Занятие 1

План

Язык JavaScript

JavaScript - это язык программирования, используемый в составе страниц HTML для увеличения функциональности и возможностей взаимодействия с пользователями. Он был разработан фирмой Netscape в сотруднечестве с Sun Microsystems на базе языка Sun's Java .С помощью JavaScript на Web-странице можно сделать то, что невозможно сделать стандартными тегами HTML. Скрипты выполняются в результате наступления каких-либо событий, инициированных действиями пользователя. Создание Web-документов, включающих программы на JavaScript, требует наличее текстового редактора и подходящего браузера. Некоторые просмоторщики включают в себе встроенные редакторы, поэтому необходимость во внешнем редакторе отпадает.

Основные понятия

1. *Язы́к программи́рования* — формальная знаковая система, предназначенная для записи программ. Программа обычно представляет собой некоторый алгоритм в форме, понятной для исполнителя (например, компьютера). Язык программирования определяет набор лексических, синтаксических и семантических правил, используемых при составлении компьютерной программы. Он позволяет программисту точно определить то, на какие события будет реагировать компьютер, как будут храниться и передаваться данные, а также какие именно действия следует выполнять над этими данными при различных обстоятельствах.

2. Языки программирования могут быть разделены на *компилируемые* и *интерпретируемые*.

    - Программа на компилируемом языке при помощи специальной программы компилятора преобразуется (компилируется) в набор инструкций для данного типа процессора (машинный код) и далее записывается в исполняемый файл, который может быть запущен на выполнение как отдельная программа. Другими словами, компилятор переводит программу с языка высокого уровня на низкоуровневый язык, понятный процессору.

    - Если программа написана на интерпретируемом языке, то интерпретатор непосредственно выполняет (интерпретирует) ее текст без предварительного перевода. При этом программа остается на исходном языке и не может быть запущена без интерпретатора. Можно сказать, что процессор компьютера — это интерпретатор машинного кода.

    - Кратко говоря, компилятор переводит программу на машинный язык сразу и целиком, создавая при этом отдельную программу, а интерпретатор переводит на машинный язык прямо во время исполнения программы.

3. *Алгори́тм* — это точный набор инструкций, описывающих порядок действий некоторого исполнителя для достижения результата, решения некоторой задачи за конечное время.

JavaScript и Java

JavaScript и Java- это два разных языка программирования. Java- это объектно-ориентированный язык программирования и запускается при помощи компилятора и вспомогательных файлов. Разрабатываемые с помощью Java программы могут работать как законченные приложения либо как встроенные в Web-страницу апплеты. И хотя они встроены в страницу HTML, они храняться на клиенской машине как отдельные файлы.

Напротив, JavaScript, размещаются внутри HTML страницы и не могут существовать, как отдельные программы и функционируют, будучи запущенными в браузерах типа Netscape Navigator или Internet Explorer.

JScript — скриптовый язык программирования компании Майкрософт, являющийся реализацией стандарта ECMAScript. Синтаксис JScript во многом аналогичен языку JavaScript компании Netscape, однако, помимо добавления клиентских скриптов на веб-страницы (что было единственной функцией JavaScript до появления проекта Mozilla), JScript может использоваться и для других целей,

ECMAScript — это интерпретируемый язык программирования, стандартизированный международной организацией ECMA в спецификации ECMA-262.

Синтаксис

Синтаксис **–** это правила построения конструкций языка программирования, в соответствии с которыми пишется исходный код программы или сценария.

Размещение сценария

Здесь есть несколько вариантов или подходов:

- скрипт может быть записан непосредственно в тексте страницы;
- скрипт может быть загружен из внешнего по отношению к html странице файла.
- команды JavaScript можно записывать так же и в обработчиках событий элементов.

В первых двух случаях используется элемент <**script**>, располагаемый в заголовке или теле страницы:

```
<html>
    <head>
        <script type="text/javascript" src="script.js"></script>
        ...
    </head>
    <body>
        <script type="text/javascript">
            текст сценария
        </script>
        ...
    </body>
</html>
```

Тэг элемента <**script**> может иметь насколько атрибутов:

- type - указывает язык программирования, на котором написан текст сценария. Для использования в качестве языка сценария JavaScript необходимо присвоить данному атрибуту значение "text/javascript".
- src - позволяет указать адрес файла, содержащего текст сценария. Файл должен иметь расширение ".js" и содержать **только строки сценария** без HTML разметки!

