указанного фрагмента текста как строчные.
`TextProperties.scriptPosition`	ScriptPosition	Устанавливает отображение символа в виде надстрочного, подстрочного знака или в нормальном режиме.
`TextProperties.textColor`	Color	Цвет указанного фрагмента документа.
`TextProperties.backgroundColor`	ColorRGBA	Цвет фона указанного фрагмента документа.
`TextProperties.characterSpacing`	Number	Размер межсимвольного интервала.

Пример:

```
local props = DocumentAPI.TextProperties()
props.fontName = "XO Oriel"
props.fontSize = 20

-- доступ к тексту третьего абзаца
local text = document:getBlocks():getParagraph(2):getRange()
```

```
-- установить свойства фрагмента текста
text:setTextProperties(props)
```

3.52 Таблица DocumentAPI.Range

Таблица `DocumentAPI.Range` предоставляет доступ к указанному фрагменту текстового документа.

3.52.1 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")
```

3.52.2 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")
```

3.52.3 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())
```

3.52.4 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())
```

3.52.5 Метод Range:lockContent

Метод запрещает изменения содержимого диапазона.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:lockContent()
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:lockContent()
```

3.52.6 Метод Range:unlockContent

Метод разрешает изменения содержимого диапазона.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:unlockContent()
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:unlockContent()
```

3.52.7 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
```

3.52.8 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")
```

3.52.9 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)
```

3.52.10 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) -- применение курсива к фрагменту текста
```

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)
```

3.52.11 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
```

3.52.12 Метод Range:enumerateTrackedChanges

Предоставляет возможность итерации по отслеживаемым изменениям. Метод может быть использован только в текстовых документах.

Пример:

```
local tracked_changes = document:getRange():enumerateTrackedChanges()
```

3.52.13 Метод Range:getComments

Обеспечивает доступ к комментариям в диапазоне.

Комментарии, примененные к одному и тому же диапазону, упорядочиваются по датам, если таковые имеются. 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
```

3.52.14 Метод Range:getParagraphs

Обеспечивает доступ к абзацам 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
```

3.52.15 Метод Range:getImages

Обеспечивает доступ к изображениям (Image) в диапазоне.

Пример:

```
local images = document:getRange():getImages()
for image in images:enumerate() do
  local frame = image:getFrame()
  print(frame:getWrapType())
end
```

3.52.16 Метод Range:getInlineObjects

Обеспечивает доступ к встроенным фигурам `MediaObject` в диапазоне.

Пример:

```
local inlineObjects = document:getRange():getInlineObjects()
for inlineObject in inlineObjects:enumerate() do
  local frame = inlineObject:getFrame()
  print(frame:getWrapType())
end
```

3.53 Таблица DocumentAPI.Table

Таблица `DocumentAPI.Table` предоставляет доступ к листу электронной таблицы или отдельной таблице в составе текстового документа.

3.53.1 Метод Table:setName

Метод устанавливает наименование листа электронной таблицы. 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" )
```

3.53.2 Метод Table:getName

Метод позволяет получить наименование листа табличного документа.

Пример:

Напечатать наименование первого листа табличного документа. Нумерация листов начинается с нуля.

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

3.53.3 Метод Table:getRowsCount

Метод позволяет получить количество строк на листе табличного документа.

Пример:

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

3.53.4 Метод Table:getColumnsCount

Метод позволяет получить количество столбцов на листе табличного документа.

Пример:

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

3.53.5 Метод Table:getCell

Метод позволяет получить доступ к управлению отдельной ячейкой таблицы.

Пример:

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

3.53.6 Метод Table:getCellRange

Метод позволяет получить доступ к диапазону ячеек таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("A1:C4")
for c in rng:enumerate() do
   print(c:getFormattedValue())
end
```

3.53.7 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 – единица.

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)
```

3.53.8 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 – единица.

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)
```

3.53.9 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 – единица.

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)
```

3.53.10 Метод Table:insertRowBefore

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

Вызов:

```
insertRowBefore( rowIndex, copyRowStyle, rowsCount )
```

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`.
- `rowsCount` – количество вставляемых строк. Значение по умолчанию – единица.

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)
```

3.53.11 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 – единица.

3.53.12 Метод Table:removeRow

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

Вызов:

```
removeRow(rowIndex, rowsCount)
```

Параметры:

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

3.53.13 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)
```

3.53.14 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)
```

3.53.15 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()
```

3.53.16 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)
```

3.53.17 Метод Table:setVisible

Метод управляет видимостью листа таблицы в табличном документе.

Вызов:

```
setVisible(visible)
```

Параметры:

`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)
```

3.53.18 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
print(tbl:isVisible())
```

3.53.19 Метод Table:__eq

Метод используется для определения эквивалентности двух объектов `Table`.

Пример:

```
local tbl1 = document:getBlocks():getTable(0)
local tbl2 = document:getBlocks():getTable("Table Name")
if tbl1 ~= nil and tbl2 ~= nil and tbl1 == tbl2 then
    print("tbl1 и tbl2 ссылаются на общую таблицу в документе")
end
```

3.53.20 Метод Table:__ne

Метод используется для определения неэквивалентности двух объектов `Table`.

Пример:

```
local tbl1 = document:getBlocks():getTable(0)
local tbl2 = document:getBlocks():getTable("Table Name")
if tbl1 ~= nil and tbl2 ~= nil and tbl1 ~= tbl2 then
    print("tbl1 и tbl2 ссылаются на разные таблицы в документе")
end
```

3.53.21 Метод Table:setPrintArea

Метод служит для установки и сброса области печати CellRangePosition.

Пример:

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

3.53.22 Метод Table:setPrintAreas

Метод `Table:setPrintAreas` задает множественные области печати / экспорта `CellRangePositions`.

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())
```

3.53.23 Метод Table:getPrintAreas

Метод `Table:getPrintAreas` возвращает коллекцию `CellRangePositions` текущих областей печати.

Пример:

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

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

3.53.24 Метод Table:getCharts

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

Пример:

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

3.53.25 Метод Table:getNamedExpressions

Для получения списка именованных выражений `NamedExpressions` используется метод `Table:getNamedExpressions`.

3.53.26 Объединение и разделение ячеек таблицы

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

Для объединения нескольких ячеек используйте метод `merge`, принадлежащий таблице `CellRange`.

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, содержавшиеся в объединенной ячейке, будут помещены в верхнюю левую ячейку диапазона.

Для разъединения ячеек используйте метод `unmerge`, принадлежащий таблице `Cell`.

Пример:

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

3.53.27 Управление видимостью строк / колонок

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

```
setColumnsVisible(first, columnsCount, visible)
```

Метод `setRowsVisible` позволяет задавать видимость `rowsCount` строк, начиная с индекса `first`. Индексация строк начинается с нуля.

```
setRowsVisible(first, rowsCount, visible)
```

3.53.28 Группировка строки 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 `setColumnsVisible` и `setRowsVisible` чтобы раскрывать и закрывать фрагменты иерархии групп.

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

Методы для группировки:

```
groupRows(first, rowsCount)
ungroupRows(first, rowsCount)
clearRowGroups(first, rowsCount)
groupColumns(first, columnsCount)
ungroupColumns(first, columnsCount)
clearColumnnGroups(first, columnsCount)
```

3.54 Таблица DocumentAPI.Cell

Таблица `DocumentAPI.Cell` предоставляет доступ к ячейке на листе табличного документа.

3.54.1 Метод Cell:getRange

Метод возвращает объект `Range` для управления содержимым ячейки.

3.54.2 Метод Cell:setBorders

Метод предназначен для установки границ ячейки. См. описание метатаблицы `DocumentAPI.Borders`.

3.54.3 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
```

3.54.4 Метод Cell:getFormat

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

3.54.5 Метод Cell:setFormat

Метод устанавливает формат ячейки. Список поддерживаемых форматов см. в разделе `CellFormat`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:getCell("A1"):setNumber(2.3) -- Формат: Общий, отображение 2,3
-- set format
-- Формат: Экспоненциальный, отображение 2,3E+00
tbl:getCell("A1"):setFormat(DocumentAPI.CellFormat_Scientific)
```

3.54.6 Метод Cell:getFormattedValue

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

Пример:

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

3.54.7 Метод Cell:setFormattedValue

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

Список поддерживаемых форматов см. в разделе CellFormat.

3.54.8 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()
```

3.54.9 Метод Cell:setContent

Определяет и устанавливает соответствующую формулу/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")
```

3.54.10 Метод Cell:getBorders

Получает границы ячейки.

Пример:

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

3.54.11 Метод Cell:getRawValue

Возвращает значение ячейки в формате «Общий» (`DocumentAPI.CellFormat_General`).

Пример:

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

3.54.12 Метод Cell:getCustomFormat

Возвращает строку формата ячейки.

Пример:

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

3.54.13 Метод Cell:setCustomFormat

Устанавливает формат ячейки.

Пример:

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

3.54.14 Метод Cell:setBool

Устанавливает для ячейки значение логического типа.

Пример:

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

3.54.15 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)
```

