Плагины и макросы.

Десктоп редакторы

Дата создания: 11 августа 2023г.

Дата последнего изменения: 10 мая 2023г.

Оглавление

Начало работы

Обзор

Р7 Сервер документов и десктопные редакторы предлагают поддержку подключаемых плагинов, позволяющих разработчикам добавлять в редакторы определенные функции, не связанные напрямую с форматом OOXML.

В этой документации описаны:

Как собрать, локализовать, запустить, протестировать и опубликовать плагин.

Как написать, отладить и запустить макрос или конвертировать макросы VBA для использования в Р7.

Как использовать методы и события Р7

Что могут плагины

Вот несколько примеров того, что вы можете делать с плагинами Р7:

- подключить сторонние сервисы, такие как Telegram, YouTube, Thesaurus, Translator, Zotero, OCR и т. д.;
- добавить пользовательские компоненты и элементы управления в пользовательский интерфейс, такие как **Trackchanges, Trackchanges_off**;

- улучшить существующий пользовательский интерфейс и функциональность редакторов: автозаполнение, поиск & замена, перемещение курсора, расширенные комментарии;
- автоматизируйте задачи в документах с помощью макросов.

Как создавать плагины

Чтобы создать свой собственный плагин, вам нужно выполнить несколько шагов:

1. Изучите основные принципы создания плагинов на странице начало работы с плагинами и напишите код, определяющий внешний вид и поведение плагина.
2. После сборки плагина тщательно протестируйте его и убедитесь, что плагин выглядит и работает так, как вы ожидали. Постарайтесь предвидеть проблемы, которые могут возникнуть, и предложите свои пути их решения.

Как получить помощь

Если у вас есть какие-либо вопросы о плагинах, попробуйте сначала найти их в разделе часто задаваемых вопросов.

Типы плагинов

Можно выделить следующие типы плагинов:

1. **Визуальный/невизуальный** (отмечен флагом isVisual в config.json):
   - Визуальные плагины («isVisual»: true) открывают окно или панель для некоторых действий.
   - Невизуальные плагины («isVisual»: false) предоставляют кнопку (или кнопки) для применения некоторых преобразований или манипуляций к документу.
   - Вспомогательный модуль ввода («события»: [«onInputHelperClear», «onInputHelperInput»]) представляет собой комбинацию как визуальных, так и невизуальных плагинов. У него есть собственное окно, которое появляется и исчезает при наборе текста. Его местоположение привязано к курсору.
2. **Системные/несистемные** (отмечены флагом isSystem в config.json):
   - Несистемные плагины (не отмеченные в config.json) запускаются при нажатии соответствующей кнопки, например.
   - Системные плагины («isSystem»: true) плагины работают в фоновом режиме, пока запущены редакторы. Вам не нужно их запускать.
3. **Улучшение редактора / улучшение пользовательского интерфейса / плагины сторонних сервисов**:
   - Плагины расширения редактора расширяют функциональные возможности редактора и улучшают существующий пользовательский интерфейс, например поиск и замена, добавление комментария в ячейку, перемещение курсора.
   - Улучшение плагина пользовательского интерфейса добавляют в пользовательский интерфейс пользовательские компоненты и элементы управления.
   - Плагины сторонних сервисов подключают к редакторам сторонние сервисы.
4. **Объект OLE** (помечен как «initDataType»: флаг «ole» в файле config.json):
   - В дополнение к простой обработке или редактированию документа плагин может встраивать OLE-объект, который позволяет стороннему разработчику получить доступ к формату документа, т. е. сохранить информацию из внешних ресурсов в результирующий файл. Например, плагин YouTube встраивает видео с YouTube в ваш документ.

Начало работы с плагинами

Вот несколько основных концепций пошагового создания плагинов:

1. Подготовка
2. Кодирование
3. Регулировка стиля плагина
4. Локализация
5. Подготовка плагина к публикации
6. Тестирование
7. Публикация

Шаг 1. Программирование

Разработать плагин. Следуйте описанной здесь структуре плагина. Папка плагина должна содержать три файла, необходимые для работы плагина: config.json, index.html, pluginCode.js.

Используйте методы и события плагина при написании кода плагина.

Шаг 2. Регулирование стиля плагина

Добавьте таблицу стилей Р7 в файл *index.html*, чтобы настроить плагин в соответствии со стилем редактора Р7:

```
<link rel="stylesheet" href="https://support.r7-office.ru/wp-content/uploads/2023/05/plugins.css">
```

Шаг 3. Локализация

Создайте папку переводов в каталоге плагинов с файлами. json для каждого языка, для которого вы хотите добавить перевод. Следуйте приведенным здесь инструкциям, чтобы локализовать и применить переводы.

Шаг 4. Подготовка плагина

Создайте информативное окно About для вашего плагина. Добавьте краткое описание и версию плагина, название компании-разработчика и ссылку на его сайт. Следуйте приведенным здесь инструкциям, чтобы создать вариант About в файле config.json.

Подготовьте иконки для плагина и поместите их в папку ресурсов. Следуйте приведенным здесь инструкциям, чтобы указать параметры значка в файле config.json.Обратите внимание, что вам необходимо подготовить 8 иконок для корректного отображения плагина в маркетплейсе: 4 типа масштабирования иконок (125%, 150%, 175%, 200%) как для светлой, так и для темной темы.

Не забудьте про файл readme, куда вы можете добавить подробное описание плагина, инструкции по установке и использованию, известные проблемы и т. д. Поместите этот файл в основную папку плагина.

Шаг 5. Тестирование

Поместите все подготовленные файлы в папку плагина и запушите в удаленный репозиторий. Вот и все! Теперь вы можете подключить его к десктопной или веб-версии редакторов Р7 и протестировать.

1. Протестируйте плагин в Р7 Desktop Editors

Вы можете запустить это приложение в режиме отладки с флагом *—ascdesktop-support-debug-info*. Для этого следуйте приведенным здесь инструкциям в зависимости от используемой операционной системы.

Методы плагина (справочник-описание)

Описание

window.Asc.plugin определяет объект, созданный при запуске плагина. Он имеет несколько методов, некоторые из которых являются необязательными и должны использоваться только в случае необходимости.

Методы и свойства

Название	Описание	Тип	Наличие
callCommand	Определяет метод, используемый для отправки данных обратно в редактор. Он заменяет метод executeCommand при работе с текстами, чтобы упростить синтаксис скрипта, который необходимо передать в редакцию.	функциональный	обязательно
callModule	Определяет метод, используемый для выполнения удаленного скрипта по ссылке.	функциональный	необязательно
createInputHelper	Определяет метод, используемый для создания помощника ввода.	функциональный	необязательно
executeCommand	Определяет метод, используемый для отправки данных обратно в редактор. Теперь этот метод в основном используется для работы с OLE-объектами и сохранен для использования с текстом, чтобы предыдущие версии плагина оставались совместимыми.	функциональный	обязательно
executeMethod	Определяет метод, используемый для выполнения определенных методов редактора с помощью плагина.	функциональный	необязательно
getInputHelper	Определяет метод, используемый для получения InputHelper-объекта.	функциональный	необязательно
loadModule	Определяет метод, используемый для загрузки удаленного/удаденно расположенного текстового ресурса.	функциональный	необязательно
resizeWindow	Определяет метод, используемый для изменения размера окна, обновляя минимальный/максимальный размеры.	функциональный	необязательно

Пример

```
(function(window, undefined){
```

```
  var text = "Hello world!";
  window.Asc.plugin.init = function() {
     Asc.scope.text = text;
     this.callCommand(function() {
        var oDocument = Api.GetDocument();
        var oParagraph = Api.CreateParagraph();
        oParagraph.AddText(Asc.scope.text);
        oDocument.InsertContent([oParagraph]);
     }, true);
  };
  window.Asc.plugin.button = function(id)
  {
  };
})(window, undefined);
```

executeMethod (справочник-описание всех методов и их свойств)

window.Asc.plugin.executeMethod (name, [args], callback)

Описание

Определяет метод, используемый для выполнения определенных методов редактора с помощью плагина.

Обратный вызов — это результат, который возвращает метод. Это необязательный параметр. В случае его отсутствия для возврата результата выполнения метода будет использована функция window.Asc.plugin.onMethodReturn.

Параметры

Название	Описание	Тип
name	Имя конкретного метода, который должен быть выполнен	строковый
args	Аргументы, которые должен использовать метод (если они есть).	массив
callback	Результат, который возвращает метод.	функциональный

См. доступные методы *window.Asc.plugin.executeMethod* ниже, чтобы узнать о них больше.

Методы и их свойства

Название	Описание
AddComment	Этот метод позволяет добавить комментарий к документу.
AddContentControl	Этот метод позволяет добавить в документ пустой элемент управления содержимым.

AddContentControlCheckBox	Этот метод позволяет добавить в документ пустой флажок управления содержим
AddContentControlDatePicker	Этот метод позволяет добавить в документ пустой указатель даты управления содержимым.
AddContentControlList	Этот метод позволяет добавить в документ пустой список элементов управления содержимым.
AddContentControlPicture	Этот метод позволяет добавить в документ пустое изображение элемента управления содержимым.
AddOleObject	Этот метод позволяет добавить OLE-объект в текущую позицию документа.
ChangeComment	Этот метод позволяет изменить указанный комментарий.
ChangeOleObject	Этот метод позволяет изменить OLE — объект OLE с помощью *InternalId*, указанного в данных OLE — объекта.
ChangeOleObjects	Этот метод позволяет изменить несколько OLE — объектов с помощью *InternalIds*, указанных в данных OLE — объекта.
CoAuthoringChatSendMessage	Этот метод позволяет отправить сообщение в чат соавторов.
ConvertDocument	Этот метод позволяет конвертировать документ в Markdown или HTML-текст.
EditOleObject	Этот метод позволяет изменить OLE — объект с помощью *InternalId*, указанного в данных объекта OLE.
EndAction	Этот метод позволяет указать конечное действие для длительных операций.
GetAllComments	Этот метод позволяет получить все комментарии из документа.
GetAllContentControls	Этот метод позволяет получить информацию обо всех элементах управления содержимым, которые были добавлены на страницу.
GetAllForms	Этот метод позволяет получить информацию обо всех формах, добавленных в документ.
GetAllOleObjects	Этот метод позволяет получить все данные OLE — объекта для объектов, которые могут быть открыты указанным плагином.
GetCurrentContentControl	Этот метод позволяет получить идентификатор выбранного элемента управления содержимым.
GetCurrentContentControlPr	Этот метод позволяет получить текущие свойства элемента управления содержимым.
GetFields	Этот метод позволяет получить все поля в виде текста.
GetFileHTML	Этот метод позволяет получить содержимое файла в формате HTML.
GetFileToDownload	Этот метод позволяет получить текущий файл для загрузки в указанном формате.
GetFontList	Этот метод позволяет получить список шрифтов.
GetFormsByTag	Этот метод позволяет получить информацию обо всех формах, добавленных в документ с указанным тегом.
GetFormValue	Этот метод позволяет получить значение заданной формы.
GetImageDataFromSelection	Этот метод позволяет получить данные изображения с первого из выбранных рисунков
GetInstalledPlugins	Этот метод позволяет получить все установленные плагины.
GetMacros	Этот метод позволяет получить макросы документа.

GetSelectedText	Этот метод позволяет получить выделенный текст из документа.
GetSelectionType	Этот метод позволяет получить тип текущего выделения.
GetVersion	Этот метод позволяет получить версию редактора.
InputText	Этот метод позволяет вставлять текст в документ.
InsertAndReplaceContentControls	Этот метод позволяет вставить элемент управления содержимым, который содержит данные.
InsertOleObject	Этот метод позволяет вставить объект OLE в текущую позицию документа.
InstallPlugin	Этот метод позволяет установить плагин по URL-адресу в конфигурации плагина.
MoveCursorToContentControl	Этот метод позволяет переместить курсор на указанный элемент управления содержимым.
MoveCursorToEnd	Этот метод позволяет переместить курсор в конечную позицию.
MoveCursorToStart	Этот метод позволяет переместить курсор в начальную позицию.
MoveToComment	Этот метод позволяет переместить курсор на указанный комментарий.
OnEncryption	Этот метод позволяет зашифровать документ.
OpenFile	Этот метод позволяет открыть файл с полями.
PasteHtml	Этот метод позволяет вставлять в документ текст в формате *html*.
PasteText	Этот метод позволяет вставлять текст в документ.
PutImageDataToSelection	Этот метод позволяет заменить первый выбранный рисунок изображением, указанным в параметрах.
RemoveComments	Этот метод позволяет удалить указанные комментарии.
RemoveContentControl	Этот метод позволяет удалить текущий выбранный элемент управления содержимым, сохранив все его содержимое.
RemoveContentControls	Этот метод позволяет удалить несколько элементов управления содержимым.
RemoveOleObject	Этот метод позволяет удалить OLE-объект из документа по его внутреннему идентификатору.
RemoveOleObjects	Этот метод позволяет удалить из документа несколько OLE-объектов по их внутренним идентификаторам.
RemovePlugin	Этот метод позволяет удалить плагин с указанным GUID.
RemoveSelectedContent	Этот метод позволяет удалить выбранный контент из документа.
ReplaceTextSmart	Этот метод позволяет заменить каждый абзац (или текст в ячейке) при выборе соответствующим текстом из массива строк.
SearchAndReplace	Этот метод позволяет найти и заменить текст.
SelectContentControl	Этот метод позволяет выбрать указанный элемент управления содержимым.
SelectOleObject	Этот метод позволяет выбрать указанный объект OLE.
SetDisplayModeInReview	Этот метод позволяет установить режим отображения для отслеживания изменений.
SetFormValue	Этот метод позволяет установить значение в указанную форму.
SetMacros	Этот метод позволяет установить макросы в документ.

SetProperties	Этот метод позволяет установить свойства документа.
ShowButton	Этот метод позволяет отображать или скрывать кнопки в шапке.
ShowInputHelper	Этот метод позволяет показать помощник ввода.
StartAction	Этот метод позволяет указать начальное действие для длительных операций.
UnShowInputHelper	Этот метод позволяет отменить отображение помощника ввода.
UpdatePlugin	Этот метод позволяет обновить плагин по URL-адресу в конфигурации плагина.

Часто задаваемые вопросы: Плагины и макросы

Как установить плагины к редакторам Р7?

Некоторые плагины установлены по умолчанию. Перейдите на вкладку «Плагины», чтобы просмотреть доступные. Чтобы установить дополнительные подключаемые модули, см. инструкции по установке на десктопном редакторе, в локальной среде.

Как настроить мой плагин в соответствии со стилем Р7?

Р7 предлагает собственную таблицу стилей в файле plugin.css, связанном с файлом index.html. Подробную инструкцию по подключению стилей Р7 к редакторам можно найти здесь.

Как локализовать плагин?

Чтобы локализовать плагин, вам нужно сделать следующее:

- Переведите разделы config.json.
- Локализуйте файлы index.html и код плагина.
- Применить переводы к плагину.

Как мне создать окно «О программе» для моего плагина?

Используйте варианты плагинов или подплагинов, чтобы создать окно «О программе» для вашего плагина или добавить дополнительные настройки плагина. Помимо двух вариаций конфига, нужно еще создать дополнительный файл *index_about.html*.

Запаковал папку с плагином в архив, поменял расширение на .plugin и добавил в редакторы.

Но это не работает. Что я должен делать?

Пожалуйста, убедитесь, что ваш архив плагинов не выглядит следующим образом:

Все файлы плагина и подпапки должны находиться в корне архива. Для этого сначала распакуйте папку плагина, а затем заархивируйте только ее элементы.

Могу ли я использовать макросы Microsoft Office в редакторах Р7?

Макросы Microsoft Office используют язык скриптов Visual Basic для приложений (VBA), а редакторы Р7 используют JavaScript. Но это не сложно преобразовать ваши ранее используемые

макросы в новый формат. Некоторые примеры преобразования макросов MS VBA вы можете увидеть здесь.

Можно ли сделать макрос глобальным?

Макросы привязаны к конкретным документам и нет возможности сделать их глобальными. Однако вы можете написать плагин, который будет загружаться для всех пользователей.

Примеры макросов

В следующих примерах показано, как использовать макросы Р7, и сравнить код JavaScript с кодом Microsoft Visual Basic для приложений, чтобы вы могли увидеть разницу и понять, что можно сделать для конвертации кода VBA в макросы Р7.

- Запись данных в ячейку рабочего листа

В этом примере мы записываем данные (фразу «Hello world») в четвертый столбец третьей строки рабочего листа.

- Изменить цвет фона ячейки

В этом примере мы устанавливаем синий цвет фона ячейки B3.

- Изменить цвет шрифта ячейки

В этом примере мы устанавливаем цвет шрифта ячейки B4 на красный.

- Сделать шрифт ячейки полужирным

В этом примере мы устанавливаем жирный шрифт ячейки A2.

- Объединить диапазон ячеек

В этом примере мы объединяем выбранный диапазон ячеек.

- Разъединить диапазон ячеек

В этом примере мы разъединяем выбранный диапазон ячеек.

- Установить ширину столбца

В этом примере мы устанавливаем ширину для второго («B») столбца.

- Отформатировать диапазон в виде таблицы

В этом примере мы форматируем диапазон ячеек в виде таблицы.

- Добавить диаграмму

В этом примере мы создаем диаграмму из данных в диапазоне ячеек «C5: D7».

- Выделять дубликаты

В этом примере мы выделяем дубликаты в выбранной области разными цветами, чтобы быстро распознавать дублирующиеся значения

- Следующая пуста строка

В этом примере мы находим следующую доступную пустую строку на рабочем листе.

Вставить текст

В этом примере мы вставляем текст в документ в текущем положении курсора.

Узнать подробнее

Курсы обмена валюты

В этом примере мы возвращаем информацию о курсах валют за последние несколько дней и заполняем таблицу полученными значениями.

Узнать подробнее

Структура плагина

Каждый плагин для версии сервера представляет собой папку с файлами. Чтобы использовать плагины с настольной версией, вам нужно запаковать эти файлы в один zip-архив (дополнительную информацию о том, как добавлять плагины в редакторы, см. в разделе добавление плагинов).

Папка плагина должна содержать три файла, необходимые для работы плагина:

- config.json -файл конфигурации плагина, содержащий информацию об основных данных плагина, необходимых для регистрации плагина в редакторах.
- index.html — точка входа плагина, соединяющая файлы config.json и plugin.js (базовый файл, необходимый для работы с плагинами).
- plugin code (.js file) — сам файл кода плагина, содержащий код JavaScript плагина, который вы хотите подключить к редакторам.

Папка плагина также может содержать другие файлы, такие как значки плагинов, стили, переводы, файл readme, сторонние сервисы и т. д. Дополнительную информацию можно найти здесь.

Макросы

Общее о макросах

Написание макросов

Теперь, когда вы знаете, как работают макросы, попробуйте написать свой собственный макрос. У нас есть таблица, и нам нужно «закрасить» чередующиеся строки таблицы (нечетные будут окрашены в зеленый цвет, четные станут красными). Таблица содержит 200 строк и столбцов от A до S. Вручную это заняло бы много времени. Таким образом, использование макросов будет лучшим решением этой проблемы.

1. Откройте редакторы Р7 и создайте новую таблицу.
2. Теперь откройте вкладку «Плагины» и выберите «Макросы». Появится окно макросов.

3. Щелкните Создать. Вам будет представлена базовая функция-обертка, которая позволит вам ввести необходимый код:

```
(function()
{
   // ... your code goes here ...
})();
```

4. Давайте обратимся к документации API Document Builder, чтобы узнать, что нам нужно для выполнения нашей задачи:
   - Сначала получите текущий рабочий лист с помощью метода GetActiveSheet:
     `var oWorksheet = Api.GetActiveSheet();`
   - Затем создайте цикл для запуска от первой до последней строки:
     ```
     for (var i = 1; i < 200; i += 2) {
     }
     ```
   - Установите две переменные: одну для нечетных строк, вторую для четных строк:
     `var rowOdd = i, rowEven = i + 1;`
   - Теперь, когда у нас есть доступ как к нечетным, так и к четным строкам, давайте раскрасим их в нужные цвета. Установите нужные цвета с помощью метода CreateColorFromRGB. Получите диапазон ячеек в строке с помощью метода GetRange и установите цвет для нечетных строк:
     ```
     oWorksheet.GetRange("A" + rowOdd + ":S" +
     rowOdd).SetFillColor(Api.CreateColorFromRGB(118, 190, 39));
     ```
     То же самое для четных рядов, но другим цветом:
     ```
     oWorksheet.GetRange("A" + rowEven + ":S" +
     rowEven).SetFillColor(Api.CreateColorFromRGB(186, 56, 46));
     ```

Теперь давайте подведем итог с полным кодом скрипта:

```
(function()
{
   var oWorksheet = Api.GetActiveSheet();
   for (var i = 1; i < 200; i += 2) {
      var rowOdd = i, rowEven = i + 1;
      oWorksheet.GetRange("A" + rowOdd + ":S" + rowOdd).SetFillColor(Api.CreateColorFromRGB(118,
190, 39));
      oWorksheet.GetRange("A" + rowEven + ":S" + rowEven).SetFillColor(Api.CreateColorFromRGB(186,
56, 46));
   }
})();
```

Вставьте приведенный выше код в окно макросов и нажмите **«Выполнить»**. Строки таблицы от 1 до 200 будут поочередно окрашены менее чем за секунду.

Назначение макросов

В табличном редакторе вы можете назначить макрос графическому объекту:

1. Щелкните правой кнопкой мыши графический объект.
2. Нажмите «Назначить макрос».
3. В появившемся окне выберите макрос. Вы можете ввести имя макроса в соответствующее поле.
4. Нажмите кнопку ОК.

Для запуска макроса достаточно кликнуть по графическому объекту и скрипт будет выполнен.

Начало работы с макросами

Макросы — это небольшие скрипты, используемые для облегчения вашей повседневной работы с различными типами документов. Макросы Р7 используют синтаксис JavaScript и нотацию сценариев API Document Builder.

Есть несколько причин, по которым Р7 использует JavaScript для макросов:

- кроссплатформенный,
- легко использовать,
- безопасность, так как макросы не имеют никакого доступа к системе. Это просто JS-код, который запускается в одном окне с редакторами.

Как начать писать свой макрос

Обратите внимание, что начиная с версии 7.1 доступ к объектам *окна* и *документа*, а также к функции оповещения ограничен из макросов, так как к макроскриптам был применен режим «*использовать строгий*». Не забудьте объявить переменные перед их использованием, чтобы макросы работали корректно.

Здесь вы можете найти готовые макросы или создать свои собственные.

Как начать писать свой макрос

1. Откройте вкладку «Плагины» и нажмите «Макросы».
2. Нажмите New в появившемся окне.
3. Обратитесь к документации API, чтобы написать скрипт.
4. Напишите код для своего макроса.
5. Переименуйте свой макрос, нажав соответствующую кнопку.
6. Когда все будет готово, нажмите. Выполнить, чтобы запустить код в документе.

Если вы хотите удалить ненужный макрос, нажмите кнопку Удалить.

Конвертация макросов

Конвертация макросов MS VBA

Макросы Р7 отличаются от макросов Microsoft, поскольку последние используют язык скрипта Visual Basic для приложений (VBA). JavaScript более гибкий и может использоваться с любой платформой (что важно, поскольку редакторы Р7 поддерживаются на платформах Windows, Linux и Mac OS).

Это может быть причиной некоторых неудобств, если вы ранее использовали Microsoft Office с макросами, так как они станут несовместимы с макросами Р7. Вы можете преобразовать ранее использовавшиеся макросы, чтобы использовать их с новыми редакторами.

Процесс не слишком сложный. Давайте посмотрим на следующий пример:

```
Sub Example()
   Dim myRange
   Dim result
   Dim Run As Long

   For Run = 1 To 3
      Select Case Run
      Case 1
         result = "=SUM(A1:A100)"
      Case 2
         result = "=SUM(A1:A300)"
      Case 3
         result = "=SUM(A1:A25)"
      End Select
      ActiveSheet.range("B" & Run) = result
   Next Run
End Sub
```

Макрос подсчитывает сумму значений из трех диапазонов ячеек столбца **A** и помещает результаты в три ячейки столбца **B**.

Того же самого можно добиться с помощью макросов Р7, код будет практически идентичен и прост для понимания, если вы знаете как Visual Basic для приложений, так и JavaScript:

```
(function()
{
   for (let run = 1; run <= 3; run++)
   {
      var result = "";
      switch (run)
      {
         case 1:
            result = "=SUM(A1:A100)";
            break;
         case 2:
            result = "=SUM(A1:A300)";
            break;
```

```
        case 3:
          result = "=SUM(A1:A25)";
          break;
        default:
          break;
      }
      Api.GetActiveSheet().GetRange("B" + run).Value = result;
    }
})();
```

Точно так же любой другой скрипт Visual Basic для приложений можно преобразовать в код JavaScript, совместимый с макросами Р7.

