КОМПЛЕКС СОЦИАЛЬНОГО РАЗВИТИЯ МОСКВЫ

Моя профессия

Введение в JavaScript

История JS, типы данных, переменные, операторы

Что такое JavaScript ?

Моя профессия

- JavaScript – это высокоуровневый, мультипарадигменый, динамически типизированный язык программирования.
- JavaScript был создан для того, чтобы сделать веб-страницы динамичными.
- Наиболее широкое применение нашел в браузерах.

JavaScript не имеет никакого отношения к **Java**

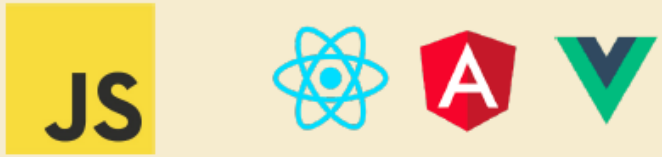
Взаимодействие Front-end технологий

Моя профессия

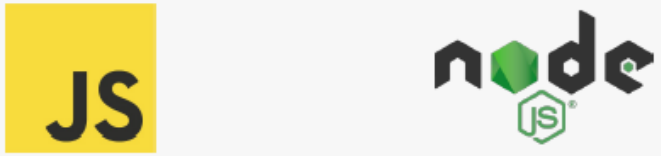
Для чего применяется JavaScript

Моя профессия

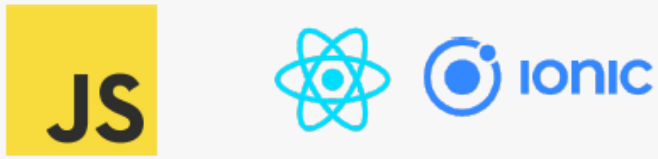
Динамические эффекты и Веб-приложения в браузере



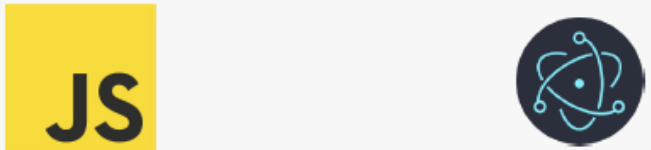
Серверные веб-приложения



Мобильные приложения



Приложения для ПК



История развития JavaScript

Моя профессия

1995

- **Бренден Эйч** создает самую первую версию JavaScript за **10** дней с названием **Mocha**, эта версия обладала фундаментальными возможностями текущей версии языка.

1996

- Mocha меняет свое название на **LiveScript**, а затем на JavaScript из-за популярности Java. В JavaScript нет почти **ничего** от Java.

- Microsoft запускает IE, копирует JavaScript у Netscape и называет его JScript.

1997

- Для стандартизации языка сообщество ECMA выпускает **ECMAScript 1** (ES1), первый официальный стандарт для JavaScript (ECMAScript – **стандарт**, JavaScript – язык программирования).

2009

- ES5 (ECMAScript 5) представлено много новых возможностей языка.

2015

- Был представлен стандарт ES6/ES2015 (ECMAScript 2015), **самое масштабное обновление стандарта**.