3.54.16 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)
```

3.54.17 Метод Cell:getFormulaAsString

Позволяет получить текст формулы в данной ячейке.

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

Пример:

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

3.54.18 Метод Cell:getCellProperties

Позволяет получить оформление ячейки (например, фон, вертикальное выравнивание и т. д.).

Пример:

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

3.54.19 Метод Cell:setCellProperties

Позволяет установить оформление ячейки (например, фон, вертикальное выравнивание и т. д.).

Пример:

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

3.54.20 Метод Cell:getParagraphProperties

Возвращает свойства абзаца `ParagraphProperties`, содержащего в ячейке.

3.54.21 Метод Cell:setParagraphProperties

Устанавливает свойства абзаца `ParagraphProperties`, содержащего в ячейке.

3.55 Таблица DocumentAPI.CurrencySignPlacement

Типы вариантов размещения знака валюты представлены в Таблице 39.

Таблица 39 – Описание полей таблицы DocumentAPI.CurrencySignPlacement

Поле	Описание	Пример
`DocumentAPI.CurrencySignPlacement_Prefix`	Размещение знака валюты до значения.	$12.00
`DocumentAPI.AccountingCellFormatting_Suffix`	Размещение знака валюты после значения.	12,00 ₽

3.56 Таблица DocumentAPI.AccountingCellFormatting

Таблица содержит параметры для финансового формата ячеек таблицы.

Описание полей таблицы `DocumentAPI.AccountingCellFormatting` представлено в Таблице 40.

Таблица 40 – Описание полей таблицы DocumentAPI.AccountingCellFormatting

Поле	Описание
`DocumentAPI.AccountingCellFormatting.decimalPlaces`	Количество десятичных позиций.

Поле	Описание
DocumentAPI.AccountingCellFormatting.symbol	Символ денежной единицы.
DocumentAPI.AccountingCellFormatting.localeCode	Идентификатор кода языка (MS-LCID).
DocumentAPI.AccountingCellFormatting.fillSymbol	Символ заполнения.
DocumentAPI.AccountingCellFormatting.useThousandsSeparator	Использовать разделитель для тысячных.
DocumentAPI.AccountingCellFormatting.currencySignPlacement	Тип размещения знака валюты CurrencySignPlacement.

3.57 Таблица DocumentAPI.PercentageCellFormatting

Содержит параметр для процентного формата ячеек таблицы.

Описание полей таблицы DocumentAPI.PercentageCellFormatting представлено в Таблице 41.

Таблица 41 – Описание полей таблицы DocumentAPI.PercentageCellFormatting

Поле	Описание
DocumentAPI.PercentageCellFormatting.decimalPlaces	Количество десятичных позиций.

3.58 Таблица DocumentAPI.NumberCellFormatting

Таблица содержит параметры для числового формата ячеек таблицы.

Описание полей таблицы DocumentAPI.NumberCellFormatting представлено в Таблица 42.

Таблица 42 – 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.useBracketsForNagative	Использовать скобки для отрицательных значений.

Поле	Описание
DocumentAPI.NumberCellFormatting.hideSign	Скрывать знак «минус» для отрицательных значений.

3.59 Таблица DocumentAPI.CurrencyCellFormatting

Таблица содержит параметры для денежного формата ячеек таблицы.

Описание полей таблицы DocumentAPI.CurrencyCellFormatting представлено в Таблице 43.

Таблица 43 – 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.useSign	Скрывать знак «минус» для отрицательных значений.

3.60 Таблица DocumentAPI.DateTimeCellFormatting

Таблица содержит параметры для формата ячеек таблицы типа Дата и Время.

Описание полей таблицы DocumentAPI.DateTimeCellFormatting представлено в Таблице 44.

Таблица 44 – Описание полей таблицы DocumentAPI.DateTimeCellFormatting

Поле	Описание
DocumentAPI.DateTimeCellFormatting.dateListId	Формат даты DatePatterns.
DocumentAPI.DateTimeCellFormatting.TimeListId	Формат времени TimePatterns.

3.61 Таблица DocumentAPI.FractionCellFormatting

Таблица содержит параметры для дробного формата ячеек таблицы.

Описание полей таблицы `DocumentAPI.FractionCellFormatting` представлено в Таблице 45.

Таблица 45 – Описание полей таблицы DocumentAPI.FractionCellFormatting

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

3.62 Таблица DocumentAPI.ScientificCellFormatting

Таблица содержит параметры для экспоненциального формата ячеек таблицы.

Описание полей таблицы `DocumentAPI.ScientificCellFormatting` представлено в Таблице 46.

Таблица 46 – Описание полей таблицы DocumentAPI.ScientificCellFormatting

Поле	Описание
`DocumentAPI.ScientificCellFormatting.decimalPlaces`	Количество десятичных позиций.
`DocumentAPI.ScientificCellFormatting.minExponentDigits`	Минимальное количество позиций экспоненты.

3.63 Таблица DocumentAPI.Bookmarks

Предоставляет доступ к операциям с закладками (`bookmarks`) в документе. Доступны следующие операции с закладками:

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

- удаление содержимого закладки.

3.63.1 Метод Bookmarks:getBookmarkRange

Возвращает объект `Range` для работы с содержимым закладки (`bookmark`) с заданным именем в документе.

Пример:

```
-- В тексте документа происходит замещение содержимого закладки на текст "Lua"
local bms = document:getBookmarks()
local bm_1 = bms:getBookmarkRange( "bm_1" )
bm_1.replaceText("Lua")
```

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")
```

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)
```

Замена текстового содержимого закладки

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

Вставка таблицы в закладку

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

3.64 Таблица DocumentAPI.Script

Таблица `DocumentAPI.Script` предназначена для управления отдельной макрокомандой.

3.64.1 Метод Script:getName

Метод возвращает имя макрокоманды.

Пример:

```
local scripts = document:getScripts()
local sc = scripts:getScript("get_app")
print(sc:getName()) -- get_app
```

3.64.2 Метод Script:setName

Метод устанавливает имя для макрокоманды.

Пример:

```
local scripts = document:getScripts()
local sc = scripts:getScript("get_app")
sc:setName("new_get_app")
```

3.64.3 Метод Script:getBody

Метод возвращает текст макрокоманды в виде строки.

Пример:

```
local scripts = document:getScripts()
local sc = scripts:getScript("get_app")
local code = sc:getBody()
```

3.64.4 Метод Script:setBody

Метод устанавливает текст макрокоманды, полностью заменяя уже имеющийся текст.

Пример:

```
local scripts = document:getScripts()
local sc = scripts:getScript("get_app")
sc:setBody( "local scripts = Application.document:getScripts()\nlocal sc =
scripts:getScript(\"new_mc\")\nprint(sc:getName())\nprint(sc:getBody())")
```

3.65 Таблица DocumentAPI.Scripts

Таблица `DocumentAPI.Scripts` предоставляет доступ к операциям управления макрокомандами.

3.65.1 Метод Scripts:getScript

Метод возвращает таблицу `Script` для управления отдельной макрокомандой.

Пример:

```
-- Макрокоманда get_app
local scripts = document:getScripts()
local sc = scripts:getScript("get_app")
print(sc:getName()) --get_app
print(sc:getBody())  --local doc = document:getRange()
```

3.65.2 Метод Scripts:setScript

Метод добавляет макрокоманду `script_name` с кодом `script_code` в текущий документ. Сохранение макрокоманд для последующего использования возможно только при работе с документами форматов DOCX, XODT, ODT, XLSX, XODS, ODS.

Пример:

```
-- Создание макрокоманды new_mc
local scripts = document:getScripts()
--
local script_name = "new_mc"
--
local script_code = "local scripts = document:getScripts()\nlocal sc =
scripts:getScript(\"new_mc\")\nprint(sc:getName())\nprint(sc:getBody())"
--
scripts:setScript( script_name, script_code)
```

3.65.3 Метод Scripts:removeScript

Метод удаляет макрокоманду с именем `script_name` из текущего документа.

Пример:

```
-- Удаление макрокоманды new_mc
local scripts = document:getScripts()
scripts:removeScript( "new_mc" )
```

3.65.4 Метод Scripts:enumerate

Метод возвращает коллекцию макрокоманд.

Пример:

```
for script in document:getScripts():enumerate() do
  -- Действие с каждым сценарием
end
```

3.66 Таблица DocumentAPI.Search

Таблица `DocumentAPI.Search` предоставляет доступ к механизму поиска и замены фрагментов документа, открытого в редакторе текста или таблиц.

3.66.1 Метод DocumentAPI:createSearch

Метод инициализирует механизм поиска для текущего документа. Возвращает ссылку на таблицу `Search`, с помощью методов которой выполняются поисковые запросы.

Пример:

```
search = DocumentAPI.createSearch(document)
ranges = search:findText("English")
```

3.66.2 Метод Search:findText

Метод выполняет поиск строки `text` без учета регистра во всем документе. Результат возвращается в виде таблицы областей `Range`, содержащих искомый фрагмент.

Если строка `text` не обнаружена, возвращается пустая таблица.

Пример:

```
search = DocumentAPI.createSearch(document)
ranges = search:findText("English")
for occurrence in ranges do
    print(occurrence:extractText())
end
```

3.67 Таблица DocumentAPI.Position

Таблица `DocumentAPI.Position` представляет определенное местоположение в текстовом документе.

3.67.1 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")
```

3.67.2 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 = begin_pos:insertTable(3,3,"Table")
```

приведет к созданию в текстовом документе таблицы с именем «Table1».

Пример вставки таблицы в текстовый документ:

```
local rng = document:getRange()
local begin_pos = rng:getBegin()
t = begin_pos:insertTable(3,3,"Table")
```

Для табличного документа 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")
```

3.67.3 Метод Position:insertPageBreak

Метод предназначен для вставки разрыва страницы в указанную позицию в документе.

Пример:

```
local rng = document:getRange()
local end_pos = rng:getEnd()
end_pos:insertPageBreak()
```

3.67.4 Метод Position:insertLineBreak

Метод предназначен для вставки перевода строки.

Пример:

```
local rng = document:getRange()
local end_pos = rng:getEnd()
end_pos:insertLineBreak()
```

3.67.5 Метод Position:insertBookmark

Вставляет закладку с наименованием в текущую позицию.

Пример:

```
document:getRange():getEnd():insertBookmark('Bookmark example')
```

3.67.6 Метод Position:insertSectionBreak

Вставляет в позицию разрыв раздела.

Пример:

```
document:getRange():getBegin():insertSectionBreak()
```

3.67.7 Метод Position:insertImage

Вставляет рисунок, размещенный в файле, в текущую позицию.

Вызов:

```
insertImage( url, size )
```

Параметры:

– `url` – полный путь к файлу;

– `size` – геометрические размеры изображения для вставки.

Пример:

```
document:getRange():getBegin():insertImage("c:\\tmp\\warning.png",
DocumentAPI.SizeU(100, 100))
```

3.67.8 Метод Position:removeBackward

Метод удаляет `count` объектов (символов, картинок и т.д.) до текущей позиции.

Пример:

```
document:getRange():getEnd():removeBackward(3)
```

3.67.9 Метод Position:removeForward

Метод удаляет `count` объектов (символов, картинок и т.д.) после текущей позиции.

Пример:

```
document:getRange():getBegin():removeForward(3)
```

3.67.10 Метод Position:__eq

Метод используется для определения эквивалентности двух объектов `Position`.

