

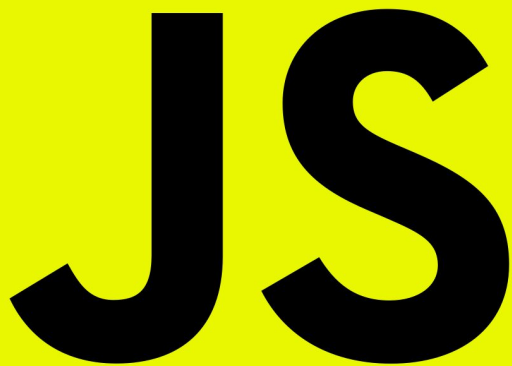


Лекция 1

Знакомимся

Цели на сегодня

- Понять, для чего нужен JavaScript;
- Узнать, как записывать данные в память;
- Разобраться со всеми примитивными типами данных;
- Пробежаться по арифметическим операторам;
- Немного поговорить про типизацию.

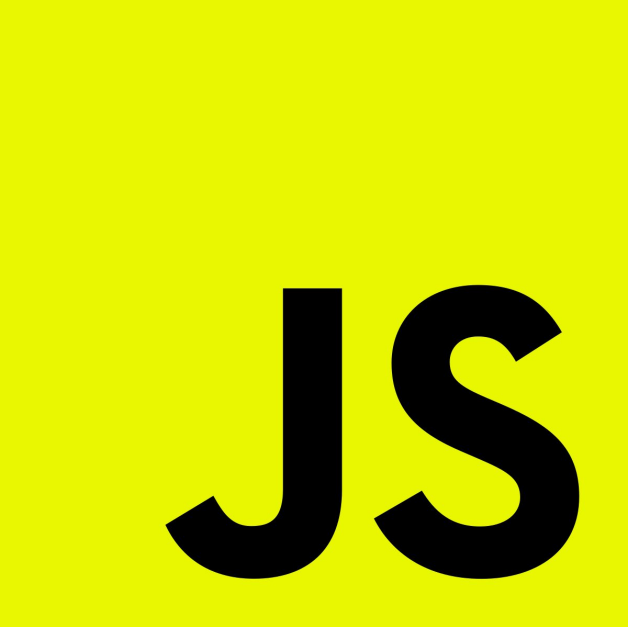
A large, bold, black 'JS' logo is centered on a bright yellow rectangular background.

JavaScript --

скриптовый язык программирования
с динамической типизацией и
автоматическим управлением
памятью.

ECMAScript --

стандарт языка JavaScript.



JS

!=



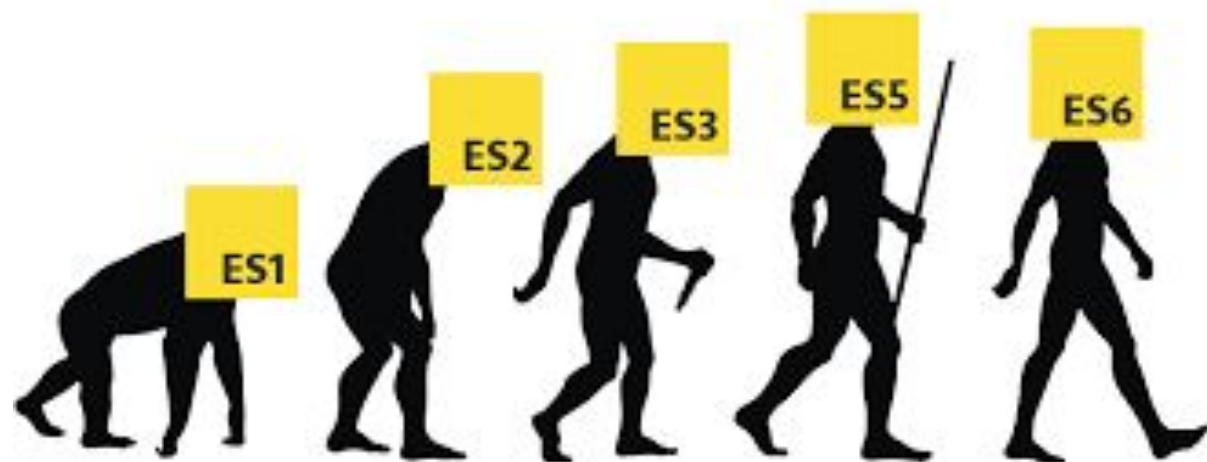
1997

1998

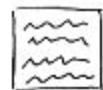
1999

2009

2015

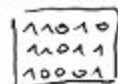


Source code:
hello.c



→ COMPILER →

Machine code:



Program (also
called binary,
executable ...)