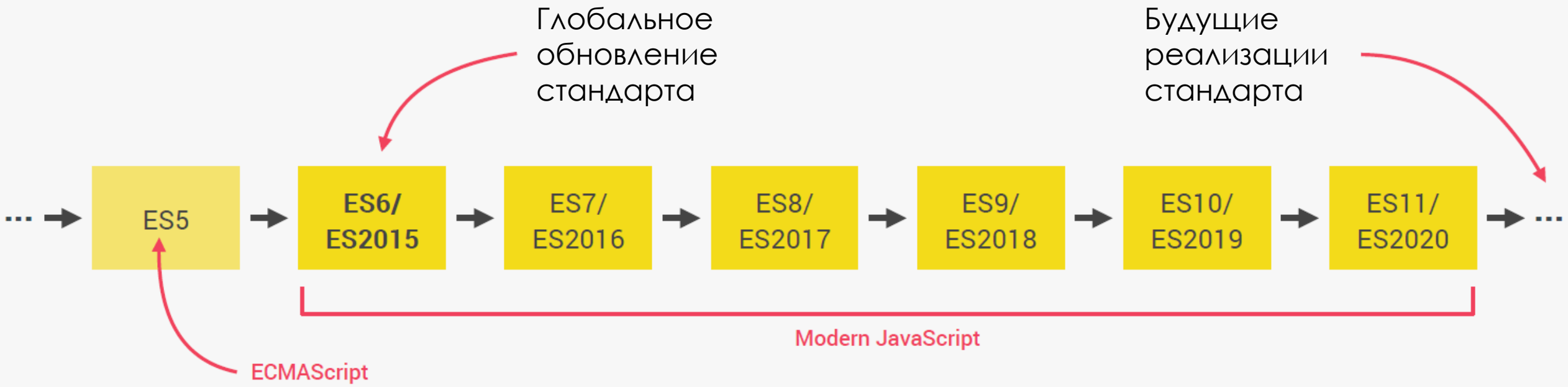
- ECMAScript переходит на ежегодный цикл обновления стандарта

2016 – ∞

- Обновления ES2016 / ES2017 / ES2018 / ES2019 / ES2020 / ES2021 / … / ES2089 😅

Стандарты JavaScript

Моя профессия



Примитивные типы данных

Моя профессия

- **Number:** Десятичные числа, числа с плавающей точкой. `let age = 23;`
- **String:** Последовательность символов, используется для текста. `let firstName = 'Jonas';`
- **Boolean:** Логический тип данных, имеет два значения, true и false. `let fullAge = true;`

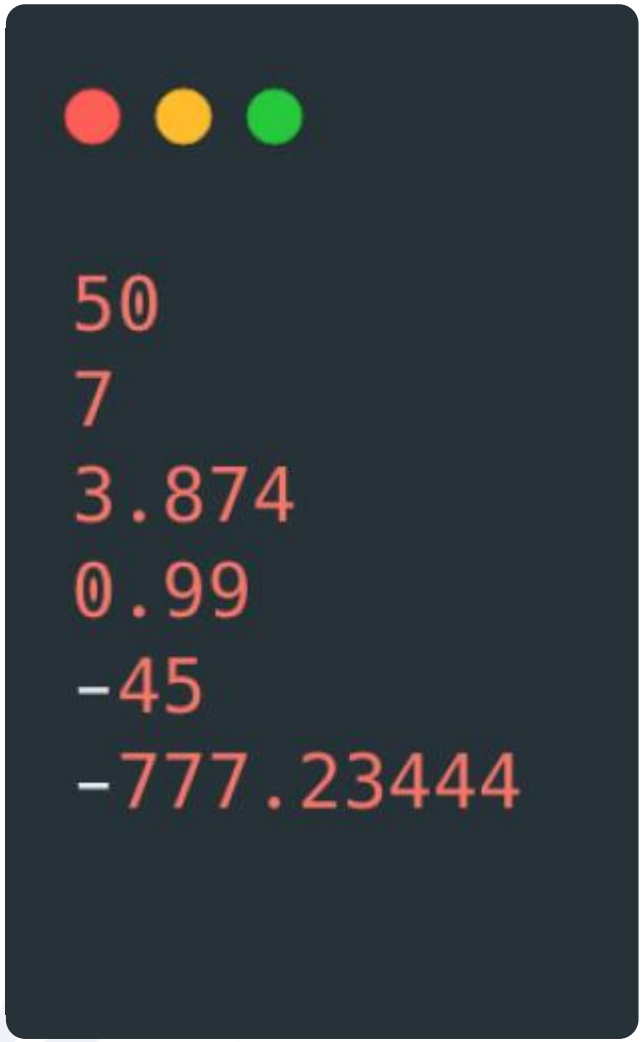
  Используется для принятия решений.
- **Undefined:** Значение переменной, которое не присвоено никакое значение. `let children;`
- **Null:** Также означает отсутствие значения.
- **Symbol:** Означает уникальное значение, которое не может быть изменено

  (Сейчас еще не используется).
- **BigInt:** Используется для больших числовых значений (больше чем Number).

