**Основы JavaScript**

Вместе научимся писать программы на JavaScript. Изучим основы языка: переменные, типы данных, условия, циклы, функции, массивы и объекты.

И самое главное — научимся решать практические задачи, с которыми регулярно сталкиваются программисты.

## Как запустить JavaScript

Написать и запустить JavaScript можно двумя способами: первый заключается в том, что мы пишем код прямо на HTML странице внутри тега **<script>**:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок тайтл</title>

<script>

var name = 'Дима';

alert('Привет, '+name);

</script>

</head>

<body>

Это основное содержимое страницы.

</body>

</html>

Тег **<script>** можно располагать в любом месте страницы - как в **<head>**, так и в **<body>**.

Второй вариант заключается в том, что JavaScript код хранится в отдельном файле (наподобие CSS) и подключается тоже с помощью тега **<script>** с атрибутом **src**, в котором указывается путь к файлу со скриптом:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок тайтл</title>

<script src="путь к файлу со скриптом"></script>

</head>

<body>

Это основное содержимое страницы.

</body>

</html>

## Строки и числа

Самыми простыми **типами данных** в JavaScript являются **строки** и **числа**.

**Числа** обозначают сами себя: 1, 12, 145, а вот **строки** требуется брать в кавычки (одинарные или двойные - без разницы):

'строка', "строка"; //это примеры строк

## Переменные

Одним из самых главных и распространенных объектов в программировании является переменная.

**Переменная** - это такой объект, который может хранить внутри себя различные данные, например, строки или числа.

**Имя переменной** должно состоять из английских букв: больших или маленьких, а также цифр и знака\_подчеркивания.

В JavaScript при **объявлении** переменной обязательно должно быть написано ключевое слово **var**:

var a; //тут мы объявили переменную

var a, a1, isVar, is\_var; //тут мы объявили группу переменных

## Операция присваивания

Очень важным элементом программирования является **операция присваивания**. Пример присваивания:

var a = 4; //мы присвоили переменной a значение 4

## Функция alert

В JavaScript существует специальная функция **alert**, которая позволяет вывести какой-либо текст в окно браузера в виде диалогового окошка.

Следующий код выводит на экран заданный текст:

alert('Привет, мир!'); //выведет на экран фразу 'Привет, мир!'

[Нажмите на эту ссылку](http://code.mu/books/javascript/base/osnovy-yazyka-javascript-dlya-novichkov.html), чтобы увидеть такое окошко.

А в следующим коде переменной **text** присваивается фраза, а затем содержимое этой переменной выводится на экран:

var text = 'Привет, мир!';

alert(text); //выведет на экран фразу 'Привет, мир!'

## Получение определенного символа строки

В JavaScript можно получить доступ к **определенному символу строки** по его номеру таким образом: **a[n]** – n-ный символ строки (учтите, что *нумерация идет с нуля*):

var a, b; //объявим наши переменные

a = 'abcde'; //в переменной а будет хранится значение 'abcde'

b = a[0]; //в переменной b будет 'a'

b = a[1]; //в переменной b будет 'b'

b = a[4]; //в переменной b будет 'e'

## Сложности с операцией присваивания

Очень часто новички не понимают, что **присваивание отличается от обычного равенства**. Посмотрите следующий пример:

var a = 1;

a = a + 2;

С точки зрения математики запись **a = a + 2** абсурдна, но не с точки зрения программирования.

В данном случае переменная **a** имела значение **1**, а затем мы переменной **a** присвоили новое значение - старое значение переменной **a** плюс **2**.

## Операции инкремента и декремента

Операция **a++** или **++a** – увеличивает переменную **a** на единицу. Эта операция называется **инкремент**.

Операция **a--** или **--a** – уменьшает переменную **a** на единицу. Эта операция называется **декремент**.

Примеры:

var a = 1;

a++; //увеличит a на 1, что соответствует коду a = a + 1;

alert(a); //выведет 2

var a = 1;

a--; //уменьшит a на 1, что соответствует коду a = a - 1;

alert(a); //выведет 0

Давайте посмотрим, в каких случаях проявляется разница между **++a** и **a++**.

Пусть у нас есть код **alert(++a)** и код **alert(a++)**.

В первом случае переменная сначала увеличится на единицу, а потом выведется, а во втором случае - сначала выведется, а потом увеличится.

## Операции +=, -=, \*=, /=

Мы уже рассматривали код, который демонстрирует сложности с операцией присваивания:

var a = 2;

a = a + 3;

В данном случае мы присваиваем переменной **a** ее текущее значение, увеличенное на 2. Однако JavaScript позволяет записать этот код еще короче с помощью оператора **+=**:

var a = 1;

a += 3; //этот код полностью эквивалентен коду a = a + 3;

Кроме того, существуют операторы -=, \*=, /=, которые эквивалентны следующему коду:

var a = 2;

a -= 3; //этот код полностью эквивалентен коду a = a - 3;

var a = 2;

a \*= 3; //этот код полностью эквивалентен коду a = a \* 3;

var a = 2;

a /= 3; //этот код полностью эквивалентен коду a = a / 3;

## Специальные значения

В JavaScript, как и в других языках программирования, существуют ключевые слова для некоторых специальных значений. Вот они: undefined, null, true, false, NaN, Infinity, -Infinity.