→ run the
program →

result



Source code:
hello.py



→ INTERPRETER → result

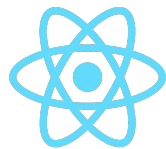


Где раньше использовался JavaScript?

- Создание всплывающих окон (попапов);
- Создание анимаций и переходов;
- Создание галерей и слайдеров;
- Создание мобильных меню;
- Работа с сервером;
- Управление HTML и CSS;

В браузере

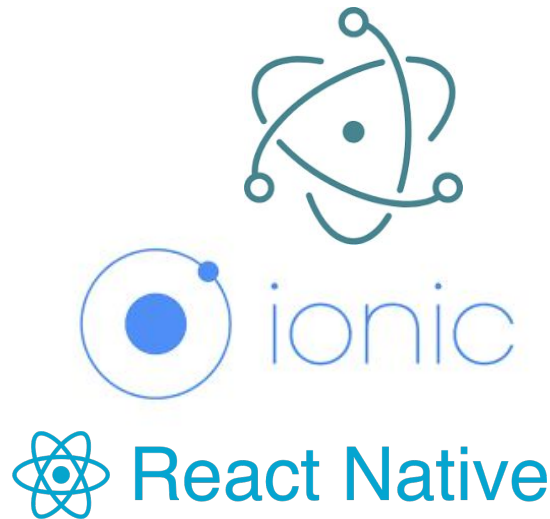
Где сейчас используется JavaScript?

The logo for ES6 (ECMAScript 2015) is a yellow square with the text "ES6" in black.

В браузере



На сервере



**На телефоне и
десктопе**

Переменные (Variables) --

это способ записать значение для дальнейшего использования. Тут подходит ассоциация с ящиком или коробкой, в которую мы ложим то, что хотим использовать позже.

```
1  /*
2   Задавать переменные можно используя var, let и const.
3
4   var -- устаревший способ, его на этом курсе использовать нельзя!
5  */
6
7  var someNumber = 1;
8  var someString = 'O Captain! my Captain! our fearful trip is done';
9
10 // Переменную, заданную через var, легко переопределить:
11
12 var number = 0;
13 console.log(number); // 0
14 var number = 1;
15 console.log(number); // 1
```

ES6

```
~/Courses/ec-develop/vars/common.js (Maket, essential-course, ec-develop) - Sublime Text (UNREGISTERED)
common.js x
1 // Предпочтительно задавать переменные через const и let.
2
3 let someNumber = 1;
4 const someString = 'O Captain! my Captain! our fearful trip is done';
5
6 // От var они отличаются тем, что их нельзя прямо переопределить:
7
8 let thing = 1; // 1
9 let thing = 2; // SyntaxError: redeclaration of let thing
10
11 // Однако, переменную, заданную через let, можно переопределить иначе:
12
13 let thing = 1; // 1
14 thing = 2; // 2
15
16 // С const так не выйдет:
17
18 const thing = 1; // 1
19 thing = 2; // TypeError: invalid assignment to const 'thing'
20
```

Line 19, Column 61 UTF-8 Tab Size: 4 JavaScript

Это не все особенности var, let и const, однако нам этого пока что достаточно

Основное правило выбора ключевого слова --

используем const во всех случаях, кроме тех, когда точно известно, что значение переменной будет изменяться.

Типы данных (Data Types)

1. Прimitives

- a. Числа (Numbers);
- b. Строки (Strings);
- c. Логические (Boolean)
- d. Символ (Symbol);
- e. null;
- f. undefined.

2. Сложные

- a. Объекты (Objects);
- b. Массивы (Arrays);
- c. Функции (Functions).

Числа (Numbers)

Числа (Numbers)

1. Целые (integer):

25

1000

25000

0x00

0x10

2. Дробные (floating points):

3.14

0.5

0.333333

Для написания дробных
чисел используется точка,
а не запятая

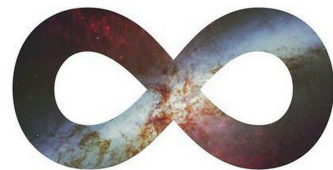
isFinite() может использоваться для проверки на бесконечность. Возвращает true или false

Бесконечность (Infinity) --

получается при попытке деления на ноль или работе с очень большими или очень маленькими числами.

```
console.log(1/0) // Infinity
```