JavaScript – язык с **динамической** типизацией. Это значит, что мы не можем задать тип значения переменной. Вместо этого тип данных определяется автоматически.

Числа в JavaScript

Моя профессия



- JavaScript имеет один числовой тип данных – Number
- Положительные числа.
- Отрицательные числа.
- Целые числа
- Десятичные числа
- Числа с плавающей точкой

Простые операции с числами

Моя профессия

```
// Сложение
50 + 5 // 55

// Вычитание
90 - 1 // 89

// Умножение
5 * 4 // 20

// Деление
60 / 5 // 12

// Деление по модулю
10 % 3 // 1
```

В JavaScript мы можем использовать все базовые математические операции

// - дает возможность написать комментарий, который игнорируется при выполнении программы

NaN – not a number

```
0 / 0 // NaN

35 + NaN // NaN
```

NaN – это **числовое** значение, которое означает, что это … не число

Переменные в JavaScript

Моя профессия

Переменные похожи на «Именованные контейнеры» для значений в JavaScript

Мы можем сохранить значение и дать ему название, а еще…

- переименовать
- использовать
- изменить значение позже

Базовый синтаксис

Моя профессия

```
let someName = value;
```

Ключевое слово — Название переменной — Значение переменной

```
let age = 55;
```

Создаем переменную с названием «age» и значением 55

Использование переменных

Моя профессия

```
// объявляем переменную cats и присваиваем значение 4
let cats = 4;

// объявляем переменную dogs и присваиваем значение 6
let dogs = 6;

// складываем значения переменных и получаем 10
cats + dogs;
```

Сохранение данных в новой переменной

Моя профессия

```
// объявляем переменную cats и присваиваем значение 4
let cats = 4;
// объявляем переменную dogs и присваиваем значение 6
let dogs = 6;
// складываем значения переменных и получаем 10
cats + dogs;

// для сохранения нового значения можно записать
// его в новую переменную
let animals = cats + dogs;
```

Изменение значения переменной

Моя профессия

```
let students = 44;

// отчисляем одного студента
students - 1; // 43
students; // все еще 44

// пробуем еще раз
students = students - 1;
students; // 43
```

Здесь не изменяется значение, сохраненное в переменной students

А здесь меняется

Константы в JavaScript

Моя профессия

```
const users = 4;
users = 30; // ERROR

const age = 12;
age = age + 1; // ERROR
```

const работает также как let, только вы не можете изменить значение

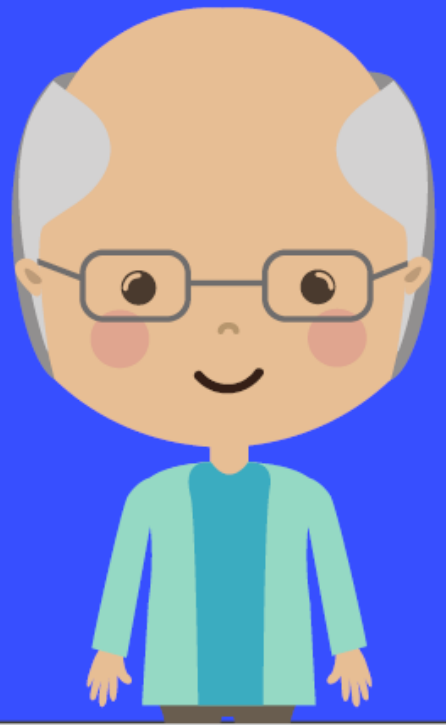
Используются для значений, которые не изменяются при работе приложения

Устаревший способ объявления переменной

Моя профессия

VAR

```
var tripDistance = 7.4;
```



До появления let и const, var был единственным способом объявления переменных. В наши дни нет причин использовать это.