3.67.11 Метод Position:__ne

Метод используется для определения неэквивалентности двух объектов `Position`.

3.68 Функция DocumentAPI.createScripting

Функция `DocumentAPI.createScripting` позволяет обеспечить выполнение макрокоманды, хранящейся в документе.

3.68.1 Метод createScripting:runScript

Метод предназначен для запуска макрокоманды, хранящейся в документе.

Пример:

```
scripting = DocumentAPI.createScripting(document)
scripting:runScript("getInlineObjects")
```

Где `OtherScriptName` – имя макрокоманды, хранящейся в документе. В данном примере (см. Рисунок 4) список макрокоманд содержит две макрокоманды, одна из которых запускает другую.

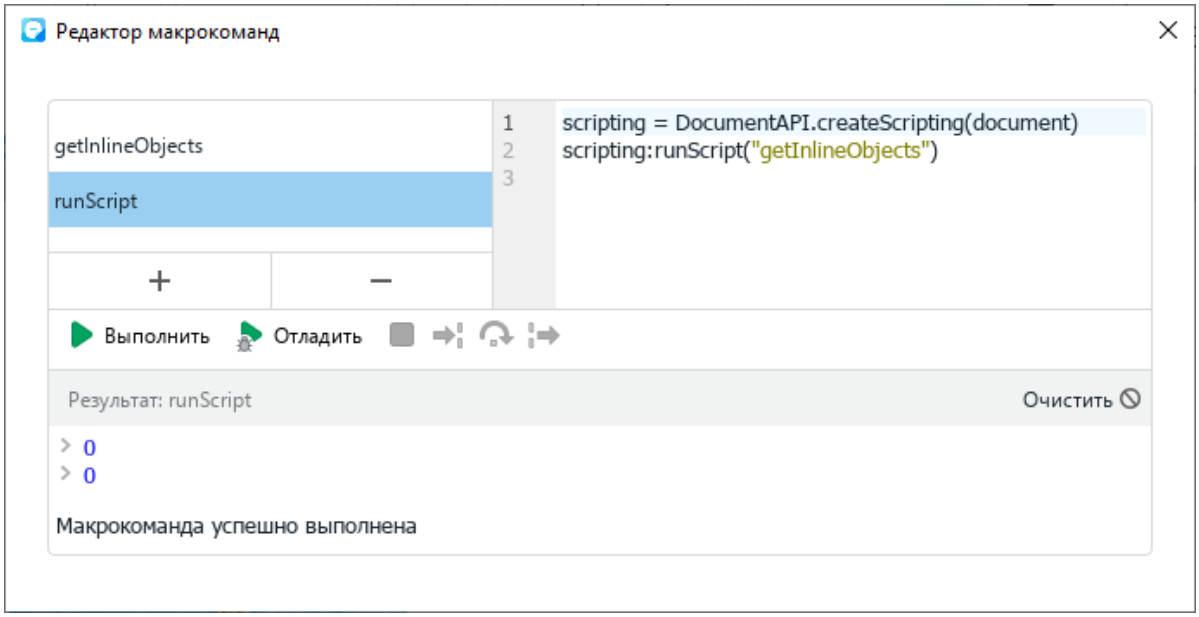


Рисунок 4 – Пример вызова метода `runScript`

3.69 Таблица DocumentAPI.Comment

Таблица `DocumentAPI.Comment` представляет доступ к свойствам примененного к диапазону комментария.

3.69.1 Метод Comment:getRange

Метод возвращает объект `Range`, соответствующего объекта `Comment`.

Пример:

```
first_comment_range = comment:getRange()
```

3.69.2 Метод Comment:getText

Метод возвращает текст комментария.

Пример:

```
first_comment_text = comment:getText()
```

3.69.3 Метод Comment:getAudioUrl

Метод возвращает путь к файлу аудиокомментария.

Пример:

```
first_comment_audio_url = comment:getAudopUrl()
```

3.69.4 Метод Comment:getInfo

Метод предоставляет доступ к отслеживаемой информации об изменении комментария `TrackedChangeInfo` (автор изменения, дата и т. д.).

Пример:

```
trackedChangeInfo = comment:getInfo()
```

3.69.5 Метод Comment:isResolved

Метод возвращает значение `true`, если комментарий решен.

3.69.6 Метод Comment:getReplies

Метод предоставляет доступ к ответам на комментарии `Comments`.

3.70 Таблица DocumentAPI.Comments

Таблица `DocumentAPI.Comments` представляет интерфейс для доступа к коллекции комментариев диапазона.

3.70.1 Метод Comments:enumerate

Метод возвращает коллекцию комментариев диапазона.

Пример:

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

3.71 Таблица DocumentAPI.TrackedChange

Таблица `DocumentAPI.TrackedChange` представляет интерфейс для отслеживания изменений в документе.

3.71.1 Метод TrackedChange:getRange

Метод возвращает объект `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
```

3.71.2 Метод TrackedChange:getType

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

Пример:

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

3.71.3 Метод TrackedChange:getInfo

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

Пример:

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

3.72 Таблица DocumentAPI.TrackedChangeInfo

Таблица `DocumentAPI.TrackedChangeInfo` содержит информацию об отслеживаемых изменениях.

Описание полей таблицы `DocumentAPI.TrackedChangeInfo` представлено в Таблице 47.

Таблица 47 – Описание полей таблицы DocumentAPI.TrackedChangeInfo

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

3.72.1 Метод TrackedChangeInfo:__eq

Метод используется для определения эквивалентности двух объектов `TrackedChangeInfo`.

3.72.2 Метод TrackedChangeInfo:__ne

Метод используется для определения неэквивалентности двух объектов `TrackedChangeInfo`.

3.73 Таблица DocumentAPI.DateTime

Таблица `DocumentAPI.DateTime` предоставляет дату и время с точностью до секунды. Описание полей таблицы `DocumentAPI.DateTime` представлено в Таблице 48.

Таблица 48 – 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

3.73.1 Метод DateTime:__eq

Метод используется для определения эквивалентности двух объектов `DateTime`.

3.73.2 Метод DateTime:__ne

Метод используется для определения неэквивалентности двух объектов `DateTime`.

3.74 Таблица DocumentAPI.Section

Таблица `DocumentAPI.Section` представляет собой раздел в документе.

3.74.1 Метод Section:setPageProperties

Метод устанавливает `PageProperties` (высоту и ширину страниц раздела).

Пример:

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

3.74.2 Метод Section:getPageProperties

Метод возвращает `PageProperties` для страниц раздела.

Пример:

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

3.74.3 Метод Section:setPageOrientation

Метод задает ориентацию `PageOrientation` страниц раздела.

Пример:

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

3.74.4 Метод Section:getPageOrientation

Метод возвращает ориентацию `PageOrientation` страниц раздела.

Пример:

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

3.74.5 Метод Section:getRange

Метод возвращает диапазон Range в документе, соответствующий данному разделу.

Пример:

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

3.74.6 Метод Section:getHeaders

Метод возвращает коллекцию 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
```

3.74.7 Метод Section:getFooters

Метод возвращает коллекцию 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
```

3.75 Таблица DocumentAPI.PageProperties

Таблица `DocumentAPI.PageProperties` предоставляет размеры страницы. Описание полей таблицы `DocumentAPI.PageProperties` представлено в Таблице 49.

Таблица 49 – Описание полей таблицы DocumentAPI.PageProperties

Поле	Описание
`DocumentAPI.PageProperties.height`	Высота страницы.
`DocumentAPI.PageProperties.width`	Ширина страницы.

3.75.1 Метод PageProperties:__eq

Метод используется для определения эквивалентности двух объектов `PageProperties`.

3.75.2 Метод PageProperties:__ne

Метод используется для определения неэквивалентности двух объектов `PageProperties`.

3.76 Таблица DocumentAPI.HeaderFooter

Таблица `DocumentAPI.HeaderFooter` определяет колонтитул текстового документа.

3.76.1 Метод HeaderFooter:getType

Метод предоставляет информацию о типе колонтитула (`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
```

3.76.2 Метод HeaderFooter:getBlocks

Метод предоставляет доступ к блокам (`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
```

3.76.3 Метод HeaderFooter:getRange

Метод предоставляет диапазон (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
```

3.77 Таблица DocumentAPI.HeadersFooters

Таблица DocumentAPI.HeadersFooters представляет коллекцию верхних и нижних колонтитулов раздела (DocumentAPI.Section).

