

# JavaScript в веб-разработке

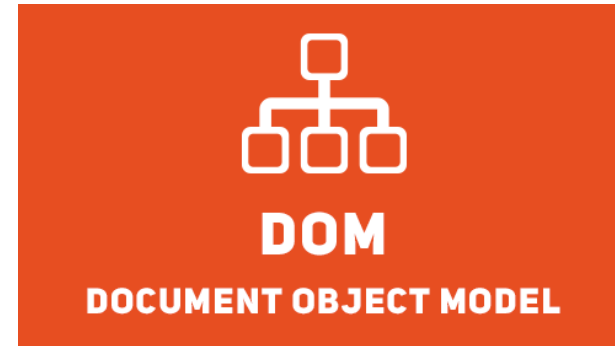
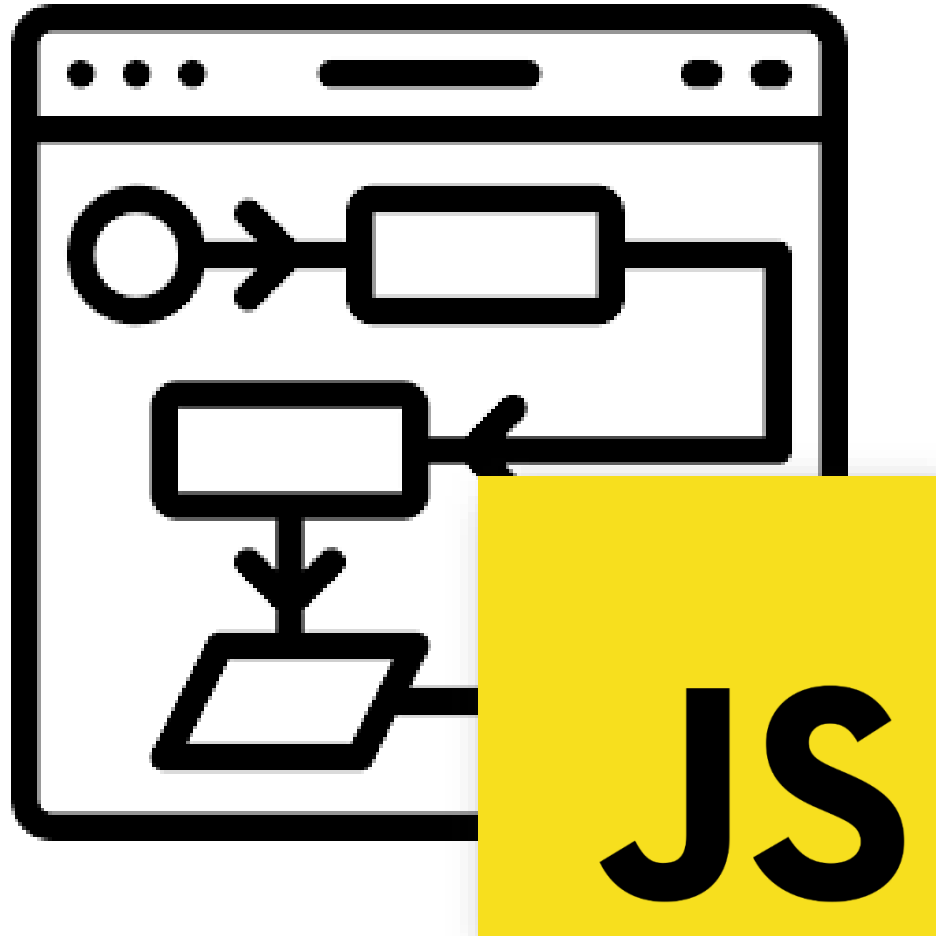


JavaScript  
Courses

[ORT.DP.UA/JS](https://ort.dp.ua/js)

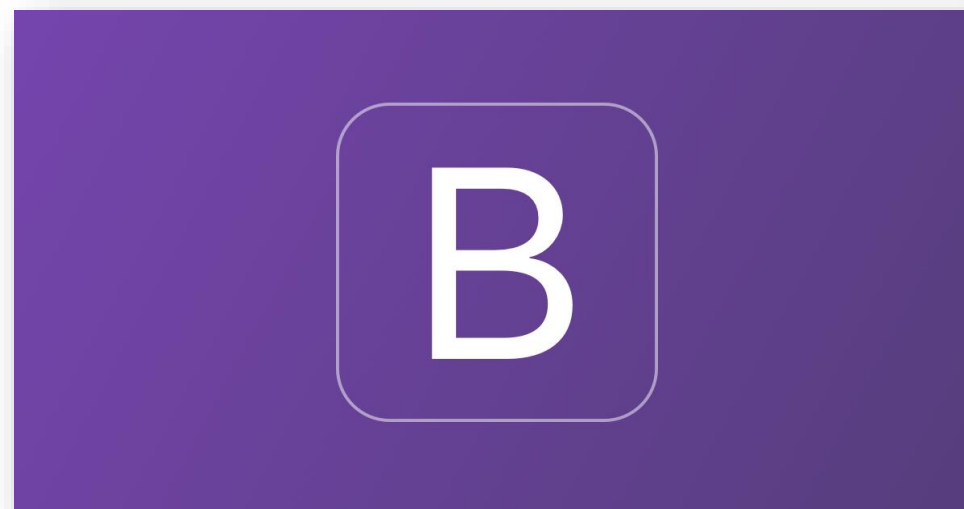
О чём курс?

