**Операционные системы**

(Методы и технологии управления ОС Windows на основе использования языка командной оболочки и сценариев)

**Лекция 9**

**Сценарии WSH как XML-документы. Схема WS XML**

Значением параметра /Новый является константа логического типа ( true или false ), поэтому для формирования строки, соответствующей этому значению, используется условный оператор языка JScript:

//Проверяем, существует ли аргумент /Новый

if (objNamedArgs.Exists("Новый"))

//Получаем с помощью условного оператора значение

//логического аргумента /Новый

s+="Новый пользователь: "+(objNamedArgs("Новый") ? "Да" : "Нет");

<job id="Named">

<runtime>

<description>

Имя: named.wsf

</description>

<named

name="Имя"

helpstring="Имя пользователя"

type="string"

required="true"

/>

<named

name="Компьютер"

helpstring="Имя рабочей станции"

type="string"

required="false"

/>

<named

name="Новый"

helpstring="Признак того, что такого пользователя раньше не было"

type="boolean"

required="true"

/>

</runtime>

<script language="JScript">

var objNamedArgs,s;

//Вызываем метод ShowUsage для вывода на экран описания сценария

WScript.Arguments.ShowUsage();

//Создаем объект WshNamed — коллекция именных аргументов сценария

objNamedArgs= WScript.Arguments.Named;

s="";

//Проверяем, существует ли аргумент /Имя:

if (objNamedArgs.Exists("Имя"))

//Получаем значение символьного аргумента /Имя

s+="Имя: "+objNamedArgs("Имя")+"\n";

//Проверяем, существует ли аргумент /Компьютер:

if (objNamedArgs.Exists("Компьютер"))

//Получаем значение символьного аргумента /Компьютер

s+="Машина: "+objNamedArgs("Компьютер")+"\n";

//Проверяем, существует ли аргумент /Новый

if (objNamedArgs.Exists("Новый"))

//Получаем с помощью условного оператора значение

//логического аргумента /Новый

s+="Новый пользователь: "+(objNamedArgs("Новый") ? "Да" : "Нет");

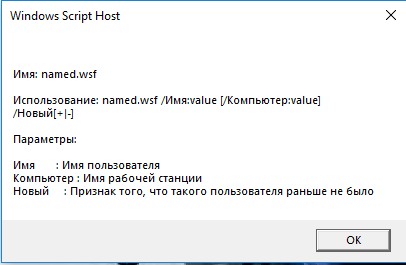
//Выводим полученные строки на экран

WScript.Echo(s);

</script>

</job>

Листинг 9.2. Файл named.wsf



**Элемент <unnamed>**

Для создания информации об использовании этого сценарии создается элемент <unnamed> следующего вида:

<job id="Unnamed">

<runtime>

<description>

Имя: unnamed.wsf

</description>

<unnamed

name="Расш"

helpstring="Расширения файлов"

many="true"

required=2

/>

</runtime>

<script language="JScript">

var objUnnamedArgs,s;

//Вызываем метод ShowUsage для вывода на экран описания сценария

WScript.Arguments.ShowUsage();

objUnnamedArgs=WScript.Arguments.Unnamed; //Создаем объект WshUnnamed

//Определяем количество безымянных аргументов

s="Передано в сценарий безымянных аргументов: "+objUnnamedArgs.length;

for (i=0; i<=objUnnamedArgs.length-1; i++)

//Формируем строки со значениями безымянных аргументов

s+="\n"+objUnnamedArgs(i);

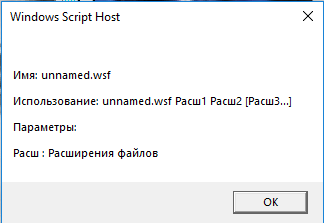
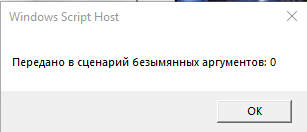
//Выводим полученные строки на экран

WScript.Echo(s);

</script>

</job>

Листинг 9.3. Файл unnamed.wsf



**Элемент <description>**

Внутри элемента <description> помещается текст (без дополнительных кавычек), описывающий назначение сценария. Как и все элементы внутри <runtime>, этот текст выводится на экран, если сценарий был запущен с ключом /? в командной строке или если в сценарии встретился вызов метода ShowUsage объекта WshArguments. При выводе текста на экран учитываются все имеющиеся в нем пробелы, символы табуляции и перевода строки.

<job id="Descrip">

<runtime>

<description>

Имя: descrip.wsf

Описание: Здесь можно привести дополнительное описание сценария

</description>

</runtime>

<script language="JScript">

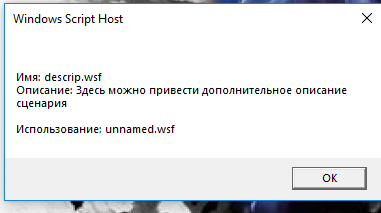
//Вызываем метод ShowUsage

WScript.Arguments.ShowUsage();

</script>

</job>

Листинг 9.4. Файл descrip.wsf



**Элемент <example>**

Внутри элемента <example> приводится текст из одной или нескольких строк, в котором можно описать примеры запуска сценария. Если сценарий был запущен с ключом /? в командной строке или в сценарии встретился вызов метода ShowUsage объекта WshArguments, то этот текст выводится в графическое диалоговое окно (при использовании wscript.exe) или на экран (в консольном режиме при использовании cscript.exe). При выводе текста на экран учитываются все имеющиеся в нем пробелы, символы табуляции и перевода строки, при этом строки из элемента <example> выводятся после строк из элемента <description>.