В случае, когда элемент <**script**> содержит атрибут src текст сценария берется из указанного файла, а содержимое самого элемента игнорируется. Такой подход используют в тех случаях, когда необходимо централизовать общий для нескольких страниц сценарий - это экономит размер страниц, а, так же, упрощает модификацию сценария.

Правила написания языка:

- JavaScript чувствителен к регистру символов.
- Операторы должны разделяться между собой символом ";" (Точка с запятой называется разделителем строки и нужна для того, чтобы сделать сценарий совместимым со стандартом ECMA, хотя и без точки с запятой ошибки не будет).
- Операторы могут объединяться в блоки. Для этого их следует размещать в фигурных скобках: "{" и "}". В этом случае несколько операторов рассматриваются интерпретатором и выполняются как один составной оператор. При этом разделитель ";" после блока не ставится.
- В текст скрипта можно включать комментарии.

```
// однострочный комментарий действует до конца строки

/*
   блочный
   комментарий (многострочный)
*/
```

Объекты, методы и свойства

Объекты

объекты находятся внутри браузера. Это, в частности, окно браузера, формы и их части, например кнопки и текстовые окна. В JavaScript также имеется собственная группа встроенных объектов, к которым относятся массивы, данные и т.д.

Методы

Метод (method) - это действия, которые может выполнять объект. Примерами других методов являются открытие и закрытие окон, нажатие кнопок.

Свойства

У всех объектов имеются свойства (properties), у такого объекта, как браузер, имеется название и номер версии и т.п.

Работа с браузерами, не поддерживающими JavaScript

Если браузер не поддерживает JavaScript, то содержимое элемента <**script**> не будет интерпретировано, а будет выведено на страницу в виде текста. Такое поведение браузера будет не совсем корректным и этого необходимо избегать. Для этого текст скрипта помещают в блок HTML комментария следующим образом:

```
<script type="text/javascript">
<!--
    текст сценария
//-->
</script>
```

Браузеры, поддерживающие JavaScript, комментарии просто игнорируют. А для браузера, не имеющего поддержки JavaScript, текст сценария будет обыкновенным комментарием и просто не будет выводиться на страницу.

Кроме того, можно вывести на страницу текст сообщения, говорящий о том, что браузер не имеет поддержки JavaScript. Для этого текст сообщения размещают в элементе

```
<noscript>

...

</noscript>.
```

Причем в этом элементе можно использовать html разметку.

Решение проблемы поддержки сценариев в браузере

```
<script>
<!--
    alert("Откроем данную страницу в Opera и выключим JavaScript
по кнопке F12,\n в Internet Explorer JavaScript включается
Сервис --> Свойства обозревателя --> Вкладка Дополнительно -->
Параметры: Разрешать запуск активного содержимого файлов на Моем
компьютере");
//-->
</script>

<noscript>
    <h3>Убедитесь, что ваш браузер поддерживает JavaScript
    и в свойствах браузера разрешено выполнение сценариев!</h3>
    Вы так же можете перейти по <a href="no-scripts.html">этой
ссылке</a>
    на вариант страницы, без скриптов.
</noscript>
```

Как вы можете видеть, в элементе <**noscript**> можно использовать HTML разметку. Это дает нам возможность вывести гиперссылку, при щелчке на которой, пользователь перейдет на страницу, не содержащую сценариев.

Классический пример реализации языка

```
<html>
  <body>
    <script type="text/javascript">

      //объявление переменной строкового типа с именем "message"
      var message="Hello World!!!";

      // вывод диалогового окна с текстом
      alert(message);

    </script>
  </body>
</html>
```

В данном примере объявляется переменная "message" и ей присваивается значение "Hello World!!!". Следующая строка скрипта выводит значение переменной на экран в диалоговом окне (функция alert).

Переменные

Часто, при выполнении сценарием некоторых вычислений, требуется сохранить результат, полученный на определенном этапе, с целью его дальнейшего использования, для этих целей используют **переменные**. По большому счету, переменная - это ячейка оперативной памяти компьютера, которая имеет имя и может хранить данные. Однако, с точки зрения JavaScript, переменную проще представлять, как **объект**, с которым ассоциированы какие-то данные.

Объявление переменных

Переменные в JavaScript объявляются при помощи ключевого слова **var**, которое располагается в начале объявления. При объявлении переменной можно присвоить значение (проинициализировать). За один раз можно объявить более одной переменной, в таком случае они разделяются запятой ",". Объявление заканчивается символом ";".