# О программировании и веб-разработке с применением языка JavaScript



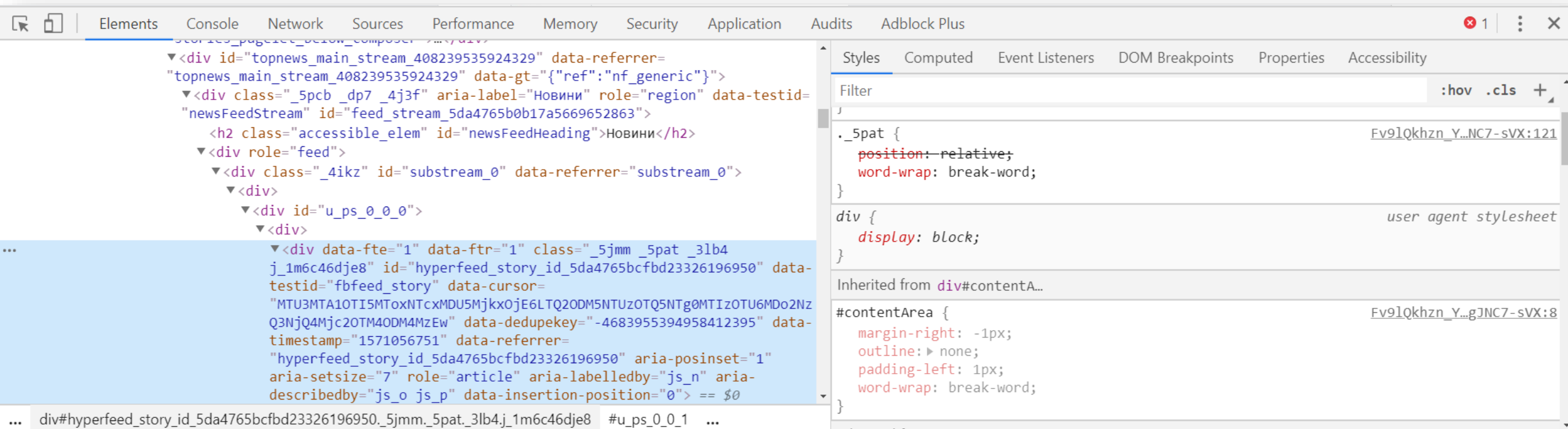
Что необходимо знать  
перед началом курса?

# 1. Вёрстка



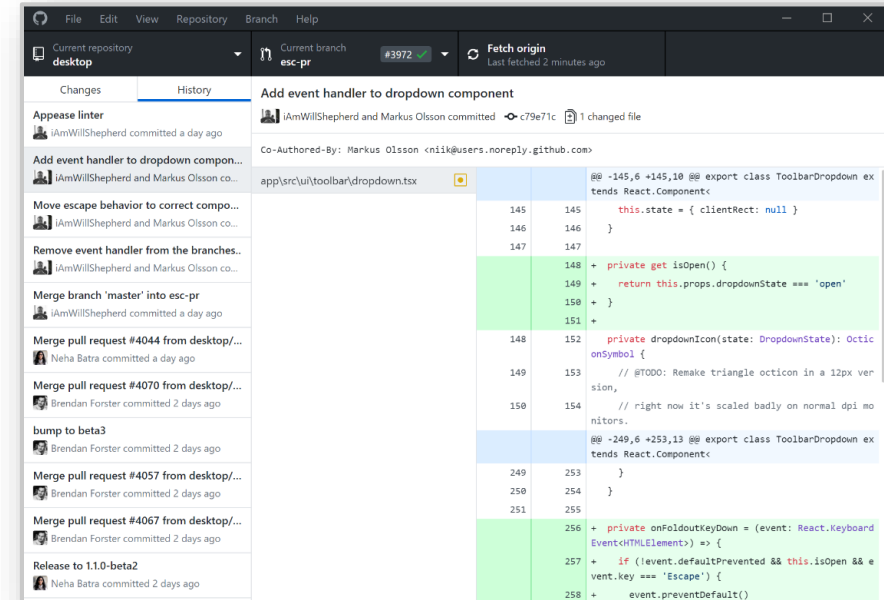
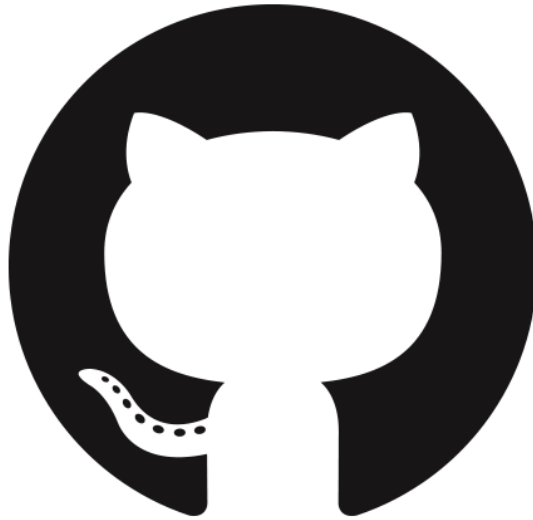
Необходим опыт в вёрстке страниц, понимание механизма **CSS селекторов** и **анимации переходов** в CSS (**transition**). Также необходимы знания и опыт в применении библиотеки **Bootstrap 4**.

