

Публикация на тему

Инструкции JavaScript

В публикации рассматриваются базовые инструкции JavaScript и тернарный оператор

Автор

[Михалькевич Александр Викторович](#)

Публикация

Наименование Инструкции JavaScript

Автор А.В.Михалькевич

Специальность В публикации рассматриваются базовые инструкции JavaScript и тернарный оператор,

Анотация

Anotation in English

Ключевые слова

Количество символов 8281

Содержание

[Введение](#)

1 [Базовые инструкции JavaScript](#)

2 [Тернарный оператор ?:](#)

[Заключение](#)

[Список использованных источников](#)

[Приложения](#)

Введение

Инструкция – выражение, в результате которого должны выполняться действия (например, объявление или инициализация переменной и т.д.). Каждая простая инструкция заканчивается символом “;”. Если этого не делать, браузер проставляет их за нас. Ориентируется он в этом случае на переносы строки.

1 Базовые инструкции JavaScript

Несколько выражений могут составлять блок инструкций:

Блок инструкций:

```
{  
  x = Math.PI;  
  y = Math.cos(x);  
}
```

Инструкции `var` и `function` являются инструкциями объявления. Инструкция `var` объявляет переменную.

Инструкция `var`:

```
var a = 'string';
```

Инструкция `function` объявляет функцию.

Инструкция функции:

```
function myFunc(){  
  //-- тело функции  
}
```

Составные инструкции заранее определены в языке и имеют свой синтаксис:

Условные инструкции (*if / else if / else*)

Инструкция переключения (*switch / case / default*)

Инструкции цикла (*while, do / while, for, for in*)

Инструкция перехвата и обработки исключения (*try / catch / finally*)

Инструкция `if / else if / else`:

```
if(n==1){  
  // выполнить первый блок  
}elseif(n==2){  
  // выполнить второй блок  
}else{  
  // выполнить блок по умолчанию.  
}
```

Если условная инструкция `if/else` принимает более одного параметра, то предпочтительнее использовать инструкцию `switch`. Рассмотрим инструкцию `switch`:

Инструкция `switch`:

```
switch(n){  
  case 1:  
    //выполнить первый блок, если n == 1  
    break;  
  case 2:  
    //выполнить второй блок, если n == 2  
    break;  
  default:  
    //выполнить блок по умолчанию.  
}
```

Рассмотрим инструкции циклов:

Инструкция while:

```
var count = 0;
while(count<10){
  console.log(count);
  count++;
}
```

Инструкция do/while во многом похожа на инструкцию while, за исключением того, что инструкция do/while сперва делает, потом проверяет условие. Таким образом, хотя бы один раз она обязательно сработает.

Инструкция do/while:

```
do {
  // что-то сделать
}while(++i)
```

Инструкция for часто оказывается более наглядной инструкцией цикла. Инициализация, проверка и инкремент (наращивание переменной) – это три операции, выполняемые с переменной цикла.

Инструкция for:

```
for(var count=0; count<10; count++){
  console.log(count);
}
```

Инструкция for/in использует ключевое слово for, но она в корне отличается от инструкции for. For/in имеет следующий синтаксис:

for(переменная in объект).

Инструкция for in:

```
for(p in o){
  console.log(o[p]);
}
```

Любая инструкция может быть помечена меткой

Идентификатор: инструкция.

Инструкция *break* приводит к немедленному выходу из внутреннего цикла или из инструкции switch.

Инструкция *continue* во многом схожа с инструкцией break, но приводит не к выходу из инструкции, а к новой итерации цикла.

Инструкция *return* внутри функции служит для определения значений, возвращаемых функцией. Имеет следующий синтаксис:

return выражение;

Инструкция throw предназначена для перехвата ошибок инструкции try/catch

Инструкция try/catch/finally:

```
try {  
    throw "myException"; // возбуждение исключительной ситуации  
} catch (e) {  
    logMyErrors(e); // вывод ошибок  
}finally{  
    // этот блок выполняется всегда, независимо от наличия либо отсутствия  
    ошибок.  
}
```

2 Тернарный оператор ?:

Инструкция вида if/else может быть представлена в виде так называемого условного или тернарного оператора

проверка_условия ? значение_1 : значение_2.

Например, данная инструкция:

Инструкция if:

```
if (x > y) {  
    var z = 'x больше y';  
} else {  
    var z = 'x меньше y';  
}
```

Может быть представлена в виде тернарного оператора:

Тернарный оператор:

```
var z = x > y ? 'x больше y' : 'x меньше y';
```

Заключение

Список использованных источников

Приложения