Отладка макросов

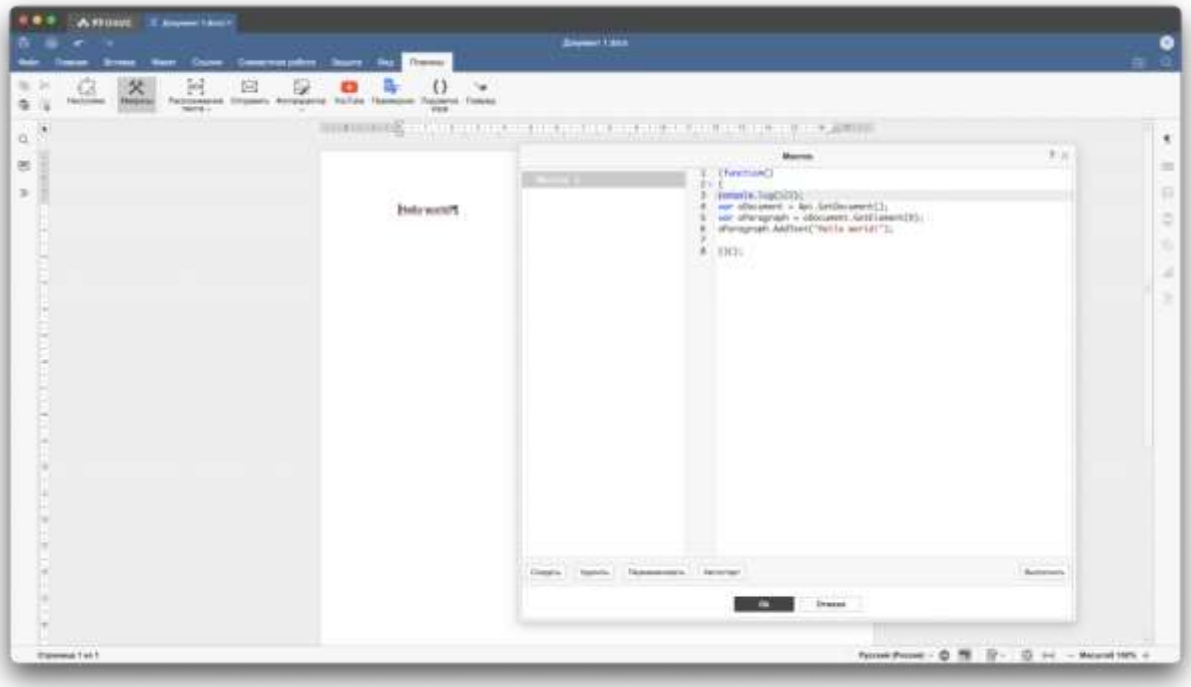
Для отладки макросов Р7 следуйте приведенным ниже инструкциям. Есть два способа запустить макрос в режиме отладки.

Вариант 1. Использование точек останова

1. Откройте вкладку «Плагины» и нажмите «Макросы».
2. Используйте метод log объекта консоли в вашем скрипте, чтобы отобразить информацию журнала в консоли отладки браузера:

```
console.log(123);
var oDocument = Api.GetDocument();
var oParagraph = oDocument.GetElement(0);
oParagraph.AddText("Hello world!");
```

3. Нажмите кнопку «Выполнить», чтобы запустить скрипт.



4. Для отладки макросов и плагинов Вы можете использовать Консоль разработчика.
   В браузере Вы можете вызвать Консоль разработчика клавишей F12.
   Для Десктоп редактора необходимо запустить приложение с параметром —ascdesktop-support-debug-info. После запуска редактора таблиц/презентаций/документов необходимо нажать клавишу F1. Лог будет выводиться на вкладке «Console»
5. Щелкните ссылку debugger:///VM(XXX) справа от строки с сообщением журнала. Файл VMXXX с вашим скриптом будет открыт.
6. Установите точку останова, щелкнув номер строки, и снова запустите скрипт.

Выполнение скрипта приостановлено, и теперь вы можете видеть текущие значения переменных, выполнять команды в консоли и т.д.

Вариант 2. С помощью команды отладчика

1. Откройте вкладку «Плагины» и нажмите «Макросы».
2. Используйте команду отладчика в своем скрипте:

```
debugger;
var oDocument = Api.GetDocument();
var oParagraph = oDocument.GetElement(0);
oParagraph.AddText("Hello world!");
```

3. Откройте консоль отладки, нажав кнопку F12.
4. Нажмите кнопку «Выполнить», чтобы запустить скрипт.

Обратите внимание, что команда отладчика будет работать, только если открыты средства разработки. В противном случае браузер его проигнорирует.

Команда отладчика работает как точка останова и приостанавливает выполнение в той точке скрипта, где эта команда вставлена.

Примеры макросов

Курсы обмена валюты

Описание

Верните информацию о курсах валют за последние несколько дней и заполните таблицу полученными значениями. Здесь представлен макрос для валютной пары USD-EUR, но вы можете получить информацию о других курсах, изменив значение переменной sCurPair («EUR_USD», «BTC_USD» и т.д.).

В этом макросе для получения информации о курсах валют используется сторонний сервис CurrencyConverterApi.com. Существует ограничение на количество запросов в час. При превышении этого лимита макрос работать не будет. Если вы хотите использовать этот макрос, то лучше зарегистрироваться на сайте сервиса и использовать свой ключ в коде макроса.

Вы можете назначить этот макрос автофигуре. При нажатии на нее выполняется макрос, таблица заполняется соответствующими данными и перестраивается соответствующий график.

```
(function()
```

```
{
   var sCurPair = "USD_EUR";

   function formatDate(d) {
      var month = '' + (d.getMonth() + 1),
         day = '' + d.getDate(),
         year = d.getFullYear();

      if (month.length < 2)
         month = '0' + month;
      if (day.length < 2)
         day = '0' + day;

      return [year, month, day].join('-');
   }

   function previousWeek(){
      var today = new Date();
      var prevweek = new Date(today.getFullYear(), today.getMonth(), today.getDate()-7);
      return prevweek;
   }

   var sDate = formatDate(previousWeek());
   var sEndDate = formatDate(new Date());
   var apiKey = 'e5ed9f0b2b3aa6f4158f';
   var sUrl = 'https://free.currconv.com/api/v7/convert?q='
         + sCurPair + '&compact=ultra' + '&date=' + sDate + "&endDate=" + sEndDate +
'&apiKey=e5ed9f0b2b3aa6f4158f';
   var xmlHttp = new XMLHttpRequest();
   xmlHttp.open("GET", sUrl, false);
   xmlHttp.send();
   if (xmlHttp.readyState == 4 && xmlHttp.status == 200) {
      var oData = JSON.parse(xmlHttp.responseText);
      for(var key in oData) {
         var sheet = Api.GetSheet("Sheet1");
         var oRange = sheet.GetRangeByNumber(0, 1);
         oRange.SetValue(key);
         var oDates = oData[key];
         var nRow = 1;
         for(var date in oDates) {
```

```
            oRange = sheet.GetRangeByNumber(nRow, 0);
            oRange.SetValue(date);
            oRange = sheet.GetRangeByNumber(nRow, 1);
            oRange.SetValue(oDates[date]);
            nRow++;
        }
    }
  }
})();
```

Вставить текст

Описание

Вставить текст в документ в текущее местоположение курсора.

```
(function()
{
  var oDocument = Api.GetDocument();
  var oParagraph = Api.CreateParagraph();
  oParagraph.AddText("Hello world!");
  oDocument.InsertContent([oParagraph]);
})();
```

Следующая пустая строка

Описание

Найдите следующую доступную пустую строку на рабочем листе. Этот макрос позволяет получить пробел в самом конце ваших данных (не между ними).

```
(function ()
{
  // Getting the active sheet
  var activeSheet = Api.ActiveSheet;
  // Minimum row index
  var indexRowMin = 0;
  // Maximum row index
  var indexRowMax = 1048576;
  // Column 'A'
  var indexCol = 0;
  // Row index for empty cell search
  var indexRow = indexRowMax;
  for (; indexRow >= indexRowMin; --indexRow) {
    // Getting the cell
```

```
    var range = activeSheet.GetRangeByNumber(indexRow, indexCol);
    // Checking the value
    if (range.GetValue() && indexRow !== indexRowMax) {
      range = activeSheet.GetRangeByNumber(indexRow + 1, indexCol);
      range.Select();
      break;
    }
  }
})();
```

Используемые методы: GetActiveSheet, GetRangeByNumber, Select

Справочный код макроса Microsoft VBA

Этот макрос VBA использовался в качестве основы.

```
Sub example()
  Range("A" & Rows.Count).End(xlUp).Offset(1).Select
End Sub
```

Выделите дубликаты

Описание

Выделите дубликаты в выбранной области разными цветами, чтобы быстро распознать повторяющиеся значения.

```
(function ()
{
  // Background color of cells with non-repeating values
  var whiteFill = Api.CreateColorFromRGB(255, 255, 255);
  // The current index of the color range
  var uniqueColorIndex = 0;
  // Color range to highlight duplicate values
  var uniqueColors = [Api.CreateColorFromRGB(255, 255, 0),
    Api.CreateColorFromRGB(204, 204, 255),
    Api.CreateColorFromRGB(0, 255, 0),
    Api.CreateColorFromRGB(0, 128, 128),
    Api.CreateColorFromRGB(192, 192, 192),
    Api.CreateColorFromRGB(255, 204, 0)];

  // Function to get color for duplicates
  function getColor(){
    // If you have chosen all the unique colors, then let's go from the beginning
    if (uniqueColorIndex === uniqueColors.length) {
```

```
            uniqueColorIndex = 0;
        }
        return uniqueColors[uniqueColorIndex++];
    }

    // Getting an active sheet
    var activeSheet = Api.ActiveSheet;
    // Getting selection on the active sheet
    var selection = activeSheet.Selection;
    // Map of values in cells with the duplicates number
    var mapValues = {};
    // All cells range
    var arrRanges = [];
    // Going through the selection
    selection.ForEach(function (range) {
        // Getting value from cell
        var value = range.GetValue();
        if (!mapValues.hasOwnProperty(value)) {
            mapValues[value] = 0;
        }
        mapValues[value] += 1;
        arrRanges.push(range);
    });
    var value;
    var mapColors = {};
    // We go through all the cells of the selection and setting the highlighting if this value is repeated
more than 1 time
    for (var i = 0; i < arrRanges.length; ++i) {
        value = arrRanges[i].GetValue();
        if (mapValues[value] > 1) {
            if (!mapColors.hasOwnProperty(value)) {
                mapColors[value] = getColor();
            }
            arrRanges[i].SetFillColor(mapColors[value]);
        } else {
            arrRanges[i].SetFillColor(whiteFill);
        }
    }
})();
```

Используемые методы: CreateColorFromRGB GetActiveSheet, GetSelection, ForEach, GetValue, SetFillColor

Справочный код макроса Microsoft VBA

Этот макрос VBA использовался в качестве основы.

```
Sub example()
  Dim xRg As Range
  Dim xTxt As String
  Dim xCell As Range
  Dim xChar As String
  Dim xCellPre As Range
  Dim xCIndex As Long
  Dim xCol As Collection
  Dim I As Long
  On Error Resume Next
  If ActiveWindow.RangeSelection.Count > 1 Then
    xTxt = ActiveWindow.RangeSelection.AddressLocal
  Else
    xTxt = ActiveSheet.UsedRange.AddressLocal
  End If
  Set xRg = Application.InputBox("please select the data range:", "Kutools for Excel", xTxt, , , , , 8)
  If xRg Is Nothing Then Exit Sub
  xCIndex = 2
  Set xCol = New Collection
  For Each xCell In xRg
    On Error Resume Next
    xCol.Add xCell, xCell.Text
    If Err.Number = 457 Then
      xCIndex = xCIndex + 1
      Set xCellPre = xCol(xCell.Text)
      If xCellPre.Interior.ColorIndex = xlNone Then xCellPre.Interior.ColorIndex = xCIndex
      xCell.Interior.ColorIndex = xCellPre.Interior.ColorIndex
    ElseIf Err.Number = 9 Then
      MsgBox "Too many duplicate companies!", vbCritical, "Kutools for Excel"
      Exit Sub
    End If
    On Error GoTo 0
  Next
End Sub
```

Добавить диаграмму

Описание

Добавить новую диаграмму в выбранный диапазон ячеек.

```
(function()
{
   Api.GetActiveSheet().AddChart("'Sheet1'!$C$5:$D$7", true, "bar", 2, 105 * 36000, 105 * 36000, 5, 2 * 36000, 1, 3 * 36000);
})();
```

Используемые методы: GetActiveSheet, AddChart.

Код ссылки макроса Microsoft VBA

```
Sub example()
   With ActiveSheet.ChartObjects.Add(Left:=300, Width:=300, Top:=10, Height:=300)
      .Chart.SetSourceData Source:=Sheets("Sheet1").Range("C5:D7")
   End With
End Sub
```

Форматировать диапазон в виде таблицы

Описание

Отформатируйте диапазон ячеек **A1:D10** в виде таблицы.

```
(function()
{
   Api.GetActiveSheet().FormatAsTable("A1:D10");
})();
```

Используемые методы: GetActiveSheet, FormatAsTable.

Код ссылки макроса Microsoft VBA

```
Sub example()
   Sheet1.ListObjects.Add(xlSrcRange, Range("A1:D10"), , xlYes).Name = "myTable1"
End Sub
```

Установить ширину столбца

Описание

Задайте ширину столбца B.

```
(function()
{
   Api.GetActiveSheet().SetColumnWidth(1, 25);
```

```
})();
```

Используемые методы: GetActiveSheet, SetColumnWidth.

Код ссылки макроса Microsoft VBA

```
Sub example()
    Columns("B").ColumnWidth = 25
End Sub
```

Разъединить диапазон ячеек

Описание

Разъединить выбранный диапазон ячеек

```
(function()
{
    Api.GetActiveSheet().GetRange("C5:D10").UnMerge();
})();
```

Используемые методы: GetActiveSheet, GetRange, UnMerge.

Код ссылки макроса Microsoft VBA

```
Sub example()
    Range("C5:D10").UnMerge
End Sub
```

Объединить диапазон ячеек

Описание

Объединить выбранный диапазон ячеек.

```
(function()
{
    Api.GetActiveSheet().GetRange("A1:B3").Merge(true);
})();
```

Используемые методы: GetActiveSheet, GetRange, Merge

Справочный код макроса Microsoft VBA

```
Sub example()
    Range("A1:B3").Merge
End Sub
```

Сделать шрифт ячейки полужирным

Описание

Установите жирный шрифт в ячейке A2.

```
(function()
{
   Api.GetActiveSheet().GetRange("A2").SetBold(true);
})();
```

Используемые методы: GetActiveSheet, GetRange, SetBold

Справочный код макроса Microsoft VBA

```
Sub example()
   Range("A2").Font.Bold = True
End Sub
```

Изменить цвет шрифта в ячейки

Описание

Установите цвет шрифта ячейки B4 на красный.

```
(function()
{
   Api.GetActiveSheet().GetRange("B4").SetFontColor(Api.CreateColorFromRGB(255, 0, 0));
})();
```

Используемый метод: GetActiveSheet, GetRange, SetFontColor

Справочный код макроса Microsoft VBA

Используемые методы: GetActiveSheet, GetRange, SetFontColor.

Код ссылки? макроса Microsoft VBA

```
Sub example()
   Range("B4").Font.Color = RGB(255, 0, 0)
End Sub
```

Изменить цвет фона ячейки

Описание

Установите цвет фона ячейки B3 на синий.

```
(function()
{
   Api.GetActiveSheet().GetRange("B3").SetFillColor(Api.CreateColorFromRGB(0, 0, 250));
})();
```

Используемые методы: GetActiveSheet, GetRange, SetFillColor, CreateColorFromRGB.

Справочный код макроса Microsoft VBA

```
Sub example()
   Range("B3").Interior.Color = RGB(0, 0, 250)
End Sub
```

Записать данные в ячейку рабочего листа

Описание

Запишите данные (фразу Hello world) в третий столбец четвертой строки рабочего листа.

```
(function()
{
   Api.GetActiveSheet().GetRange("C4").SetValue("Hello world");
})();
```

Используемые методы: GetActiveSheet, GetRange, SetValue.

Справочный код макроса Microsoft VBA

```
Sub example()
   Cells(3, 4)="Hello world"
End Sub
```

Управление настройками редактора для макросов

Запрет просмотра, создания и запуска макросов в документах

Внимание! Приведенные ниже рекомендации могут привести к утере работоспособности редакторов и может потребоваться их переустановка! Изменения следует проводить с обязательным резервным сохранением файлов, которые подвергаются модификации, в исходном виде. Для приведенных ниже изменений текущий пользователь должен обладать административными привилегиями.

Вызов диалогового окна с перечнем уже существующих макросов в документе с возможностью их запуска, а также инструмент по созданию новых макросов вызывается при помощи кнопки «Макросы» на вкладке «Плагины».

Данную кнопку можно удалить из интерфейса редакторов, для этого удалите папку «{E6978D28-0441-4BD7-8346-82FAD68BCA3B}», которая расположена по адресу:

ОС Windows — C:\Program Files\R7-Office\Editors\editors\sdkjs-plugins

Или C:\Program Files (x86)\R7-Office\Editors\editors\sdkjs-plugins

ОС Linux — ./opt/r7-office/desktopeditors/editors/sdkjs-plugins

ОС Mac — /Applications/R7-Office.app/Contents/Resources/editors/sdkjs-plugins

Перезагрузите редакторы Р7-Офис, после старта приложения кнопка «Макросы» будет отсутствовать.

Запрет установки и удаления дополнительных плагинов

Внимание! Приведенные ниже рекомендации могут привести к утере работоспособности редакторов и может потребоваться их переустановка! Изменения следует проводить с обязательным резервным сохранением файлов, которые подвергаются модификации, в исходном виде. Для приведенных ниже изменений текущий пользователь должен обладать административными привилегиями.

Пользователь десктоп-редакторов Р7-Офис может устанавливать дополнительные плагины и удалять их (за исключением плагинов, которые были установлены вместе с приложением).

Вызов диалогового окна для управления плагинами осуществляется при нажатии на кнопку «Настройки» по вкладке «Плагины».

Кнопку управления установленными плагинами можно удалить из интерфейса редакторов, при этом все уже установленные в редакторы плагины останутся установленными. Для этого, удалите папку «{8D67F3C5-7736-4BAE-A0F2-8C7127DC4BB8}», которая расположена по адресу:

ОС Windows — C:\Program Files\R7-Office\Editors\editors\sdkjs-plugins

ОС Linux — ./opt/r7-office/desktopeditors/editors/sdkjs-plugins

ОС Mac — /Applications/R7-Office.app/Contents/Resources/editors/sdkjs-plugins

Перезагрузите редакторы Р7-Офис, после старта приложения кнопка «Настройки» будет отсутствовать.

Установите запрет обычным пользователям на запись в папку с персональными плагинами в профиле пользователя:

ОС Windows — C:\Users\%username%\AppData\Local\R7-Office\Editors\data\sdkjs-plugins

ОС Linux — /home/%username%/.local/share/r7-office/editors/sdkjs-plugins

ОС Mac — /Users/%username%/Library/Application Support/ru.nkt.r7-office-signature/data/sdkjs-plugins

Запрет автозапуска и изменения настройки режима автозапуска

Внимание! Приведенные ниже рекомендации могут привести к утере работоспособности редакторов и может потребоваться их переустановка! Изменения следует проводить с обязательным резервным сохранением файлов, которые подвергаются модификации, в исходном виде. Для приведенных ниже изменений текущий пользователь должен обладать административными привилегиями.

Откройте в текстовом редакторе файл api.js, расположенный в папке:

Windows — C:\Program Files\R7-Office\Editors\editors\web-apps\apps\api\documents\api.js

или C:\Program Files (x86)\R7-Office\Editors\editors\web-apps\apps\api\documents\api.js

ОС Linux — /opt/r7-office/desktopeditors/editors/web-apps/apps/api/documents/api.js

ОС Mac — /Applications/R7-Office.app/Contents/Resources/editors/web-apps/apps/api/documents/api.js

Начиная со строки 780, приведены настройки запуска редакторов:

```
DocsAPI.DocEditor.defaultConfig = {
    type: 'desktop',
    width: '100%',
    height: '100%',
    editorConfig: {
        lang: 'en',
        canCoAuthoring: true,
        customization: {
            about: true,
            feedback: false
        }
    }
};
```

Внутри блока customization можно добавить следующие параметры:

Параметр	Возможные значения	Назначение
macros	true | false Значение по умолчанию — true	Разрешает (true) или запрещает (false) пользователям менять настройки режима автозапуска макросов. При установке в значение false, пункт «Настройки макросов» в диалоговом окне «Дополнительные настройки» не отображается, режим автозапуска устанавливается в значение «Отключить все»

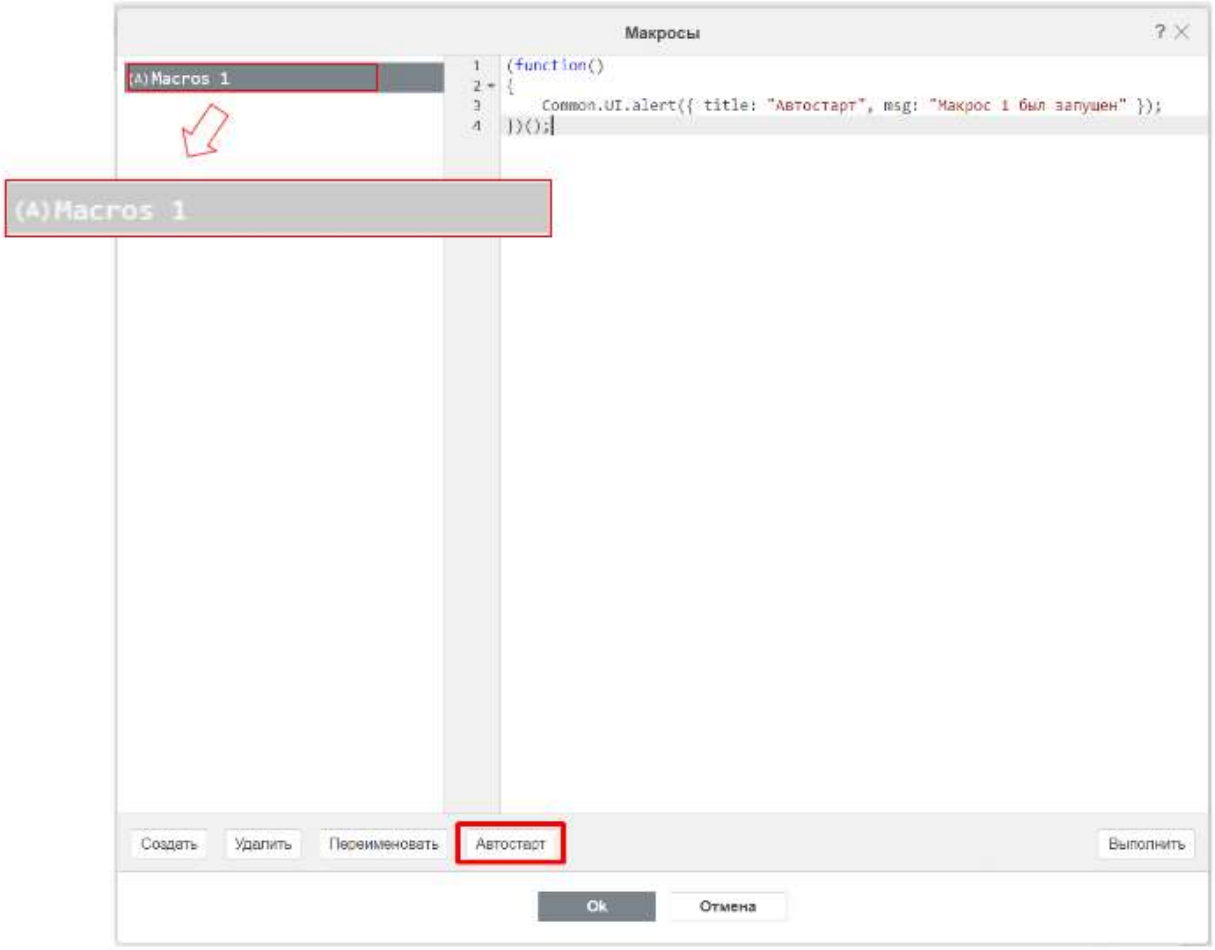
Например, для запрета автозапуска макросов и изменения пользователем настройки в диалоговом окне «Дополнительные настройки» приведите описание объекта defaultConfig к виду:

```
DocsAPI.DocEditor.defaultConfig = {
    type: 'desktop',
    width: '100%',
    height: '100%',
    editorConfig: {
        lang: 'ru',
        canCoAuthoring: true,
        customization: {
            about: true,
            feedback: false,
            macros: false
        }
    }
};
```

После сохранения изменений перезапустите редакторы Р7-Офис.