## 2. Инструменты разработчика



Необходимо знание и умение пользоваться инструментами **DevTools** (он же *Web Developers Tools*, он же *Консоль разработчика*, он же *Инспектор объектов*). В частности разделами **Elements** (исследование дерева документа) и **Network** (сетевая активность приложения).

# 3. GitHub



Необходимо уметь, **создавать свои репозитории**, выгружать в них код, выполнять **обновление** своего кода в репозиториях (**commit**'ы), а также уметь загружать себе сторонние репозитории.

**Рекомендации:** вы можете использовать программу **GitHub Desktop**

Как будем  
взаимодействовать



**Наша группа: JS9**

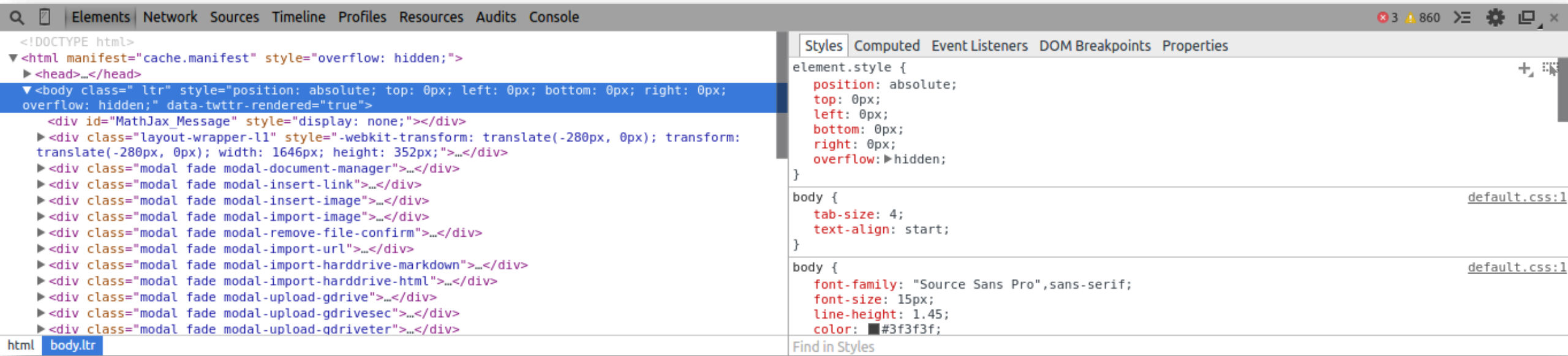
**[js9ort.github.io](https://js9ort.github.io)**



Общение при  
помощи  
мессенджера **Slack**,  
а для материалов и  
домашних заданий  
будем использовать  
**GitHub**

Что нам понадобится

# Браузер(ы) и инструменты разработчика



# Редактор кода



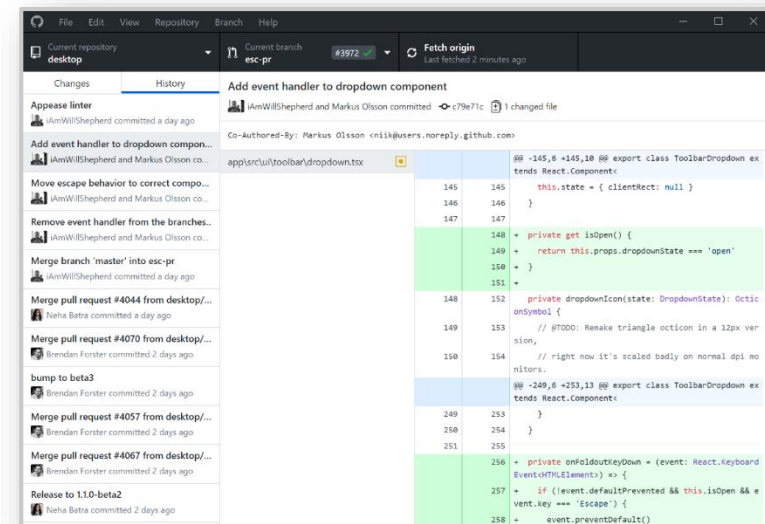
На ваше усмотрение

# А также...



# slack

\*есть веб-версия



**GitHub Desktop**  
**[WinX64, MacOS]**  
(или консоль...)

Первым делом!

