PhotoshopCS : Actions Vs. JavaScript

Те, кому приходилось заниматься автоматизацией работы в Photoshop наверняка столкнулись с проблемой ограниченности возможностей Actions. Да, они позволяют автоматизировать то, что требует многократного повторения. Но в Actions нет возможности изменения параметров операций. Официально описанная объектная модель JavaScript ещё беднее по функциональным возможностям, чем Actions.

Ни одним из указанных инструментов задача генерации кнопочек с разными надписями, но одинаковыми эффектами не решается. А если этих кнопочек 64 шт и каждая в 3-х состояниях? Вручную задача решаема, но процесс довольно нудный. А если нужно будет изменить стиль? Переделывать заново?

Совершенно случайно наткнулся в интернете на описание занятного плагина к Photoshop. ScriptListener. Изначально ставится с Photoshop, но отключён по умолчанию. После подключения этого плагина Photoshop создаёт **лог в форматах JavaScript/VBScript** и последовательно записывает туда все действия, которые производятся. В том числе **всё, что можно записать в Actions**. Только **в отличие от Actions JavaScript-файл можно отредактировать**.

В логе используется слабодокументированная внутренняя объектная модель Photoshop. Одной команде соответствует целый блок строк. В логе такие блоки разделены комментариями, что позволяет зная последовательность действий в Photoshop однозначно установить какому действию соответствует какой блок. Названия вызываемых команд мнемонические (сокращённые, легко запоминающиеся и однозначно определяющие полное название команды), например Layer обозначается как Lyr, Select как Slct и т.д. Аргумент-параметр такой команды в JavaScript можно заменить на переменную. Это может быть практически что угодно: текст, вводимый текстовым инструментом, название стиля, который применяется к слою, параметры выравнивания слоя относительно связанного и др. Также можно вызывать JavaScript с параметрами, а именно: заменить интересующие значения в теле скрипта на входные параметры, с которыми вызывается скрипт.

В процессе экспериментов и анализа файлов сформировалась методика:

1. Подключить ScriptListener
* Закрыть PhotoshopCS
* Скопировать файл ScriptListener.8li из папки [PhostoshopCS]\Scripting Guide\Utilities\ в папку [PhotoshopCS]\Plug-Ins\Adobe Photoshop Only\Automate (где [PhotoshopCS] - папка, в котрую установлен Photoshop)
* Запустить Photoshop

2. Проделать все необходимые действия (*и только их, потому как лишние операци также записываются в лог и затрудняют его анализ*)

3. Открыть лог на JavaScript и, сверяясь с последовательностью действий в п.2, обнаружить блоки, в которых обрабатываются интересующие команды
* Лог сохраняется сразу в два файла C:\ScriptingListenerJS.log и C:\ScriptingListenerVB.log соответственно

4. Заменить интересующие параметры операции на входные параметры скрипта
* Примечание : Записанная в лог команда открытия файла вызывает ошибку, вместо неё я использовал команду из описанной объектной модели (параметр arguments[0].toString() соответствует первому входному параметру, с которым в дальнейшем будет вызываться скрипт)

```
var docToOpen=File(arguments[0].toString());
open(docToOpen);
```

Числовые параметры можно вписывать просто как argument[x], где x - порядковый номер параметра, начиная с 0, а строковые в виде arguments[x].toString()

5. Переименовать ScriptingListenerJS.log в файл с расширением JSX и положить его куда удобно (например C:\test.jsx)

6. Написать простейший скрипт на VBScript (например C:\test.vbs), из которого будет осуществляться вызов JavaScript файла с соответствующими параметрами

```
Set appRef = CreateObject("Photoshop.Application")
appRef.DoJavaScriptFile x, Array(y, z)
```

В первой строке создаём объект Photoshop, визуально это приводит к его запуску, если он закрыт. Во второй строке передаём Photoshop команду запуска скрипта, где X - имя файла JSX с полным путём, Y,Z - входные параметры, которые будут доступны внутри JSX скрипта как элементы массива arguments[], описанного в п.4

7. Запустить скрипт, описанный в п.5 и с нескрываемым удовольствием наблюдать как Photoshop в автоматическом режиме за несколько минут проделывает операции, которые в ручном режиме обработки могли бы стоить часов, а то и дней утомительной работы. Теперь это время можно потратить на что-нибудь более полезное.
* Двойным кликом запускаем VBS файл.

На данный момент методика использована
* при генерации кнопочек для интерфейса программы (описанный случай - 64 кнопки в трёх состояниях, в том числе с заданием Alpha канала) - условия большого объёма ручной работы и возможности изменения стиля кнопок в процессе разработки
* при генерации кнопочек для сайта - условия изменения стиля всех кнопок до полного удовлетворения заказчика и возможность создания новой кнопки вписыванием всего одной строчки в скрипт

Ссылки (источники использованные при анализе и написании статьи, могут быть полезны при разборе скриптов, сгенерированных Photoshop):

Свойства в объектной модели Photoshop: http://objjob.phrogz.net

Александр Макаров
a.makarov.1985@inbox.ru