Режим автозапуска макросов документов

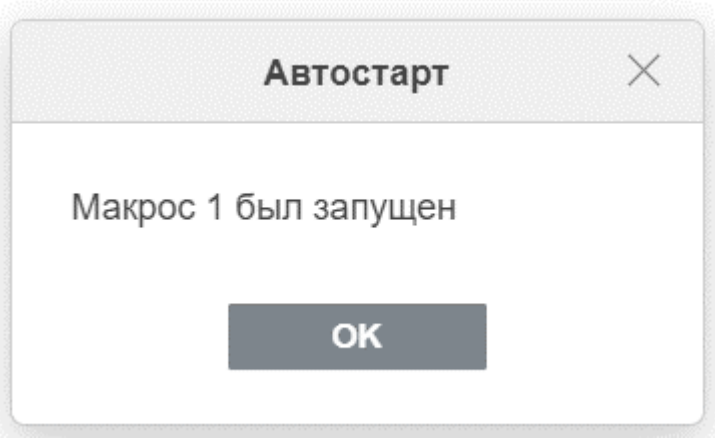
Создаваемым в Р7-Офис макросам можно установить признак «Автостарт», при этом отмеченные макросы могут автоматически запускаться при открытии документа.




```
(function()
```

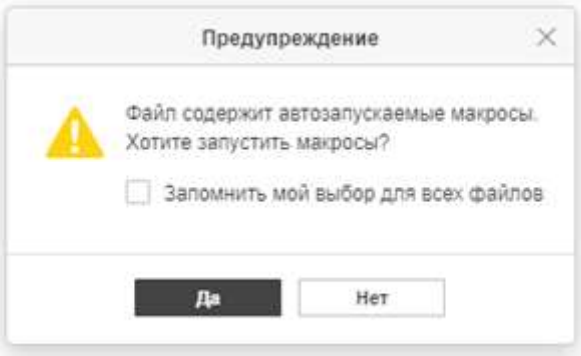
```
{
    Common.UI.alert({ title: "Автостарт", msg: "Макрос 1 был запущен" });
})();
```

При следующем открытии документа, все макросы с установленным признаком (Символ A в круглых скобкам перед наименованием) «Автостарт» будут автоматически запущены (на скриншоте отображен пример выполнения макроса, вызывающего диалоговое окно-уведомление):

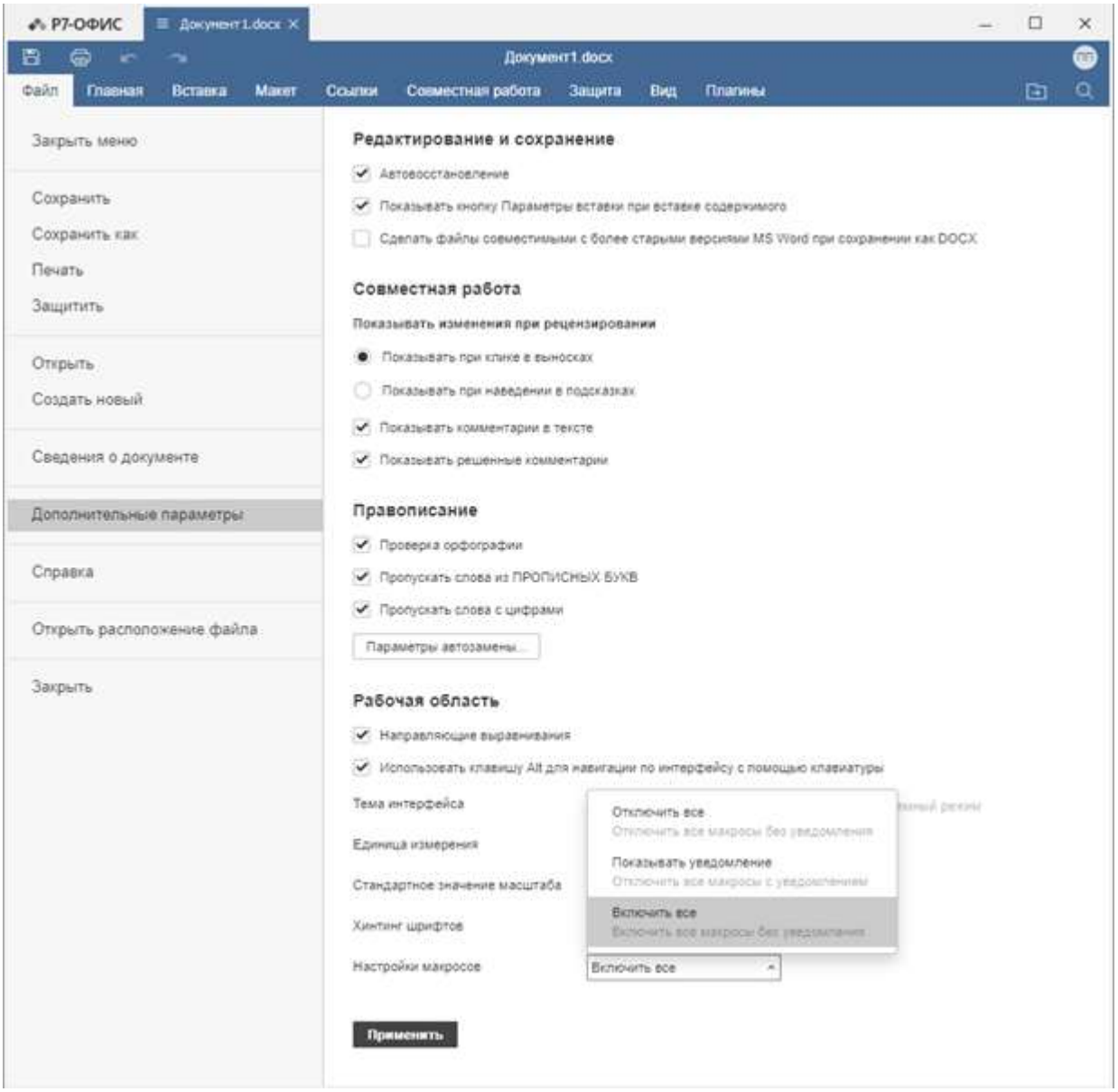


Поведение редакторов по автоматическому запуску содержащихся в документах макросов можно регулировать, выбрав одну из настроек:

- **Включить все** (включить автозапуск всех макросов без уведомления),
- **Показывать уведомления** (перед автозапуском макросов отображается системный диалог, предупреждающий о наличии в документе макросов, и предлагающий разрешить/запретить их запуск) (см. скриншот ниже),
- **Отключить все** (отключить автозапуск всех макросов без уведомления)



Данная настройка расположена в «Дополнительных параметрах», для перехода в которые нужно открыть вкладку «Файл» в любом документе:



Инструкция по отключению возможности установки пароля на документы

Внимание! Приведенные ниже рекомендации могут привести к утере работоспособности редакторов и может потребоваться их переустановка! Изменения следует проводить с обязательным резервным сохранением файлов, которые подвергаются модификации, в исходном виде. Для приведенных ниже изменений текущий пользователь должен обладать административными привилегиями.

Откройте в текстовом редакторе файл api.js, расположенный в папке:

Windows — C:\Program Files\R7-Office\Editors\editors\web-apps\apps\api\documents\api.js

Или

ОС Linux — /opt/r7-office/desktopeditors/editors/web-apps/apps/api/documents/api.js

ОС Mac — /Applications/R7-Office.app/Contents/Resources/editors/web-apps/apps/api/documents/api.js

В строке 64, приведены настройки возможности установки в редакторах:

```
«protect: <can protect document> // default = true. show/hide protect tab or protect buttons»
```

После блока editorConfig (строка 801) нужно добавить параметр:

Параметр	**Возможные значения**	**Назначение**
protect	true | false Значение по умолчанию — true	Разрешает (true) или запрещает (false) пользователям устанавливать пароль на документы (файлы). При установке в значение false, пункт «Защитить» в диалоговом окне «Файл» и пункт «Шифровать» в диалоговом окне «Защита» не отображается.

Например

```
document: {
   permissions: {
      protect: false,
      },
```

```
797     DocsAPI.DocEditor.defaultConfig = {
798         type: 'desktop',
799         width: '100%',
800         height: '100%',
801         editorConfig: {
802             lang: 'en',
803             createUrl: 'desktop://create.new', // R7 - Anakhaev D
804             isSupportCreateNew: true, // R7 - Anakhaev D
805             canCoAuthoring: true,
806             customization: {
807                 about: true,
808                 feedback: false,
809             },
810             templates: [{
811                 title: 'Новый шаблон',
812                 image: '',
813                 url: 'create:new'
814             }],  // R7 - Anakhaev D
815         },
816         document: {
817             permissions: {
818                 protect: false,
819                 },
820         }
821     };
```

После сохранения изменений перезапустите редакторы Р7-Офис.

Плагины

Добавление плагинов

P7 серверные решения

Добавление плагинов в локальную версию P7 Document Server

Добавить плагины можно тремя способами: через папку sdkjs-plugins, через файл config.json или через менеджер плагинов.

Добавление плагинов через папку sdkjs-plugins

Поместите папку с кодом плагина в папку P7 Document Server. Путь к папке зависит от используемой операционной системы:

- **Для Linux** — /var/www/P7/documentserver/sdkjs-plugins/
- **Для Windows** — %ProgramFiles%\P7\DocumentServer\sdkjs-plugins\

Плагины будут доступны всем пользователям P7 Document Server в локальной среде. В некоторых случаях требуется перезапуск сервиса.

- Для отладки запустите P7 Document Server вместе с общей папкой sdkjs-plugins:
- docker run -itd -p 80:80 -v /absolutly_path_to_work_dir:/var/www/P7/documentserver/sdkjs-plugins/plugin P7/documentserver-ee:latest

Добавление плагинов через файл config.json

В конфигурацию P7 Document Server добавьте относительный путь к файлу *config.json* созданного плагина в параметр *plugins.pluginsData*:

```
var docEditor = new DocsAPI.DocEditor("placeholder", {
  "editorConfig": {
    "plugins": {
      "autostart": [
        "asc.{0616AE85-5DBE-4B6B-A0A9-455C4F1503AD}",
        "asc.{FFE1F462-1EA2-4391-990D-4CC84940B754}",
        ...
      ],
      "pluginsData": [
        "https://example.com/plugin1/config.json",
        "https://example.com/plugin2/config.json",
        ...
      ]
    },
    ...
  },
```

```
  ...
});
```

Где

example.com — это имя сервера, на котором установлены менеджер документов и служба хранения документов, а также размещены плагины.

https://example.com/plugin1/config.json — путь к плагину

Если в аонфигурации уже есть тестовый пример, замените следующую строку /etc/P7/documentserver-example/local.json на путь к нужной конфигурации плагина.

См. документацию P7 Document Server API для получения дополнительной информации о том, где найти конфигурацию и что и как там можно изменить.

- В файле index.html всегда есть ссылка по умолчанию на файл pluginBase.js. Не забудьте добавить его в папку плагина.
- Если плагины, помещенные в папку, и плагины, указанные в конфигурации, совпадают, то будут использованы последние.
- Если вы хотите загрузить плагин на серверы S3 или Nginx, вам необходимо разрешить междоменные запросы с адреса вашего сервера документов. Или вы можете просто позволить всем использовать ваш плагин:

```
add_header 'Access-Control-Allow-Origin' '*';
add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';
```

Удаление плагинов из P7 Document Server on-premises

Вы можете удалить плагины тремя способами:

Вариант 1. Перейдите в папку sdkjs-plugins и удалите соответствующую папку плагина из этого каталога.

Путь к папке зависит от используемой операционной системы:

Для Linux — */var/www/P7/documentserver/sdkjs-plugins/*

Для Windows — *%ProgramFiles%\P7\DocumentServer\sdkjs-plugins*

Вариант 2. Отредактируйте конфигурацию P7 Document Server, удалив соответствующий плагин. Плагин станет недоступен для всех пользователей портала при следующем запуске редактора:

```
var docEditor = new DocsAPI.DocEditor("placeholder", {
  "editorConfig": {
    "plugins": {
      "autostart": [
        "asc.{0616AE85-5DBE-4B6B-A0A9-455C4F1503AD}",
        "asc.{FFE1F462-1EA2-4391-990D-4CC84940B754}",
        ...
```

```
        ],
        "pluginsData": [
          "https://example.com/plugin1/config.json",
          "https://example.com/plugin2/config.json",
          ...
        ]
      },
      ...
    },
    ...
});
```

Плагины будут удалены для всех пользователей. В некоторых случаях требуется перезапуск службы.

В таблице ниже приведены руководства по плагинам по умолчанию, чтобы вам было проще их найти:

Руководство по плагинам по умолчанию

Фото-редактор	{07FD8DFA-DFE0-4089-AL24-0730933CC80A}
Настройки	{8D67F3C5-7736-4BAE-A0F2-8C7127DC4BB8}
Распознавание текста	{440EBF13-9B19-4BD8-8621-05200E58140B}
Главред	{B631E142-E40B-4B4C-90B9-2D00222A286E}
Подсветка кода	{BE5CBF95-C0AD-4842-B157-AC40FEDD9841}
Макросы	{E6978D28-0441-4BD7-8346-82FAD68BCA3B}
Речь	{D71C2EF0-F15B-47C7-80E9-86D671F9C595}

Обратите внимание, что после обновления приложения снова появятся плагины по умолчанию, и вам нужно будет их удалить.

Вариант 3. Чтобы удалить только что добавленный плагин, выполните следующие действия:

1. Откройте вкладку «Плагины».
2. Зайдите в менеджер плагинов.
3. Нажмите кнопку «Удалить» рядом с соответствующим плагином.

Р7 Десктопные редакторы

Добавление плагинов в десктопные редакторы Р7

Добавить плагины можно двумя способами: через менеджер плагинов и через папку плагинов.

Добавление плагинов через менеджер плагинов

Шаг 1. Запакуйте все файлы плагины в папке плагина в zip-архив и измените его расширение на .plugin (все файлы плагина и подпапки должны быть в корне архива).

Шаг 2. Нажмите кнопку «Настройки» на вкладке «Плагины».

Шаг 3. В окне «Настройки плагина» нажмите кнопку «Добавить плагин», чтобы загрузить созданный архив:

Плагин будет добавлен в открытые редакторы, и все редакторы, которые вы откроете после слов, будут отображаться на вкладке плагинов.

Добавление плагинов через папку плагинов

Шаг 1. Создайте новую папку в каталоге sdkjs-plugins. Путь к папке зависит от используемой операционной системы:

Для Linux — */opt/P7/desktopeditors/editors/sdkjs-plugins/*

Для Windows — *%ProgramFiles%\P7\DesktopEditors\sdkjs-plugins*

Для Mac OS — */Applications/P7.app/Contents/Resources/editors/sdkjs-plugins/*

Шаг 2. Используйте GUID плагина в качестве имени папки. Вы можете найти GUID плагина в файле config.json.

Например, для плагина **Расширенные комментарии** параметр guid будет выглядеть следующим образом:

```
{
   "name" : "Extended comments",
   "guid" : "asc.{91EAC419-EF8B-440C-A960-B451C7DF3A37}",
   ...
}
```

Итак, имя папки этого плагина будет {91EAC419-EF8B-440C-A960-B451C7DF3A37}.

Шаг 3. Поместите все файлы плагина в эту папку:

Шаг 4. Запустите десктопные редакторы P7. Если все сделано правильно, плагин отобразится на вкладке «Плагины»:

Удаление плагинов из десктопных редакторов P7

Чтобы удалить недавно добавленные плагины, выполните следующие действия:

1. Откройте вкладку «Плагины».
2. Зайдите в настройки плагина.
3. Нажмите кнопку «Удалить» рядом с соответствующим плагином.

Эта опция недоступна для плагинов по умолчанию и плагинов, добавленных через папку плагинов. Если вам нужно удалить плагины по умолчанию, перейдите в папку sdkjs-plugins и удалите соответствующую папку плагинов из этого каталога.

Обратите внимание, что удаление подключаемых модулей по умолчанию из десктопных редакторов P7 в Mac OS нарушает целостность пакета и может привести к сбою приложения. Будьте осторожны и не делайте этого без крайней необходимости.

В таблице ниже приведены руководства по плагинам по умолчанию, чтобы вам было проще их найти:

Обратите внимание, что удаление подключаемых модулей по умолчанию из десктопных редакторов Р7 в Mac OS нарушает целостность пакета и может привести к сбою приложения. Будьте осторожны и не делайте этого без крайней необходимости.

В таблице ниже приведены руководства по плагинам по умолчанию, чтобы вам было проще их найти:

Руководство по плагинам по умолчанию

Фото-редактор	{07FD8DFA-DFE0-4089-AL24-0730933CC80A}
Настройки	{8D67F3C5-7736-4BAE-A0F2-8C7127DC4BB8}
Распознавание текста	{440EBF13-9B19-4BD8-8621-05200E58140B}
Главред	{B631E142-E40B-4B4C-90B9-2D00222A286E}
Отправить	{B509123E-6335-40BD-B965-91EB799346E3}
Подсветка кода	{BE5CBF95-C0AD-4842-B157-AC40FEDD9841}
Макросы	{E6978D28-0441-4BD7-8346-82FAD68BCA3B}

Обратите внимание, что после обновления приложения снова появятся плагины по умолчанию, и вам нужно будет их удалить.

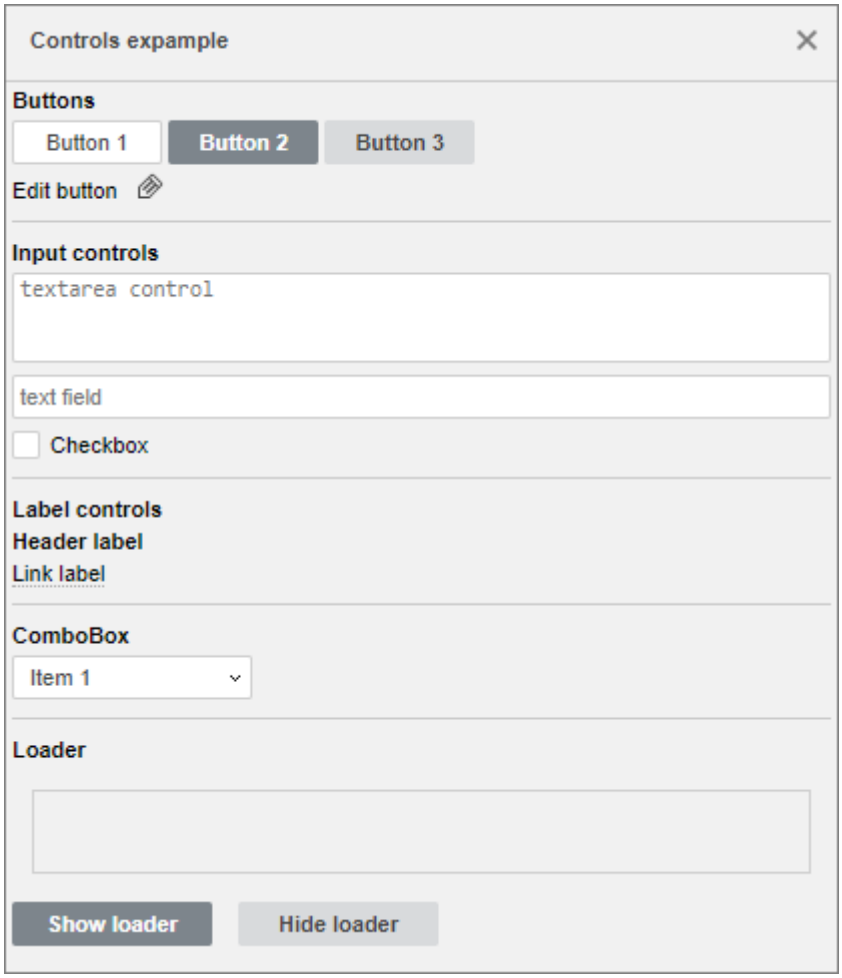
Общее

Стили плагинов

Р7 предоставляет таблицу стилей для различных элементов интерфейса. Чтобы настроить интерфейс вашего плагина в стиле Р7, подключите plugin.css к файлу index.html по следующей ссылке:

```
<link rel="stylesheet" href="https://support.r7-office.ru/wp-content/uploads/2023/05/plugins.css">
```

Все доступные элементы управления отображаются в примере плагина Controls:All the available controls are displayed within the Controls example plugin:



Кнопки

1. Используйте класс *btn-text-default,* чтобы добавить **кнопку 1** в ваш плагин
Этот класс имеет следующие параметры:

```
.btn-text-default {
   background:#fff;
   border:1px solid #cfcfcf;
   border-radius:2px;
   color:#444444;
   font-size:11px;
   font-family:"Helvetica Neue", Helvetica, Arial, sans-serif;
   height:22px;
   cursor: pointer;
}
```

Button 1

2. Используйте класс *btn-text-default.submit.primary*, чтобы добавить **кнопку 2** в свой плагин:

```
<button class="btn-text-default submit primary" style="width:75px;">Button 2</button>
```

Этот класс имеет следующие параметры:

```
.btn-text-default.submit.primary {
   color:#fff;
   background-color:#7d858c;
}
```

3. Используйте класс *btn-text-default.submit*, чтобы добавить **кнопку 3** в свой плагин:

```
<button class="btn-text-default submit" style="width:75px;">Button 3</button>
```

Этот класс имеет следующие параметры:

```
.btn-text-default.submit {
   font-weight: bold;
   background-color:#d8dadc;
   border:1px solid transparent;
}
```

Цвет кнопки выбирает разработчик. Чем темнее кнопка, тем она важнее. Как правило, для подтверждения действия и отправки результата используется кнопка класса *btn-text-default.submit.primary* (например, кнопка **Ok**).