3.77.1 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
```

3.78 Таблица DocumentAPI.LineEndingProperties

Таблица DocumentAPI.LineEndingProperties содержит свойства оформления окончаний линий. Описание полей таблицы DocumentAPI.LineEndingProperties представлено в Таблице 50.

Таблица 50 – Описание полей таблицы DocumentAPI.LineEndingProperties

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

3.79 Таблица DocumentAPI.DocumentSettings

Таблица `DocumentAPI.DocumentSettings` предоставляет общие настройки документа.

Описание полей таблицы `DocumentAPI.DocumentSettings` представлено в Таблице 51.

Таблица 51 – Описание полей таблицы DocumentAPI.DocumentSettings

Поле	Тип	Описание
`DocumentAPI.DocumentSettings.documentType`	`DocumentType`	Тип документа .
`DocumentAPI.DocumentSettings.userInfo`	`UserInfo`	Информация о пользователе.
`DocumentAPI.DocumentSettings.localeInfo`	`LocaleInfo`	Информация о локализации.
`DocumentAPI.DocumentSettings.timeZone`	`TimeZone`	Информация о временной зоне .
`DocumentAPI.DocumentSettings.formulaType`	`FormulaType`	Система адресации ячеек .

3.80 Таблица DocumentAPI.UserInfo

Таблица `DocumentAPI.UserInfo` предоставляет информацию о пользователе.

Описание полей таблицы `DocumentAPI.UserInfo` представлено в Таблице 52.

Таблица 52 – Описание полей таблицы DocumentAPI.UserInfo

Поле	Описание
`DocumentAPI.UserInfo.name`	Наименование пользователя.
`DocumentAPI.UserInfo.email`	Адрес электронной почты пользователя.

3.81 Таблица DocumentAPI.LocaleInfo

Таблица `DocumentAPI.LocaleInfo` предоставляет информацию о локализации.

Описание полей таблицы `DocumentAPI.LocaleInfo` представлено в Таблице 53.

Таблица 53 – Описание полей таблицы DocumentAPI.LocaleInfo

Поле	Описание
`DocumentAPI.LocaleInfo.localName`	Название локализации, представлено в формате **<language> <REGION>**, где языковой код соответствует стандарту ISO-639, а код региона стандарту ISO-3166.
`DocumentAPI.LocaleInfo.decimalSeparator`	Десятичный разделитель, отделяет целые и дробные части чисел.

Поле	Описание
DocumentAPI.LocaleInfo.thousandSeparator	Символ, разделяющий группы цифр в числовых значениях.
DocumentAPI.LocaleInfo.listSeparator	Символ, отделяющий элементы в списке.
DocumentAPI.LocaleInfo.currencySymbol	Символ валюты, используемой в текущей стране или регионе.
DocumentAPI.LocaleInfo.currencyFormat	Расположение знака валюты в текущем регионе.
DocumentAPI.LocaleInfo.shortDatePattern	Заданный «короткий» формат отображения даты в текущем регионе (например, 'm/d/yy' для en_US).
DocumentAPI.LocaleInfo.longDatePattern	Заданный «длинный» формат отображения даты в текущем регионе (например, 'dddd, mmmm d, yyyy' for en_US).
DocumentAPI.LocaleInfo.timePattern	Заданный формат отображения времени в текущем регионе (например, 'h:mm AM/PM' для en_US).

3.82 Таблица DocumentAPI.TimeZone

Таблица DocumentAPI.TimeZone предоставляет настройки, необходимые для экспорта текстовых документов.

Поле таблицы DocumentAPI.TimeZone.offsetInSecondsToUTC – содержит значение помощью которого задается смещение или разность между временем в данном часовом поясе и временем в формате UTC (Всемирное координированное время).

3.83 Таблица DocumentAPI.DSVSettings

Таблица DocumentAPI.DSVSettings предоставляет настройки, необходимые для работы с файлами CSV (comma-separated value) и DSV (delimiter-separated value).

Описание полей таблицы DocumentAPI.DSVSettings представлено в Таблице 54.

Таблица 54 – Описание полей таблицы DocumentAPI.DSVSettings

Поле	Описание
DocumentAPI.DSVSettings.separator	Символ-разделитель полей данных, значение по умолчанию – запятая.
DocumentAPI.DSVSettings.escapeChar	Символ-разделитель для текстовых строк, значение по умолчанию – двойная кавычка.
DocumentAPI.DSVSettings.autofit	Признак необходимости автоматического подстраивания ширины столбца под размер данных в ячейке.
DocumentAPI.DSVSettings.startBlockIndex	Индекс блока документа для начала сохранения.

Поле	Описание
DocumentAPI.DSVSettings.lastBlockIndex	Индекс блока документа для окончания сохранения.

3.84 Таблица DocumentAPI.Application

Таблица DocumentAPI.Application управляет параметрами и объектами приложения. Предоставляет интерфейс для создания и загрузки документов. Допустимо использование только одного объекта Application для всего сеанса обработки документа.

3.85 Таблица DocumentAPI.Messenger

3.85.1 Метод Messenger:subscribe

Метод служит для подписки на события лога.

3.85.2 Метод Messenger:notify

Метод используется для создания события лога

3.86 Таблица DocumentAPI.Connection

Таблица Connection реализует соединение между Messenger и клиентом. Содержит один метод unsubscribe для разрыва соединения.

3.87 Таблица DocumentAPI.Message

3.87.1 Таблица Message.Severity

Таблица Message.Severity (Таблица 55) описывает уровни сообщений лога (информация, предупреждение, ошибка).

Таблица 55 – Описание уровней лога Message.Severity

Поле	Описание
DocumentAPI.Message.Severity_Info	Информация
DocumentAPI.Message.Severity_Warning	Предупреждение
DocumentAPI.Message.Severity_Error	Ошибка

3.87.2 Метод Message:__eq

Метод используется для определения эквивалентности двух объектов Message.

3.87.3 Метод Message:__ne

Метод используется для определения неэквивалентности двух объектов Message.

3.87.4 Метод Message:getSeverity

Метод возвращает уровень лога DocumentAPI.Message_Severity.

3.87.5 Метод Message:getText

Метод возвращает текст сообщения.

3.87.6 Метод Message:makeInfo

Метод создает сообщение типа DocumentAPI.Message.Severity_Info с заданным текстом.

3.87.7 Метод Message:makeWarning

Метод создает сообщение типа DocumentAPI.Message.Severity_Warning с заданным текстом.

3.87.8 Метод Message:makeError

Метод создает сообщение типа DocumentAPI.Message.Severity_Error с заданным текстом.

3.88 Таблица DocumentAPI.SaveDocumentSettings

Таблица DocumentAPI.SaveDocumentSettings предоставляет настройки, используемые для сохранения документа в файл. Описание полей таблицы DocumentAPI.SaveDocumentSettings представлено в Таблице 56.

Таблица 56 – Описание полей таблицы DocumentAPI.SaveDocumentSettings

Поле	Описание
DocumentAPI.SaveDocumentSettings.documentFormat	Формат документа DocumentFormat.
DocumentAPI.SaveDocumentSettings.documentType	Тип документа DocumentType.
DocumentAPI.SaveDocumentSettings.documentPassword	Пароль для защиты электронного документа от несанкционированного доступа.
DocumentAPI.SaveDocumentSettings.isTemplate	Флаг, обозначающий, что документ должен быть сохранен как шаблон.
DocumentAPI.SaveDocumentSettings.dsvSettings	Структура DSVSettings, необходимая для сохранения в формате DSV.

3.89 Таблица DocumentAPI.LoadDocumentSettings

Таблица `DocumentAPI.LoadDocumentSettings` предоставляет дополнительные настройки, необходимые для загрузки документов из файла.

Описание полей таблицы `DocumentAPI.LoadDocumentSettings` представлено в Таблице 57.

Таблица 57 – Описание полей таблицы DocumentAPI.LoadDocumentSettings

Поле	Описание
`DocumentAPI.LoadDocumentSettings.commonDocumentSettings`	Экземпляр таблицы, общие настройки документа `DocumentSettings`.
`DocumentAPI.LoadDocumentSettings.encoding`	Кодировка документа `Encoding`.
`DocumentAPI.LoadDocumentSettings.dsvSettings`	Экземпляр класса `DSVSettings`, настройки, необходимые для работы с файлами CSV и DSV .
`DocumentAPI.LoadDocumentSettings.documentPassword`	Пароль для защиты электронного документа от несанкционированного доступа. Механизм парольной защиты поддерживается только для семейства ОС Microsoft Windows.

3.90 Таблица DocumentAPI.TextExportSettings

Таблица `DocumentAPI.TextExportSettings` предоставляет настройки, необходимые для экспорта текстовых документов. Поле таблицы `DocumentAPI.TextExportSettings.pageNumbers` содержит таблицу `DocumentAPI.PageNumbers`, в которой содержатся настройки страниц для экспорта текстовых документов.

3.91 Таблица DocumentAPI.PageNumbers

Таблица `DocumentAPI.PageNumbers` представляют собой коллекцию страниц для экспорта.

Позволяет установить следующие типы страниц для экспорта:

- нечетные, четные страницы `PageParity`;
- список номеров страниц;
- диапазон страниц, с указанием начальной и конечной страницы.

3.91.1 Метод PageNumbers:contains

Метод `PageNumbers:contains` служит для проверки вхождения заданного номера страницы в коллекцию номеров страниц для открытого документа.

Пример:

```
local isPageContained = DocumentAPI.PageNumbers():contains(10)
```

3.91.2 Метод PageNumbers:getLast

Метод `PageNumbers:getLast` возвращает последний номер страницы.

Пример:

```
local lastPageNum = DocumentAPI.PageNumbers():getLast()
```

3.92 Таблица DocumentAPI.WorkbookExportSettings

Таблица `DocumentAPI.WorkbookExportSettings` предоставляет настройки, необходимые для экспорта табличных документов.

Описание полей таблицы `DocumentAPI.WorkbookExportSettings` представлено в Таблице 58.

Таблица 58 – Описание полей таблицы DocumentAPI.WorkbookExportSettings

Поле	Описание
`DocumentAPI.WorkbookExportSettings.sheetNames`	Представляет коллекцию имен листов для экспорта. Если коллекция пуста, экспортируются все листы.
`DocumentAPI.WorkbookExportSettings.printingScope`	Представляет область печати (весь документ, область печати, пользовательский диапазон и т. д.) `PrintingScope`.
`DocumentAPI.WorkbookExportSettings.pageProperties`	Представляют свойства страницы для выходного документа (высота и ширина страницы в пунктах pt) `PageProperties`.
`DocumentAPI.WorkbookExportSettings.scale`	Представляет масштаб экспорта выходного документа в процентах (например, 50,0%, 150,63%, 400,0% и т. д.).

3.93 Таблица DocumentAPI.PrintingScope

Таблица `DocumentAPI.PrintingScope` содержит настройки для экспорта табличных документов.

Позволяет создать области печати следующих типов:

- выбранная область печати (по умолчанию) (см. `PrintingScope`);
- выбранная область печати либо весь документ (см. `PrintingScope`);

- указанный диапазон ячеек.

3.93.1 Метод PrintingScope:getCellRange

Метод `PrintingScope:getCellRange` возвращает диапазон ячеек таблицы (см. `CellRangePosition`).

3.93.2 Метод PrintingScope:usePrintArea

Метод `PrintingScope:usePrintArea` возвращает `true`, если область печати должна использоваться во время печати, экспорта и т. д.

3.94 Таблица DocumentAPI.Color

Таблица `DocumentAPI.Color` представляет либо цветовой объект RGBA, либо заданные цвета идентификатора темы.

Пример:

```
local rgbaColor = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0, 255))
local themeColor = DocumentAPI.Color(DocumentAPI.ThemeColorID_Text1)
```

3.94.1 Метод Color:getRGBAColor

Метод возвращает цвет RGBA.

Пример:

```
local rgbaColor = color:getRGBAColor()
if rgbaColor then
  print(rgbaColor.r, rgbaColor.g, rgbaColor.b, rgbaColor.a)
end
```

3.94.2 Метод Color:getThemeColorID

Метод возвращает цвет идентификатора темы (см. `ThemeColorID`).

Пример:

```
local themeColorID = color:getThemeColorID()
if themeColorID then
  print(themeColorID)
end
```

3.94.3 Метод Color:__eq

Метод используется для определения эквивалентности двух объектов `Color`.

Пример:

```
if color1 == color2 then
  print("Два одинаковых цвета")
end
```

3.94.4 Метод Color:__ne

Метод используется для определения неэквивалентности двух объектов `Color`.

Пример:

```
if color1 == color2 then
  print("Два одинаковых цвета")
end
```

3.95 Таблица DocumentAPI.TextAnchoredPosition

Таблица `DocumentAPI.TextAnchoredPosition` представляет позицию объекта на странице текстового документа. Описание полей таблицы представлено в Таблице 59.

Таблица 59 – Описание полей таблицы DocumentAPI.TextAnchoredPosition

Поле	Описание
`DocumentAPI.TextAnchoredPosition.horizontal`	Позиция по горизонтали `HorizontalTexAnchoredPosition`.
`DocumentAPI.TextAnchoredPosition.vertical`	Позиция по вертикали `VerticalTextAnchoredPosition`.

3.95.1 Метод TextAnchoredPosition:__eq

Метод используется для определения эквивалентности двух объектов `TextAnchoredPosition`.

3.95.2 Метод TextAnchoredPosition:__ne

Метод используется для определения неэквивалентности двух объектов `TextAnchoredPosition`.

3.96 Таблица DocumentAPI.AnchoredPosition

Таблица `DocumentAPI.AnchoredPosition` представляет позицию с относительными смещениями на странице текстового документа. Описание полей таблицы представлено в Таблице 60.

Таблица 60 – Описание полей таблицы DocumentAPI.AnchoredPosition

Поле	Описание
DocumentAPI.AnchoredPosition.textPosition	Позиция на странице текстового документа TextAnchoredPosition.

3.96.1 Метод AnchoredPosition:__eq

Метод используется для определения эквивалентности двух объектов AnchoredPosition.

3.96.2 Метод AnchoredPosition:__ne

Метод используется для определения неэквивалентности двух объектов AnchoredPosition.

3.97 Таблица DocumentAPI.Frame

Таблица DocumentAPI.Frame представляет рамку для встроенного объекта. Объект рамки может быть использован для изменения положения и геометрии встроенного объекта, поворота, управления переносом текста вокруг объекта и т. д.

3.97.1 Метод Frame:setPosition

Метод задает положение встроенного объекта. Для текстовых документов позиция может быть установлена только для встроенных объектов, тип переноса текста которых не является DocumentAPI.TextWrapType_Inline.

Пример:

```
local pos = DocumentAPI.TextAnchoredPosition()

-- Установка смещения по горизонтали относительно края колонки
pos.horizontal =
DocumentAPI.HorizontalTextAnchoredPosition
(DocumentAPI.HorizontalRelativeTo_Column)
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
frame:setPosition(pos)
```

3.97.2 Метод FrameFrame:getPosition

Метод возвращает позицию встроенного объекта на странице в виде таблицы (см. AnchoredPosition).

3.97.3 Метод Frame:setDimensions

Метод задает размеры (изменяет размер) встроенного объекта.

Вызов:

```
setDimensions( size )
```

Параметры:

`size` – размеры встроенного объекта.

Пример:

```
frame:setDimensions(DocumentAPI.SizeU(100, 100))
```

3.97.4 Метод Frame:getDimensions

Метод возвращает задает размеры встроенного объекта.

3.97.5 Метод Frame:setWrapType

Метод устанавливает вариант обтекания текстом встроенного объекта (см. TextWrapType).

Пример:

```
frame:setWrapType(DocumentAPI.TextWrapType_Square)
```

3.97.6 Метод Frame:getWrapType

Метод возвращает вариант обтекания текстом встроенного объекта (см. TextWrapType).

Пример:

```
local inlineObjects = document:getRange():getInlineObjects()
for inlineObject in inlineObjects:enumerate() do
    local frame = inlineObject:getFrame()
    print(frame:getWrapType())
end
```

3.98 Таблица DocumentAPI.Image

Таблица `DocumentAPI.Image` представляет собой изображение, как встроенный объект.

3.98.1 Метод Image:getFrame

Метод возвращает рамку встроенного объекта `InlineFrame`.

3.99 Таблица DocumentAPI.Images

Таблица `DocumentAPI.Images` предоставляет интерфейс для доступа к коллекции изображений.

3.99.1 Метод Images:enumerate

Метод возвращает коллекцию изображений.

Примеры:

```
for image in EditorAPI.getSelection():getImages():enumerate() do
  -- Действия с объектом image
end
```

```
local images = document:getRange():getImages()
for image in images:enumerate() do
  -- Действия с объектом image
end
```

3.100 Таблица DocumentAPI.InlineObject

Таблица `DocumentAPI.MediaObject` представляет собой встроенный объект, который позиционируется как символ в строке текста.

3.100.1 Метод InlineObject:toImage

Метод возвращает изображение `Image`, связанное со встроенным объектом.

Пример:

```
for inlineObject in EditorAPI.getSelection():getInlineObjects():enumerate() do
  local image = inlineObject:toImage()
  if image then
    print("Текущий объект является изображением")
```

```
  end
end
```

3.100.2 Метод InlineObject:getFrame

Метод возвращает рамку встроенного объекта `Frame`.

3.101 Таблица DocumentAPI.InlineObjects

Таблица `DocumentAPI.InlineObjects` предоставляет интерфейс для доступа к коллекции встроенных объектов.

3.101.1 Метод InlineObjects:enumerate

Метод возвращает коллекцию встроенных объектов.

Пример:

```
for inlineObject in EditorAPI.getSelection():getInlineObjects():enumerate() do
  -- Действия со встроенным объектом
end
```

3.102 Таблица DocumentAPI.HorizontalTextAnchoredPosition

Таблица `DocumentAPI.HorizontalTextAnchoredPosition` предназначена для управления относительным положением объекта со смещением или выравниванием по горизонтали.

Описание полей таблицы `DocumentAPI.HorizontalTextAnchoredPosition` представлено в Таблице 61.

Таблица 61 – Описание полей таблицы DocumentAPI.HorizontalTextAnchoredPosition

Поле	Описание
`DocumentAPI.HorizontalTextAnchoredPosition.relativeTo`	Тип размещения объекта относительно закрепленной позиции по горизонтали `HorizontalRelativeTo`.
`DocumentAPI.HorizontalTextAnchoredPosition.offset`	Смещение объекта.
`DocumentAPI.HorizontalTextAnchoredPosition.aligment`	Тип выравнивания объекта относительно закрепленной позиции по горизонтали `HorizontalAnchorAlignment`.

3.102.1 Метод HorizontalTextAnchoredPosition:__eq

Метод используется для определения эквивалентности двух объектов `HorizontalTextAnchoredPosition`.

3.102.2 Метод HorizontalTextAnchoredPosition:__ne

Метод используется для определения неэквивалентности двух объектов `HorizontalTextAnchoredPosition`.

3.103 Таблица DocumentAPI.VerticalTextAnchoredPosition

Таблица `DocumentAPI.VerticalTextAnchoredPosition` предназначена для управления относительным положением объекта со смещением или выравниванием по вертикали.

Описание полей таблицы `DocumentAPI.VerticalTextAnchoredPosition` представлено в Таблице 62.

Таблица 62 – Описание полей таблицы DocumentAPI.VerticalTextAnchoredPosition

Поле	Описание
`DocumentAPI.VerticalTextAnchoredPosition.relativeTo`	Тип размещения объекта относительно закрепленной позиции по вертикали `VerticalRelativeTo`.
`DocumentAPI.VerticalTextAnchoredPosition.offset`	Смещение объекта.
`DocumentAPI.VerticalTextAnchoredPosition.alignment`	Тип выравнивания объекта относительно закрепленной позиции по вертикали `VerticalAnchorAlignment`.

3.103.1 Метод VerticalTextAnchoredPosition:__eq

Метод используется для определения эквивалентности двух объектов `VerticalTextAnchoredPosition`.

3.103.2 Метод VerticalTextAnchoredPosition:__ne

Метод используется для определения неэквивалентности двух объектов `VerticalTextAnchoredPosition`.

3.104 Таблица DocumentAPI.PrintSettings

Таблица `DocumentAPI.PrintSettings` представляет установки, используемые при печати документов. Описание полей таблицы `DocumentAPI.PrintSettings` представлено в Таблице 63.

Таблица 63 – 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 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.

Поле	Тип	Описание
DocumentAPI.PrintSettings.bottomMargin	number	Ширина нижнего поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
DocumentAPI.PrintSettings.parity	PageParity	Тип выбора 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.

3.105 Таблица DocumentAPI.SizeU

Таблица `DocumentAPI.SizeU` представляет размер объекта в двухмерном пространстве. Описание полей таблицы `DocumentAPI.SizeU` представлено в Таблице 64.

Таблица64 – Описание полей таблицы DocumentAPI.SizeU

Поле	Тип	Описание
`DocumentAPI.SizeU.width`	number	Ширина объекта в двухмерном пространстве.
`DocumentAPI.SizeU.height`	number	Высота объекта в двухмерном пространстве.

3.105.1 Метод SizeU:toString

Возвращает информацию о размерах объекта в виде строкового значения формата `(width: <value>, height: <value>)`.

3.106 Именованные выражения

Именованное выражение – это выражение (являющееся описанием диапазона или формулой), которому присвоено имя. Преимуществом именованного выражения является его информативность. Именованные выражения упрощают работу с ячейками, также их удобно использовать при работе с формулами. На данный момент доступна возможность работы с именованными выражениями, представляющими собой ссылки на диапазоны ячеек.

3.106.1 Таблица DocumentAPI.NamedExpression

Класс описывает структуру именованного выражения.

3.106.1.1 Метод NamedExpression:getName

Возвращает имя именованного выражения.

3.106.1.2 Метод NamedExpression:getExpression

Возвращает текст выражения (формулы).

3.106.1.3 Метод NamedExpression:getCellRange

Возвращает диапазон ячеек `CellRange`, если выражение является ссылкой на диапазон.

3.106.2 Таблица DocumentAPI.NamedExpressions

Класс для представления списка именованных выражений.

3.106.2.1 Метод NamedExpression:NamedExpressions:get

Возвращает именованное выражение `NamedExpression` по имени `name`, если оно существует.

3.106.2.2 Метод NamedExpression:NamedExpressions:getEnumerator

Возвращает объект типа `Enumerator`, позволяющий получить доступ ко всему списку именованных выражений.

3.106.2.3 Метод NamedExpression:addExpression

Добавляет новое выражение в список именованных выражений, возвращает результат операции `NamedExpressionsValidationResult`.

3.106.2.4 Метод NamedExpressions:removeExpression

Удаляет выражение по заданному имени `name`, возвращает результат операции `NamedExpressionsValidationResult`.

3.106.3 Таблица DocumentAPI.NamedExpressionsValidationResult

Таблица `DocumentAPI.NamedExpressionsValidationResult` описывает результат операции, связанной с именованным выражением. Описание полей таблицы представлено в Таблице 65.

Таблица 65 – 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.

3.107 Сводные таблицы

3.107.1 Таблица DocumentAPI.PivotTable

Таблица для представления сводной таблицы.

3.107.1.1 Метод PivotTable:remove

Метод удаляет сводную таблицу.

3.107.1.2 Метод PivotTable:getSourceRangeAddress

Метод возвращает текстовое представление диапазона исходных данных сводной таблицы.

3.107.1.3 Метод PivotTable:getSourceRange

Метод возвращает диапазон `CellRange` исходных данных сводной таблицы.

3.107.1.4 Метод PivotTable:getPivotRange

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

3.107.1.5 Метод PivotTable:changeSourceRange

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

3.107.1.6 Метод PivotTable:isRowGrandTotalEnabled

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

3.107.1.7 Метод PivotTable:isColumnGrandTotalEnabled

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

3.107.1.8 Метод PivotTable:getPivotTableCaptions

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

3.107.1.9 Метод PivotTable:getPivotTableLayoutSettings

Метод возвращает настройки отображения `PivotTableLayoutSettings` сводной таблицы.

3.107.1.10 Метод PivotTable:getUnsupportedFeatures

Метод возвращает неподдерживаемые свойства `PivotTableUnsupportedFeature` сводной таблицы.

3.107.1.11 Метод PivotTable:getFieldsList

Метод возвращает список `PivotTableField` всех полей сводной таблицы.

3.107.1.12 Метод PivotTable:getRowFields

Метод возвращает список полей `PivotTableCategoryField` из области строк.

3.107.1.13 Метод PivotTable:getColumnFields

Метод возвращает список полей `PivotTableCategoryField` из области колонок.

3.107.1.14 Метод PivotTable:getValueFields

Метод возвращает список полей `PivotTableValueField` из области значений.

3.107.1.15 Метод PivotTable:getPageFields

Метод возвращает список полей `PivotTablePageField` из области фильтров.

3.107.1.16 Метод PivotTable:getFieldCategories

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

3.107.1.17 Метод PivotTable:getFieldItems

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

3.107.1.18 Метод PivotTable:getFieldItemsByName

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

3.107.1.19 Метод PivotTable:getFilter

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

3.107.1.20 Метод PivotTable:getFilters

Метод возвращает список фильтров `PivotTableFilter` сводной таблицы.

3.107.1.21 Метод PivotTable:update

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

3.107.1.22 Метод PivotTable:createPivotTableEditor

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

3.107.2 Таблица DocumentAPI.PivotTableCaptions

Таблица DocumentAPI.PivotTableCaptions хранит все пользовательские заголовки сводной таблицы. Описание полей таблицы представлено в Таблице 66.

Таблица 66 – Описание полей таблицы `DocumentAPI.PivotTableCaptions`

Поле	Описание
`PivotTableCaptions.errorCaption`	Алиас для значений, которые возвращают ошибку.

Поле	Описание
PivotTableCaptions.emptyCaption	Алиас для значений, которые возвращают пустое значение.
PivotTableCaptions.grandTotalCaption	Алиас общих итогов.
PivotTableCaptions.valuesHeaderCaption	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 имеет тип ‘outline’ или ‘tabular’.
PivotTableCaptions.rowHeaderCaption	Алиас заголовка строк (виден только при включенном компактном макете, это алиас по умолчанию).
PivotTableCaptions.columnHeaderCaption	Алиас заголовка колонок (виден только при включенном компактном макете, это алиас по умолчанию).

3.107.3 Таблица DocumentAPI.PivotTableLayoutSettings

Таблица DocumentAPI.PivotTableLayoutSettings содержит настройки отображения сводной таблицы. Описание полей таблицы представлено в Таблице 67.

Таблица 67 – Описание полей таблицы DocumentAPI.PivotTableLayoutSettings

Поле	Описание
PivotTableLayoutSettings.reportLayout	Настройка вида макета сводной таблицы (PivotTableReportLayout: компактный, табличный, структурный).
PivotTableLayoutSettings.valueFieldsOrientation	Позволяет расположить 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.

Поле	Описание
PivotTableLayoutSettings.displayFieldCaptions	Флаг, отвечающий за отображение заголовков полей.

3.107.4 Таблица DocumentAPI.PivotTableUnsupportedFeature

Таблица DocumentAPI.PivotTableUnsupportedFeature описывает неподдерживаемую функциональность сводных таблиц. Описание полей таблицы представлено в Таблице 68.

Таблица 68 – Описание полей таблицы DocumentAPI.PivotTableUnsupportedFeature

Поле	Описание
DocumentAPI.PivotTableUnsupportedFeature_CalculatedField	Вычисляемые поля.
DocumentAPI.PivotTableUnsupportedFeature_CalculatedItem	Вычисляемые элементы.
DocumentAPI.PivotTableUnsupportedFeature_CollapsedValues	Свернутые поля.
DocumentAPI.PivotTableUnsupportedFeature_ShowDataAs	Вычисления (Show data как в MS Excel).

3.107.5 Таблица DocumentAPI.PivotTableReportLayout

Таблица DocumentAPI.PivotTableReportLayout описывает внешний вид отчетов сводной таблицы. Описание полей таблицы представлено в Таблице 69.

Таблица 69 – Описание полей таблицы DocumentAPI.PivotTableReportLayout

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

3.107.6 Таблица DocumentAPI.ValueFieldsOrientation

Таблица DocumentAPI.ValueFueldsOrientation описывает варианты ориентации в случае, когда в сводной таблице более, чем одно поле из области значений. Описание полей таблицы представлено в Таблице 70.

Таблица 70 – Описание полей таблицы `DocumentAPI.ValueFueldsOrientation`

Поле	Описание
`DocumentAPI.ValueFueldsOrientation.ByRows`	По строкам.
`DocumentAPI.ValueFueldsOrientation.ByColumns`	По столбцам.

3.107.7 Таблица DocumentAPI.PageFieldOrder

Таблица `DocumentAPI.PageFieldOrder` описывает вид отображения полей из области фильтров. Описание полей таблицы представлено в Таблице 71.

Таблица 71 – Описание полей таблицы `DocumentAPI.PageFieldOrder`

Поле	Описание
`DocumentAPI.PageFieldOrder.DownThenOver`	Вниз, затем поперек.
`DocumentAPI.PageFieldOrder.OverThenDown`	Поперек, затем вниз.

3.107.8 Таблица DocumentAPI.PivotTableFieldCategory

Таблица `DocumentAPI.PivotTableFieldCategory` описывает флаги, которые задают категорию области полей. Описание полей таблицы представлено в Таблице 72.

Таблица 72 – Описание полей таблицы `DocumentAPI.PivotTableFieldCategory`

Поле	Описание
`DocumentAPI.PivotTableFieldCategory_Pages`	Область фильтров.
`DocumentAPI.PivotTableFieldCategory_Rows`	Область строк.
`DocumentAPI.PivotTableFieldCategory_Columns`	Область колонок.
`DocumentAPI.PivotTableFieldCategory_Values`	Ообласть значений.

3.107.9 Таблица DocumentAPI.PivotTableFieldCategories

Класс обеспечивает доступ к списку категорий полей сводных таблиц.

3.107.9.1 Метод getEnumerator

Возвращает объект `Enumerator` для доступа к коллекции категорий.

3.107.10 Таблица DocumentAPI.PivotTableFunction

Таблица `DocumentAPI.PivotTableFunction` описывает функции, которые могут быть использованы в сводных таблицах. Описание полей таблицы представлено в Таблице 73.

Таблица 73 – 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.
`DocumentAPI.PivotTableFunction_StdDeviationPopulation`	Стандартное несмещенное отклонение.
`DocumentAPI.PivotTableFunction_Variance`	Смещенная дисперсия.
`DocumentAPI.PivotTableFunction_VariancePopulation`	Несмещенная дисперсия.

3.107.11 Таблица DocumentAPI.PivotTableFilter

Представляет собой интерфейс для доступа к списку фильтров таблицы, каждый из которых обладает свойством видимости. При любом изменении фильтров они должны быть применены к сводной таблице посредством использования метода `setFilter` класса `PivotTableEditor`.

3.107.11.1 Метод getFieldName

Возвращает имя поля, с которым ассоциирован фильтр.

3.107.11.2 Метод getCount

Возвращает количество фильтруемых полей.

3.107.11.3 Метод getName

Возвращает имя поля для заданного индекса.

3.107.11.4 Метод isHidden

Возвращает видимость поля для заданного индекса `itemIndex`, если `true`, то поле

скрыто.

3.107.11.5 Метод setHidden

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

3.107.12 Таблица DocumentAPI.PivotTableFilters

Таблица обеспечивает доступ к списку фильтров.

3.107.12.1 Метод getEnumerator

Возвращает объект `Enumerator` для доступа к коллекции фильтров.

3.107.13 Таблица DocumentAPI.PivotTableFieldProperties

`DocumentAPI.PivotTableFieldProperties` содержит основные свойства полей сводной таблицы (см.Таблицу 74).

Таблица 74 – Описание полей таблицы `DocumentAPI.PivotTableFieldProperties`

Поле	Описание
`PivotTableFieldProperties.fieldName`	Имя поля.
`PivotTableFieldProperties.fieldAlias`	Псевдоним поля (пользовательское имя поля).
`PivotTableFieldProperties.subtotalAlias`	Псевдоним подытогов конкретного поля.

3.107.14 Таблица DocumentAPI.PivotTableField

Таблица `DocumentAPI.PivotTableField` содержит свойства полей сводной таблицы (см.Таблицу 75).

Таблица 75 – Описание полей таблицы `DocumentAPI.PivotTableField`

Поле	Описание
`PivotTableField.fieldProperties`	Свойства полей сводной таблицы `PivotTablFieldProperties`.
`PivotTableField.fieldCategories`	Категории полей сводной таблицы `PivotTablFieldCategories`.
`PivotTableField.customFormula`	Вычисляемая формула (строка).

3.107.15 Таблица DocumentAPI.PivotTableCategoryField

`DocumentAPI.PivotTableCategoryField` содержит свойства поля сводной таблицы, использующегося как строка / столбец (см.Таблицу 76).

Таблица 76 – Описание полей таблицы `DocumentAPI.PivotTableCategoryField`

Поле	Описание
`PivotTableCategoryField .fieldProperties`	Свойства поля `PivotTableFieldProperties`.
`PivotTableCategoryField .subtotalFunctions`	Список функций `PivotTableFunctions` для вычисления подытога.

3.107.16 Таблица DocumentAPI.PivotTableValueField

`DocumentAPI.PivotTableValueField` содержит свойства поля сводной таблицы, использующегося как значение столбец (см.Таблицу 77).

Таблица 77 – Описание полей таблицы `DocumentAPI.PivotTableValueField`

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

3.107.17 Таблица DocumentAPI.PivotTablePageField

Содержит свойства поля из области фильтров (см. Таблицу 78).

Таблица 78 – Описание полей таблицы `DocumentAPI.PivotTablePageField`

Поле	Описание
`PivotTablePageField.fieldProperties`	Свойства поля `PivotTableFieldProperties`.

3.107.18 Таблица DocumentAPI.PivotTableItem

`DocumentAPI.PivotTableItem` описывает элемент сводной таблицы.

3.107.18.1 Метод getName

Метод возвращает имя элемента сводной таблицы, тип - строка.

3.107.18.2 Метод getAlias

Метод возвращает псевдоним элемента (идентификатор, созданный пользователем), тип - строка.

3.107.18.3 Метод getItemType

Метод возвращает тип `PivotTableItemType` элемента сводной таблицы.

3.107.18.4 Метод isCollapsed

Метод возвращает `true`, если элемент сводной таблицы свернут.

3.107.19 Таблица DocumentAPI.PivotTableItemType

Таблица `DocumentAPI.PivotTableItemType` содержит возможные типы элементов сводной таблицы. Описание полей таблицы представлено в Таблице 79.

Таблица 79 – 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.

3.107.20 Таблица DocumentAPI.PivotTableEditor

3.107.20.1 Метод PivotTableEditor:addField

Метод добавляет новое поле в сводную таблицу, используя параметры: `fieldName` - имя поля, `toCategory` - категория поля (тип - `PivotTableFieldCategory`), `index` - позиция в категории. Метод возвращает объект `PivotTableEditor`.

3.107.20.2 Метод PivotTableEditor:moveField

Метод перемещает поле между категориями. Параметры: `fieldName` - имя поля, `toCategory` - область, в которую перемещается поле (тип - `PivotTableFieldCategory`), `index` - позиция в новой категориии. Метод возвращает объект `PivotTableEditor`.

3.107.20.3 Метод PivotTableEditor:removeField

Метод удаляет поле из категории. Параметры: `fieldName` - имя поля, `fromCategory` - область, из которой удаляется поле (тип - `PivotTableFieldCategory`). Метод возвращает объект `PivotTableEditor`.

3.107.20.4 Метод PivotTableEditor:reorderField

Метод изменяет позицию поля в пределах категории. Параметры: `fieldName` - имя поля, `category` - область (тип - `PivotTableFieldCategory`), `toIndex` - новая позиция поля. Метод возвращает объект `PivotTableEditor`.

3.107.20.5 Метод PivotTableEditor:enableField

Метод добавляет поле в область, зависящую от типа поля. Параметр `fieldName` - имя поля. Метод возвращает объект `PivotTableEditor`.

3.107.20.6 Метод PivotTableEditor:disableField

Метод удаляет поле из всех областей. Параметр `fieldName` - имя поля (тип - строка). Метод возвращает объект `PivotTableEditor`.

3.107.20.7 Метод PivotTableEditor:setSummmarizeFunction

Метод задает суммирующую функцию для поля из области значений. Параметр `valueFieldName` - имя поля (тип - строка), `summarizeFunction` – суммирующая функция, тип - `PivotTableFunction`. Метод возвращает объект `PivotTableEditor`.

3.107.20.8 Метод PivotTableEditor:setFilter

Метод задает фильтр `PivotTableFilter` сводной таблицы. Если фильтр не может быть применен, вызывается исключение `PivotTableError`. Метод возвращает объект `PivotTableEditor`.

3.107.20.9 Метод PivotTableEditor:setFilters

Метод задает фильтры `PivotTableFilters` сводной таблицы. Если какой-то из фильтров не может быть применен, он пропускается. Метод возвращает объект `PivotTableEditor`.

3.107.20.10 Метод PivotTableEditor:setCaptions

Метод задает заголовки сводной таблицы `PivotTableCaptions`, возвращает объект `PivotTableEditor`.

3.107.20.11 Метод PivotTableEditor:setLayoutSettings

Метод устанавливает настройки отображения `PivotTableLayoutSettings` сводной таблицы, возвращает объект `PivotTableEditor`.

3.107.20.12 Метод PivotTableEditor:setGrandTotalSettings

Метод задает настройки отображения общего итога. Параметры: `isRowGrandTotalEnabled` – показывать общие итоги для строк, `isColGrandTotalEnabled` – показывать общие итоги для столбцов.

3.107.20.13 Метод PivotTableEditor:apply

Метод обновляет сводную таблицу с заданными свойствами и возвращает результат `PivotTableUpdateResult`.

3.107.21 Таблица DocumentAPI.PivotTableUpdateResult

В Таблице 80 приведены константы, которые соответствуют возможным результатам обновления сводной таблицы.

Таблица 80 – 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_NoSuchFieldInPivotTable`	Не найдено поле в сводной таблице.
`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_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/создать сводную таблицу на именованном диапазоне, который не содержит ссылку, а содержит константу.
`DocumentAPI_PivotTableUpdateResult_AnotherPivotInsideDataSource`	Найдена другая сводная таблица в этом же диапазоне.

3.107.22 Таблица DocumentAPI.PivotTablesManager

3.107.22.1 Метод PivotTablesManager:create

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

Если местоположение (параметр `destination`, тип `Cell`) не задано, создается новый лист (таблица), и сводная таблица будет расположена по умолчанию.

3.108 Диаграммы

3.108.1 Таблица DocumentAPI.Chart

Таблица `DocumentAPI.Chart` представляет диаграмму в документе со всеми элементами (заголовок, легенда, тип, данные, диапазон и т.д.).

3.108.1.1 Метод Chart:getType

Метод возвращает тип диаграммы `ChartType`.

Пример:

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

3.108.1.2 Метод Chart:setType

Метод устанавливает тип диаграммы `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())
```