Шаблон объявления переменных имеет следующий вид (в квадратные скобки [] взяты необязательные части конструкции):
var имя_переменной [= значение] [, имя_переменной [= значение] ...];

Примеры:
var angle; //переменная объявлена но не проинициализирована
var counter_1 = 0; //переменная объявлена и проинициализирована значением 0
var FirstName, LastName = "Иванов", age=20; //объявление нескольких переменных одновременно

Правила объявления переменных:

- имена переменных чувствительны к регистру;
- имена переменных должны состоять из букв латинского алфавита, цифр и знаков "_";
- имена переменных не могут начинаться с цифры;

- переменная должна быть объявлена до того, как вы начнете ее использовать;

Если переменная не была проинициализирована при объявлении, она будет содержать специальное значение - **undefined** (неопределено).

Типы данных
Тип переменной указывает, какого рода данные содержатся в переменной.
В JavaScript используются следующие типы данных:
- строковый;
- числовой;
- логический;
- объектный;

Совет:
- Переменные JavaScript не имеют строго определенного типа.
- Тип переменной определяется присвоенным ей значением и может измениться на протяжении программы.
- При выполнении операций над разнотипными данными происходит автоматическое преобразование типа.

В JavaScript, как и во многих других языках программирования, существует понятие **литеральный тип данных** - это специальный тип, касающийся фиксированных значений.
JavaScript поддерживает три литеральных типа данных:
- строковый;
- числовой;
- логический;

Числовые литералы
Целые числовые литералы представляют собой числа, не содержащие дробной части. Записываются такие литералы в виде простой последовательности цифр:
0
-1
135
Числовые литералы с плавающей точкой представляют собой числа с дробной частью. Таким образом, сначала нужно указать целую часть, затем десятичную точку (используется символ "."), затем дробную часть:
0.5
-1.5667
5.234e+3
5.234E-8
Последние два значения в примере представлены в так называемой **экспоненциальной форме**. Такая запись числа используется для записи действительно больших или маленьких чисел с плавающей точкой. За дробной частью числа идет символ "e" или "E", после чего следует степень числа 10.
Разберем число 5.234e+3. Его можно представить в следующей форме: $5.234*10^3$ то есть 5.234*1000 или 5234.

Шестнадцатеричные значения.
В шестнадцатеричной системе счисления используются цифры от 0 до 9, а ,так же, буквы от "a" до "f" (или от "A" до "F") латинского алфавита. Число, равное 8 в десятичной системе, в восьмеричной системе будет равно 10. Для записи шестнадцатеричного числа в JavaScript перед числом записывается "0x" (или "0X"):
0xD45
0X1FF

Строковые литералы
Строковый литерал представляет собой последовательность символов (абсолютно любых), заключенных в двойные (") или одинарные (') кавычки. Ниже приводятся несколько примеров строковых литералов:
"Строковый литерал"
'другой строковый литерал'
"1 января 2004 г."
""
В последнем случае строка не содержит ни одного символа и называется **пустой**.

Наиболее популярные **управляющие последовательности** внутри строк:

Последовательность	**Символ**
\'	Символ ' (одинарная кавычка)
\"	Символ " (двойная кавычка)
\\	Символ \ (обратный "слэш")
\b	Символ "BackSpace"
\n	Перевод строки (повая строка)
\r	Возврат каретки
\t	Символ "Tab"

Пример применения управляющих последовательностей в строках:

Применение управляющих последовательностей

```
        <script type="text/javascript">
            //объявление переменной строкового типа с именем
"message"
            var message="Первая строка\nВторая строка";
            alert(message);// вывод диалогового окна с текстом
        </script>
```

Логические (булевы) литералы
Это тип литералов принимает всего два значения: **true** или **false**.

Операторы
Операторы позволяют связывать данные различного рода отношениями (математическими, логическими и т.п.) и составлять **выражения**. Данные, над которыми выполняются действия, называются **операндами**. Операторы классифицируются по количеству участвующих в отношении операндов следующим образом:

- унарные (одноместные) - один операнд;
- бинарные (двухместные) - два операнда;
- тернарные (трехместные) - три операнда;
- и т.д.

Существует несколько форм записи операторов:

- префиксная,
- постфиксная,
- инфиксная.

Каждая форма записи определяет порядок следования операндов по отношению к знаку операции. Ниже на примере бинарного оператора арифметического сложения показаны различные формы записи операторов:
+ операнд1 операнд2 //префиксная запись
операнд1 операнд2 + //постфиксная запись
операнд1 + операнд2 //инфиксная (традиционная) запись