4. Используйте класс *btn-edit,* чтобы добавить кнопку «Редактировать» в свой плагин:

```
<label class="for-combo">Edit button</label><div class="btn-edit" style="display: inline-block; margin-left: 10px;"></div>
```

Этот класс имеет следующие параметры:

```
.btn-edit {
   width:13px;
   height:13px;
   cursor: pointer;
   background-
image:url('
IWXMAAAsTAAALEwEAmpwYAAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAGZSURBVHgBfZI/y4
FRGMavx7+SRQaTTQab74CVlBKL/FukDGQhEgsDNh/Apiw+gcXm70DJoEikKMUk7vec8/Yi75O7Tj2d+/4913
Wuc6Tz+UyQqev1itvtBr1e/6+nkgP2+z0qlYr4DgaDsNls36HtdotisYhoNAqLxYJyuSz230HFO7DZbISC0+lEp9
OBRqNBLpdDq9XCeDx+DfIz8TWZTIhZodFoRMvlknw+H8XjcdrtarRarYgpU6/XE7DtgcRzgcRq1Ww3A4FKparRbspyJRo9H4G4TD4R
9tNbICOxyOVSiVuVzU7/epXq9TIBAQtrkzxeVygclQrfbhd1uRzgcRq1Ww3A4FKparRbspyJRo9H4G4TD4R
```

```
D06XQS3pkt8nq9NJ1OiSVGsVjsqfC3nvekVCrxeDxgMBgQiUTEa2g2m8hkMi8FuXtSq9VIJBK43+8iHB7GJ8B
L4vY+N3U6HQqFAsxmM+TqB5Je/SVNoN18AAAAAElFTkSuQmCC');
}
```

Элементы управления вводом

1. Используйте элемент формы *textarea*, чтобы добавить **текстовую область** в ваш плагин:

```
<textarea style="height:45px;width: 100%;" class="form-control" placeholder="textarea
control"></textarea>
```

Этот класс имеет следующие параметры:

```
textarea.form-control {
   resize: none;
}
```

2. Используйте элемент формы ввода с текстовым типом, чтобы добавить **текстовое поле** в ваш плагин:

```
<input type="text" class="form-control" placeholder="text field" style="width: 100%;margin-bottom:
2px;">
```

Этот класс имеет следующие параметры:

```
.form-control {
   border:1px solid #cfcfcf;
   border-radius:2px;
   box-sizing: border-box;
   color:#444444;
   font-size:11px;
   height:22px;
   padding:1px 3px;
   -webkit-box-shadow: none;
   box-shadow: none;
   -webkit-user-select: text;
   -moz-user-select: text;
   -ms-user-select: text;
   user-select: text;
}
```

3. Используйте элемент формы *ввода* с типом флажка, чтобы добавить **флажок** в ваш плагин:

```
<input type="checkbox" class="form-control" style="vertical-align: middle;"><label style="margin-left: 5px;vertical-align: middle;">Checkbox</label>
```

Этот класс имеет следующие параметры:

```
input[type='checkbox '].form-control {
   height: auto;
   margin:0;
}
```

Элементы управления ярлыками

1. Используйте класс *label.header*, чтобы добавить **жирный заголовок** к вашему плагину:

```
<label class="header">Header label</label>
```

Этот класс имеет следующие параметры:

```
label.header {
   font-weight: bold;
}
```

2. Используйте класс *label.link*, чтобы добавить ссылку на ваш плагин:

```
<label class="link">Link label</label>
```

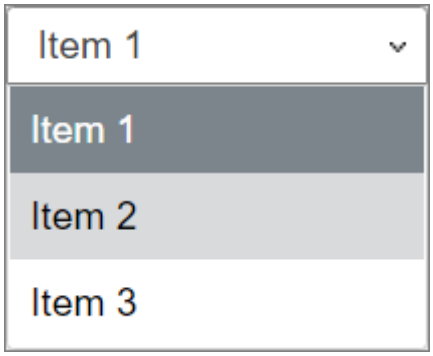
Этот класс имеет следующие параметры:

```
label.link {
   border-bottom:1px dotted #aaa;
   cursor: pointer;
}
```

Поле со списком

Используйте функцию select2, чтобы добавить **поле со списком** в ваш плагин:

```
<select id="select_example" class="" ></select>
$('#select_example').select2({
   data : [{id:0, text:'Item 1'}, {id:1, text:'Item 2'}, {id:2, text:'Item 3'}],
   minimumResultsForSearch: Infinity,
   width : '120px'
});
```



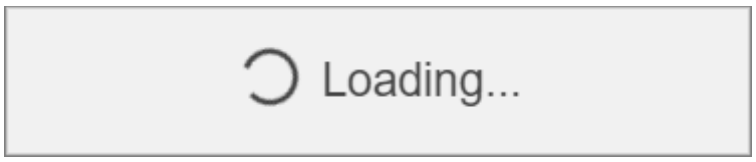
Loader

Используйте класс *asc-loader-container*, чтобы добавить **контейнер загрузчика** в ваш плагин:

```
<div id="loader-container" class="asc-loader-container" style="margin: 10px; height: 40px; border: 1px solid #cfcfcf;"></div>
```

Этот класс имеет следующие параметры:

```
.asc-loader-container {
   position: relative;
}
```



Пример

Давайте посмотрим, как добавить элементы интерфейса в стиле P7 в плагин YouTube

1. Чтобы добавить простую белую кнопку **OK**, используйте класс *btn-text-default* (ширина кнопки — 30 пикселей):

```
<button class="btn-text-default" style="width:30px;">OK</button>
```

2. Для добавления поля используйте класс *form-control* (ширина формы — 100%, т.е. подстраивается под ширину родительского элемента):

```
<input type="text" class="form-control" style="width:100%;">
```

3. Чтобы добавить темно-серую кнопку **Ok**, используйте класс *btn-text-default.submit.primary* (ширина кнопки — 90 пикселей):

```
<button class="btn-text-default submit primary" style="width:90px;">Ok</button>
```

4. Чтобы добавить светло-серую кнопку «**Отмена**», используйте класс *btn-text-default.submit* (ширина кнопки — 90 пикселей):

```
<button class="btn-text-default submit" style="margin-left:5px; width:90px;">Cancel</button>
```

Локализация плагина

Локализация плагинов на ваш язык

Все плагины созданы на английском языке по умолчанию. Если вы хотите, чтобы они были доступны на вашем языке, вы можете добавить к ним переводы.

Перевод разделов config.json

Сначала вы можете перевести файл config.json. Для этого откройте его и найдите все строки на английском языке. Обычно это узлы name, variations.description и variations.buttons.text (название параметров) объекта конфигурации.

Добавьте новые узлы с именем ключа и Региональными настройками, равными объекту, который будет иметь локаль языка в качестве ключа и перевод в качестве значения. Например, для ключа name объект локализации будет выглядеть так:

```
"name": "Highlight code",
"nameLocale": {
   "de": "Code hervorheben",
   "es": "Resaltar el código",
   "fr": "Code en surbrillance",
   "ru": "Подсветка кода"
}
```

Полные переводы в config.json для плагина подсветки кода будут выглядеть следующим образом:

```
{
   "name": "Highlight code",
   "nameLocale": {
      "de": "Code hervorheben",
      "es": "Resaltar el código",
      "fr": "Code en surbrillance",
      "ru": "Подсветка кода"
   },
   ...,
   "variations": [
      {
         "description": "Highlight code",
         "descriptionLocale": {
            "de": "Code hervorheben",
            "es": "Resaltar el código",
            "fr": "Code en surbrillance",
            "ru": "Подсветка кода"
         },
         ...,
         "buttons": [
```

```
        {
          "text": "Cancel",
          "textLocale": {
            "de": "Abbrechen",
            "es": "Cancelar",
            "fr": "Annuler",
            "ru": "Отмена"
          },
          ...
        }
      ],
      ...
    }
  ]
}
```

Локализация

Найдите все строки, которые вы хотите локализовать, из файлов index.html и pluginCode.js и создайте их список. Затем создайте папку translations в каталоге плагина, чтобы структура выглядела следующим образом:

```
..
[translations]
config.json
index.html
pluginCode.js
```

Создайте языковые файлы .json для каждого языка, для которого вы хотите добавить переводы, с четырехбуквенным кодом языка для его имени (например, de-DE.json). Эти файлы будут содержать объекты со словами и фразами исходного (английского) языка в качестве ключей и переводами на выбранный язык в качестве значений. Вот пример перевода с немецкого:

```
{
  "Language": "Sprache",
  "Highlight": "Hervorheben",
  "Style": "Stil",
}
```

Начиная с версии 7.2, вы можете добавить файл langs.json в папку translations. Он содержит имена файлов с переводами на определенный язык в формате массива:

```
[
  "cs-CZ",
  "de-DE",
  "es-ES",
  "fr-FR",
```

```
  "ru-RU"
]
```

Сначала будет запрошен файл langs.json, и будет выполнен поиск полного совпадения языка и имени файла. Если совпадений не найдено, проверяются первые два символа до знака «-». Если в файле langs.json нет нужного имени файла, плагин будет работать на английском языке. Если файла langs.json нет или есть проблемы при его парсинге, то переводы будут работать по старой схеме перевода.

После добавления всех файлов локализации файловая структура плагина будет выглядеть так:

```
..
[translations]
  de-DE.json
  es-ES.json
  fr-FR.json
  langs.json
  ru-RU.json
config.json
index.html
pluginCode.js
```

Теперь вы можете заменить строки в файлах их переведенными значениями.

Применение переводов к плагину

Чтобы применить переводы, вам нужно будет добавить уникальные идентификаторы к каждому элементу, который имеет строковое значение, которое вы хотите локализовать. Например, вы хотите локализовать кнопку «Создать» в этой части кода:

```
<button>New</button>
```

Добавьте к нему атрибут id, чтобы он выглядел так:

```
<button id="button_new">New</button>
```

После этого добавьте функцию window.Asc.plugin.onTranslate в файл pluginCode.js:

```
window.Asc.plugin.onTranslate = function()
  {
    var label = document.getElementById("button_new");
    if (label)
      label.innerHTML = window.Asc.plugin.tr("New");
  }
```

Функция *window.Asc.plugin.onTranslate* будет вызываться сразу после запуска плагина и позже в случае изменения языка плагина.

Если вам нужно локализовать более одного слова/фразы, функция *window.Asc.plugin.onTranslate* может иметь следующий вид:

window.Asc.plugin.onTranslate = function()

```
  {
    document.getElementById("button_delete").innerHTML = window.Asc.plugin.tr("Delete");
    document.getElementById("button_new").innerHTML    = window.Asc.plugin.tr("New");
    document.getElementById("button_rename").innerHTML = window.Asc.plugin.tr("Rename");
    document.getElementById("button_run").innerHTML    = window.Asc.plugin.tr("Run");
  }
```

Где каждая строка будет представлять локализованный элемент, доступ к которому осуществляется с использованием соответствующего идентификатора.

Обратите внимание, что в переводе используется метод .innerHTML, поэтому он может содержать не только слова и фразы, но и некоторые элементы HTML (теги, ссылки и т. д.). Не забывайте экранировать любые кавычки в переводе (как и в любом .json файле), чтобы они работали корректно.

Теперь, когда редакторы запущены, текущий язык интерфейса будет использоваться для определения того, имеет ли плагин такой же перевод локали. Если это так, язык плагина будет изменен в соответствии с языком интерфейса редактора и будет применен перевод.

Структура плагинов

config.json

Иконки плагинов

Файлы иконок плагинов, которые указаны в файле config.json для отображения плагинов на вкладке Плагины или в окне настроек плагина. Здесь вы можете найти все доступные параметры иконок.

Масштабирование

Для иконок плагинов может быть несколько типов масштабирования: 100%, 125%, 150%, 175%, 200% и т. д. Для каждого типа значок имеет свое нормальное состояние:

```
"icons": [
  {
    "100%": { "normal": "icon.png" },
    "125%": { "normal": "icon@1.25.png" },
    "150%": { "normal": "icon@1.5x.png" },
    "175%": { "normal": "icon@1.75.png" },
    "200%": { "normal": "icon@2x.png" }
  }
]
```

Редактор документов выбирает необходимые иконки следующим образом:

1. получить информацию о текущем масштабировании и найти для него иконку;
2. если такой иконки в конфигурации нет, берем ту, что ближе всего к нужному размеру и округляем в большую сторону (150% вместо 140%).

Стиль

Параметр стиля также используется для указания внешнего вида иконки:

Название	Описание	Тип	По умолчанию
style	Тип темы иконок плагина. Он может иметь светлые или темные значения.	строковый	«dark»

```
"icons": [
  {
    "style" : "dark"
  }
]
```

Этот параметр используется только тогда, когда иконки различаются в светлой и темной темах.

Вариация плагина

Вариации плагинов — субплагины, которые создаются внутри исходного плагина для следующих целей:

- выполнять основные действия плагина;
- содержать настройки плагина;
- для отображения окна «О программе» и т. д.

Давайте посмотрим на плагин перевода. Сам плагин не нуждается в визуальном окне для перевода, так как это можно сделать нажатием одной кнопки, но его настройки (направление перевода) и окно «О программе» должны быть визуальными. Таким образом, нам потребуется как минимум две вариации плагина (сам перевод и настройки) или три, если мы хотим добавить окно «О программе» с информацией о плагине и его авторах или программном обеспечении, используемом для создания плагина.

Для создания вариаций плагина необходимо указать его параметры в файле config.json и собрать файлы index.html для каждой вариации.

Файлы .html для всех вариантов должны быть помещены в корневую папку плагина вместе с файлом конфигурации config.json для корректной работы плагина.

config.json

Описание

Файл config.json — это файл конфигурации плагина, содержащий информацию об основных данных плагина, необходимых для регистрации плагина в редакторах.

Параметры

Название	Описание	Тип	Пример
baseUrl	Путь к плагину. Все остальные пути рассчитываются относительно этого пути. В случае, если плагин установлен на сервере, туда добавляется дополнительный параметр (путь к плагинам). Если baseUrl == «», будет использоваться путь ко всем плагинам.	строковый	«»
guid	Идентификатор плагина. Он должен быть типа asc.{UUID}.	строковый	«asc.{FFE1F462-1EA2-4391-990D-4CC84940B754}»
minVersion	Минимальная поддерживаемая версия редактора.	строковый	«6.3.0»
version	Версия плагина.	строковый	«1.0»
help	Путь к странице описания плагина. Если параметр указан, в окне плагина отображается кнопка помощи. Когда пользователь нажимает кнопку, он переходит по ссылке на страницу описания плагина.	строковый	«»
name	Имя плагина, которое будет отображаться на панели инструментов плагина.	строковый	«plugin name»
nameLocale	Переводы поля имени. Ключи объектов — это двухбуквенные коды языков (ru, de, it и т. д.), а значения — это перевод названия плагина для каждого языка.	объект	
variations	Вариации плагинов или субплагины, которые создаются внутри исходного плагина. Более подробную информацию вы можете найти здесь.	массив объектов	
variations.buttons	Список кнопок плагинов с изменяемым оформлением, используемых в интерфейсе плагина (используется только для визуальных плагинов с собственным окном, т. е. isVisual == true && isInsideMode == false). Объект кнопки может иметь следующие параметры: • **текст** — метка, которая отображается на кнопке,	variations.buttons	

	тип: строковый, **пример:** «Cancel»; • **primary** — определяет, является ли кнопка основной или нет. Основной флаг влияет только на оформление кнопки, **тип:** логический, **пример:** true; • isViewer -определяет, будет ли кнопка отображаться только в режиме просмотра или нет, **тип:** логический, **пример:** false; • textLocale — переводы для текстового поля. Ключи объектов — это двухбуквенные коды языков (ru, de, it и т. д.), а значения — перевод меток кнопок для каждого языка. **тип:** объект		
ariations.description	Описание, то есть то, что лучше всего описывает ваш плагин.	строковый	«plugin description»
variations.descriptionLocale	Переводы для поля описания. Ключи объектов — это двухбуквенные коды языков (ru, de, it и т. д.), а значения — это перевод описания плагина для каждого языка.	объект	
variations.EditorsSupport	Редакторы, для которых доступен плагин (word — редактор текстовых документов, ячейка — редактор электронных таблиц, слайд — редактор презентаций).	массив строк	
variations.icons	Файлы изображений значков плагинов, используемые в редакторах. Более подробную информацию вы можете найти здесь.	массив строк	
variations.initData	Обычно равен «» — это данные, которые отправляются из редактора в плагин при запуске плагина (например, если initDataType == «text», плагин при запуске получит выделенный текст). Это также может быть эквивалентно шифрованию в плагинах шифрования.	строковый	«»

variations.initDataType	Тип данных, выбираемый в редакторе и отправляемый в плагин: text — текстовые данные, html — код в формате HTML, ole — данные объекта OLE, desktop — данные десктопного редактора, destop-external — данные главной страницы десктопного приложения (системные сообщения), none — в плагин не будут отправляться данные из редактора.	строковый	«ole»
variations.initOnSelectionChanged	Указывает, отслеживает ли плагин события выделения текста в окне редактора.	логический	true
variations.cryptoMode	Идентификатор плагина шифрования, так как существует несколько плагинов шифрования.	строковый	«1»
variations.isDisplayedInViewer	Указывает, будет ли плагин отображаться в режиме просмотра, а также в режиме редактирования (isDisplayedInViewer == true) или только в режиме редактирования (isDisplayedInViewer == false).	логический	true
variations.isInsideMode	Указывает, должен ли плагин отображаться внутри панели редактора вместо собственного окна.	логический	true
variations.isModal	Указывает, является ли открытое окно плагина модальным (используется только для визуальных плагинов, и если isInsideMode не равно true).	логический	true
variations.isCustomWindow	Указывает, использует ли плагин пользовательское окно без стандартных полей и кнопок (используется только для модальных плагинов).	логический	true
variations.isSystem	Указывает, если плагин не отображается в интерфейсе редактора и запускается в фоновом режиме с сервером (или запускаются десктопные редакторы), не мешая другим плагинам, чтобы они могли работать одновременно.	логический	false
variations.isUpdateOleOnResize	Указывает, должен ли объект OLE перерисовываться при изменении размера в редакторе с использованием типа рисования векторного объекта или нет	boolean	true

	(используется только для объектов OLE, т. е. initDataType == «ole»).		
variations.isViewer	Указывает, работает ли плагин, когда документ доступен только в режиме просмотра или нет. Значение по умолчанию false.	логический	false
variations.isVisual	Указывает, является ли плагин визуальным (откроет окно для какого-либо действия или внесет некоторые дополнения в интерфейс панели редактора) или невизуальным (предоставит кнопку (или кнопки), которая будет применять некоторые преобразования или манипуляции к документу).	логический	true
variations.url	Точка входа плагина, т.е. файл HTML, который подключает файл plugin.js (базовый файл, необходимый для работы с плагинами) и запускает код плагина. См. раздел index.html для получения подробной информации.	строковый	«index.html»
variations.size	Размер плагинного окна.	массив целых чисел	
variations.events	События плагина	массив строк	
variations.cryptoDisabledForInternalCloud	Указывает, отключено ли шифрование для облаков Р7.	строковый	«true»
variations.cryptoDisabledForExternalCloud	Указывает, отключено ли шифрование для сторонних облаков.	строковый	«true»
variations.cryptoDisabledOnStart	Указывает, сбрасывается ли режим при перезапуске или нет.	строковый	«true»

Пример

```
{
    "baseUrl": "",
    "guid": "asc.{FFE1F462-1EA2-4391-990D-4CC84940B754}",
    "version": "1.0",
    "minVersion": "6.3.0",
    "help": "",
    "name": "plugin name",
    "nameLocale": {
       "fr": "french plugin name",
       "es": "spanish plugin name"
    },
```

```
"variations": [
  {
    "buttons": [
      {
        "text": "Cancel",
        "primary": false,
        "isviewer": false,
        "textLocale": {
          "fr": "Annuler",
          "es": "Cancelar"
        }
      }
    ],
    "description": "plugin description",
    "descriptionLocale": {
      "fr": "french plugin description",
      "es": "spanish plugin description"
    },
    "EditorsSupport": ["word", "cell", "slide"],
    "icons": [
      {
        "100%": { "normal": "icon.png" },
        "125%": { "normal": "icon@1.25.png" },
        "150%": { "normal": "icon@1.5x.png" },
        "175%": { "normal": "icon@1.75.png" },
        "200%": { "normal": "icon@2x.png" }
      },
      {
        "style" : "dark"
      }
    ],
    "initData": "",
    "initDataType": "ole",
    "initOnSelectionChanged": true,
    "cryptoMode": "1",
    "isDisplayedInViewer": true,
    "isInsideMode": false,
    "isModal": true,
    "isCustomWindow": true,
    "isSystem": false,
```

```
            "isUpdateOleOnResize": true,
            "isViewer": true,
            "isVisual": false,
            "url": "index.html",
            "size": [600, 700],
            "events": ["onClick"],
            "cryptoDisabledForInternalCloud": "true",
            "cryptoDisabledForExternalCloud": "true",
            "cryptoDisabledOnStart": "true"
        }
    ]
};
```

index.html

index.html

ExampleКаждый плагин действует в своем собственном iframe. Редактор подключит файл *index.html*, указанный в конфигурационном файле плагина config.json. Файл *index.html* — это точка входа плагина, соединяющая файл plugin.js — базовый файл, необходимый для работы с плагинами.

Пример

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title>Plugin name</title>
    <script type="text/javascript" src="https://support.r7-office.ru/wp-content/uploads/2023/05/plugins.js"></script>
    <script type="text/javascript" src="https://support.r7-office.ru/wp-content/uploads/2023/05/plugins-ui.js"></script>
    <link rel="stylesheet" href="https://support.r7-office.ru/wp-content/uploads/2023/05/plugins.css">
    <script type="text/javascript" src="plugin.js"></script>
  </head>
  <body style="width: 100%; height: 100%; margin: 0; padding: 0; overflow: hidden;">
    <div id="plugin name" style="margin: 0; padding: 0;"></div>
  </body>
</html>
```

Раздел <head>…</head> содержит ссылки на все скрипты и таблицы стилей, необходимые для корректной работы плагина (как локальные, так и удаленные, если плагин их использует). Он

также содержит ссылку на базовый файл plugins.js, необходимый для корректной работы с редакторами и содержащий работу базового метода плагина.

Тело может содержать теги … с заполнителями, в которые будут вставлены компоненты плагина.

Plugin code

Events

События

Раздел событий позволяет изменить все функции, относящиеся к событиям.

- button
- init
- inputHelper_onSelectItem
- onBlurContentControl
- onChangeContentControl
- onClick
- onCommandCallback
- onDocumentContentReady
- onEnableMouseEvent
- onExternalMouseUp
- onExternalPluginMessage
- onFocusContentControl
- onInputHelperClear
- onInputHelperInput
- onMethodReturn
- onTargetPositionChanged
- onTranslate

События и их описание:

button

функция, вызываемая при нажатии любой из кнопок плагина. Он определяет кнопки, используемые с плагином, и поведение плагина при нажатии на них.

Параметры

Параметр	Описание	Тип
id	Определяет индекс кнопки в массиве кнопок файла config.json. Если id == -1, то плагин считает, что крестик Закрыть окно был нажат или его работа была каким-то образом прервана.	численный

Пример

```
window.Asc.plugin.button = function (id) {
```

```
    this.executeCommand("close", '');
};
```

init

функция, вызываемая при запуске плагина. Он определяет данные, отправляемые в плагин, описывающие, какие действия должны быть выполнены и как они должны быть выполнены.

Параметры

Параметр	Описание	Тип
data	Определяет параметр данных, зависящий от параметра initDataType, указанного в файле config.json.	строковый

Пример

```
window.Asc.plugin.init = function () {
    this.callCommand(function() {
        var oDocument = Api.GetDocument();
        var oParagraph = Api.CreateParagraph();
        oParagraph.AddText("Hello world!");
        oDocument.InsertContent([oParagraph]);
    }, true);
};
```

inputHelper_onSelectItem

функция, вызываемая, когда пользователь пытается выбрать элемент из помощника ввода.

Параметры

Параметр	Описание	Тип
item	Определяет выбранный элемент: • **text** — текст предмета, **тип:** строковый, **пример:** «name»; • **id** — индекс предмета, **тип:** строковый, **пример:** «1».	объект

Пример

```
window.Asc.plugin.inputHelper_onSelectItem = function(item) {
    if (!item)
        return;

    window.Asc.plugin.executeMethod("InputText", [item.text, window.Asc.plugin.currentText]);
    window.Asc.plugin.getInputHelper().unShow();
};
```

onBlurContentControl

функция, вызываемая, чтобы показать, какой элемент управления содержимым был размыт.

Параметры

Параметр	Описание	Тип
control	Определяет элемент управления содержимым, который был размыт: • **Tag** -тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы вы могли ссылаться на них в своем коде. **тип:** строковый, **пример:** «{tag}»; • **Id** — уникальный идентификатор управления контентом. Его можно использовать для поиска определенного элемента управления содержимым и ссылки на него в коде. **тип:** численный, **пример:** 0; • **Lock** — значение, которое определяет, возможно ли удалить и/или изменить элемент управления содержимым или нет, **тип:** численный, **пример**: 0; • **InternalId** — уникальный внутренний идентификатор элемента управления контентом, **тип:** строковый, **пример:** «5_665».	объект

Параметр *Lock* может принимать следующие значения:

Численное значение	Редактировать	Удалить
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Пример

```
connector.attachEvent("onBlurContentControl", function(oPr)
{
  if (oPr && "BankBIC" === oPr["Tag"])
  {
    connector.executeMethod("GetFormValue", [oPr["InternalId"]], function(value)
    {
      if ("12345678" !== value)
        return;

      connector.executeMethod("GetFormsByTag", ["BankAccount"], function(forms)
      {
```

```
            for (let i = 0; i < forms.length; ++i)
            {
               connector.executeMethod("SetFormValue", [forms[i]["InternalId"],
"10101110100000000123"], null);
            }
         });

         connector.executeMethod("GetFormsByTag", ["BankName"], function(forms)
         {
            for (let i = 0; i < forms.length; ++i)
            {
               connector.executeMethod("SetFormValue", [forms[i]["InternalId"], "P7 BANK"], null);
            }
         });

         connector.executeMethod("GetFormsByTag", ["BankPlace"], function(forms)
         {
            for (let i = 0; i < forms.length; ++i)
            {
               connector.executeMethod("SetFormValue", [forms[i]["InternalId"], "Himalayas"], null);
            }
         });
      });
   }
   console.log("event: onBlurContentControl");
});
```

onChangeContentControl

функция, вызываемая, чтобы показать, какой элемент управления содержимым был изменен.

Параметры

Параметр	Описание	Тип
control	Определяет измененный элемент управления содержимым: • **Tag** — тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы вы могли ссылаться на них в своем коде. **тип:** строковый, **пример:** «{tag}»; • **Id** — уникальный идентификатор управления контентом. Его можно использовать для поиска определенного	объект

	элемента управления содержимым и ссылки на него в коде. **тип:** численный, **пример:** 0; • **Lock** — значение, которое определяет, возможно ли удалить и/или изменить элемент управления содержимым или нет, **тип:** численный, **пример:** 0; • **InternalId** — уникальный внутренний идентификатор элемента управления контентом, **тип:** строковый, **пример:** «5_665».	

Параметр *Lock* может принимать следующие значения:

Численные значения	Редактировать	Удалить
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Пример