3.108.1.3 Метод Chart:getRangesCount

Метод возвращает количество серий диаграммы.

Пример:

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

3.108.1.4 Метод Chart:getRange

Метод возвращает диапазон ячеек `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)
```

3.108.1.5 Метод getTitle

Метод возвращает заголовок диаграммы.

Пример:

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

3.108.1.6 Метод Chart:setRange

Метод задает диапазон 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)
```

3.108.1.7 Метод Chart:setRect

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

3.108.1.8 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())
```

3.108.1.9 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())
```

3.108.1.10 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())
```

3.108.1.11 Метод Chart:getDirectionType

Метод возвращает направление ChartSeriesDirectionType серий диаграммы.

Пример:

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

3.108.1.12 Метод getChartLabels

Метод возвращает коллекцию меток диаграммы типа ChartLabelsInfo.

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,
chartLabels.isOneColumnRowChart)
```

3.108.1.13 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())
```

3.108.1.14 Метод Chart:applySettings

Метод позволяет обновить параметры текущей выбранной диаграммы.

Вызов:

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

Параметры:

– `cellRange` – обновленный диапазон исходных данных диаграммы CellRange;

– `directionType` – направление серий ChartSeriesDirectionType;

– `title` – заголовок диаграммы (тип - строка);

– `labelsInfo` – информация о метках диаграммы `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)
```

3.108.2 Таблица DocumentAPI.Charts

Таблица `DocumentAPI.Charts` обеспечивает доступ к списку диаграмм документа.

3.108.2.1 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())
```

3.108.2.2 Метод Charts :getChart

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

Пример:

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

3.108.2.3 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))
```

3.108.3 Таблица DocumentAPI.ChartLabelsDetectionMode

Таблица `DocumentAPI.ChartLabelsDetectionMode` описывает режимы автоматического определения меток диаграмм. Описание полей таблицы представлено в Таблице 81.

Таблица 81 – Описание полей таблицы `DocumentAPI.ChartLabelsDetectionMode`

Поле	Описание
`DocumentAPI.ChartLabelsDetectionMode_Unknown`	Неопределенный тип.
`DocumentAPI.ChartLabelsDetectionMode_FirstRow`	Метка на первой строке.
`DocumentAPI.ChartLabelsDetectionMode_FirstColumn`	Метка на первой колонке.
`DocumentAPI.ChartLabelsDetectionMode_NoLabels`	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)
```

3.108.4 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, `seriesNameMode` – режим автоматического определения меток для серий, `oneColumnRow` – передается `true`, если диапазон диаграммы содержит только одну строку или одну колонку. Описание полей таблицы представлено в Таблице 82.

Таблица 82 – Описание полей таблицы DocumentAPI.ChartLabelsInfo

Поле	Описание
ChartLabelsInfo_categoriesMode	Режим автоматического определения меток для категорий.
ChartLabelsInfo_seriesNameMode	Режим автоматического определения меток для серий.
ChartLabelsInfo_isOneColumnRowChart	Поле содержит true, если диапазон диаграммы содержит только одну строку или одну колонку.

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)
```

3.108.5 Таблица DocumentAPI.ChartRangeInfo

Таблица DocumentAPI.ChartRangeInfo описывает серию диаграммы. Инициализируется конструктором, в который передаются следующие параметры: tableRangeInfo - диапазон ячеек, color – цвет серии диаграммы, hidden – видимость серии, rangeType – тип диапазона исходных данных диаграммы. Описание полей таблицы представлено в Таблице 83.