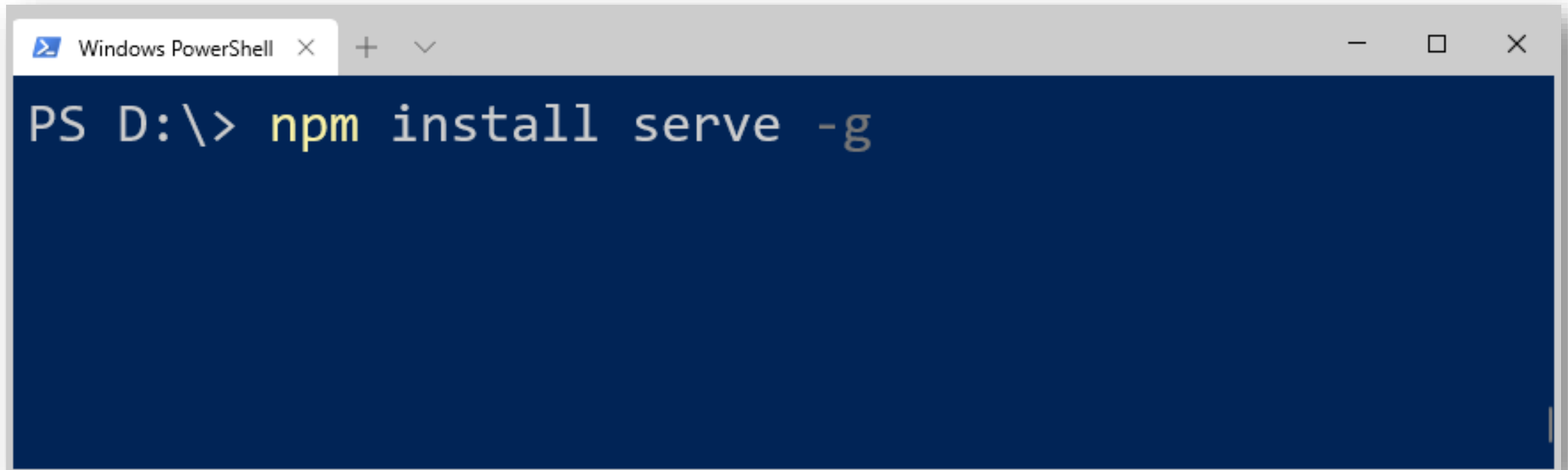
# Установим Node.js



<https://nodejs.org/>



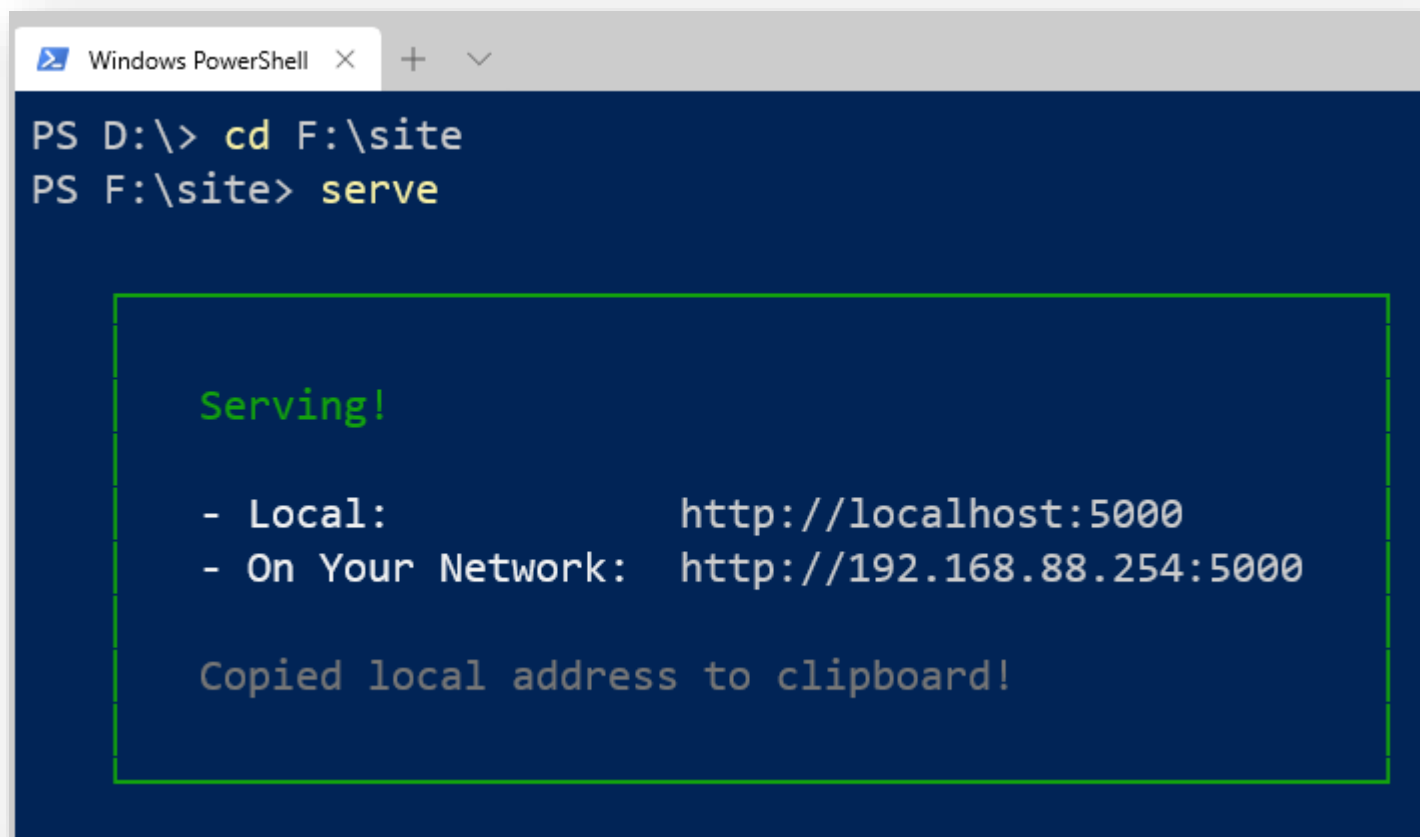
# Установим веб-сервер

A screenshot of a Windows PowerShell terminal window. The title bar at the top shows 'Windows PowerShell' with a close button. The terminal has a dark blue background. The prompt 'PS D:\>' is followed by the command 'npm install serve -g' in a light blue font. The 'npm' part of the command is highlighted in yellow. The cursor is at the end of the command.

```
PS D:\> npm install serve -g
```

Для этого нам понадобится **консоль** (PowerShell, bash...) и менеджер пакетов **NPM** (идёт в комплекте **Node.JS**). С их помощью установим статический сервер Serve при помощи консольной команды **npm install serve -g** при использовании **\*nix**'ов не забудьте про **sudo**.

# Запустим веб-сервер



```
Windows PowerShell
PS D:\> cd F:\site
PS F:\site> serve

Serving!

- Local: http://localhost:5000
- On Your Network: http://192.168.88.254:5000

Copied local address to clipboard!
```

Для запуска веб-сервера нам понадобится **каталог** в котором будут лежать файлы нашего сайта, например это будет **F:\site\** При помощи консоли переходим в этот каталог командой **cd F:\site\** Далее при помощи команды **serve** запустим веб-сервер и корневым каталогом в нём будет каталог нашего сайта. П.С. не забудьте в каталоге сайта создать файл **index.html** В консоли будет указано по какому адресу посещать наш сайт.

Поехали!