```
window.Asc.plugin.event_onClick = function(isSelectionUse) {
  window.Asc.plugin.executeMethod("GetCurrentContentControlPr", [], function(obj) {
    window.Asc.plugin.currentContentControl = obj;
    var controlTag = obj ? obj.Tag : "";
    if (isSelectionUse)
      controlTag = "";
    ...
  });
};
```

onClick

функция, вызываемая, когда пользователь нажимает на элемент.

Параметры

Параметр	Описание	Тип
isSelectionUse	Определяет, используется ли выделение или нет.	логический

Пример

```
window.Asc.plugin.event_onClick = function(isSelectionUse) {
  window.Asc.plugin.executeMethod("GetCurrentContentControlPr", [], function(obj) {
    window.Asc.plugin.currentContentControl = obj;
```

```
    var controlTag = obj ? obj.Tag : "";
    if (isSelectionUse)
        controlTag = "";
    ...
  });
};
```

onCommandCallback

функция, вызываемая для возврата результата ранее выполненной команды. Его можно использовать для возврата данных после выполнения метода executeCommand.

Пример

```
window.Asc.plugin.onCommandCallback = function() {
  var plugin = window.Asc.plugin;
  plugin.executeCommand("close", "");
};
```

onDocumentContentReady

функция вызывается, когда документ полностью загружен.

Пример

```
window.Asc.plugin.event_onDocumentContentReady = function() {
  var oProperties = {
    "searchString"  : "P7",
    "replaceString" : "P7 is cool",
    "matchCase"     : false
  };

  window.Asc.plugin.executeMethod("SearchAndReplace", [oProperties], function() {
      window.Asc.plugin.executeCommand("close", "");
  });
};
```

onEnableMouseEvent

функция, вызываемая для включения/выключения событий мыши или тачпада.

Параметры

Параметр	Описание	Тип
isEnabled	Определяет, включена ли мышь или тачпад.	логический

Пример

```
window.Asc.plugin.onEnableMouseEvent = function(isEnabled) {
```

```
  var _frames = document.getElementsByTagName("iframe");
  if (_frames && _frames[0]) {
    _frames[0].style.pointerEvents = isEnabled ? "none" : "";
  }
};
```

onExternalMouseUp

функция, вызываемая при отпускании кнопки мыши за пределами iframe плагина.

Пример

```
window.Asc.plugin.onExternalMouseUp = function () {
  var evt = document.createEvent("MouseEvents");
  evt.initMouseEvent("mouseup", true, true, window, 1, 0, 0, 0, 0, false, false, false, false, 0, null);
  document.dispatchEvent(evt);
};
```

onExternalPluginMessage

функция, вызываемая для отображения сообщения интегратора редактора.

Параметры

Параметр	Описание	Тип
data	Определяет сообщение интегратора редактора: • Тип -тип сообщения, **тип:** строковый, **пример:** «close»; • text — текст сообщения, **тип:** строковый, **пример:** «text».	объект

Пример

```
window.Asc.plugin.onExternalPluginMessage = function(data) {
  switch (data.type) {
    case "close": {
      this.executeCommand("close", "");
      break;
    }
    case "insertText": {
      Asc.scope.text = data.text;
      this.callCommand(function() {
        var oDocument = Api.GetDocument();
        var oParagraph = Api.CreateParagraph();
```

```
            oParagraph.AddText(Asc.scope.text);
            oDocument.InsertContent([oParagraph]);
        }, false);
        break;
    }
  }
};
```

onFocusContentControl

функция, вызываемая для отображения того, какой элемент управления содержимым был сфокусирован.

Параметры

Параметр	Описание	Тип
control	Определяет элемент управления содержимым, который был сфокусирован: • **Tag** — тег, назначенный элементу управления содержимым. Один и тот же тег может быть назначен нескольким элементам управления контентом, чтобы вы могли ссылаться на них в своем коде, **тип:** строковый, **пример:** «{tag}»; • Id -уникальный идентификатор элемента управления контентом. Его можно использовать для поиска определенного элемента управления контентом и ссылки на него в вашем коде, **тип:** численный, **пример:** 0; • Lock — значение, определяющее, можно ли удалять и/или редактировать элемент управления содержимым или нет, **тип:** численный, **пример:** 0; • **InternalId** -уникальный внутренний идентификатор элемента управления контентом, **тип:** строковый, **пример:** «5_665».	объект

Параметр *Lock* может иметь следующие значения:

Численное значение	**Редактировать**	**Удалить**
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Пример

```
connector.attachEvent("onFocusContentControl", function()
```

```
{
   console.log("event: onFocusContentControl");
});
```

onInputHelperClear

функция, вызываемая, когда пользователь пытается очистить текст, и помощник ввода исчезает.

Пример

```
window.Asc.plugin.event_onInputHelperClear = function() {
   window.Asc.plugin.currentText = "";
   window.Asc.plugin.getInputHelper().unShow();
};
```

onInputHelperInput

функция, вызываемая, когда пользователь пытается ввести текст и появляется помощник ввода.

Параметры

Название	Описание	Тип
data	Определяет текст, который вводит пользователь: • add — определяет, добавляется ли текст к текущему тексту или это начало текста, **тип:** логический, **пример:** true; • text — текст, который вводит пользователь, **тип:** строковый, **пример:** «text».	объект

Пример

```
window.Asc.plugin.event_onInputHelperInput = function(data) {
   if (data.add)
      window.Asc.plugin.currentText += data.text;
   else
      window.Asc.plugin.currentText = data.text;
      ...
}
```

onMethodReturn

функция, вызываемая для возврата результата ранее выполненного метода. Его можно использовать для возврата данных после выполнения метода executeMethod.

Параметры

Параметр	Описание	Тип
returnValue	Определяет значение, которое будет возвращено.	

Пример

```
window.Asc.plugin.executeMethod("InsertAndReplaceContentControls", [_obj]);
window.Asc.plugin.onMethodReturn = function(returnValue) {
    if (window.Asc.plugin.info.methodName == "InsertAndReplaceContentControls") {
        window.Asc.plugin.executeMethod("GetAllContentControls");
    } else if ("GetAllContentControls") {
        window.Asc.plugin.executeCommand("close", console.log(JSON.stringify(returnValue)));
    }
};
```

onTargetPositionChanged

функция, вызываемая при изменении целевой позиции в редакторе.

Пример

```
window.Asc.plugin.event_onTargetPositionChanged = function() {
    if (!fClickLabel) {
        window.Asc.plugin.executeMethod("GetCurrentContentControl");
    }
    fClickLabel = false;
};
```

onTranslate

функция, вызываемая сразу после запуска плагина или позже, в случае изменения языка плагина.

Пример

```
window.Asc.plugin.onTranslate = function() {
    var label = document.getElementById("button_new");
    if (label)
        label.innerHTML = window.Asc.plugin.tr("New");
}
```

Вспомогательные объекты

Вспомогательные объекты

Описание

Вспомогательные объекты, определяющие дополнительные настройки для работы плагина.

Объекты

Название	Описание
info object	Определяет объект, в котором хранится вся информация о редакторе, использующем плагин, и дополнительные настройки для OLE-объектов.
Asc.scope object	Определяет объект, который используется для передачи любых дополнительных данных (объектов, параметров, переменных и т. д.) методу window.Asc.plugin.callCommand, который выполняется в своем собственном изолированном контексте.
InputHelper object	Определяет свойства помощника ввода.

Пример

```
(function(window, undefined){
    var text = "Hello world!";
    window.Asc.plugin.init = function() {
        Asc.scope.text = text;
        this.callCommand(function() {
            var oDocument = Api.GetDocument();
            var oParagraph = Api.CreateParagraph();
            oParagraph.AddText(Asc.scope.text);
            oDocument.InsertContent([oParagraph]);
        }, true);
    };
    window.Asc.plugin.button = function(id)
    {
    };
})(window, undefined);
```

InputHelper object

Описание

Определяет вспомогательный объект window.Asc.plugin.InputHelper, который представляет свойства помощника ввода.

Методы и свойства:

- **createWindow** – функция, вызываемая для создания вспомогательного окна ввода.

Пример

```
window.Asc.plugin.init = function(text) {
    if (!window.isInit) {
        window.isInit = true;
        window.Asc.plugin.currentText = "";
        window.Asc.plugin.createInputHelper();
```

```
        window.Asc.plugin.getInputHelper().createWindow();
    }
};
```

- **getItems** — функция, вызываемая для возврата массива объектов InputHelperItem, содержащих все элементы из помощника ввода.

Пример

```
function getInputHelperSize () {
    var _size = window.Asc.plugin.getInputHelper().getScrollSizes();
    var _width = 150;
    var _height = _size.h;
    var _heightMin = window.Asc.plugin.getInputHelper().getItemsHeight(Math.min(5,
window.Asc.plugin.getInputHelper().getItems().length));
    if (_width > 400)
        _width = 400;
    if (_height > _heightMin)
        _height = _heightMin;
        _width += 30;
        return { w: _width, h : _height };
}
```

- **getScrollSizes** — функция, вызываемая для получения размеров прокручиваемого окна помощника по вводу. Возвращает объект с параметрами ширины и высоты.

Пример

```
function getInputHelperSize () {
    var _size = window.Asc.plugin.getInputHelper().getScrollSizes();
    var _width = 200;
    var _height = _size.h;
    var _heightMin = window.Asc.plugin.getInputHelper().getItemsHeight(Math.min(5,
window.Asc.plugin.getInputHelper().getItems().length));
    if (_width > 400)
        _width = 400;
    if (_height > _heightMin)
        _height = _heightMin;
        _width += 30;
        return { w: _width, h : _height };
}
```

- **setItems** — функция, вызываемая для установки элементов в вспомогательный инструмент ввода.

Параметры

Параметр	Описание	Тип

InputHelperItem	Определяет массив объектов InputHelperItem, который содержит все элементы для помощника ввода. Этот объект может иметь следующие параметры: • **id** — индекс предмета, **тип:** строковый, **пример:** «1»; • **text** -текст предмета, **тип:** строковый, **пример:** «name»	объект

Пример

```
{
   var items = [
      { text: "Name1.Family1", id : "0" },
      { text: "Name2.Family2", id : "1" },
      { text: "Name3.Family3", id : "2" },
      { text: "Name4.Family4", id : "3" },
      { text: "Name5.Family5", id : "4" },
      { text: "Name6.Family6", id : "5" },
      { text: "Name7.Family7", id : "6" },
      { text: "Name8.Family8", id : "7" },
      { text: "Name9.Family9", id : "8" },
      { text: "Name10.Family10", id : "9" },
      { text: "Name11.Family11", id : "10" },
      { text: "Name12.Family12", id : "11" },
      { text: "Name13.Family13", id : "12" }
   ];
   window.Asc.plugin.getInputHelper().setItems(items);
   var _sizes = getInputHelperSize();
   window.Asc.plugin.getInputHelper().show(_sizes.w, _sizes.h, true);
}
```

- **show** — функция, вызываемая для отображения вспомогательного средства ввода.

Параметры

Параметры	**Описание**	**Тип**
width	Ширина окна помощника ввода измеряется в миллиметрах.	численный
height	Высота окна помощника ввода измеряется в миллиметрах.	численный
isCaptureKeyboard	Определяет, захвачена ли клавиатура (true) или нет (false).	логический

Пример

```
{
   var items = [
      { text: "Name1.Family1", id : "0" },
      { text: "Name2.Family2", id : "1" },
      { text: "Name3.Family3", id : "2" },
      { text: "Name4.Family4", id : "3" },
      { text: "Name5.Family5", id : "4" },
      { text: "Name6.Family6", id : "5" },
      { text: "Name7.Family7", id : "6" },
      { text: "Name8.Family8", id : "7" },
      { text: "Name9.Family9", id : "8" },
      { text: "Name10.Family10", id : "9" },
      { text: "Name11.Family11", id : "10" },
      { text: "Name12.Family12", id : "11" },
      { text: "Name13.Family13", id : "12" }
   ];
   window.Asc.plugin.getInputHelper().setItems(items);
   var _sizes = getInputHelperSize();
   window.Asc.plugin.getInputHelper().show(_sizes.w, _sizes.h, true);
}
```

- **unShow** — функция, вызываемая для скрытия помощника ввода.

Пример

```
window.Asc.plugin.executeMethod ("SelectContentControl",
[window.Asc.plugin.currentContentControl.InternalId], function() {
   window.Asc.plugin.executeMethod("InputText", [item.text]);
   window.Asc.plugin.getInputHelper().unShow();
});
```

info object

Описание

Определяет вспомогательный объект window.Asc.plugin.info, который доступен при работе плагина. В нем хранится вся информация о редакторе, который использует плагин (используемый editorType — текстовые документы, таблицы, презентации) и дополнительные настройки для OLE-объектов (их ширина, высота, отношение миллиметра к пикселю для векторной отрисовки OLE-объектов и некоторых других OLE-объектов). параметры объекта).

Этот объект используется для изменения данных объекта и отправки дополнительных параметров при выполнении метода window.Asc.plugin.executeCommand. Например, если содержимое документа изменилось и требуется пересчет, параметр пересчета должен быть установлен в значение true. Это действие необходимо, поскольку процесс пересчета является асинхронным. Кроме того, может потребоваться загрузка некоторых других данных (например, шрифт или что-то еще).

Параметры

Название	Описание	Тип	Пример
data	Данные объекта OLE, которые необходимо отправить в объект window.Asc.plugin.info и использовать плагин.	строковый	«{data}»
editorType	Тип редактора, в котором в данный момент запущен плагин.	строковый	«word»
guid	GUID OLE-объекта в текущем документе.	строковый	«asc.{UUID}»
height	Высота OLE-объекта измеряется в миллиметрах.	численный	70
imgSrc	Ссылка на изображение (его визуальное представление), хранящееся в OLE-объекте и используемое плагином.	строковый	«./resources/image.png»
mmToPx	Коэффициент конвертации миллиметра в пиксель для отрисовки вектора объекта OLE.	численный	3
objectId	Идентификатор объекта OLE в текущем документе.	строковый	1
recalculate	Заставляет документ пересчитывать данные своего содержимого.	логический	true
resize	Проверяет, был ли изменен размер объекта OLE.	логический	true
width	Ширина объекта OLE измеряется в миллиметрах.	численный	70

Пример параметров *data, height, imgSrc, mmToPx, objectId* и *width*

```
window.Asc.plugin.button = function (id) {
  var _info = window.Asc.plugin.info;
  var _method = (_info.objectId === undefined) ? "asc_addOleObject" : "asc_editOleObject";
  _info.width = _info.width ? _info.width : 70;
  _info.height = _info.height ? _info.height : 70;
  _info.widthPix = (_info.mmToPx * _info.width) >> 0;
  _info.heightPix = (_info.mmToPx * _info.height) >> 0;
  _info.imgSrc = window.g_board.getResult(_info.widthPix, _info.heightPix).image;
  _info.data = window.g_board.getData();
  var _code = "Api." + _method + "(" + JSON.stringify(_info) + ");";
```

```
    this.executeCommand("close", _code);
};
```

Пример для параметра *editorType*

```
function createScriptFromArray (aSelected) {
    var sScript = '';
    if (aSelected.length > 0) {
        switch (window.Asc.plugin.info.editorType) {
            case 'word': {
                sScript += 'var oDocument = Api.GetDocument();';
                sScript += '\noDocument.CreateNewHistoryPoint();';
                sScript += '\nvar oParagraph, oRun, arrInsertResult = [], oImage;';
                sScript += '\noDocument.InsertContent(arrInsertResult);';
                break;
            }
        }
    }
    return sScript;
}
```

Пример для параметра *guid*

```
window.Asc.plugin.init = function () {
    var plugin_uuid = window.Asc.plugin.info.guid;
};
```

Пример *пересчета* параметра

```
window.Asc.plugin.init = function () {
    var sScript = 'var oDocument = Api.GetDocument();';
    sScript += 'oDocument.CreateNewHistoryPoint();';
    sScript += 'oParagraph = Api.CreateParagraph();';
    sScript += 'oParagraph.AddText(\'Hello word!\');';
    sScript += 'oDocument.InsertContent([oParagraph]);';
    window.Asc.plugin.info.recalculate = true;
    this.executeCommand("close", sScript);
};
```

Пример параметра изменения */resize* размера

```
if (window.Asc.plugin.info.resize === true) {
```

```
    return window.Asc.plugin.button(0);
}
```

Asc.scope object

Описание

Объект используется для передачи любых дополнительных данных (объектов, параметров, переменных и т. д.) методу window.Asc.plugin.callCommand, который выполняется в своем собственном изолированном контексте.

Функции нельзя передать методу callCommand с помощью объекта Asc.scope.

Пример

```
(function(window, undefined){
    var scopeText = ["Hello World!", "This is me!", "I'm glad to see you!!!"];
    window.Asc.plugin.init = function() {
        Asc.scope.st = scopeText;
        this.callCommand(function() {
            var oDocument = Api.GetDocument();
            var oParagraph = Api.CreateParagraph();
            for (var i = 0; i < Asc.scope.st.length; i++)
            {
                oParagraph.AddText(Asc.scope.st[i] + "<br />");
            }
            oDocument.InsertContent([oParagraph]);
        }, true);
    };
    window.Asc.plugin.button = function(id)
    {
    };
})(window, undefined);
```

Методы

resizeWindow

window.Asc.plugin.resizeWindow (width, height, minW, minH, maxW, maxH)

Определяет метод, используемый для изменения размера окна, обновляя минимальный/максимальный размеры.

Этот метод используется только для визуальных модальных плагинов.

Параметры

Название	Описание	Тип
width	Ширина окна.	численный
height	Высота окна.	численный
minW	Минимальная ширина окна.	численный
minH	Минимальная высота окна.	численный
maxW	Максимальная ширина окна.	численный
maxH	Максимальная высота окна.	численный

Пример

```
window.Asc.plugin.init = function() {
   this.resizeWindow(392, 147, 392, 147, 392, 147);
};
```

loadModule

window.Asc.plugin.loadModule (url, callback)

Определяет метод, используемый для загрузки удаленно расположенного текстового ресурса.

Параметры

Название	Описание	Тип
url	Ссылка на код ресурса.	строковый
callback	Результат, который возвращает метод.	функциональный

Пример

```
window.Asc.plugin.loadModule("./vendor/highlight/styles/" + e.params.data.id , function(content) {
   var style_value = content;
   if (isDE || isFF) {
      $("#jq_color").spectrum("set", (hexc($(container).css('backgroundColor'))));
   } else {
      background_color.value = hexc($(container).css('backgroundColor'));
   }
});
```

getInputHelper

Определяет метод, используемый для получения объекта InputHelper.

Пример

```
window.Asc.plugin.init = function(text) {
```

```
    if (!window.isInit) {
        window.isInit = true;
        window.Asc.plugin.currentContentControl = null;
        window.Asc.plugin.createInputHelper();
        window.Asc.plugin.getInputHelper().createWindow();
    }
};
```

executeCommand

window.Asc.plugin.executeCommand (type, data, callback)

Определяет метод, используемый для отправки данных обратно в редактор.

Этот метод устарел, используйте метод callCommand, который запускает код из параметра строки данных. Сейчас этот метод в основном используется для работы с OLE-объектами или закрытия плагина без каких-либо других команд. Он также сохраняется для использования с текстом, чтобы предыдущие версии плагина оставались совместимыми.

Обратный вызов — это результат, который возвращает команда. Это необязательный параметр. В случае его отсутствия для возврата результата выполнения команды будет использована функция window.Asc.plugin.onCommandCallback.

Второй параметр — это код JavaScript для работы с Document Builder API, который позволяет плагину отправлять структурированные данные, вставленные в результирующий файл документа (форматированные абзацы, таблицы, текстовые части и отдельные слова и т. д.).

Команды Document Builder можно использовать только для создания контента и его вставки в редактор документов (используя Api.GetDocument().InsertContent(…)). Это ограничение существует из-за функции совместного редактирования в онлайн-редакторах. Если необходимо создать плагин для десктопных редакторов для работы с локальными файлами, такое ограничение не применяется.

Параметры

Название	Описание	Тип
type	Определяет тип команды. Закрытие используется для закрытия окна плагина после выполнения функции в параметре данных. Команда используется для выполнения команды и оставления окна открытым в ожидании следующей команды.	строковый
data	Определяет команду, написанную в коде JavaScript, целью которой является формирование структурированных данных, которые могут быть вставлены в результирующий файл документа (форматированные абзацы, таблицы, текстовые части и отдельные слова и т. д.). Затем данные отправляются в редакцию. Команда должна быть совместима с синтаксисом Document Builder.	строковый
callback	Результат, который возвращает метод.	функциональный

При создании/редактировании OLE-объектов для работы с ними используются два расширения:

- Api.asc_addOleObject (window.Asc.plugin.info) — используется для создания OLE — объекта в документе;
- Api.asc_editOleObject (window.Asc.plugin.info) -используется для редактирования созданного OLE — объекта.

При создании/редактировании объектов их свойства могут быть переданы в объект window.Asc.plugin.info, который определяет, как должен выглядеть объект.

Пример OLE-объекта

```
window.Asc.plugin.button = function (id) {
    var _info = window.Asc.plugin.info;
    var _method = (_info.objectId === undefined) ? "asc_addOleObject" : "asc_editOleObject";
    _info.width = _info.width ? _info.width : 70;
    _info.height = _info.height ? _info.height : 70;
    _info.widthPix = (_info.mmToPx * _info.width) >> 0;
    _info.heightPix = (_info.mmToPx * _info.height) >> 0;
    _info.imgSrc = window.g_board.getResult(_info.widthPix, _info.heightPix).image;
    _info.data = window.g_board.getData();
    var _code = "Api." + _method + "(" + JSON.stringify(_info) + ");";
    this.executeCommand("close", _code);
};
```

Пример текста (не используется, но сохранен для совместимости)

```
window.Asc.plugin.init = function () {
    var sScript = 'var oDocument = Api.GetDocument();';
    sScript += 'oDocument.CreateNewHistoryPoint();';
    sScript += 'oParagraph = Api.CreateParagraph();';
    sScript += 'oParagraph.AddText(\'Hello word!\');';
    sScript += 'oDocument.InsertContent([oParagraph]);';
    window.Asc.plugin.info.recalculate = true;
    this.executeCommand("close", sScript);
};
```

createInputHelper

window.Asc.plugin.createInputHelper ()

Определяет метод, используемый для создания помошника ввода — окна, которое появляется и исчезает при вводе текста. Его местоположение привязано к курсору.

Пример

```
window.Asc.plugin.init = function(text) {
```

```
    if (!window.isInit) {
        window.isInit = true;
        window.Asc.plugin.currentContentControl = null;
        window.Asc.plugin.createInputHelper();
        window.Asc.plugin.getInputHelper().createWindow();
    }
};
```

callModule

window.Asc.plugin.callModule (url, callback, isClose)

Определяет метод, используемый для выполнения удаленного скрипта по ссылке.

Параметры

Название	Описание	Тип
url	URL-адрес кода ресурса.	строковый
callback	Результат, который возвращает метод.	функциональный
isClose	Определяет, должно ли окно плагина закрываться после выполнения кода или оставаться открытым в ожидании другого действия. Значение true используется для закрытия окна плагина после выполнения удаленного скрипта. Значение false используется для выполнения кода и оставления окна открытым в ожидании следующего действия.	логический

Пример

```
window.Asc.plugin.callModule("./templates/" + _templates[_index][0] + "/script.txt", function(content)
{
    _templates_code[_index] = content;
});
```

callCommand

window.Asc.plugin.callCommand (func, isClose, isCalc, callback)

Определяет метод, используемый для отправки данных обратно в редактор. Он заменяет метод executeCommand при работе с текстами, чтобы упростить синтаксис скрипта, который необходимо передать редакторам с помощью Document Builder API. Это позволяет плагину отправлять структурированные данные, которые можно вставить в результирующий файл документа (форматированные абзацы, таблицы, текстовые части, отдельные слова и т. д.).

Обратный вызов — это результат, который возвращает команда. Это необязательный параметр. В случае его отсутствия для возврата результата выполнения команды будет использована функция window.Asc.plugin.onCommandCallback.

Команды Document Builder можно использовать только для создания контента и вставки его в редактор документов (используя Api.GetDocument().InsertContent(…)). Это ограничение существует из-за функции совместного редактирования в онлайн-редакторах. Если необходимо создать плагин для десктопных редакторов для работы с локальными файлами, такое ограничение не применяется.

Параметры

Название	Описание	Тип
func	Определяет команду, написанную на JavaScript, целью которой является формирование структурированных данных, которые могут быть вставлены в результирующий файл документа (форматированные абзацы, таблицы, текстовые части, отдельные слова и т. д.). Затем данные отправляются в редакцию. Команда должна быть совместима с синтаксисом Document Builder.	функциональный
isClose	Определяет, должно ли окно плагина закрываться после выполнения кода или оставаться открытым в ожидании другой команды или действия. Значение true используется для закрытия окна плагина после выполнения функции в параметре func. Значение false используется для выполнения команды и оставления окна открытым в ожидании следующей команды.	логический
isCalc	Определяет, будет ли документ пересчитываться или нет. Значение true используется для пересчета документа после выполнения функции в параметре func. Значение false не приведет к пересчету документа (используйте его только в том случае, если ваши правки точно не потребуют пересчета документа).	логический
callback	Результат, который возвращает метод.	функциональный

Этот метод выполняется в своем контексте, изолированном от других данных JavaScript. Если в этот метод нужно передать какие-то параметры или другие данные, используйте объект Asc.scope.

Пример

```
window.Asc.plugin.init = function () {
    this.callCommand(function() {
        var oDocument = Api.GetDocument();
        var oParagraph = Api.CreateParagraph();
        oParagraph.AddText("Hello world!");
        oDocument.InsertContent([oParagraph]);
    }, true);
};
```

Профили. Краткое описание методов

Название	Ресурс	Описание

Добавить пользователя	POST api/2.0/people	Добавляет нового пользователя портала с именем, фамилией, адресом электронной почты и несколькими необязательными параметрами, указанными в запросе.
Удалить пользователя	DELETE api/2.0/people/{userid}	Удаляет с портала пользователя с идентификатором, указанным в запросе.
Удалить пользователей	PUT api/2.0/people/delete	Удаляет список пользователей с идентификаторами, указанными в запросе.
Получить профиль по электронной почте пользователя	GET api/2.0/people/email	Возвращает подробную информацию о профиле пользователя с адресом электронной почты, указанным в запросе.
Получить профиль по имени пользователя	GET api/2.0/people/{username}	Возвращает подробную информацию о профиле пользователя с именем, указанным в запросе.
Получить все профили	GET api/2.0/people	Возвращает список профилей для всех пользователей портала.
Получить мой профиль	GET api/2.0/people/@self	Возвращает подробную информацию о текущем профиле пользователя.
Импорт пользователей	POST api/2.0/people/import/save	Импортирует новых пользователей портала с именем, фамилией и адресом электронной почты.
Повторно отправить активационное письмо	PUT api/2.0/people/invite	Повторно отправляет электронные письма пользователям, которые не активировали свои электронные письма.
Отправить инструкции по удалению	PUT api/2.0/people/self/delete	Отправляет инструкции по удалению профиля пользователя.
Обновить пользователя	PUT api/2.0/people/{userid}	Обновляет данные для выбранного пользователя портала с указанием имени, фамилии, адреса электронной почты и/или дополнительных параметров, указанных в запросе.

executeMethod

UpdatePlugin

window.Asc.plugin.executeMethod («UpdatePlugin», [args], callback)

Определяет метод, который позволяет обновлять плагин по URL-адресу конфигурации плагина.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("UpdatePlugin", [url]);
```

Параметры

Параметр	Описание	Тип	Пример
url	URL-адрес для обновления конфигурации плагина.	строковый	«https://example.com/plugin/config.json»

Возвращает

Метод возвращает объект с информацией о результате.