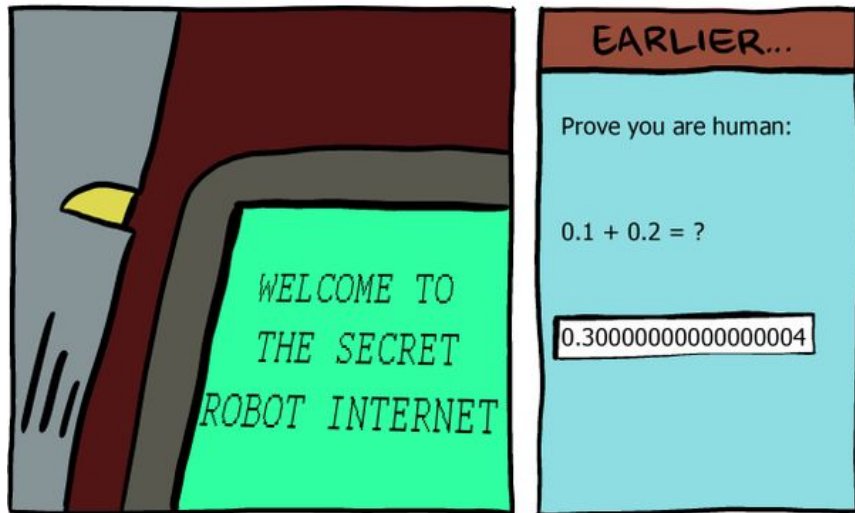
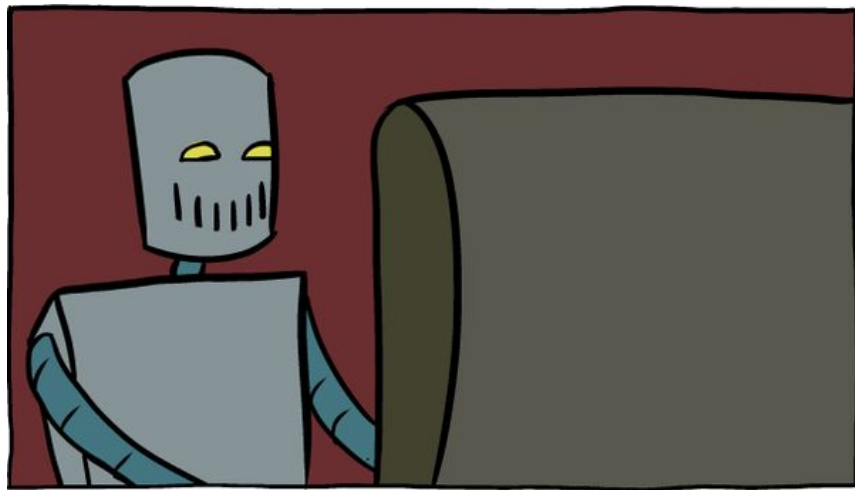
```
console.log(-1/0) // -Infinity
```



common.js x

```
1  /*
2   NaN (Not a Number) -- значение, которое появляется, когда мы пытаемся
3   произвести арифметическую операцию со значением, которое нельзя
4   превести к числовому типу
5  */
6
7  console.log('Maximumstart' * 2); // NaN
8  console.log('Maximumstart' - 2); // NaN
9  console.log('Maximumstart' / 2); // NaN
10 console.log('Maximumstart' + 2); // Maximumstart2
11
12 console.log(0/0); // NaN
13 console.log(Infinity/-Infinity); // NaN
14 console.log(undefined + 2); // NaN
```

```
1  /*
2   Проверка через оператор сравнения не работает
3   NaN === NaN // false
4   "someNoteANumberText" === NaN // false
5   Поэтому используем специальную функцию isNaN
6  */
7
8  isNaN(2); // false
9  isNaN('2'); // false, потому что приводится к числовому типу
10
11  isNaN(true); // false, поскольку true = 1
12  isNaN(false); // false, поскольку false = 0
13  isNaN(null); // false
14  isNaN(undefined); // true
15
16  isNaN('Maximumstart'); // true
17
18  isNaN(NaN); // true
19  isNaN(Infinity); // ?
```



Стоит учитывать, что из-за особенностей преобразования чисел с плавающей точкой, операции над дробными числами могут возвращать непредсказуемый результат.

Строки (Strings)

В домашних заданиях и выпускных проектах нужно будет выбрать между одинарными и двойными

"Строка" --

в JavaScript это все, что записано в кавычках.

