Основы языка JavaScript

Функции JavaScript

Функция – это блок кода на языке JavaScript, который определяется однажды, и в дальнейшем может выполняться или вызываться сколько угодно раз.

Определение и вызов функций

Функция определяется с помощью ключевого слова **function**, за которым следуют:

1. Имя функции
2. Круглые скобки, которые могут содержать список имен параметров, разделенных запятыми
3. Выполняемые действия, которые составляют тело функции, заключенные в фигурные скобки

Для вызова функции используется оператор вызова, представляющий из себя пару круглых скобок.

Пример

```
//определение функции
function starline() {
  for(var i = 0; i < 45; i++){
    document.write("*");
  }
  document.write("<br>");
}
 starline();  //вызов функции
document.write("<p>это абзац</p>");
starline();
```

Функция starline() вызывается дважды.

Область видимости глобальных и локальных переменных

Область видимости переменной – это та часть программы, для которой эта переменная определена. **Глобальная переменная** имеет глобальную область видимости – она определена для всей JavaScript программы.

Переменные, объявленные внутри функции называются **локальными** и имеют локальную область видимости т.е. они видны только в теле функции. Параметры функции также являются ее локальными переменными и видны только в ее теле.

Внутри тела функции локальная переменная имеет преимущество перед глобальной переменной с тем же именем. Если объявить локальную переменную или параметр функции с тем же именем, что у глобальной переменной, то фактически глобальная переменная будет скрыта:

Пример

```
var x = "глобальная";
function checkscope() {
  var x = "локальная";
document.write(x);
}
checkscope();
```

Параметры и аргументы

Параметры указываются в определении функции внутри круглых скобок и являются ее локальными переменными, т.е. видны только в ее теле. При вызове функция может получать аргументы, которыми инициализируются параметры.

Аргумент - это выражение, которое может быть переменной, литералом, или выражением, состоящим из одного или нескольких операторов, которое служит для инициализации параметров.

Пример

```
function getstr(car, place) {
document.write('Моя машина - ' + car + ' и я еду на ней ' + place);
}
 function showbook(title) {
  str = '<p>книга называется: "' + title + '"</p>';
  document.write(str);
}
var house = "на дачу";
 getstr('BMW', house);
showbook('Машина времени');
```

В JavaScript при установке параметров для функции не указывается ключевое слово **var**. JavaScript объявляет переменные автоматически, когда они устанавливаются в качестве параметров функции, поэтому ключевое слово **var** здесь не используется.

Параметры можно использовать как и все другие переменные.

У функции может быть сколько угодно параметров. При назначении функции нескольких параметров не обязательно использовать все параметры, можно воспользоваться только одним или не использовать их вообще.

Когда функция вызывается с меньшим количеством используемых параметров, чем описывается в определении, недостающие параметры получают значение undefined.

Оператор return

Оператор **return** применяется для возврата результата работы функции основному сценарию с целью дальнейшего его использования. Он может находиться в любом месте функции. Обычно оператор **return** записывается в последней строке функции перед закрывающей фигурной скобкой и заканчивается точкой с запятой.

Пример

```
return имя_переменной;
return "строка1 " + "строка 2";
return true;
return null;
return a + (b - 31) * 2;
```

Оператор **return** может использоваться без значения, чтобы прекратить выполнение и выйти из функции:

```
function calc(a) {
  if(!a) return;

  document.write(a);
}
```

В функции может использоваться не один, а несколько операторов **return**:

```
function check(a, b) {
  if(a > b) {
    return a;
  } else {
      return b;
    }
}
```

Выбор имени функции

Имя функции следует выбирать по тем же правилам что и имя переменной.

Единственное различие - это то, что в качестве имени функции обычно выбирают глагол, т.к. функция выполняет действие.

Диалоговые окна

1. **alert("Текст сообщения");**

   Выводит диалоговое окно с заданным сообщением и кнопкой ОК.

2. **confirm("Текст сообщения");**

   Выводит диалоговое окно с заданным сообщением и кнопками ОК и ОТМЕНА.

3. **prompt("Текст сообщения", ["текст по умолчанию в поле ввода"]);**

   Все три типа окна являются модальными.

   Также могут использоваться для вывода результатов.