Пример

```
window.Asc.plugin.executeMethod ("UpdatePlugin", ["https://example.com/plugin/config.json"]);
```

UnShowInputHelper

window.Asc.plugin.executeMethod («UnShowInputHelper», [args], callback)

Определяет метод, позволяющий скрыть помощник ввода.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("UnShowInputHelper", [guid, isclear]);
```

Параметры

Параметр	Описание	Тип	Пример
guid	Строковое значение, указывающее идентификатор плагина, который должен иметь тип asc.{UUID}.	строковый	«asc.{UUID}»

Возвращает

Метод возвращает *неопределенное* значение.

Пример

```
window.Asc.plugin.executeMethod("UnShowInputHelper", ["asc.{UUID}", true]);
```

StartAction

window.Asc.plugin.executeMethod («StartAction», [args], callback)

Определяет метод, позволяющий указать начало действия для длительных операций.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("StartAction", [type, description]);
```

Параметры

Параметр	Описание	Тип	Пример
type	Значение, определяющее тип действия, которое может принимать 0, если это действие Information, или 1, если это действие BlockInteraction.	численный	1
description	Строковое значение, задающее текст описания начала действия операции	строковый	«Save to local storage…»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("StartAction", [1, "Save to local storage..."]);
```

ShowInputHelper

window.Asc.plugin.executeMethod («ShowInputHelper», [args], callback)

Определяет метод, который позволяет отображать помощник ввода.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("ShowInputHelper", [guid, w, h, isKeyboardTake]);
```

Параметры

Параметры	**описание**	**Тип**	**Пример**

guid	Строковое значение, указывающее идентификатор плагина, который должен иметь тип asc.{UUID}.	строковый	«asc.{UUID}»
w	Число, указывающее ширину окна, измеренную в миллиметрах.	строковый	70
h	Число, указывающее высоту окна, измеренную в миллиметрах.	строковый	70
isKeyboardTake	Определяет, захвачена ли клавиатура (true) или нет (false).	логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("ShowInputHelper", ["asc.{UUID}", 70, 70, true]);
```

ShowButton

window.Asc.plugin.executeMethod («ShowButton», [args], callback)

Определяет метод, позволяющий отображать или скрывать кнопки в заголовке.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("ShowButton", [id, bShow]);
```

Параметры

Параметр	Описание	Тип	Пример
id	Идентификатор кнопки.	строковый	«1»
bShow	Флаг указывает, отображается ли кнопка (true) или скрыта (false).	логический	false

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("ShowButton", ["1", false]);
```

SetMacros

window.Asc.plugin.executeMethod («SetMacros», [args], callback)

Определяет метод, позволяющий устанавливать макросы в документ.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("SetMacros", [data]);
```

Параметры

Параметр	Описание	Тип	Пример
data	Объект Macros, содержащий данные обо всех макросах из документа в следующем виде (JSON): • **macrosArray** — массив кодов макросов ([{«name»: «Macros1», «value»: «{macrosCode}»}]), **тип:** массив объектов; • **current** — a current macro index, **тип:** численные, **пример:** 1	объект	

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("SetMacros", [{"macrosArray": [{"name": "Macros 1","value": "(function()\n{oDocument = Api.GetDocument();\noParagraph = Api.CreateParagraph();\noParagraph.AddText(\"This is a new paragraph\");\noDocument.Push(oParagraph);\n})();"},{"name": "Macros 2","value": "(function()\n{oDocument = Api.GetDocument();\noParagraph = oDocument.GetElement(0);\noParagraph.AddText(\"Document Builder\");\noRange = oDocument.GetRange(0, 24);\noRange.SetBold(true);\n})();"}],"current": 1}]);
```

SetFormValue

window.Asc.plugin.executeMethod («SetFormValue», [args], callback)

Определяет метод, позволяющий установить значение в указанную форму.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("SetFormValue", [internalId, value]);
```

Параметры

Параметр	Описание	Тип	Пример
internalId	Уникальный внутренний идентификатор формы	строковый	«1_713»
value	Устанавливаемое значение формы. Его тип зависит от типа формы	строковый или логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("SetFormValue", ["1_713", true]);
```

SetDisplayModeInReview

window.Asc.plugin.executeMethod («SetDisplayModeInReview», [args], callback)

Определяет метод, позволяющий установить режим отображения для отслеживания изменений.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("SetDisplayModeInReview", [sMode]);
```

Параметры

Параметр	Описание	Тип	Пример
sMode	Режим отображения: «редактировать» — отображаются все изменения, «простой» — отображаются все изменения, но всплывающие подсказки отключены, «окончательный» — отображаются все принятые изменения, «исходный» — отображаются все отклоненные изменения. Значение по умолчанию — «редактировать».	строковый	«edit»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("SetDisplayModeInReview", ["edit"]);
```

SelectOleObject

window.Asc.plugin.executeMethod («SelectOleObject», [args], callback)

Определяет метод, позволяющий выбрать указанный OLE — объект.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("SelectOleObject", [id]);
```

Параметры

Параметр	Описание	Тип	Пример
id	Идентификатор OLE-объекта, который используется для работы с OLE — объектом, добавленным в документ.	строковый	«5_665»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("SelectOleObject", ["5_665"]);
```

SelectContentControl

window.Asc.plugin.executeMethod («SelectContentControl», [args], callback)

Определяет метод, позволяющий выбрать указанный элемент управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("SelectContentControl", [InternalId]);
```

Параметры

Параметр	Описание	Тип	Пример
InternalId	Уникальный внутренний идентификатор элемента управления содержимым.	строковый	«5_665»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("SelectContentControl", ["5_665"]);
```

SearchAndReplace

SearchAndReplace(oProperties, oProperties.searchString, oProperties.replaceString, [oProperties.matchCase])

Находит и заменяет текст.

Данный метод применим только для Документов и нет возможности применить для Таблиц и Презентаций.

Параметры:

Название	**Тип**	**По умолчанию**	**Описание**
oProperties	объект		Объект, содержащий строки поиска и замены.
oProperties.searchString	строковый		Строка поиска.
oProperties.replaceString	строковый		Строка замены.
oProperties.matchCase	строковый	true	Чувствителен к регистру или нет.

Возвращает:

Этот метод не возвращает никаких данных.

Пример

```
window.Asc.plugin.executeMethod ("SearchAndReplace", [
   "searchString": "text1",
   "replaceString": "text2",
   "matchCase": true
]);
```

ReplaceTextSmart

window.Asc.plugin.executeMethod («ReplaceTextSmart», [args], callback)

Определяет метод, который позволяет заменить каждый параграф (или текст в ячейке) в выделении соответствующим текстом из массива строк.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("ReplaceTextSmart", [arrString, sParaTab, sParaNewLine]);
```

Параметры

Параметр	Описание	Тип	Пример
arrString	Массив замещения строк.	массив строк	[«test_1», «test_2»]
sParaTab	Символ, который используется для указания табуляции в исходном тексте.	строковый	» «
sParaNewLine	Символ, который используется для указания символа разрыва строки в исходном тексте..	строковый	» «

Возвращает

Метод всегда возвращает логическое значение *true*.

Пример

```
window.Asc.plugin.executeMethod("ReplaceTextSmart", [["test_1", "test_2"], " ", " "]);
```

RemoveSelectedContent

window.Asc.plugin.executeMethod («RemoveSelectedContent», callback)

Определяет метод, позволяющий удалить выбранный контент из документа.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RemoveSelectedContent");
```

Возвращает

Метод возвращает неопределенное значение.

RemovePlugin

window.Asc.plugin.executeMethod («RemovePlugin», [args], callback)

Определяет метод, позволяющий удалить плагин с указанным GUID.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RemovePlugin", [guid]);
```

Параметры

Параметр	Описание	Тип	Пример
guid	Идентификатор плагина. Он должен быть типа asc.{UUID}.	строковый	«asc.{FFE1F462-1EA2-4391-990D-4CC84940B754}»

Возвращает

Метод возвращает объект с информацией о результате.

Пример

```
window.Asc.plugin.executeMethod ("RemovePlugin", ["asc.{FFE1F462-1EA2-4391-990D-4CC84940B754}"]);
```

RemoveOleObjects

window.Asc.plugin.executeMethod («RemoveOleObjects», [args], callback)

Определяет метод, позволяющий удалить из документа несколько OLE-объектов по их внутренним идентификаторам.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RemoveOleObjects", [arrObjects]);
```

Параметр

Параметры	Описание	Тип	Пример
arrObjects	В документ добавляется массив данных OLE-объекта с идентификаторами, которые используются для работы с OLE-объектами.	массив объектов	[{«InternalId»: «5_556»}]

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("RemoveOleObjects", [[{"InternalId": "5_556"}]]);
```

RemoveOleObject

window.Asc.plugin.executeMethod («RemoveOleObject», [args], callback)

Определяет метод, позволяющий удалить OLE-объект из документа по его внутреннему идентификатору.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RemoveOleObject", [sInternalId]);
```

Параметры

Параметр	Описание	Тип	Пример
sInternalId	Идентификатор объекта OLE, который используется для работы с объектом OLE, который был добавлен в документ.	строковый	«5_556»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("RemoveOleObject", ["5_556"]);
```

RemoveContentControls

window.Asc.plugin.executeMethod («RemoveContentControls», [args], callback)

Определяет метод, позволяющий удалить несколько элементов управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RemoveContentControls", [arrDocuments]);
```

Параметры

Параметр	Описание	Тип	Пример
arrDocuments	Массив объектов управления содержимым с их внутренними идентификаторами.	массив объектов	[{«InternalId»: «5_556»}]

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("RemoveContentControls", [[{"InternalId": "5_556"}]])
```

RemoveContentControl

window.Asc.plugin.executeMethod («RemoveContentControl», [args], callback)

Определяет метод, который позволяет удалить текущий выбранный элемент управления содержимым, сохранив все его содержимое. Элемент управления содержимым, в котором в данный момент находится курсор мыши, будет удален.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RemoveContentControl", [InternalId]);
```

Параметры

Параметр	**Описание**	**Тип**	**Пример**

InternalId	Уникальный внутренний идентификатор элемента управления содержимым.	строковый	«5_665»

Возвращает

Метод возвращает объект, который содержит следующие значения:

```
{
   "Parent" : object,
   "Pos" : integer,
   "Count" : integer
}
```

Параметры

Параметр	Описание	Тип	Пример
Parent	Родительский элемент управления содержимым.	объект	oParagraph
Pos	Позиция элемента управления содержимым в родительском объекте.	целочисленное	0
Count	Количество элементов в родительском объекте.	целочисленное	1

Пример

```
window.buttonIDChangeState_click = undefined;
   if (null == returnValue) {
      window.Asc.plugin.executeMethod("AddContentControl", [1, {"Id" : 7, "Lock" : 0, "Tag" : "{some text}"}]);
   }
   else {
      window.Asc.plugin.executeMethod("RemoveContentControl", [returnValue]);
   }
```

RemoveComments

window.Asc.plugin.executeMethod («RemoveComments», [args], callback)

Определяет метод, позволяющий удалить указанные комментарии.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RemoveComments", [arrIds]);
```

Параметры

Параметр	Описание	Тип	Пример
arrIds	Массив, содержащий ID указанных комментариев.	массив строк	

Возвращает

Метод возвращает *неопределенное* значение.

```
window.Asc.plugin.executeMethod ("RemoveComments", [["1_631", "1_632"]]);
```

RejectReviewChanges

window.Asc.plugin.executeMethod («RejectReviewChanges», [args], callback)

Определяет метод, позволяющий отклонить изменения обзора.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("RejectReviewChanges", [isAll]);
```

Параметры

Параметр	Описание	Тип	Пример
isAll	Указывает, будут ли отклонены все изменения (true) или только изменения текущего выбора (false). Значение по умолчанию false.	логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("RejectReviewChanges", [true]);
```

PutImageDataToSelection

window.Asc.plugin.executeMethod («PutImageDataToSelection», [args], callback)

Определяет метод, позволяющий заменить первый выбранный рисунок изображением, указанным в параметрах. Если рисунки не выбраны, метод вставляет изображение в текущую позицию.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("PutImageDataToSelection", [oImageData]);
```

Параметры

Параметр	Описание	Тип	Пример
oImageData	Информация об изображении PNG в кодировке base64: • **src** -исходник изображения в формате base64, **тип:** строковый, **пример:** «-in-the-base64-format»; • **nWidth** — ширина изображения в пикселях, **тип:** численный, **пример:** 300; • **nHeight** -высота изображения в пикселях, **тип:** численный, **пример:** 200		

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.saveImage = function () {
  let sImageSrc = imageEditor.toDataURL();
  let editorDimension = imageEditor.getCanvasSize();
  let nWidth = editorDimension.width;
  let nHeight = editorDimension.height;
  let oImageData = {
```

```
        "src": sImageSrc,
        "width": nWidth,
        "height": nHeight
    };
    window.Asc.plugin.executeMethod ("PutImageDataToSelection", [oImageData]);
    window.Asc.plugin.executeCommand("close", "");
};
```

PasteText

window.Asc.plugin.executeMethod («PasteText», [args], callback)

Определяет метод, позволяющий вставлять текст в документ.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("PasteText", [text]);
```

Параметры

Параметр	Описание	Тип	Пример
text	Строковое значение, указывающее текст, который нужно вставить в документ.	строковый	«P7 for developers»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("PasteText", ["P7 for developers"]);
```

PasteHtml

window.Asc.plugin.executeMethod («PasteHtml», [args], callback)

Определяет метод, позволяющий вставлять в документ текст в формате HTML.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("PasteHtml", [htmlText]);
```

Параметры

Параметр	Описание	Тип	Пример
htmlText	Строковое значение, указывающее текст в формате HTML, который нужно вставить в документ	строковый	«html text»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("PasteHtml", ["<p><b>Plugin methods for OLE objects</b></p><ul><li>AddOleObject</li><li>EditOleObject</li></ul>"]);
```

OpenFile

window.Asc.plugin.executeMethod («OpenFile», [args], callback)

Определяет метод, позволяющий открыть файл с полями.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("OpenFile", [binaryFile, fields]);
```

Параметры

Параметр	Описание	Тип	Пример
binaryFile	Файл в формате 8-битного массива целых чисел без знака.	массив	[Uint8Array]
fields	Список значений полей.	список	[«id», «name»]

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("OpenFile", [[Uint8Array], ["id", "name"]]);
```

OnEncryption

window.Asc.plugin.executeMethod («OnEncryption», [args], callback)

Определяет метод, позволяющий зашифровать документ.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("OnEncryption", [obj]);
```

Параметры

Параметр	Описание	Тип
obj	Свойства шифрования. Этот объект может иметь следующие значения: • **type** — тип операции шифрования (generatePassword — генерирует пароль к документу, getPasswordByFile — отправляет пароль при открытии документа, setPasswordByFile — устанавливает пароль к документу, encryptData — шифрует изменения при совместном редактировании, decryptData — расшифровывает изменения при совместном редактировании редактирование), **тип:** строковый, **пример:** «encryptData»; • password -строковое значение, указывающее пароль для доступа к документу, **тип:** строковый, **пример:** «password»; • data — зашифрованные/расшифрованные изменения, **тип:** строковый, **пример:** «{data}»; • check — проверяет успешность операции шифрования/дешифрования (используется только для типов encryptData/decryptData), **тип:** логический, **пример:** true; • docinfo — незашифрованная часть зашифрованного файла, **тип:** строковый, **пример:** «{docinfo}»; • hash — строковое значение, указывающее хэш файла (по умолчанию sha256), **тип:** строковый, **пример:** «sha256»; • error — строковое значение, указывающее возникшую ошибку (значение «» означает, что операция прошла успешно), **тип:** строковый, **пример:** «no_build».	объект

Возвращает

Метод возвращает *неопределенное* значение.

Пример

```
window.Asc.plugin.executeMethod("OnEncryption", [{"type": "getPasswordByFile", "hash": "sha256", "docinfo": "{docinfo}"}]);
```

MoveToNextReviewChange

window.Asc.plugin.executeMethod («MoveToNextReviewChange», [args], callback)

Определяет метод, который позволяет перемещаться по изменениям в обзоре.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("MoveToNextReviewChange", [isForward]);
```

Параметры

Параметр	Описание	Тип	Пример
isForward	Определяет, следует ли переходить к следующему (true) или предыдущему (false) изменению обзора. Значение по умолчанию — true.	логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("MoveToNextReviewChange", [true]);
```

MoveToComment

window.Asc.plugin.executeMethod («MoveToComment», [args], callback)

Определяет метод, позволяющий переместить курсор на указанный комментарий.

Применение

Этот метод следует использовать следующим образом:

window.Asc.plugin.executeMethod ("MoveToComment", [sId]);

Параметры

Параметр	Описание	Тип	Пример
sId	Идентификатор комментария.	строковый	«ID»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("MoveToComment", ["ID"]);
```

MoveCursorToStart

window.Asc.plugin.executeMethod («MoveCursorToStart», [args], callback)

Определяет метод, позволяющий переместить курсор в начальную позицию.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("MoveCursorToStart", [isMoveToMainContent]);
```

Параметры

Параметр	Описание	Тип	Пример
isMoveToMainContent	Определяет, перемещается ли курсор в начало документа (true) или в начало текущего элемента (false).	логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("MoveCursorToStart", [true]);
```

MoveCursorToEnd

window.Asc.plugin.executeMethod («MoveCursorToEnd», [args], callback)

Определяет метод, позволяющий переместить курсор в конечную позицию.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("MoveCursorToEnd", [isMoveToMainContent]);
```

Параметры

Параметр	Описание	Тип	Пример
isMoveToMainContent	Определяет, перемещается ли курсор в конец документа (true) или в конец текущего элемента (false).	логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("MoveCursorToEnd", [true]);
```

MoveCursorToContentControl

window.Asc.plugin.executeMethod («MoveCursorToContentControl», [args], callback)

Определяет метод, позволяющий перемещать курсор к указанному элементу управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("MoveCursorToContentControl", [InternalId, isBegin]);
```

Параметры

Параметр	**Описание**	**Тип**	**Пример**

InternalId	Уникальный внутренний идентификатор элемента управления содержимым.	строковый	«2_839»
isBegin	Определяет, изменяется ли положение курсора в элементе управления содержимым. По умолчанию курсор будет помещен в начало элемента управления содержимым (false).	логический	false

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("MoveCursorToContentControl", ["2_839", false]);
```

InstallPlugin

window.Asc.plugin.executeMethod («InstallPlugin», [args], callback)

Определяет метод, который позволяет установить плагин по URL-адресу конфигурации плагина.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("InstallPlugin", [url]);
```

Параметры

Параметр	Описание	Тип	Пример
url	URL-адрес конфигурации плагина для установки.	строковый	«https://example.com/plugin/config.json»

Возвращает

Метод возвращает объект с информацией о результате.

Пример

```
window.Asc.plugin.executeMethod ("InstallPlugin", ["https://example.com/plugin/config.json"]);
```

InsertOleObject

window.Asc.plugin.executeMethod («InsertOleObject», [args], callback)

Определяет метод, позволяющий вставить OLE — объект в текущую позицию документа.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("InsertOleObject", [NewObject, bSelect]);
```

Параметры

Параметр	Описание	Тип	Пример
NewObject	Объект OLEObjectData, содержащий следующие параметры: Объект OLEObjectData, позволяет получить параметры: • **Data** — Данные объекта OLE (внутренний формат), **тип:** строковый, **пример:** «{data}»; • ImageData — изображение в формате base64, хранящееся в объекте OLE и используемое подключаемым модулем, **тип:** строковый, **пример:** «-in-the-base64-format»; • ApplicationId -идентификатор подключаемого модуля, который может редактировать текущий объект OLE и должен быть типа asc.{UUID}, **тип:** строковый, **пример:** «asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}»; • InternalId -идентификатор OLE объекта, который используется для работы с объектом OLE, добавленным в документ, **тип:** строковый, **пример:** «5_556»; • ParaDrawingId — идентификатор объекта рисования, содержащего текущий OLE объект, **тип:** строковый, **пример:** «1_713»; • Width — ширина объекта OLE, измеренная в миллиметрах, **тип:** целочисленный, **пример:** 70; • Height — высота объекта OLE, измеренная в миллиметрах,	объект	

	тип: целочисленный, **пример:** 70; • WidthPix — ширина изображения объекта OLE в пикселях, **тип:** целочисленный, **пример:** 60 * 36000; • HeightPix -высота изображения объекта OLE в пикселях, **тип:** целочисленный, **пример:** 60 * 36000.		
Data — Данные объекта OLE (внутренний формат),	Определяет, будет ли OLE объект выбран после вставки в документ (true) или нет (false).	логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("InsertOleObject", [{"Data": "{data}", "ImageData": "-in-the-base64-format", "ApplicationId": "asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}", "Width": 70, "Height": 70, "WidthPix": 60 * 36000, "HeightPix": 60 * 36000}, true]);
```

InsertAndReplaceContentControls

window.Asc.plugin.executeMethod(«InsertAndReplaceContentControls», [args], callback)

Определяет метод, позволяющий вставить элемент управления содержимым, содержащий данные. Данные задаются js-кодом для Document Builder или ссылкой на общий документ.

Использование

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod("InsertAndReplaceContentControls", [arrDocuments]);
```

Параметры

Параметр	Описание	Тип
arrDocuments	Массив свойств и содержимого элемента управления содержимым. Каждый объект из этого массива может иметь следующие значения: • Props -свойства управления содержимым,	массив объектов

	тип: объект; • Script — скрипт, который будет выполняться для генерации данных в элементе управления содержимым (можно заменить параметром Url), **тип:** строковый, **пример:** «{script}»; • Url — ссылка на общий файл (может быть заменена параметром Script), **тип:** строковый, **пример:** «https://example.com/script.docbuilder».	