# ES

**ECMAScript**

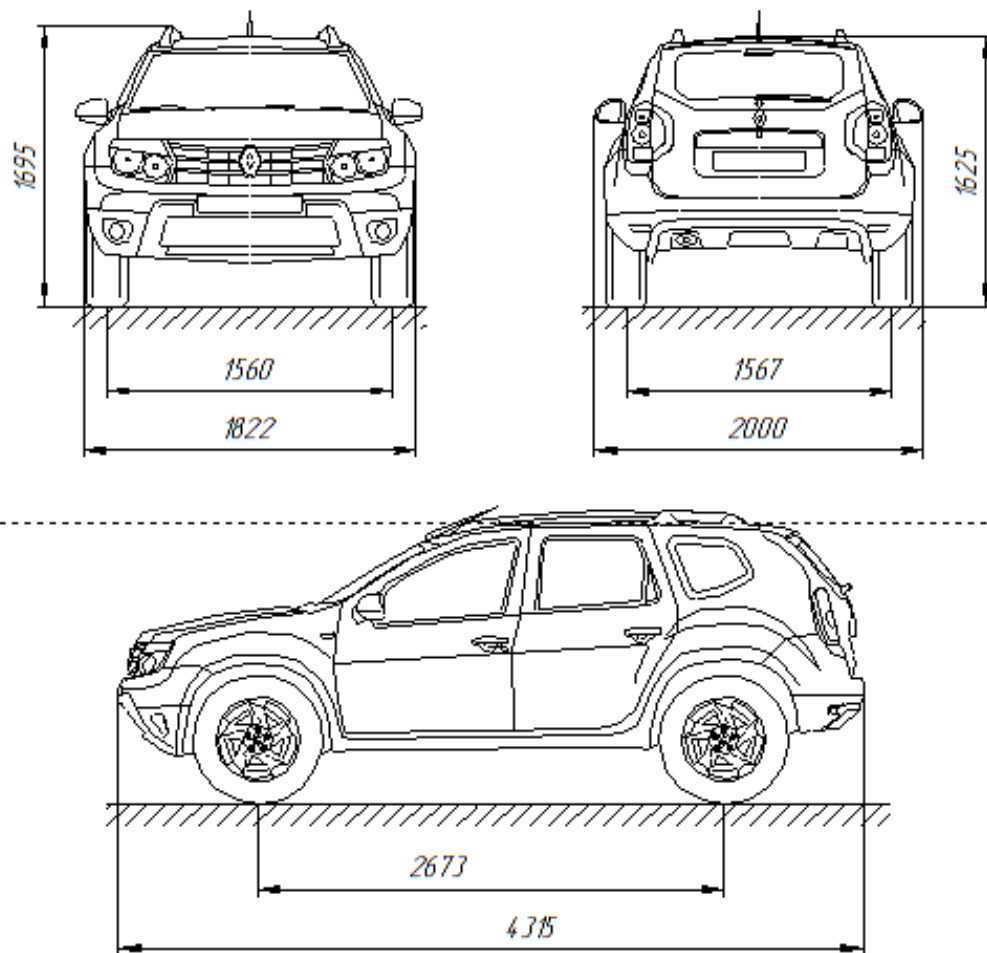
vs

# JS

**JavaScript**

# ECMAScript

# JavaScript



*Спецификация...*



*...и её реализация*

Начнём с простой задачи

или

О важности алгоритма

**«Задача банкомата»** Программа спрашивает у пользователя сумму, а в ответ сообщает купюры каких номиналов, и в каком количестве необходимо выдать. При этом *суммарное количество купюр должно быть минимально возможным*. Для простоты будем считать, что в банкомате есть только купюры по **1, 5, 50** гривен (при этом из количество не ограничено).

*Например: Пользователь вводит сумму: 552 грн.*

*В ответ программа выдаёт:*

50 грн. x 11;

5 грн. x 0;

1 грн. x 2;

Первая часть нашего  
курса: «Основы  
программирования  
на базе JavaScript»



# Часть 1. Основы программирования на базе JavaScript

1. Переменные и типы данных, операции над ними;
2. Условные и логические операторы, ветвление кода;
3. Структуры данных, коллекции, массивы и их обработка;
4. Циклы и перебирающие методы;
5. Функции и таймеры в JavaScript;
6. Объекты и классы в JS, принципы объектно-ориентированного программирования;
7. AJAX;
8. EventLoop и асинхронность в JS (Promise/Async/Await);
9. Обработка ошибок в JS.

# JavaScript без полной разметки

F: > site > <> index.html > ...

```
1  <script>
2
3      let message = 'Hello world!';
4
5      console.log(message);
6
7  </script>
8
```

В начале обучения мы можем ограничиться только тегами `<script></script>` для написания кода, и опускать полную разметку документа.

# **JavaScript – язык программирования**

**1. Императивный**

**2. Интерпретируемый**

**3. Чувствительный к регистру**

# Служебные функции для ввода/вывода данных

*\* которые нам помогут прожить без разметки))*

**console.log(...);** - вывод в консоль браузера;

**alert(...);** - вывод во всплывающем окне;

**prompt(...);** - окно с запросом информации;

**confirm(...);** - окно для подтверждения;

**document.write()** – вывести данные в документ.

*\*\* кроме **console.log()** применение перечисленных методов считается плохой практикой, но они могут нам помочь в процессе обучения.*

# Переменные и типы данных

# Переменные

```
1
2  var user_name    = "Elena";
3
4  let user_age     = 27;
5
6  const user_inn   = 3252873450;
7
8  console.log(user_name, typeof user_name);
9  console.log(user_age,  typeof user_age);
10 console.log(user_inn,  typeof user_inn);
11
```

