Р7-ОФИС

Руководство по работе с макросами

Введение

В редакторах **Р7-Офис** макросы - это это скрипты, с помощью которых можно автоматизировать рутинные операции с разными типами документов. В макросах используется синтаксис языка JavaScript и скрипты API Генератора документов, поэтому макросы поддерживают методы, доступные в JavaScript и методы, которые поддерживает Генератор документов.

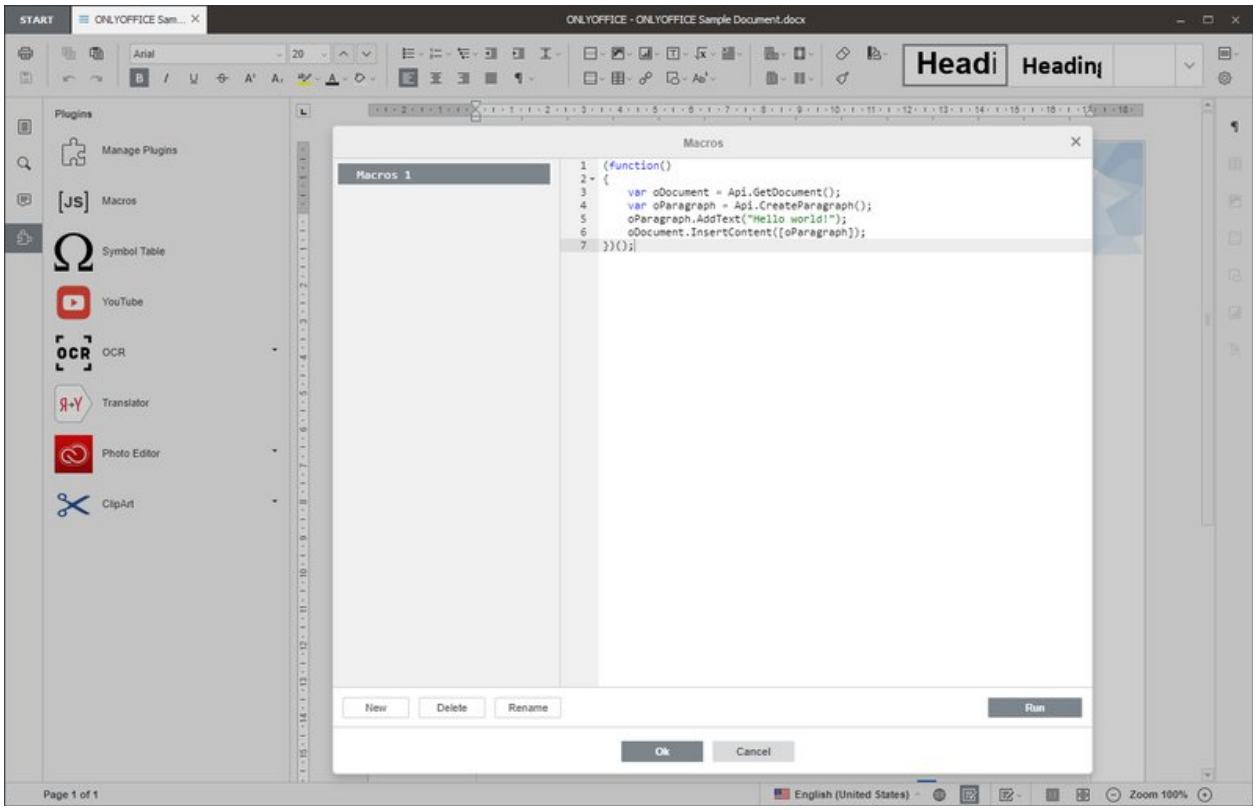
Вы можете добавлять в документы свои собственные макросы (поддерживаются все типы документов: текстовые документы, электронные таблицы , презентации), редактировать их и сохранять, чтобы сделать работу с документами еще проще и удобнее.

Начало работы с макросами

Макрос - это функция JavaScript, связанная с документом. Самая простая функция, при запуске вставляющая в документ текст "Hello world!", выглядит следующим образом:

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

В интерфейсе десктопных редакторов это выглядит так:



Для доступа к макросу, откройте или создайте документ нужного типа, нажмите кнопку плагинов и выберите **Макросы**. Откроется окно макросов. Нажмите **Создать** и введите ваш код скрипта в окно с правой стороны. Когда закончите, нажмите **Выполнить** для запуска кода в документе.

Вы также можете переименовать ваш макрос для того, чтобы отличать его от других в случае, если в документе несколько макросов, либо удалить ненужные.