Объект Props может иметь следующие значения:

Параметры

Параметр	Описание	Тип	Пример
Id	Уникальный идентификатор элемента управления содержимым. Он может быть использован для поиска определенного элемента управления содержимым и ссылаться на него в коде.	целочисленный	2
Tag	Тег, назначенный элементу управления содержимым. Один и тот же тег может быть присвоен нескольким элементам управления содержимым, чтобы на них можно было ссылаться в коде.	строковый	«{tag}»
Lock	Значение, определяющее, можно ли удалять и/или редактировать элемент управления содержимым или нет.	целочисленный	0
InternalId	Уникальный внутренний идентификатор элемента управления содержимым.	строковый	«1_713»
Alias	Атрибут псевдонима.	строковый	«№1»
PlaceHolderText	Текст плейсхолдера элемента управления содержимым.	строковый	«placeholder text»
Appearance	Определяет, отображается ли элемент управления содержимым в виде ограничивающей рамки (1) или нет (2).	целочисленный	1
Color	Цвет текущего элемента управления содержимым в формате RGB: • **R** — значение компонента красного цвета, **тип:** целочисленный, **пример:** 0; • G — значение компонента зеленого цвета, **тип:** целочисленный, **пример:** 0; • B — значение компонента синего цвета,		

	тип: целочисленный, **пример:** 255;		

Параметр Lock может принимать следующие значения:

Численное значение	Редактирование	Удалить
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Возвращает

Метод возвращает данные, которые содержит созданный элемент управления содержимым (в формате JSON):

```
[
  {
    "Tag": "Document",
    "Id": 0,
    "Lock": 0,
    "InternalId": "1_713"
  }
]
```

Пример изменения существующего элемента управления содержимым

```
var arrDocuments = [{
  "Props": {
    "InternalId": "2_803"
  },
  "Script": "var oParagraph = Api.CreateParagraph();oParagraph.AddText('New text');Api.GetDocument().InsertContent([oParagraph]);"
}]
window.Asc.plugin.executeMethod("InsertAndReplaceContentControls", [arrDocuments]);
```

Пример добавления нового элемента управления содержимым

```
var arrDocuments = [{
  "Props": {
    "Id": 100,
    "Tag": "CC_Tag",
```

```
    "Lock": 3
  },
  "Script": "var oParagraph = Api.CreateParagraph();oParagraph.AddText('Hello world!');Api.GetDocument().InsertContent([oParagraph]);"
}]
window.Asc.plugin.executeMethod("InsertAndReplaceContentControls", [arrDocuments]);
```

InputText

window.Asc.plugin.executeMethod («InputText», [args], callback)

Определяет метод, позволяющий вставлять текст в документ.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("InputText", [text, textReplace]);
```

Параметры

Параметр	Описание	Тип	Пример
text	Строковое значение, определяющее текст, который будет вставлен в документ.	строковый	«P7 Plugins»
textReplace	Строковое значение, указывающее текст, который должен быть заменен.	строковый	«P7 for developers»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("InputText", ["P7 Plugins", "P7 for developers"]);
```

GetVersion

window.Asc.plugin.executeMethod («GetVersion», callback)

Определяет метод, позволяющий получить версию редактора.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetVersion");
```

Возвращает

Метод возвращает версию редактора в строковом формате.

Пример

```
window.Asc.plugin.executeMethod ("GetVersion", [], function(version) {
    if (version === undefined) {
        window.Asc.plugin.executeMethod ("PasteText", [ifr.contentDocument.getElementById
("google_translate_element").outerText]);
    }
    else {
        window.Asc.plugin.executeMethod ("GetSelectionType", [], function(sType) {
            switch (sType) {
                case "none":
                case "drawing":
                    window.Asc.plugin.executeMethod("PasteText",
[ifr.contentDocument.getElementById("google_translate_element").outerText]);
                    break;
                case "text":
                    window.Asc.plugin.callCommand(function() {
                        Api.ReplaceTextSmart(Asc.scope.arr);
                    });
                    break;
            }
        });
    }
});
```

GetSelectionType

window.Asc.plugin.executeMethod («GetSelectionType», callback)

Определяет метод, позволяющий получить тип текущего выделения.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetSelectionType");
```

Возвращает

Метод возвращает тип выделения в строковом формате: «нет», «текст», «рисунок» или «слайд».

Пример

```
window.Asc.plugin.executeMethod ("GetSelectionType", [], function(sType) {
    switch (sType) {
        case "none":
        case "drawing":
            window.Asc.plugin.executeMethod ("PasteText", [$("#txt_shower")[0].innerText]);
            break;
        case "text":
            window.Asc.plugin.callCommand (function() {
                Api.ReplaceTextSmart (Asc.scope.arr);
            });
            break;
    }
});
```

GetSelectedText

window.Asc.plugin.executeMethod («GetSelectedText», [args], callback)

Определяет метод, позволяющий получить выделенный текст из документа.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetSelectedText", [numbering]);
```

Параметры

Параметр	Описание	Тип

numbering	Определяет результирующие свойства отображения строки: • **NewLine** — определяет, будет ли результирующая строка включать границы строк или нет, **тип:** логический, **пример:** true; • NewLineParagraph — определяет, будет ли результирующая строка включать границы строки абзаца или нет, **тип:** логический, **пример:** true; • Numbering — определяет, будет ли результирующая строка включать нумерацию или нет, **тип:** логический, **пример:** true. • Math — определяет, будет ли результирующая строка включать математические выражения или нет, **тип:** логический, **пример:** true; • TableCellSeparator -определяет, как разделитель ячеек таблицы будет указан в результирующей строке, **тип:** строковый, **пример:** '\n'; • TableRowSeparator — определяет, как разделитель строк таблицы будет указан в результирующей строке, **тип:** строковый, **пример:** '\n'; • ParaSeparator — определяет, как будет указан разделитель абзаца в результирующей строке, **тип:** строковый, **пример:** '\n'; • TabSymbol -определяет, как вкладка будет указана в результирующей строке, **тип:** строковый, **пример:** '\t'.	массив объектов

Возвращает

Метод возвращает выделенный текст в строковом формате.

Пример

```
function CorrectText() {
  switch (window.Asc.plugin.info.editorType) {
    case 'word':
```

```
      case 'slide': {
         window.Asc.plugin.executeMethod("GetSelectedText", [{"Numbering": false, "Math": false,
"TableCellSeparator": '\n', "ParaSeparator": '\n', "TabSymbol": String.fromCharCode(9)}], function(data)
{
            sText = data;
            ExecTypograf(sText);
         });
         break;
      }
      case 'cell': {
         window.Asc.plugin.executeMethod("GetSelectedText", [{"Numbering": false, "Math": false,
"TableCellSeparator": '\n', "ParaSeparator": '\n', "TabSymbol": String.fromCharCode(9)}], function(data)
{
            if (data == ''){
               sText = sText.replace(/\t/g, '\n');
               ExecTypograf(sText);
            }
            else {
               sText = data;
               ExecTypograf(sText);
            }
         });
         break;
      }
   }
}
```

GetMacros

window.Asc.plugin.executeMethod («GetMacros», callback)

Определяет метод, позволяющий получить макросы документа.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetMacros");
```

Возвращает

Метод возвращает объект Macros, содержащий данные обо всех макросах из документа в следующем виде (JSON):

```
{
   "macrosArray" : array,
   "current" : number
}
```

Параметры

Параметр	Описание	Тип	Пример
macrosArray	Массив кодов макросов ([{«name»: «Macros1», «value»: «{macrosCode}»}]).	массив объектов	
current	Индекс текущих макросов.	численный	1

Пример

```
{"macrosArray": [{"name": "Macros 1","value": "(function()\n{oDocument = Api.GetDocument();\noParagraph = Api.CreateParagraph();\noParagraph.AddText(\"This is a new paragraph\");\noDocument.Push(oParagraph);\n})();"},{"name": "Macros 2","value": "(function()\n{oDocument = Api.GetDocument();\noParagraph = oDocument.GetElement(0);\noParagraph.AddText(\"Document Builder\");\noRange = oDocument.GetRange(0, 24);\noRange.SetBold(true);\n})();"}],"current": 1};
```

GetInstalledPlugins

window.Asc.plugin.executeMethod («GetInstalledPlugins», callback)

Определяет метод, позволяющий вернуть все установленные плагины.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetInstalledPlugins");
```

Возвращает

Метод возвращает массив всех установленных плагинов.

Пример

```
window.Asc.plugin.executeMethod ("GetInstalledPlugins");
```

GetImageDataFromSelection

window.Asc.plugin.executeMethod («GetImageDataFromSelection», callback)

Определяет метод, позволяющий получить данные изображения с первого из выбранных рисунков. Если чертежи не выбраны, метод возвращает белый прямоугольник.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetImageDataFromSelection");
```

Возвращает

Метод возвращает объект oImageData, содержащий информацию об изображении png, закодированном в base64:

```
{
   "src": string,
   "nWidth": number,
   "nHeight": number
}
```

Параметры

Параметр	Описание	Тип	Пример
src	Источник изображения в формате base64.	строковый	«-in-the-base64-format»
nWidth	Ширина изображения в пикселях.	численный	300
nHeight	Высота изображения в пикселях.	численный	200

Пример

```
window.Asc.plugin.executeMethod ("GetImageDataFromSelection", [], function(oResult) {
   oImage = document.createElement("img");
   oImage.src = oResult.src;
   oImage.width = oResult.width;
   oImage.height = oResult.height;
   CreateImageEditor();
   initializationDone = true;
   var imageHeight = null;
   oImage.height > 500 ? imageHeight = 500 : imageHeight = oImage.height;
   window.Asc.plugin.resizeWindow(undefined, undefined, 870, imageHeight + 300, 0, 0);
});
```

GetFormValue

window.Asc.plugin.executeMethod («GetFormValue», [args], callback)

Определяет метод, позволяющий получить значение заданной формы.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetFormValue", [internalId]);
```

Параметры

Параметр	Описание	Тип	Пример
internalId	Уникальный внутренний идентификатор формы.	строковый	«1_713»

Возвращает

Метод возвращает значение формы в строковом или логическом формате в зависимости от типа формы. Нулевое значение означает, что форма заполнена плейсхолдером.

Пример

```
window.Asc.plugin.executeMethod ("GetFormValue", ["1_713"]);
```

GetFormsByTag

window.Asc.plugin.executeMethod («GetFormsByTag», [args], callback)

Определяет метод, позволяющий вернуть информацию обо всех формах, которые были добавлены в документ с указанным тегом.

Использование

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetFormsByTag", [tag]);
```

Параметры

Параметр	Описание	Тип	Пример
tag	Тег формы.	строковый	«{tag}»

Возвращает

Метод возвращает массив со всеми формами из документа с указанным тегом.

Пример

```
window.Asc.plugin.executeMethod ("GetFormsByTag", ["{tag}"]);
```

GetFontList

window.Asc.plugin.executeMethod («GetFontList», callback)

Определяет метод, позволяющий получить список шрифтов.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetFontList");
```

Возвращает

Метод возвращает массив объектов FontInfo, содержащих данные об используемых шрифтах, в следующем виде (JSON):

```
{
   "m_wsFontName" : "string",
   "m_wsFontPath" : "string",
   "m_lIndex" : integer,
   "m_bBold": boolean,
   "m_bItalic" : boolean,
   "m_bIsFixed" : boolean,
   "m_aPanose" : integer[],
   "m_ulUnicodeRange1" : integer,
   "m_ulUnicodeRange2" : integer,
   "m_ulUnicodeRange3" : integer,
   "m_ulUnicodeRange4": integer,
   "m_ulCodePageRange1" : integer,
   "m_ulCodePageRange2" : integer,
   "m_usWeigth" : integer,
   "m_usWidth" : integer,
   "m_sFamilyClass" : integer,
   "m_eFontFormat" : integer,
   "m_shAvgCharWidth": integer,
   "m_shAscent" : integer,
```

```
  "m_shDescent" : integer,
  "m_shLineGap" : integer,
  "m_shXHeight" : integer,
  "m_shCapHeight" : integer
}
```

Параметры

Параметр	Описание	Тип	Пример
m_wsFontName	Название шрифта.	строковый	«Open Sans»
m_wsFontPath	Путь к файлу с текущим шрифтом.	строковый	«OpenSans-Bold.ttf»
m_lIndex	The font number in the file if there is more than one font in the file. Номер шрифта в файле, если в файле более одного шрифта.	целочисленный	0
m_bBold	Указывает, являются ли символы шрифта полужирными или нет.	логический	true
m_bItalic	Указывает, являются ли символы шрифта курсивными или нет.	логический	false
m_bIsFixed	Указывает, является ли текущий шрифт моноширинным или нет.	логический	false
m_aPanose	Классификационный номер шрифта PANOSE, компактное 10-байтовое описание важнейших визуальных характеристик шрифта, таких как контрастность, вес и стиль с засечками.	массив целых чисел	[2, 11, 8, 6, 3, 5, 4, 2, 2, 4]
m_ulUnicodeRange1	Диапазон Unicode, охватываемый файлом шрифта (биты 0–31).	целочисленный	3758097135
m_ulUnicodeRange2	Диапазон Unicode, охватываемый файлом шрифта (биты 32–63).	целочисленный	1073750107
m_ulUnicodeRange3	Диапазон Unicode, охватываемый файлом шрифта (биты 64–95).	целочисленный	40
m_ulUnicodeRange4	Диапазон Unicode, охватываемый файлом шрифта (биты 96-127).	целочисленный	0
m_ulCodePageRange1	Кодовые страницы, включенные в файл шрифта (биты 0–31).	целочисленный	536871327
m_ulCodePageRange2	Кодовые страницы, включенные в файл шрифта (биты 32–63).	целочисленный	0
m_usWeigth	Визуальный вес (чернота или толщина штриха) символов шрифта (1-1000).	целочисленный	700
m_usWidth	Относительное изменение нормального соотношения сторон (отношение ширины к высоте).	целочисленный	5
m_sFamilyClass	Класс семейства шрифтов, значения которого IBM	целочисленный	2050

	присваивает каждому семейству шрифтов..		
m_eFontFormat	Конкретные типы файлов, используемые для хранения данных шрифта: 0— *.fon, 1 — *.ttf, 2 — *.ttf, *.otf (CFF), 3 — неизвестный формат шрифта.	целочисленный	1
m_shAvgCharWidth	The arithmetic average of the escapement (width) of all non-zero width glyphs in the font.	целочисленный	632
m_shAscent	Высота над базовой линией области отсечения.	целочисленный	765
m_shDescent	Протяженность по вертикали ниже базовой линии области отсечения.	целочисленный	-240
m_shLineGap	Типографский межстрочный интервал для текущего шрифта.	целочисленный	64
m_shXHeight	Расстояние между базовой линией и приблизительной высотой невозрастающих строчных букв, измеренное в единицах FUnits.	целочисленный	545
m_shCapHeight	Расстояние между базовой линией и приблизительной высотой прописных букв, измеренное в единицах FUnit.	целочисленный	713

Пример

{"m_wsFontName": "Open Sans", "m_wsFontPath": "OpenSans-Bold.ttf", "m_lIndex": 0, "m_bBold": true, "m_bItalic": false, "m_bIsFixed": false, "m_aPanose": [2, 11, 8, 6, 3, 5, 4, 2, 2, 4],

"m_ulUnicodeRange1": 3758097135, "m_ulUnicodeRange2": 1073750107, "m_ulUnicodeRange3": 40, "m_ulUnicodeRange4": 0, "m_ulCodePageRange1": 536871327, "m_ulCodePageRange2": 0,

"m_usWeigth": 700, "m_usWidth": 5, "m_sFamilyClass": 2050, "m_eFontFormat": 1, "m_shAvgCharWidth": 632, "m_shAscent": 765, "m_shDescent": -240, "m_shLineGap": 64, "m_shXHeight": 545,

"m_shCapHeight": 713}

GetFileToDownload

window.Asc.plugin.executeMethod («GetFileToDownload», [args], callback)

Определяет метод, позволяющий вернуть текущий файл для его загрузки в указанном формате.

Использование

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetFileToDownload", [format]);
```

Параметры

Параметр	Описание	Тип	Пример
format	Формат, в котором вам нужно скачать файл. Значение по умолчанию — «».	строковый	«pdf»

Возвращает

Метод возвращает URL-адрес в строковом формате для загрузки файла в указанном формате или же ошибку.

Пример

```
window.Asc.plugin.executeMethod ("GetFileToDownload", ["pdf"]);
```

GetFileHTML

window.Asc.plugin.executeMethod («GetFileHTML», callback)

Определяет метод, позволяющий получить содержимое файла в формате HTML.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetFileHTML");
```

Возвращает

Метод возвращает содержимое HTML-файла в строковом формате.

GetFields

window.Asc.plugin.executeMethod («GetFields», callback)

Определяет метод, позволяющий получить все поля в формате текста.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetFields");
```

Возвращает

Метод возвращает список значений полей.

GetCurrentContentControlPr

Определяет метод, позволяющий получить текущие свойства элемента управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetCurrentContentControlPr", [contentFormat]);
```

Параметры

Параметр	Описание	Тип	Пример
contentFormat	Формат содержимого («нет», «текст», «html», «ole» или «рабочий стол»)	строковый	«text»

Возвращает

Метод возвращает объект ContentControlProperties, содержащий свойства элемента управления содержимым, в следующей форме (JSON):

```
{
   "Id" : integer,
   "Tag" : string,
   "Lock" : integer,
   "InternalId" : string,
   "Alias" : string,
   "PlaceHolderText" : string,
   "Appearance" : integer,
   "Color" : {
      "R": integer,
      "G": integer,
      "B": integer
   }
}
```

Параметры

Параметр	**Описание**	**Тип**	**Пример**
Id	Уникальный идентификатор элемента управления содержимым. Его можно	целочисленный	2

	использовать для поиска определенного элемента управления содержимым и ссылки на него в коде.		
Tag	Тег, назначенный элементу управления содержимым. Один и тот же тег может быть присвоен нескольким элементам управления содержимым, чтобы на них можно было ссылаться в коде.	строковый	«{tag}»
Lock	Значение, которое определяет, возможно ли удалить и/или изменить элемент управления содержимым или нет.	целочисленный	0
InternalId	Уникальный внутренний идентификатор элемента управления содержимым.	строковый	«1_713»
Alias	Атрибут псевдонима.	строковый	«№1»
PlaceHolderText	Текст-заполнитель элемента управления содержимым.	строковый	«placeholder text»
Appearance	Определяет, отображается ли элемент управления содержимым в виде ограничивающей рамки (1) или нет (2).	целочисленный	1
Color	Цвет текущего элемента управления содержимым в формате RGB: • **R** — значение компонента красного цвета, **тип:** целочисленный, **пример:** 0; • **G** -значение компонента зеленого цвета, **тип:** целочисленный, **пример:** 0; • **B** -значение компонента синего цвета, **тип:** целочисленный, **пример:** 255;	объект	

Параметр *Lock* может принимать следующие значения:

Числовое значение	Редактировать	Удалить
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Пример

```
window.Asc.plugin.event_onClick = function(isSelectionUse) {
  window.Asc.plugin.executeMethod("GetCurrentContentControlPr", [], function(obj) {
    window.Asc.plugin.currentContentControl = obj;
    var controlTag = obj ? obj.Tag : "";
    if (isSelectionUse)
```

```
        controlTag = "";
    ...
  });
};
```

GetCurrentContentControl

window.Asc.plugin.executeMethod («GetCurrentContentControl», callback)

Определяет метод, позволяющий получить идентификатор выбранного элемента управления содержимым (то есть элемента управления содержимым, в котором в данный момент находится курсор мыши).

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetCurrentContentControl");
```

Возвращает

Метод возвращает внутренний идентификатор элемента управления содержимым в строковом формате.

Пример

```
document.getElementById("buttonIDChangeState").onclick = function () {
  var _Control = [];
  window.buttonIDChangeState_click = true;
  window.Asc.plugin.executeMethod("GetCurrentContentControl");
};
```

GetAllOleObjects

window.Asc.plugin.executeMethod («GetAllOleObjects», [args], callback)

Определяет метод, позволяющий получить все данные объекта OLE для объектов, которые могут быть открыты указанным плагином. Если sPluginIdis не определен, этот метод возвращает все объекты OLE, содержащиеся в текущем документе.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetAllOleObjects", [sPluginId]);
```

Параметры

Параметр	Название	Тип	Пример
sPluginId	Идентификатор плагина. Он должен быть типа asc.{UUID}.	строковый	«asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}»

Возвращает

Метод возвращает массив объектов OLEObjectData, содержащих данные о параметрах объекта OLE:

```
{
   "Data" : string,
   "ImageData" : string,
   "ApplicationId" : string,
   "InternalId" : string,
   "ParaDrawingId" : string,
   "Width" : integer,
   "Height" : integer,
   "WidthPix" : integer,
   "HeightPix" : integer
}
```

Параметры

Параметр	Описание	Тип	Пример
Data	Данные объекта OLE (внутренний формат).	строковый	«{data}»
ImageData	Изображение в формате base64, хранящееся в объекте OLE и используемое подключаемым модулем.	строковый	«-in-the-base64-format»
ApplicationId	Идентификатор плагина, который может редактировать текущий OLE-объект, должен иметь тип asc.{UUID}.	строковый	«asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}»
InternalId	Идентификатор объекта OLE, который используется для работы с объектом OLE, добавленным в документ.	строковый	«5_556»
ParaDrawingId	Идентификатор объекта рисования, содержащего текущий объект OLE.	строковый	«1_713»
Width	Ширина объекта OLE измеряется в миллиметрах.	целочисленный	70
Height	Высота объекта OLE измеряется в миллиметрах.	целочисленный	70

WidthPix	Ширина изображения объекта OLE в пикселях.	целочисленный	60 * 36000
HeightPix	Высота изображения объекта OLE в пикселях.	целочисленный	60 * 36000

Пример

```
window.Asc.plugin.executeMethod("GetAllOleObjects", ["asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}"]);
```

GetAllForms

window.Asc.plugin.executeMethod («GetAllForms», callback)

Определяет метод, позволяющий вернуть информацию обо всех формах, которые были добавлены в документ.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetAllForms");
```

Возвращает

Метод возвращает массив со всеми формами из документа.

Пример

```
window.Asc.plugin.executeMethod ("GetAllForms");
```

GetAllContentControls

window.Asc.plugin.executeMethod («GetAllContentControls», callback)

Определяет метод, позволяющий получить информацию обо всех элементах управления содержимым, которые были добавлены на страницу.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetAllContentControls");
```

Возвращает

Метод возвращает массив объектов управления содержимым:

```
[
  {
    "Tag": "Document",
    "Id": 0,
    "Lock": 0,
    "InternalId": "1_713"
  }
]
```

Параметры

Параметр	Описание	Тип	Пример	Параметр
Tag	Тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы можно было ссылаться на них в коде.	строковый	«{tag}»	Tag
Id	Уникальный идентификатор элемента управления содержимым. Его можно использовать для поиска определенного элемента управления содержимым и ссылаться на него в коде.	целочисленный	2	Id
Lock	Значение, которое определяет, возможно ли удалить и/или изменить элемент управления содержимым или нет.	целочисленный	0	Lock

Пример

```
var flagInit = false;
window.Asc.plugin.init = function (text) {
  if (!flagInit) {
    this.executeMethod ("GetAllContentControls", null, function(data) {
      for (var i = 0; i < data.length; i++) {
        if (data[i].Tag == 11) {
          this.Asc.plugin.executeMethod("SelectContentControl",[data[i].InternalId]);
          break;
        }
      }
    });
    flagInit = true;
```

```
    ...
  }
};
```

GetAllComments

window.Asc.plugin.executeMethod («GetAllComments», callback)

Определяет метод, позволяющий получить все комментарии из документа.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("GetAllComments");
```

Возвращает

Метод возвращает массив объектов комментариев, содержащих данные комментариев, в следующем виде:

```
{
  "Id": "1_631",
  "Data": {
    "UserName": "John Smith",
    "Text": "comment",
    "Time": "1662737941471",
    "Solved": true,
    "Replies": [{"UserName": "Mark Potato", "Text": "reply 1", "Time": "1662740895892", "Solved": false}]
  }
}
```

Параметры

Параметр	Описание	Тип	Пример
Id	Идентификатор комментария.	строковый	«1_631»
Data	Объект, который содержит данные комментария: • **UserName** — автор комментария, **тип:** строковый, **пример:** «John Smith»; • **Text** — текст комментария, **тип:** строковый, **пример:** «comment»; • **Time** — время публикации комментария (в миллисекундах), **тип:** строковый, **пример:** «1662737941471»;		

	• **Solved** -указывает, разрешен ли комментарий (true) или нет (false), **тип:** логический, **пример:** true; • **Replies** — массив, содержащий ответы на комментарии, представленные в виде объектов oCommentData, **тип:** массив объектов.		