Таблица 83 – Описание полей таблицы DocumentAPI.ChartRangeInfo

Поле	Описание
ChartRangeInfo_tableRangeInfo	Исходный диапазон ячеек TableRangeInfo для серии.
ChartRangeInfo_rangeColor	Цвет ColorRGBA для отрисовки серии.
ChartRangeInfo_isHidden	Задает видимость серии диаграммы.
ChartRangeInfo_rangeType	Тип диапазона диаграммы 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)
```

3.108.6 Таблица DocumentAPI.ChartRangeType

Таблица `DocumentAPI.ChartRangeType` описывает тип диапазона исходных данных диаграммы. Описание полей таблицы представлено в Таблице 84.

Таблица 84 – 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])
```

3.108.7 Таблица DocumentAPI.ChartSeriesDirectionType

Таблица `DocumentAPI.ChartSeriesDirectionType` описывает направление серий диаграмм. Описание полей таблицы представлено в Таблице 85.

Таблица 85 – 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())
```

3.108.8 Таблица DocumentAPI.ChartType

Таблица `DocumentAPI.ChartType` описывает все поддерживаемые типы диаграмм. Описание полей таблицы представлено в Таблице 86.

Таблица 86 – 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_Pie AreaPercentStacked`	Круговая диаграмма.
`DocumentAPI.ChartType_PieExploded`	Круговая диаграмма с отделенными секторами.
`DocumentAPI.ChartType_Scatter`	Диаграмма рассеяния.

Пример:

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

4 СПРАВОЧНИК ФУНКЦИЙ ГЛОБАЛЬНОЙ ТАБЛИЦЫ EDITORAPI

Глобальная таблица `EditorAPI` содержит функции доступа к внешней функциональности редактора.

4.1 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 функция `EditorAPI.getSelection` возвращает `CellRange`.

Примеры:

Использование функции `EditorAPI.getSelection` в редакторе текста для печати выделенного фрагмента текста.

```
local text = EditorAPI.getSelection():extractText()
print(text)
```

Использование функции `EditorAPI.getSelection` в редакторе таблиц для печати значений ячеек в выделенном фрагменте таблицы.

```
for cell in EditorAPI.getSelection():enumerate() do
    print(cell:getFormattedValue())
end
```

4.2 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`.

Вызов функции для табличного документа:

```
EditorAPI.setSelection(range)
```

Где:

– `range` – выделяемый в табличном документе фрагмент таблицы типа `DocumentAPI.CellRange`.

Примеры:

Использование функции `EditorAPI.setSelection` в текстовом документе:

```
EditorAPI.setSelection(document:getBlocks():getParagraph(0):getRange())
```

Использование функции `EditorAPI.setSelection` в табличном документе:

```
EditorAPI.setSelection(document:getBlocks():getTable(0):getCellRange("A1:E5"))
```

4.3 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("Message", "Title")
```

4.4 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 `PrintDocumentResult`.

Пример:

```
EditorAPI.showPrintDialog()
```

4.5 Функция EditorAPI.printDocument

Функция `EditorAPI.printDocument` предоставляет возможность печати документа с заданными параметрами печати. Описание параметров печати представлено в разделе `PrintSettings`.

Значения, возвращаемые функцией `EditorAPI.printDocument`, перечислены в разделе `PrintDocumentResult`.

Пример:

```
local printSettings = {}
printSettings.printSelection = true
EditorAPI.printDocument(printSettings)
```

4.6 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 ФУНКЦИИ ДЛЯ РАБОТЫ СО СТРОКАМИ В ФОРМАТЕ ЮНИКОД (UTF-8)

5.1 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 (`string`) в формате UTF-8

Возвращает:

`string`

5.2 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 (`string`) в формате UTF-8.

Возвращает:

`string`

5.3 Функция utf8.substr

Функция `utf8.substr` возвращает подстроку в формате UTF-8, полученную из исходной строки путем выборки с заданной позиции первого символа и до позиции последнего указанного символа.

Вызов:

```
utf8.substr(str, first[, last])
```

Параметры:

- `str` – исходная строка (`string`) в формате UTF-8;
- `first` – позиция (`number`) первого символа строки для выборки.
- `last` – позиция (`number`) последнего символа строки для выборки.

Возвращает:

`string`

5.4 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` согласно алгоритму сортировки по Юникоду:

- -1 – если `str1` < `srt2`;
- 0 – если `str1` = `str2`;
- 1 – если `str1` > `str2`.

5.5 Функция utf8.islower

Функция `utf8.islower` проверяет, находится ли в нижнем регистре переданный символ или число.

Вызов:

```
utf8.islower(char)
```

Параметры:

- `char` – строка (string) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код символа в нижнем регистре.

5.6 Функция utf8.isupper

Функция `utf8.islower` проверяет, находится ли в верхнем регистре переданный символ или число.

Вызов:

```
utf8.isupper(char)
```

Параметры:

- `char` – строка (string) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код символа в верхнем регистре.

5.7 Функция utf8.isdigit

Функция `utf8.isdigit` проверяет, является ли цифровым символом переданный символ или число.

Вызов:

```
utf8.isdigit(char)
```

Параметры:

– `char` – строка (`string`) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код цифрового символа.

5.8 Функция utf8.isalpha

Функция `utf8.isalpha` проверяет, является ли буквенным символом переданный символ или число.

Вызов:

```
utf8.isalpha(char)
```

Параметры:

– `char` – строка (`string`) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код буквенного символа.

5.9 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.

6 ФУНКЦИИ ДЛЯ РАБОТЫ С РЕГУЛЯРНЫМИ ВЫРАЖЕНИЯМИ

6.1 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`.

6.2 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`.

6.2.1 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 87

Таблица 87 – 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 ”\A” и "\`" не должны совпадать с подмножеством *[first, first)*.
`NotEOB`	Указывает, что выражения "\`", "\ z" и "\Z" не должны совпадать с подмножеством *[last, last)*.
`NotBOL`	Указывает, что выражение "^" не должны совпадать с подмножеством *[first, first)*.
`NotEOL`	Указывает, что выражение "$" не должно совпадать с подмножеством *[last, last)*.
`NotBOW`	Указывает, что выражения "\<" и "\b" не должны совпадать с подмножеством *[first, first)*.
`NotEOW`	Указывает, что выражения "\>" и "\b" не должны совпадать с подмножеством *[last, last)*.
`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 *[from, last)*, при этом *from! = last*, если может существовать более длинная последовательность символов *[from, to)*, из которых *[from, last)* - это префикс, который приведет к полному совпадению. Этот флаг используется при сопоставлении неполных или очень длинных текстов; дополнительную информацию см. документацию 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 `match_results::captures()` или `sub_match_captures()`.
`SingleLine`	Эквивалентно обратному модификатору "m/" языка Perl; предотвращает поиск ^ после встроенного символа новой строки (для того, чтобы он совпадал только в начале исходного текста) и $ от поиска перед встроенным символом новой строки (чтобы он совпадал только в конце исходного текста).

Флаг	Описание
`PrevAvail`	Указывает, что `--first` является допустимой позицией итератора, когда этот флаг установлен, тогда флаги `match_not_bol` и `match_not_bow` игнорируются алгоритмами регулярных выражений (RE.7) и итераторов (RE.8).
`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 *"leftmost-longest"*, независимо от того, какое выражение было скомпилировано. Эти правила плохо работают со многими специфичными для Perl моментами, например, такими как ленивые (*"non-greedy"*) 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.

6.3 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`

6.4 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`.

6.5 Флаги, используемые для замены

Эти флаги определены в пространстве имен `Re.Replace`. Они используются в алгоритме, используемом методом `Re.replace()` и находятся в Таблице 88

Таблица 88 – Описание флагов для функции `Re.replace()`

Флаг	Описание
`FormatDefault`	Когда к исходному тексту применяется регулярное выражение, и находится очередной фрагмент для замены, новая строка формируется с использованием функции замены *ECMAScript*, ECMA-262, Спецификация языка ECMAScript, глава 15, часть 5.4.11 String.prototype.replace. Эта функциональность идентична описанной в руководстве Perl Format String Syntax. После того, как все неперекрывающиеся вхождения регулярного выражения найдены и заменены, фрагменты исходного текста, не соответствующие регулярному выражению, копируются в результирующую строку без изменений.
`FormatSed`	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 SystemInterface (POSIX), Shells and Utilities. См. также Sed Format String Syntax.
`FormatPerl`	В случае, когда регулярное выражение применяется для замены в строке, новая строка формируется по правилам Perl 5.
`FormatLiteral`	В случае, когда регулярное выражение применяется для замены в строке, новая строка будет являться строковой копией заменяемого текста.
`FormatNoCopy`	В случае, когда регулярное выражение применяется для замены в строке, фрагменты, не соответствующие регулярному выражению, не будут копироваться в результирующую строку.

Флаг	Описание
`FormatFirstOnly`	Когда данный флаг установлен при операции поиска или замены, то заменяется только первое вхождение регулярного выражения.
`FormatAll`	Все синтаксические расширения включены, включая условные замены (*?ddexpression1:expression2*). Для дополнительных деталей см. Руководство по форматированию строк Boost.

7 КЛАСС MATCHES

Класс `Matches` содержит результат функций `Re.match()` и `Re.search()`.

7.1 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`.

7.2 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`.

7.3 Метод getSize

Вызов:

```
size, err = matches:getSize()
```

Возвращает:

- `size (int)` – количество найденных групп;
- `err (string)` - сообщение об ошибке или `nil`.

7.4 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`.

7.5 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)
```