Написание собственных макросов

Теперь, когда вы знаете, как работают макросы, давайте попробуем написать собственный макрос. К примеру, у нас есть таблица, в которой нужно раскрасить строки в разные цвета: нечётные строки в зелёный, а чётные в красный. В таблице 200 строк (если делать всё вручную, это займёт немало времени) и столбцы от **A** до **S**.

1. Откройте десктопные редакторы и создайте новую таблицу.
2. Откройте плагины и выберите **Макросы**. Откроется окно макросов.
3. Нажмите **Создать**. Создастся базовый шаблон функции, в который можно вставить нужный код:

```
(function()
{
    // . . . ваш код здесь. . .
})();
```

4. Что нужно для выполнения нашей задачи:

- Сначала нужно получить текущий лист с помощью метода **GetActiveSheet**:

```
var oWorksheet = Api.GetActiveSheet();
```

- Затем создадим цикл, который пройдёт от первого до последней строки таблицы:

```
for (var i = 1; i < 200; i += 2) {
}
```

- Теперь зададим две переменных: одну для нечётных строк, вторую для чётных:

```
var row0dd = i, rowEven = i + 1;
```

- Теперь мы можем раскрасить нечётные и чётные строки в нужные цвета. Установим эти цвета с помощью метода **CreateColorFromRGB**. Диапазон ячеек внутри ряда можно получить с помощью метода **GetRange**, цвет для нечётных строк устанавливается так:

```
oWorksheet.GetRange("A" + rowOdd + ":S" + row0dd) .SetFillColor(Api.CreateColorFromRGB(118, 190,
39));
```

То же самое для чётных строк, но цвет будет другим:

```
oWorksheet.GetRange("A" + rowEven + ":S" + rowEven).SetFillColor(Api.CreateColorFromRGB(186, 56,
46));
```

Давайте подведём итог и запишем весь код:

```
(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 код из примера выше в окно макросов и нажмите **Выполнить**. Строки таблицы с 1 по 200 будут раскрашены в разные цвета менее чем за секунду.

Конвертация макросов MS VBA

Макросы в редакторах отличаются от макросов, используемых в программах Microsoft, поскольку в программных продуктах Microsoft используется скриптовый язык Visual Basic

для приложений (VBA). Язык JavaScript является более гибким и может использоваться на любой платформе (что важно, поскольку наши редакторы доступны на разных платформах).

Это может быть причиной определённых неудобств в том случае, если до этого вы работали с Microsoft Office с использованием макросов, поскольку они несовместимы с макросами на JavaScript. Однако вы можете сконвертировать ваши старые макросы и использовать их с новыми редакторами.

Это не сложный процесс. Рассмотрим 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 выше считает сумму значений из трёх диапазонов ячеек столбца **A** и помещает результат в три ячейки столбца **B**.

Всё то же самое можно выполнить с помощью макросов JavaScript, код будет выглядеть почти идентично и легко читаем, если вы знаете и Visual Basic для приложений 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;
    }
})();
```

Таким же образом можно сконвертировать любой другой скрипт, написанные на Visual Basic для приложений, в код JavaScript, который будет совместим с нашими редакторами.

Полезные примеры макросов

Примеры на этой странице продемонстрируют использование макросов и дадут возможность сравнить макросы, написанные на JavaScript, с макросами, написанными на Microsoft Visual Basic для приложений, чтобы вы могли увидеть разницу и понять, как можно сконвертировать уже имеющиеся макросы.

Запись данных в ячейку таблицы

В этом примере мы запишем данные (фразу "Hello world") в ячейку, находящуюся в четвёртом столбце третьего ряда

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

Изменение цветов шрифта и фона ячейки, выделение шрифта жирным

В этом примере мы установим жирный шрифт, изменим его цвет и цвет фона ячейки

```
(function()
{
Api.GetActiveSheet().GetRange("A2").SetBold(true);
Api.GetActiveSheet().GetRange("B4").SetFontColor(Api.CreateColorFromRGB(255, 0, 0));
Api.GetActiveSheet().GetRange("B3").SetFillColor(Api.CreateColorFromRGB(0, 0, 250));
}) ();
```

Объединение и отмена объединения ячеек выбранного диапазона

В этом примере мы объединим ячейки одного диапазона и отменим объединение ячеек другого диапазона

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

Установка ширины столбца

В этом примере мы установим ширину второго столбца ("B")

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

Форматирование диапазона как таблицы

В этом примере мы отформатируем диапазон ячеек в таблицу

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

Добавление новой диаграммы для выбранного диапазона ячеек

В этом примере мы создадим диаграмму для данных из диапазона ячеек "C5:D7"

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