Строки в JavaScript

Моя профессия

STRINGS

В JavaScript, строки – это кусочки текста или последовательность символов.

Строки оборачиваются в двойные или одинарные кавычки.

Особенности применения кавычек в строках

Моя профессия

```
let firstname = "Arnold"; // Строка обернута в двойные кавычки

let message = "Однажды в студеную зимнюю пору...";

let animal = 'Dumbo Octopus'; // Строка в одинарных кавычках

let bad = "Это не правильно!!!'; Так работать не будет!
```

Когда вы используете двойные или одинарные кавычки, просто будьте уверены в том, что в начале и в конце строки они одинаковые.

Шаблоны в строковых литералах

Моя профессия

```
let message = 'I Like JavaScript.'; // обычный строковый литерал
// * литерал - это некое простое значение

let born = 1990;

let age = `Мне ${2022 - born} лет`; // Мне 32 лет
```

Шаблоны позволяют встраивать выражения JavaScript в строки, в результате формируется строка, содержащая результат выполнения кода в шаблоне.

Для создания шаблона используются апострофы, обычно находятся над клавишей TAB, значение или выражение заключается в фигурные скобки, перед которыми идет знак доллара, $.

Шаблоны в строковых литералах

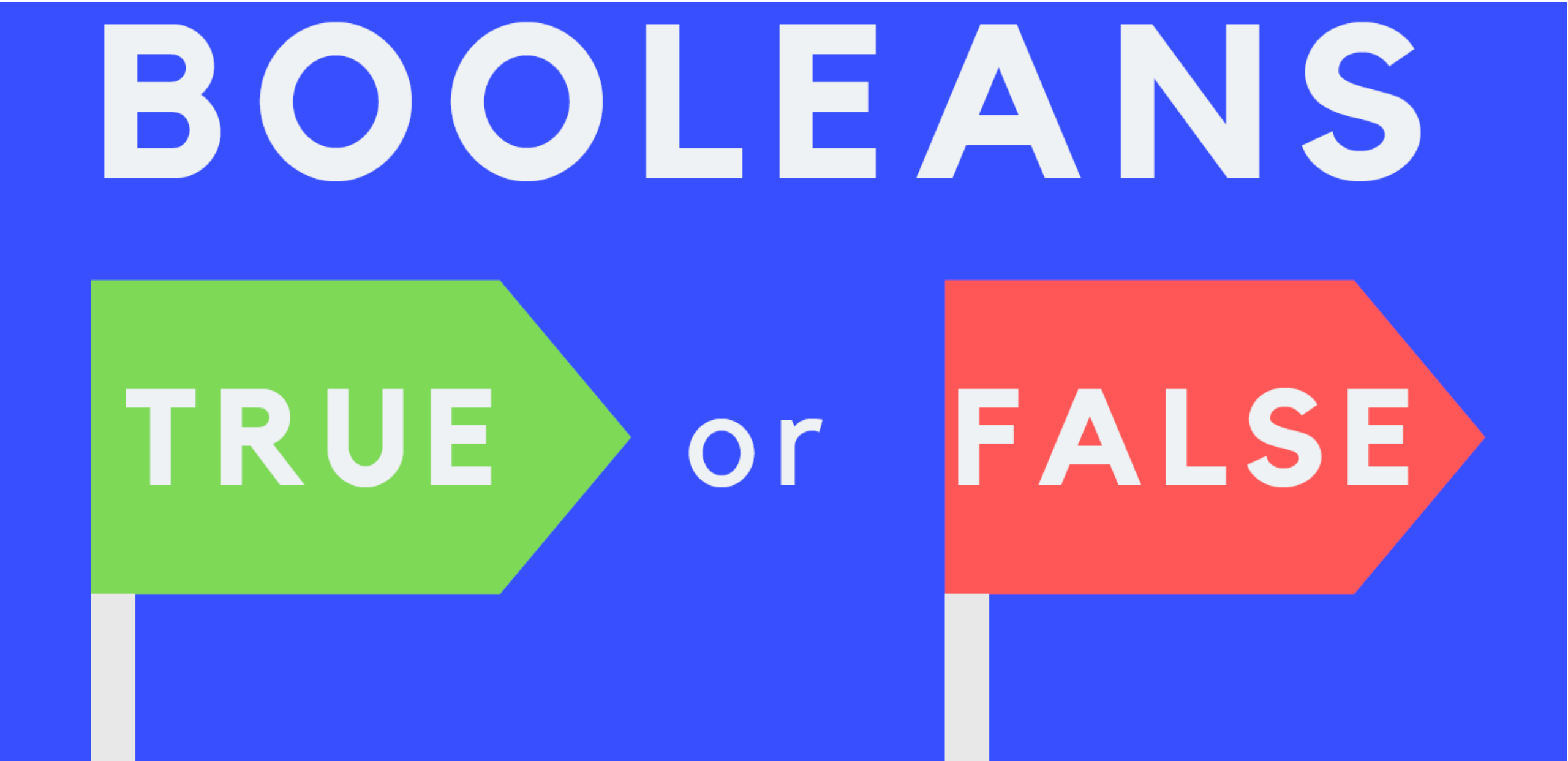
Моя профессия

Еще несколько примеров

```
let userName = 'Ziggy31';
`Добро пожаловать, ${userName}`; // Добро пожаловать, Ziggy31

`Game over, ${userName}`; // Game over, Ziggy31



let item = 'Огурцы';
let price = 2.50;
let quantity = 4;