Использование двойных или одинарных кавычек -- дело вкуса.

common.js x

```
1  /*
2   В JavaScript любые текстовые данные являются строками.
3   Можно использовать одинарные (''), двойные (") и косые (``) кавычки.
4
5   Одинарные и двойные кавычки абсолютно равноценны, при использовании
6   косых появляются дополнительные плюшки.
7  */
8
9  const firstPart = "Все равно любовь моя — тяжкая гиря ведь — ";
10 const secondPart = 'висит на тебе, куда ни бежала б.';
11
12 const phrase = firstPart + secondPart;
13 // "Все равно любовь моя — тяжкая гиря ведь — висит на тебе, куда ни бежала б."
14
15 const fakeBoolean = 'true';
16 const date = '20.03.2018';
17 const pi = '3.14';
```

```
common.js x
1 // Сложение строк:
2
3 const date = "20.03.2018";
4 console.log("Указанная дата:" + " " + date + "."); // Указанная дата: 20.03.2018.
5
6
7 // При использовании одинарных или двойных строк, переносы не учитываются:
8
9 const poetry = 'O Captain! my Captain! our fearful trip is done,
10 The ship has weather'd every rack, the prize we sought is won,
11 The port is near, the bells I hear, the people all exulting,';
12
13 // Раньше перенести строку можно было только с помощью управляющего символа:
14
15 const poetry = 'O Captain! my Captain! our fearful trip is done, \n' +
16 'The ship has weather\'d every rack, the prize we sought is won, \n' +
17 'The port is near, the bells I hear, the people all exulting,';
18
19
```

ES6

~/Courses/ec-develop/vars/common.js (Maket, essential-course, ec-develop) - Sublime Text (UNREGISTERED)

common.js

```
1  /*
2     Сейчас намного чаще используется новый синтаксис с использованием косых
3     кавычек (`), в котором исправлены перечисленные недостатки
4  */
5
6  // Переносы поддерживаются и управляющие символы не нужны!
7
8  const poetry = `0 Captain! my Captain! our fearful trip is done,
9  The ship has weather'd every rack, the prize we sought is won,
10 The port is near, the bells I hear, the people all exulting`;
11
12 // Работать с переменными и выражениями стало удобнее:
13
14 const date = '20.03.2018';
15 const string = `Указанная дата: ${date}.` // "Указанная дата: 20.03.2018."
16
17 const math = `Каждый школьник знает, что 2 * 2 = ${2 * 2}.`
18 // "Каждый школьник знает, что 2 * 2 = 4."
```

Line 18, Column 43

UTF-8

Tab Size: 4

JavaScript

Логический (булев) тип данных (Boolean) --

тип данных, принимающий два возможных значения -- **true** или **false**. Используется повсеместно, поскольку на его основе легко строить логику приложения.

Специальные значения `null` и `undefined`

`null` --

означает отсутствие какого-то значения;

`undefined` --

указывает на то, что значение не задано. Часто встречается при оглашении переменной без значения или обращении к несуществующей переменной.

Как узнать тип данных?

common.js

```
1 // typeof -- правильный способ узнать тип данных
2
3 typeof 2; // "number"
4 typeof '2'; // "string"
5
6 typeof 'true'; // "string"
7 typeof true; // "boolean"
8
9 typeof Symbol("name"); // "symbol"
10
11 typeof undefined; // "undefined"
12 typeof NaN; // "number"
13 typeof Infinity; // ?
14
15 typeof null; // "object"
16
```

Математические операторы

```
1 // Математические операторы
2
3 console.log(1 + 2); // 3
4 console.log(2 - 3); // -1
5 console.log(4 / 2); // 2
6 console.log(10 * 10); // 100
7 console.log(8 % 2); // 0
8
9
10 // Дополнительные операторы и константы есть в объекте Math
11
12 Math.sqrt(25); // 5
13 Math.PI; // 3.141592653589793
14 Math.E; // 2.718281828459045
15 Math.pow(2,4); // 16
```

```
1 // Операторы сравнения возвращают булевые значения
2
3 console.log(10 > 5); // true
4 console.log(2 >= 2); // true
5 console.log(5 > 10); // false
6 console.log(5 <= 5); // true
7
8 // Оператор сравнения, который сравнивает значения, но
9 // не сравнивает типы данных
10 console.log(3 == '3'); // true
11
12 // Оператор сравнения, который сравнивает значения и типы данных
13 console.log(3 === '3'); // false
14
15 // Нестрогий оператор "не равно"
16 console.log(3 != '3'); // false
17
18 // Строгое "не равно"
19 console.log(3 !== '3'); // true
20
```

Контрольные вопросы

1. Как определить, четное число или нечетное?
2. Как получить бесконечность?
3. Что такое “динамическая типизация”?
4. Как сложить две строки?
5. Почему сравнение:

`100 === "100"`

Вернет `false`?

Самостоятельная работа

1. Сейчас мы будем прибавлять число 100 к заданному пользователем числу. Создайте переменные `number`, `result`, `string`. В `string` должна быть записана фраза "Результат сложения числа (`number`) и 100 = (`result`)". В `string` мы можем передать строку ("5", "100") и все должно считаться корректно.