Переменные объявляются при помощи ключевых слов **var**, **let** и **const**. Первые два способа отличаются областью видимости переменной которая создаётся. Третий создаёт переменную у которой нельзя заменить значения после инициализации.

# Типы данных в JavaScript

Переменные могут хранить значение одного из поддерживаемых типов данных. В ходе выполнения кода может меняться как содержимое переменной так и его тип. Тип влияет на то какие операции могут быть выполнены с переменной. Тип переменной можно получить при помощи оператора **typeof**.

**BigInt** – не стандартизирован до конца и имеет слабую поддержку в браузерах.

Подробнее: <https://learn.javascript.ru/types-intro>

# Преобразование типов

```
3 let data = "12.35";
4
5 let a = Number(data);
6
7 let b = parseFloat(data);
8 // let b = parseInt(...);
9
10 let c = +data;
11
12 console.log(a, b, c);
13
```

Несмотря на наличие механизма автоматического приведения типов может возникать ситуации требующие принудительного преобразования типов (чаще всего **string** к **number**), для этого есть ряд возможностей.



# Операторы, операнды и операции

# Оператор присвоения

Чтобы сказать компьютеру, что именно нужно записать в переменную используется оператор присвоения =

**a = 2 + 3 \* 5;**



Оператор присвоения берёт то что справа от него и записывает в переменную имя которой расположено слева от него.

# Операторы, операнды и операции...

Для выполнения действий (**операций**) над переменными или значениями (операндами) используются **операторы**, операторов существует много. С некоторыми из них все знакомы, например с арифметические операторами.

---

**Унарный оператор** – тот который взаимодействует только с одной переменной (операндом).

```
a++;  
b--;
```

**Бинарный оператор** – тот который взаимодействует с двумя переменными (операндами).

```
a = b**2 + 4*a*c;
```

---

У операторов есть приоритеты, какой приоритет выше, какой ниже запомнить непросто. Поэтому в случае сомнений какая операция будет первой а какая второй – смело используйте скобки. Принцип их применения такой же как и в математике – скобки повышают приоритет операции в них записанной.

```
a = (2+2) * 2;
```

«Скобками программу не испортишь» (с)

Подробнее: <https://learn.javascript.ru/operators>

## Операторы, операнды и операции...

### Что получится?

```
2  
3  let x = 5;  
4  
5      x = x++ + ++x;  
6  
7  console.log(x);  
8
```

?!?