<job id="Example">

<runtime>

<description>

Имя: example.wsf

Описание: Здесь можно привести дополнительное описание сценария

</description>

<example>

Здесь приводится пример запуска сценария

(с параметрами командной строки, например)

</example>

</runtime>

<script language="JScript">

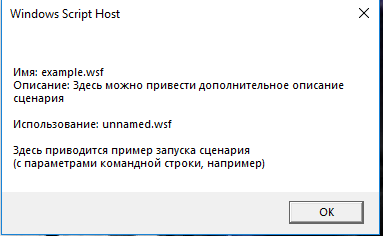
//Вызываем метод ShowUsage

WScript.Arguments.ShowUsage();

</script>

</job>

Листинг 9.5. Файл example.wsf



**Элемент <resource>**

Элемент <resource> позволяет отделить символьные или числовые константы (ресурсы) от остального кода сценария. Например, таким образом удобно собрать в одном месте строки, которые используются в сценарии для вывода каких-либо стандартных сообщений. Если после этого понадобится изменить сообщения в сценарии (например, перевести их на другой язык), то достаточно будет внести соответствующие корректировки в строки, описанные в элементах <resource>.

WScript.Echo(getResource("MyName"));

<job id="Resource">

<runtime>

<description>

Имя: resource.wsf

Описание: Пример использования в сценарии ресурсных строк

</description>

</runtime>

<resource id="MyName">

Меня зовут Андрей Попов

</resource>

<script language="JScript">

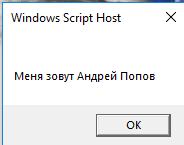
//Выводим на экран значение ресурса "MyName"

WScript.Echo(getResource("MyName"));

</script>

</job>

Листинг 9.6. Файл resource.wsf



**Элемент <object>**

Элемент <object> предлагает еще один способ создания экземпляра COM-объектов для использования их внутри сценариев. Напомним, что ранее для этого мы использовали методы CreateObject и GetObject объекта WScript, объект ActiveXObject и функцию GetObjectязыка JScript, а также функцию CreateObject языка VBScript. Элемент <object> может заменить эти средства.

**Несколько заданий в одном файле**

Каждое отдельное задание в WS-файле должно находиться внутри элементов <job> и </job>. В свою очередь, все элементы <job>являются дочерними элементами контейнера <package>.

В качестве примера рассмотрим сценарий multijob.wsf, приведенный в листинге 9.7. Здесь описываются два задания с идентификаторами "VBS"(сценарий на языке VBScript) и "JS" (сценарий на языке JScript).

<package>

<job id="VBS">

<!-- Описываем первое задание (id="VBS") -->

<runtime>

<description>

Имя: multijob.wsf

Описание: Первое задание из multijob.wsf

</description>

</runtime>

<script language="VBScript">

WScript.Echo "Первое задание (VBScript)"

</script>

</job>

<job id="JS">

<!-- Описываем второе задание (id="JS") -->

<runtime>

<description>

Имя: multijob.wsf

Описание: Второе задание из multijob.wsf

</description>

</runtime>

<script language="JScript">

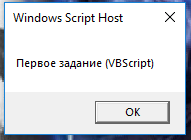
WScript.Echo("Второе задание (JScript)");

</script>

</job>

</package>

Листинг 9.7. Файл multijob.wsf



**Использование констант внешних объектов**

Для того чтобы в сценарии обращаться по имени к константам, определенным во внешних объектах, не создавая экземпляров самих объектов, необходимо сначала получить ссылку на эти объекты с помощью элемента <reference>.

В листинге 9.8 приведен сценарий refer.wsf, в котором с помощью элемента <reference> производится доступ к трем константам объекта FileSystemObject ( ForReading, ForWriting и ForAppending ), которые определяют режим работы из сценария с внешним текстовым файлом. В результате выполнения сценария refer.wsf на экран выведется диалоговое окно с информацией о значениях констант объекта FileSystemObject.

<job id="Example">

<runtime>

<description>

Имя: refer.wsf

Описание: Использование констант внешних объектов

</description>

</runtime>

<!-- Получаем ссылку на объект FileSystemObject -->

<reference object="Scripting.FileSystemObject"/>

<script language="JScript">

var s;

s="Значения констант объекта FileSystemObject:\n\n";

//Получаем значение константы ForReading

s+="ForReading="+ForReading+"\n";

//Получаем значение константы ForWriting

s+="ForWriting="+ForWriting+"\n";

//Получаем значение константы ForAppending

s+="ForAppending="+ForAppending;

//Выводим полученные строки на экран

WScript.Echo(s);

</script>

</job>

Листинг 9.8. Использование в сценарии констант внешних объектов (файл refer.wsf)