`Вы купили ${quantity}кг ${item}, итоговая сумма $${price * quantity}`;
// Вы купили 4кг Огурцы, итоговая сумма $10
```

Булев тип данных

Моя профессия



Применение булева типа данных

Моя профессия

```
let isLoggedIn = true;

let gameOver = false;

const isWaterWet = true;
```

Булев тип имеет всего два значения – это true и false.
True – истина, false – ложь.

Undefined

Моя профессия

Значение undefined имеют переменные, которые объявлены, но которым еще не присвоено значение.

```
let message; // значение переменной не задано
message; // undefined
message = 'Завершено успешно';

let age; // значение переменной не задано
age; // undefined
age = 32;
```

Null

Моя профессия

Null означает полное отсутствие значения.

```
// еще никто не зарегистрирован
let loggedUser = null; // отсутствие значения

// пользователь регистрируется
loggedUser = 'Alan Rickman';
```

Булева логика, операторы И(&&), ИЛИ(||), НЕ(!)

Моя профессия

A И B

У меня есть водительские права
И у меня хорошее зрение

B \ A	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

👇

Выражение истинно, когда оба значения истинны

A ИЛИ B

У меня есть сахар ИЛИ деньги
на его покупку

B \ A	TRUE	FALSE
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

👇

Выражение истинно, когда хотя бы одно значение истинно

НЕ A, НЕ B

👇

Изменяет значение на противоположное.
Истина становится ложью, а ложь истиной.

Если значение A было Истинно, оно меняется на ложно, то же самое в обратную сторону.