# Выражения

По правую сторону от оператора присвоения может быть конкретное значение или же **выражение**, одна или несколько операций, результат выполнения которых будет записан в переменную имя которой стоит слева от знака присвоения. В выражении могут участвовать как и конкретные значения так и другие переменные.

$$a = b**2 + 4*a*c;$$

# NaN – Not a Number

Значение NaN (`typeof:number`) – в результате выполнения операции(й) означает, что среди операндов есть тот кто не являются `number`'ом или не может быть приведено к типу `number`.

# Математические функции

# Объект Math

```
2  
3   let data = 144;  
4  
5   let result = Math.sqrt(data);  
6  
7   console.log(data, result);  
8
```

**Math** - это встроенный объект с полями и методами для реализации математических постоянных и функций (в частности функции округления чисел).

Подробнее: <https://javascript.ru/math>



# Полезные методы объекта Number

# Методы объекта **Number** и переменных типа **number**

`Number.isNaN()` 

Определяет, является ли переданное значение значением `NaN`.

`Number.isFinite()` 

Определяет, является ли переданное значение конечным числом.

`Number.isInteger()` 

Определяет, является ли тип переданного значения «числом», а само число — целым значением.

`Number.isSafeInteger()` 

Определяет, является ли переданное значение безопасным целым числом (числом в диапазоне от  $-(2^{53} - 1)$  до  $2^{53} - 1$ ).

...

Подробнее: [https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global\\_Objects/Number](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/Number)

Тип string

# Строки – текстовый тип данных

```
2  
3   let name    = 'Jane';  
4  
5   let phrase  = `Hello ${name}! Welcome!`;  
6  
7   console.log(phrase);  
8
```

Строки могут быть заданы при помощи одинарных и двойных кавычек. А с помощью обратных («косых») кавычек можно создать строку с подстановкой в неё значений переменных или выражений - т.н. шаблонные строки.

Подробнее: <https://learn.javascript.ru/string>

# Строки – текстовый тип данных

```
2  
3 let str = 'Hello world!';  
4  
5 console.log(str.length);  
6  
7 console.log(str[2], str[7]);  
8
```

У строк есть понятие длины (количества символов), узнать которую можно при помощи свойства **.length**, также есть возможность обращаться к конкретному символу по его номеру (индексу), при помощи оператора **[]**.

Подробнее: <https://learn.javascript.ru/string>

# Преобразование к строке

```
2
3   let a = 42;
4   let b = true;
5
6   console.log( a.toString() );
7   console.log( b.toString() );
8   console.log( String(a) );
9   console.log( String(b) );
10
```

Любые типы можно привести к строке, для этих целей можно вызвать метод **.toString()** или воспользоваться функцией **String()**

# Тип `boolean` и условные операторы

# Тип Boolean

```
2  
3   let a    = true;  
4   let b    = false;  
5  
6   console.log(a, typeof(a));  
7   console.log(b, typeof(b));  
8
```

Переменная типа **boolean** содержит одно из двух возможных значений: истина (**true**) или ложь (**false**).



```

2
3 Boolean(undefined); //false;
4
5 // Number (and BigInt) to Boolean
6 Boolean(42); //true;
7 Boolean(-23.45); //true;
8 Boolean(0); //false;
9 Boolean(0.000001); //true;
10 Boolean(NaN); //false;
11
12 //String to Boolean
13 Boolean("Hello"); //true;
14 Boolean(" "); //true;
15 Boolean(""); //false (if zero length);
16
17 //Object to Boolean
18 Boolean(null); //false;
19 Boolean({ name: 'Jane' }); //true;
20 Boolean({}); //true;
21
22 //Symbol to Boolean
23 Boolean(Symbol('my-symbol')); //true;
24

```

## Откуда берётся **boolean**?

Из преобразования типов, явного (при помощи **Boolean()** или **!!**) или неявного (в условных операторах, циклах...).

# Оператор if-else – основной клиент boolean

```
2
3  if( some_boolean ){
4      //if some_boolean is true;
5  }else{
6      //if some_boolean is false;
7      //false branch is optional;
8  }
```

Оператор **if-else** в зависимости от переданного (**true** или **false**) значения выполняет один из двух блоков кода (**первый** или **второй**, соответственно), другой блок при этом не выполняется. Если значение переданное оператору **if** не является **boolean**'ом будет выполнено неявное преобразование. Ветка **else** не является обязательной.

# Откуда берётся boolean?

## Операторы сравнения

>	<	>=	<=	==	!=	===	!==
---	---	----	----	----	----	-----	-----

```
2  
3     var a = 8;  
4     var b = 7;  
5  
6     var c = a >= b;  
7  
8     console.log(c, typeof(c));  
9
```

Подробнее: <https://learn.javascript.ru/comparison>

# Откуда берётся boolean?

>	<	>=	<=	==	!=	===	!==
---	---	----	----	----	----	-----	-----

```
2
3   let a = 6;
4   let b = 500;
5
6   let c = "6";
7   let d = "500";
8
9   console.log(a > b);
10  console.log(a > d);
11  console.log(c > d);
12
```

?!?