### Значения undefined и null

Значение **undefined** обозначает неопределенность. К примеру, если мы попробуем обратиться к переменной, которой мы еще не задали значение - то ее значение и будет **undefined**.

var a;

alert(a); //выведет undefined

Значение **null** обозначает 'ничего'. К примеру, мы можем присвоить переменной значение **null** в знак того, что там ничего не лежит.

Это значение очень похоже на **undefined**, отличие в том, что **undefined** - это не определенное значение, а **null** - определенное - ничего.

### Значения true и false

Значения **true** и **false** обозначают истину и ложь соответственно. Они используются для таких вещей, которые предполагают два варианта ответа - да или нет.

К примеру, на вопрос 'вам уже есть 18 лет?' в можете ответить да, то есть **true**, или нет, то есть **false**.

### Значение NaN

Значение **NaN** (*Not-A-Number*) обозначает не число. Оно может получится, к примеру, в таком случае - когда вы умножаете строку с буквами на число:

alert('abc'\*3); //выведет NaN

### Значения Infinity и -Infinity

Значения **Infinity** и **-Infinity** обозначают соответственно бесконечность и минус бесконечность. Они получаются если какое-то число поделить на ноль - в этом случает JavaScript не выдает ошибку, как в других языках программирования, а возвращает эти значения.

Если мы делим на ноль положительное число, то получаем **Infinity**, а если отрицательное - то **-Infinity**.

## Функция prompt

Кроме функции alert, которая выдает диалоговое окошко, существует функция **prompt**, которая не только выдает окошко с текстом, но и позволяет получить от пользователя какой-либо текст.

Этот текст можно записать в переменную и затем выполнить над ним какие-либо операции.

В следующем примере мы спросим имя пользователя, запишем его в переменную **name** и с помощью функции **alert** выведем на экран:

var name = prompt('Ваше имя?');

alert('Ваше имя: '+name);

[Нажмите на эту ссылку](http://code.mu/books/javascript/base/osnovy-yazyka-javascript-dlya-novichkov.html), чтобы запустить код примера.

## Функция confirm

Если вам нужно просто спросить у пользователя 'Да' или 'Нет', не давая ему возможность ввести иной текст - используйте функцию **confirm**.

Эта функция вызывает окошко с вопросом, на который нужно ответить пользователю, и двумя кнопками для ответа: с кнопкой 'ОК' и с кнопкой 'Отмена'.

Если пользователь нажмет 'ОК' - то функция вернет **true**, а если 'Отмена' - то вернет **false**.

В следующем примере функция **confirm** выведет диалоговое окно с вопросом '*Вам уже есть 18 лет?*'.

Если вы нажмете 'Ок', то в переменную ok запишется **true** и выведется на экран функцией alert, а если нажмете 'Отмена' - то **false**:

var ok = confirm('Вам уже есть 18 лет?');

alert(ok);

[Нажмите на эту ссылку,](http://code.mu/books/javascript/base/osnovy-yazyka-javascript-dlya-novichkov.html) чтобы запустить код примера.

## Типизация переменных

Что будет, если попробовать перемножить, к примеру, число и строку, вот так: **3 \* '3'**? В результате вы получите число **9**. Это значит, что JavaScript автоматически осуществляет преобразование типов при необходимости, вам не нужно за это переживать.

Однако, есть нюанс: если мы попытаемся **сложить** строку и число, то JavaScript сложит их как строки, а не как числа, вот так: **'3' + 3** получится строка **'33'**, а не число 6.

В случае, например, с умножением JavaScript понимал, что нельзя перемножить строки, поэтому строки переводил в числа и перемножал их. А случай со сложением можно трактовать двояко: складывать как строки или как числа (плюс-то используется как для сложения строк, так и чисел).

Бороться с этим можно следующем способом: нужно сделать недопустимую для строк операцию, например, так: **+'3' + 3** - поставим плюс перед строкой и она преобразуется к числу.

Второй вариант такой: можно сказать яваскрипту, что мы хотим явно преобразовать строку к числу. Это делается с помощью функции Number, вот так: **Number('3') + 3**. В результате получится 6, а не '33'.

К числам могут преобразовываться не только строки, но и любые другие типы данных, например true тоже можно преобразовать к числу таким образом: **Number(true)**.

Можно преобразовывать и к другим типам с помощью функций **Boolean**, **String** и других подобных.