EndAction

window.Asc.plugin.executeMethod («EndAction», [args], callback)

Определяет метод, позволяющий указать конечное действие для длительных операций.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("EndAction", [type, description]);
```

Параметры

Параметр	Описание	Тип	Пример
type	Значение, определяющее тип действия, которое может принимать значение 0, если это действие Information, или 1, если это действие BlockInteraction.	численный	1
description	Строковое значение, задающее текст описания для действия окончания операции.	строковый	«Save to local storage…»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("EndAction", [1, "Save to local storage..."]);
```

EditOleObject

window.Asc.plugin.executeMethod («EditOleObject», [args], callback)

Определяет метод, позволяющий редактировать объект OLE в документе.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("EditOleObject", [data]);
```

Параметры

Параметр	Описание	Тип
data	Свойства объекта OLE: • **data** — Данные объекта OLE (внутренний формат), **тип:** строковый, **пример:** «{data}»; • **imgSrc** — ссылка на изображение (его визуальное представление), хранящееся в OLE-объекте и используемое плагином, **тип:** строковый, **пример:** «https://link-to-the-image.jpg»; • **objectId** — идентификатор объекта OLE, **тип:** строковый, **пример:** «5_556»; • **width** — ширина объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • **height** — высота объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • **widthPix** — ширина изображения объекта OLE в пикселях **тип:** целое число, **пример:** 60 * 36000; • **heightPix** — ширина изображения объекта OLE в пикселях, **тип:** целое число, **пример:** 60 * 36000.	объект

Возвращает

Метод возвращает *неопределенное* значение.

Пример

```
window.Asc.plugin.executeMethod("EditOleObject", [{"data": "{data}", "imgSrc": "https://link-to-the-image.jpg", "objectId": "5_556", "width": 70, "height": 70, "widthPix": 60 * 36000, "heightPix": 60 * 36000}]);
```

AcceptReviewChanges

window.Asc.plugin.executeMethod («AcceptReviewChanges», [args], callback)

Определяет метод, позволяющий принимать изменения в обзоре.

Использование

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("AcceptReviewChanges", [isAll]);
```

Параметры

Параметр	Описание	Тип	Пример
isAll	Указывает, будут ли приняты все изменения (true) или только изменения текущего выбора (false). Значение по умолчанию false.	логический	true

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("AcceptReviewChanges", [true]);
```

ConvertDocument

window.Asc.plugin.executeMethod («ConvertDocument», [args], callback)

Определяет метод, который позволяет преобразовать документ в текст Markdown или HTML.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("ConvertDocument", [sConvertType, bHtmlHeadings, bBase64img, bDemoteHeadings, bRenderHTMLTags]);
```

Параметры

Параметр	**Описание**	**Тип**	**Пример**
sConvertType	Тип конвертации («markdown» или «html»).	строковый	«markdown»

bHtmlHeadings	Определяет, будут ли создаваться заголовки и идентификаторы HTML, когда средство визуализации Markdown вашей целевой платформы не обрабатывает идентификаторы в стиле Markdown.	логический	false
bBase64img	Определяет, будут ли изображения создаваться в формате base64.	логический	false
bDemoteHeadings	Определяет, будут ли понижены все уровни заголовков в вашем документе для соответствия следующему стандарту: один H1 в качестве заголовка, H2 в качестве заголовка верхнего уровня в основном тексте.	логический	true
bRenderHTMLTags	Определяет, будут ли сохранены теги HTML в вашем Markdown. Если вы просто хотите использовать случайный HTML-тег, вы можете избежать использования открывающей угловой скобки следующим образом: \<tag>text\</tag>. По умолчанию угол раскрытия	логический	false

Возвращает

Метод возвращает текст Markdown/HTML в строковом формате.

Пример

```
var sInfo = "";
window.Asc.plugin.executeMethod ("ConvertDocument", ["markdown", false, false, true, false], function (sOutput) {
    document.getElementById ("text-area").value = sInfo + sOutput;
});
```

CoAuthoringChatSendMessage

window.Asc.plugin.executeMethod («CoAuthoringChatSendMessage», [args], callback)

Определяет метод, позволяющий отправить сообщение в чат соавторства.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("CoAuthoringChatSendMessage", [sText]);
```

Параметры

Параметр	Описание	Тип	Пример
sText	Текст сообщения.	строковый	«message»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("CoAuthoringChatSendMessage", ["message"]);
```

CoAuthoringChatSendMessage

window.Asc.plugin.executeMethod («CoAuthoringChatSendMessage», [args], callback)

Определяет метод, позволяющий отправить сообщение в чат соавторства.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("CoAuthoringChatSendMessage", [sText]);
```

Параметры

Параметр	Описание	Тип	Пример
sText	Текст сообщения.	строковый	«message»

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("CoAuthoringChatSendMessage", ["message"]);
```

ChangeOleObjects

window.Asc.plugin.executeMethod («ChangeOleObjects», [args], callback)

Определяет метод, позволяющий изменять несколько объектов OLE с помощью InternalIds, указанных в данных объекта OLE.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("ChangeOleObjects", [arrObjectData]);
```

Параметры

Параметр	Описание	Тип
arrObjectData	Массив данных объекта OLE, который содержит следующие параметры: • **Data** — Данные объекта OLE (внутренний формат), **тип:** строковый, **пример:** «{data}»; • ImageData -изображение в формате base64, хранящееся в объекте OLE и используемое подключаемым модулем, **тип:** строковый, **пример:** «-in-the-base64-format»; • **ApplicationId** -идентификатор подключаемого модуля, который может редактировать текущий объект OLE и должен быть типа asc.{UUID}, **тип:** строковый, **пример:** «asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}»; • InternalId — идентификатор OLE — объекта OLE, который используется для работы с OLE — объектом, добавленным в документ, **тип:** строковый, **пример:** «5_556»; • ParaDrawingId — идентификатор объекта рисования, содержащего текущий объект OLE, **тип:** строковый, **пример:** «1_713»; • Width — ширина объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • Height -высота объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • WidthPix — ширина изображения объекта OLE в пикселях, **тип:** целое число, **пример:** 60 * 36000; • HeightPix — высота изображения объекта OLE в пикселях, **тип:** целое число, **пример:** 60 * 36000.	массив объектов

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("ChangeOleObjects", [[{"Data": "{data}", "ImageData": "-in-the-base64-format", "ApplicationId": "asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}", "InternalId": "5_556", "ParaDrawingId": "1_713", "Width": 70, "Height": 70, "WidthPix": 60 * 36000, "HeightPix": 60 * 36000}]]);
```

ChangeOleObject

window.Asc.plugin.executeMethod («ChangeOleObject», [args], callback)

Определяет метод, позволяющий изменить объект OLE с помощью InternalId, указанного в данных объекта OLE.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("ChangeOleObject", [ObjectData]);
```

Параметры

Параметр	Описание	Тип
ObjectData	Объект OLEObjectData, содержащий следующие параметры: • **Data** — Данные объекта OLE (внутренний формат), **тип:** строковый, **пример:** «{data}»; • **ImageData** — изображение в формате base64, хранящееся в объекте OLE и используемое подключаемым модулем, **тип:** строковый, **пример:** «-in-the-base64-format»; • **ApplicationId** — идентификатор плагина, который может редактировать текущий объект OLE и должен быть типа asc.{UUID}, **тип:** строковый, **пример:** «asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}»; • **InternalId** -идентификатор объекта OLE, который используется для работы с объектом OLE, добавленным в документ, **тип:** строковый, **пример:** «5_556»; • **ParaDrawingId** — идентификатор объекта рисования, содержащего текущий объект OLE, **тип:** строковый,	объект

	пример: «1_713»; • **Width** -ширина объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • **Height** -высота объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • **WidthPix** -ширина изображения объекта OLE в пикселях, **тип:** целое число, **пример:** 60 * 36000; • **HeightPix** — высота изображения объекта OLE в пикселях, **тип:** целое число, **пример:** 60 * 36000.	

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("ChangeOleObject", [{"Data": "{data}", "ImageData": "-in-the-base64-format", "ApplicationId": "asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}", "InternalId": "5_556", "ParaDrawingId": "1_713", "Width": 70, "Height": 70, "WidthPix": 60 * 36000, "HeightPix": 60 * 36000}]);
```

ChangeComment

window.Asc.plugin.executeMethod («ChangeComment», [args], callback)

Определяет метод, позволяющий изменить указанный комментарий.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("ChangeComment", [sId, oCommentData]);
```

Параметры

Параметр	Описание	Тип	Пример
sId	Идентификатор комментария.	строковый	«ID»
oCommentData	Объект, который содержит новые данные комментария: • **UserName** -автор комментария, **тип:** строковый, **пример:** «John Smith»; • **Text** — текст комментария, **тип:** строковый,		

	пример: «comment»; • **Time** — время, когда был опубликован комментарий, **тип:** строковый, **пример:** «1662737941471»; • **Solved** -указывает, разрешен ли комментарий (true) или нет (false), **тип:** логический, **пример:** true; • **Replies** -массив, содержащий ответы на комментарии, представленные в виде объектов oCommentData, **тип:** массив объектов		

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("ChangeComment", ["ID", {"UserName": "John Smith", "Text": "comment", "Time": "1662737941471", "Solved": true, "Replies": [{"UserName": "Mark Potato", "Text": "reply 1", "Time": "1662740895892", "Solved": false}]}]);
```

AddOleObject

window.Asc.plugin.executeMethod («AddOleObject», [args], callback)

Определяет метод, позволяющий добавить объект OLE в текущую позицию документа.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("AddOleObject", [data]);
```

Параметры

Параметр	Описание	Тип
data	Свойства объекта OLE: • **data** — Данные объекта OLE (внутренний формат), **тип:** строковый, **пример:** «{data}»; • **imgSrc** — ссылка на изображение (его визуальное представление), хранящееся в OLE-объекте и используемое плагином, **тип:** строковый, **пример:** «https://link-to-the-image.jpg»;	объект

	• **guid** — идентификатор плагина, который может редактировать текущий объект OLE и должен быть типа asc.{UUID}, **тип:** строковый, **пример:** «asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}»; • width — ширина объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • **height** — высота объекта OLE, измеренная в миллиметрах, **тип:** целое число, **пример:** 70; • **widthPix** — ширина изображения объекта OLE в пикселях, **тип:** целое число, **пример:** 60 * 36000; • **heightPix** — высота изображения объекта OLE в пикселях, **тип:** целое число, **пример:** 60 * 36000	

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod("AddOleObject", [{"data": "{data}", "imgSrc": "https://link-to-the-image.jpg", "guid": "asc.{38E022EA-AD92-45FC-B22B-49DF39746DB4}", "width": 70, "height": 70, "widthPix": 60 * 36000, "heightPix": 60 * 36000}]);
```

AddContentControlPicture

window.Asc.plugin.executeMethod («AddContentControlPicture», [args], callback)

Определяет метод, позволяющий добавить в документ пустое изображение для управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("AddContentControlPicture", [commonPr]);
```

Параметры

Параметр	Описание	Тип
commonPr	Определяет общие свойства элемента управления содержимым: • Id — никальный идентификатор элемента управления содержимым. Его можно использовать для поиска	объект

	определенного элемента управления содержимым и ссылки на него в коде. **тип:** целое число, **пример:** 2; • **Tag** — тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы можно было ссылаться на них в коде. **тип:** строковый, **пример:** «{tag}»; • Lock — значение, которое определяет, возможно ли удалить и/или отредактировать элемент управления содержимым или нет, **тип:** целое число, **пример:** 0; • **InternalId** -уникальный внутренний идентификатор элемента управления контентом, **тип:** строковый, **пример:** «1_713»; • **Alias** — атрибут псевдонима, **тип:** строковый, **пример:** «№1»; • **PlaceHolderText** — текст заполнителя элемента управления содержимым **тип:** строковый, **пример:** «placeholder text»; • **Appearance** — определяет, отображается ли элемент управления содержимым в виде ограничивающей рамки (1) или нет (2), **тип:** целое число, **пример:** 1; • **Color** — цвет для текущего элемента управления содержимым в формате RGB (R — значение составляющей красного цвета, G — значение составляющей зеленого цвета, B — значение составляющей синего цвета). Например: {«R»: 0, «G»: 0, «B»: 255}, **тип:** объект	

Параметр *Lock* может принимать следующие значения:

Числовое значение	Редактировать	Удалить
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("AddContentControlPicture", [{"Id" : 7, "Tag" : "{tag}", "Lock" : 0}]);
```

AddContentControlList

window.Asc.plugin.executeMethod («AddContentControlList», [args], callback)

Определяет метод, позволяющий добавить в документ пустой список управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("AddContentControlList", [type, List, commonPr]);
```

Параметры

Параметр	Описание	Тип	Пример
type	Числовое значение, указывающее тип элемента управления содержимым. Может принимать одно из следующих значений: 1 (comboBox) или 0 (раскрывающийся список).	численный	0
List	Список элементов управления содержимым, состоящий из двух элементов: • **Display** -элемент, который будет отображаться пользователю в списке управления содержимым, **тип:** строковый, **пример:** «Item1_D»; • **Value** -значение каждого элемента из списка управления содержимым, **тип:** строковый, **пример:** «Item1_V».	массив объектов	
commonPr	Определяет общие свойства элемента управления содержимым: • **Id** — уникальный идентификатор элемента управления содержимым. Его можно использовать для поиска определенного элемента управления содержимым и ссылки на него в коде. **тип:** целое число, **пример:** 2; • **Tag** — тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить	объект	

	нескольким элементам управления содержимым, чтобы можно было ссылаться на них в коде. **тип:** строковый, **пример:** «{tag}»; • **Lock** -значение, которое определяет, возможно ли удалить и/или отредактировать содержимоecontrol or not, **тип:** целое число, **пример:** 0; • **InternalId** — уникальный внутренний идентификатор элемента управления контентом, **тип:** строковый, **пример:** «1_713»; • **Alias** -атрибут псевдонима, **тип:** строковый, **пример:** «№1»; • **PlaceHolderText** -текст заполнителя элемента управления содержимым, **тип:** строковый, **пример:** «placeholder text»; • **Appearance** -определяет, отображается ли элемент управления содержимым в виде ограничивающей рамки (1) или нет (2), **тип:** целое число, **пример:** 1; • **Color** — цвет для текущего элемента управления содержимым в формате RGB (R — значение составляющей красного цвета, G — значение составляющей зеленого цвета, B — значение составляющей синего цвета). Например: {«R»: 0, «G»: 0, «B»: 255}, **тип:** объект.		

Параметр *Lock* может принимать следующие значения:

Числовое значение	Редактировать	Удалить
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("AddContentControlList", [0, [{Display: "Item1_D", Value: "Item1_V"}, {Display: "Item2_D", Value: "Item2_V"}], {"Id" : 7, "Tag" : "{tag}", "Lock" : 0}]);
```

AddContentControlDatePicker

window.Asc.plugin.executeMethod («AddContentControlDatePicker», [args], callback)

Определяет метод, который позволяет добавить в документ указатель даты элемента управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("AddContentControlDatePicker", [datePickerPr, commonPr]);
```

Параметры

Параметр	Описание	Тип
datePickerPr	Определяет свойства выбора даты элемента управления содержимым: • **DateFormat** — формат, в котором будет отображаться дата. Например:«MM/DD/YYYY», «dddd\,\ mmmm\ dd\,\ yyyy», «DD\ MMMM\ YYYY», «MMMM\ DD\,\ YYYY», «DD-MMM-YY», «MMMM\ YY», «MMM-YY», «MM/DD/YYYY\ hh:mm\ AM/PM», «MM/DD/YYYY\ hh:mm:ss\ AM/PM», «hh:mm», «hh:mm:ss», «hh:mm\ AM/PM», «hh:mm:ss:\ AM/PM», **тип:** строковый, **пример:** «DD\ MMMM\ YYYY»; • **Date** — текущая дата и время, **тип:** объект, **пример:** Date	объект
commonPr	Определяет общие свойства элемента управления содержимым: • **Id** — уникальный идентификатор элемента управления содержимым. Его можно использовать для поиска определенного элемента управления содержимым и ссылки на него в коде. **тип:** целое число, **пример:** 2; • **Tag** — тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы можно было ссылаться на них в коде. **тип:** строковый, **пример:** «{tag}»; • **Lock** — значение, которое определяет, возможно ли удалить и/или отредактировать элемент управления содержимым или нет, **тип:** целое число, **пример:** 0; • **InternalId** -уникальный внутренний идентификатор элемента управления контентом,	объект

	тип: строковый, **пример:** «1_713»; • **Alias** — атрибут псевдонима, **тип:** строковый, **пример:** «№1»; • **PlaceHolderText** -текст заполнителя элемента управления содержимым **тип:** строковый, **пример:** «placeholder text»; • Appearance — определяет, отображается ли элемент управления содержимым в виде ограничивающей рамки (1) или нет (2) **тип:** целое число, **пример:** 1; • **Color** — цвет для текущего элемента управления содержимым в формате RGB (R — значение составляющей красного цвета, G — значение составляющей зеленого цвета, B — значение составляющей синего цвета). Например: {«R»: 0, «G»: 0, «B»: 255}, **тип:** объект	

Параметр *Lock* может принимать следующие значения:

Числовое значение	Редактировать	Удалить
0	нет	да
1	нет	нет
2	да	нет
3	да	да

Возвращает

Метод возвращает неопределенное значение.

Пример

```
var Date = new window.Date();

window.Asc.plugin.executeMethod ("AddContentControlDatePicker", [{"DateFormat" : "DD\ MMMM\ YYYY", "Date" : Date}, {"Id" : 7, "Tag" : "{tag}", "Lock" : 0}]);
```

AddContentControlCheckBox

window.Asc.plugin.executeMethod («AddContentControlCheckBox», [args], callback)

Определяет метод, позволяющий добавить в документ пустой флажок управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("AddContentControlCheckBox", [checkBoxPr, commonPr]);
```

Параметры

Параметр	Описание	Тип
checkBoxPr	Определяет свойства флажка управления содержимым: • **Checked** — определяет, установлен ли флажок управления содержимым или нет, **тип:** логический, **пример:** false; • **CheckedSymbol** — символ в формате кода HTML, который используется при установленном флажке, **тип:** численный, **пример:** 9746; • **UncheckedSymbol** — символ в формате кода HTML, который используется, когда флажок не установлен, **тип:** численный, **пример:** 9744.	объект
commonPr	Определяет общие свойства элемента управления содержимым: • **Id** — уникальный идентификатор элемента управления содержимым. Его можно использовать для поиска определенного элемента управления содержимым и ссылки на него в коде. **тип:** целое число, **пример:** 2; • **Tag** — тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы можно было ссылаться на них в коде. **тип:** строковый, **пример:** «{tag}»; • **Lock** — значение, которое определяет, возможно ли удалить и/или отредактировать элемент управления содержимым или нет, **тип:** целое число, **пример:** 0; • **InternalId** — уникальный внутренний идентификатор элемента управления контентом, **тип:** строковый, **пример:** «1_713»; • **Alias** — атрибут псевдонима, **тип:** строковый, **пример:** «№1»; • **PlaceHolderText** — текст заполнителя элемента управления содержимым **тип:** строковый, **пример:** «placeholder text»;	объект

	• **Appearance** — определяет, отображается ли элемент управления содержимым в виде ограничивающей рамки (1) или нет (2), **тип:** целое число, **пример:** 1; • **Color** — цвет для текущего элемента управления содержимым в формате RGB (R — значение составляющей красного цвета, G — значение составляющей зеленого цвета, B — значение составляющей синего цвета). Например: {«R»: 0, «G»: 0, «B»: 255}, **тип:** объект.	

Параметр *Lock* может принимать следующие значения:

Числовое значение	Редактировать	Удалить
0	Нет	Да
1	Нет	Нет
2	Да	Нет
3	Да	Да

Возвращает

Метод возвращает неопределенное значение.

Пример

```
window.Asc.plugin.executeMethod ("AddContentControlCheckBox", [{"Checked" : false, "CheckedSymbol" : 9756, "UncheckedSymbol" : 9744}, {"Id" : 7, "Tag" : "{tag}", "Lock" : 0}]);
```

AddContentControl

window.Asc.plugin.executeMethod («AddContentControl», [args], callback)

Определяет метод, позволяющий добавить в документ пустой элемент управления содержимым.

Применение

Этот метод следует использовать следующим образом:

```
window.Asc.plugin.executeMethod ("AddContentControl", [type, commonPr]);
```

Параметры

Параметр	Описание	Тип	Пример
type	Числовое значение, указывающее тип элемента управления содержимым. Он может иметь одно из	целое число	2

	следующих значений: 1 (блок), 2 (встроенный), 3 (строка) или 4 (ячейка).		
commonPr	Определяет общие свойства элемента управления содержимым: • **Id** — уникальный идентификатор элемента управления содержимым. Его можно использовать для поиска определенного элемента управления содержимым и ссылки на него в коде. **тип:** целое число, **пример:** 2; • **Tag** — тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы можно было ссылаться на них в коде. **тип:** строковый, **пример:** «{tag}»; • **Lock** — значение, которое определяет, возможно ли удалить и/или изменить элемент управления содержимым или нет, **тип:** целое число, **пример:** 0; • **InternalId** — уникальный внутренний идентификатор элемента управления контентом, **тип:** строковый, **пример:** «1_713»; • **Alias** -атрибут псевдонима, **тип:** строковый, **пример:** «№1»; • **PlaceHolderText** — текст-заполнитель элемента управления содержимым, **тип:** строковый, **пример:** «placeholder text»; • **Appearance** — определяет, отображается ли элемент управления содержимым в виде ограничивающей рамки (1) или нет (2), **тип:** целое число, **пример:** 1; • **Color** -цвет для текущего элемента управления содержимым в формате RGB (R — значение составляющей красного цвета, G — значение составляющей зеленого цвета, B — значение составляющей синего цвета). Например: {«R»: 0, «G»: 0, «B»: 255}, **тип:** объект.	объект	

Параметр *Lock* может принимать следующие значения:

Числовое значение	Редактировать	Удалить
0	Нет	Да
1	Нет	Нет

2	Да	Нет
3	Да	Да

Возвращает

Метод возвращает JSON — объект, содержащий данные о созданном элементе управления содержимым, в следующем виде:

```
{
   "Tag": "{tag}",
   "Id": 0,
   "Lock": 0,
   "InternalId": "1_713"
}
```

Параметры

Параметр	Описание	Тип	Пример
Tag	Тег, назначенный элементу управления содержимым. Один и тот же тег можно назначить нескольким элементам управления содержимым, чтобы можно было ссылаться на них в коде.	строковый	«{tag}»
Id	Уникальный идентификатор элемента управления содержимым. Его можно использовать для поиска определенного элемента управления содержимым и ссылки на него в коде.	целое число	2
Lock	Значение, которое определяет, возможно ли удалить и/или изменить элемент управления содержимым или нет.	целое число	0
InternalId	Уникальный внутренний идентификатор элемента управления содержимым.	строковый	«1_713»

Пример

```
window.Asc.plugin.executeMethod ("AddContentControl", [1, {"Id" : 7, "Tag" : "{tag}", "Lock" : 0}]);
```

Параметры

Параметр	Описание	Тип
oCommentData	Объект, который содержит данные комментария: • **UserName** — автор комментария, **тип:** сроковый, **пример:** «John Smith»; • **Text** — текст комментария,	объект

	тип: строковый, **пример:** «comment»; • **Time** — время публикации комментария (в миллисекундах), **тип:** строковый, **пример:** «1662737941471»; • **Solved** — указывает, разрешен ли комментарий (true) или нет (false), **тип:** логический, **пример:** true; • **Replies** — массив, содержащий ответы на комментарии, представленные в виде объектов oCommentData, **тип:** массив объектов.	

Возвращает

Метод возвращает идентификатор комментария в строковом формате или *null*, если комментарий не может быть добавлен.

Пример

```
window.Asc.plugin.executeMethod ("AddComment", [{"UserName": "John Smith", "Text": "comment", "Time": "1662737941471", "Solved": true, "Replies": [{"UserName": "Mark Potato", "Text": "reply 1", "Time": "1662740895892", "Solved": false}]}]);
```