Примеры использования булевой логики

Моя профессия

```
age = 16
```

- A: age больше 20? = `false`
- B: age меньше 30? = `true`

Примеры

- `!A` (`false`) ⇨ `true`
- `A AND B` (`false`, `true`) ⇨ `false`
- `A OR B` (`false`, `true`) ⇨ `true`
- `!A AND B` (`true`, `true`) ⇨ `true`
- `A OR !B` (`false`, `false`) ⇨ `false`

B \ A	TRUE	FALSE
AND		
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

B \ A	TRUE	FALSE
OR		
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

Операторы сравнения

Моя профессия

```
<   // меньше

>   // больше

<=  // меньше либо равно

>=  // больше либо равно
```

Операторы сравнения, примеры

Моя профессия

```
10 > 1;      // true
0.2 > 0.3;   // false
-10 < 0;     // true
50.5 < 5;    // false
0.5 <= 0.5;  // true
99 >= 4;     // true
99 >= 99;    // true
'a' < 'b';   // true
'A' > 'a';   // false
```

Операторы сравнения возвращают значения булевого типа!

Хоть это и необычно, но вы можете сравнивать строки. Просто будьте осторожны, результат может быть неожиданный, особенно когда вы используете специальные символы.

Операторы сравнения

Моя профессия

```
==  // равенство

!=  // не равно

===  // строгое равенство

!==  // строго не равно
```

Строгое равенство и неравенство учитывают тип данных сравниваемых значений.

Операторы сравнения, примеры

Моя профессия

```
5 == 5;                 // true
'b' == 'c';             // false
7 == '7';               // true
0 == '';                // true
true == false;          // false
0 == false;             // true
null == undefined;      // true
9 != '9';               // false
```

Нестрогое равенство приводит значения к одному типу, а затем сравнивает.

Псевдо-истинные и псевдо-ложные значения

Моя профессия

Все значения в логических операциях приводятся к булевому типу, TRUE либо FALSE.

Псевдо-ложные значения:

- false
- 0
- “” (пустая строка)
- null
- undefined
- NaN

Все остальные значения преобразуются в true.

Операторы сравнения, примеры

Моя профессия

```
5 === 5;                // true
1 === 2;                // false
7 === '7';              // false
0 === '';               // false
0 === false;            // false
null === undefined; // false

9 != '9';               // false
9 !== '9';               // true
```

Операторы строгого равенства и неравенства сравнивают значения без преобразования типов.

Потенциально опасная операция

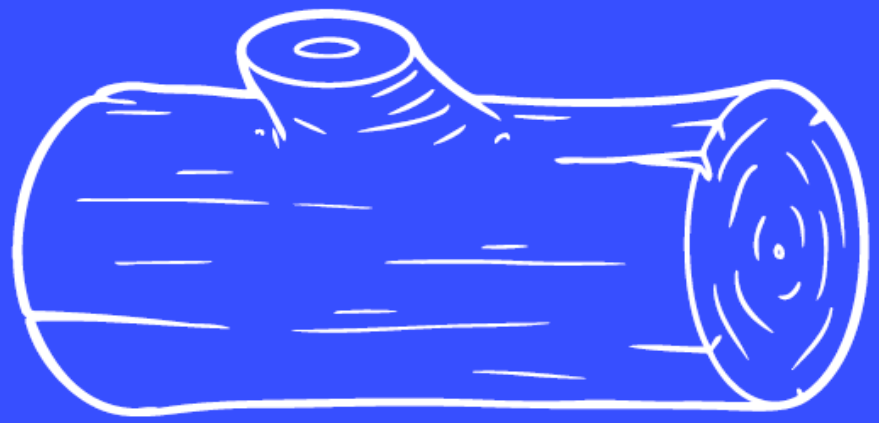
Вывод данных в консоль

Моя профессия

console.log()

Печатает аргументы в консоль браузера

(это необходимо, если вы начинаете работу с файлами)



Использование console.log()

Моя профессия

app.js

```
let firstName = 'Mark';
let lastName = 'Twain';

let fullName = `${firstName} ${lastName}`;

console.log(fullName); // Mark Twain
```

Пишем код в файле app.js

index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>JS Demo</title>
    <script src="app.js"></script>
</head>
<body>

</body>
</html>
```

Подключаем скрипт в файле index.html