Есть нюансы с типами...  
При сравнении разных  
типов происходит их  
преобразование к  
**number**

# Откуда берётся boolean?

>	<	>=	<=	==	!=	===	!==
---	---	----	----	----	----	-----	-----

```
2
3   let a = "Ivan";
4   let b = "Iven";
5
6   console.log(a > b);
7   console.log(a < b);
8   console.log(a == b);
9
```

?!?

Сравнение строк  
осуществляется  
посимвольно.

Выполняется сравнение  
кодов символов.

# Логические операторы

Когда нужны «сложные» условия

&&		
----	--	--

```
2
3      let a = 80;
4      let b = 500;
5
6      if( a > 1 && b < 1000 ){
7          //...
8      }else{
9          //...
10     }
11
```

Подробнее: <https://learn.javascript.ru/logical-ops>

# Логические операторы

## Таблицы истинности

<b>&amp;&amp;</b>	<b>False</b>	<b>True</b>
<b>False</b>	False	False
<b>True</b>	False	True

<b>  </b>	<b>False</b>	<b>True</b>
<b>False</b>	False	True
<b>True</b>	True	True

<b>!</b>	<b>False</b>	<b>True</b>
	True	False

Операторы логическое И (**&&**) и логическое ИЛИ (**||**) работают по такой схеме: Приводят левый операнд к **boolean**. Если по нему можно сделать выводы (будет выражение, в целом, верным или ложным), то возвращают левый операнд (в том типе в котором он и был). Если нет, то возвращают правый операнд (в том типе в котором он и был). Логические операторы **&&** и **||** могут не проверять правый операнд, если значение левого операнда уже достаточно для итогового результата выражения (*это важно если правый операнд - вызов функции*).

# «Многоэтажный» if-else-if

```
2
3   let t = 18; //temperature;
4
5   if( t < 0 ){ //from (-∞ to 0)
6       | //...very cold...
7   }else if(t < 14){ //from [0 to 14)
8       | //...cold...
9   }else if(t < 24){ //from [14 to 24)
10      | //...comfort...
11   }else if(t < 32){ //from [24 to 32)
12      | //...hot...
13   }else{ //from [32 to ∞)
14      | //...very hot...
15   }
```

Многоэтажных **if-else-if...**

Хорош для задач в которых больше чем два вариант развития событий, или когда значения нужно распределить по диапазонам.



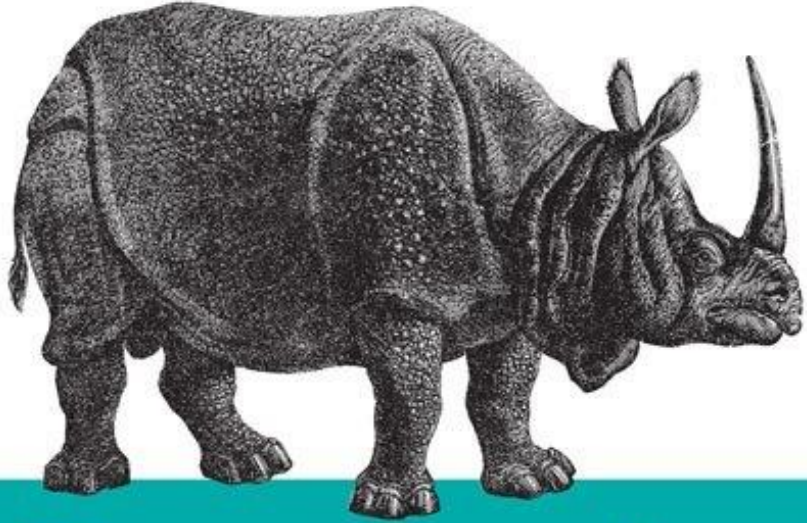
Немного практики

***Задача:*** Мы знаем день, месяц и год рождения человека. Мы также знаем сегодняшний день, месяц и год, необходимо рассчитать сколько человеку полных лет на сегодняшний день.

«Источники знаний»

Создание активных веб-страниц

6-е издание  
Включает ECMAScript 5 и HTML5



# JavaScript

*Подробное руководство*

 **О'REILLY®**

Дэвид Флэнаган

## Дэвид Флэнаган

### JavaScript Подробное руководство

# Современный учебник Javascript

Перед вами учебник по JavaScript, начиная с основ, включающий в себя много тонкостей и фишек JavaScript/DOM.

[смотреть на Github](#)

Поделиться:

[НАЙТИ](#)

## Содержание

Первые две части посвящены JavaScript и его использованию в браузере. Затем идут дополнительные циклы статей на разные темы.

<http://learn.javascript.ru/>

Ресурси для розробників, від розробників.

Веб-технології



Вивчення веб-розробки



Інструменти розробника



<https://developer.mozilla.org/uk/>

Узнайте  
к следующему занятию...

# Современный учебник Javascript

Перед вами учебник по JavaScript, начиная с основ, включающий в себя много тонкостей и фишек JavaScript/DOM.



смотреть на Github

Поделиться:



НАЙТИ

## Содержание

Первые две части посвящены JavaScript и его использованию в браузере. Затем идут дополнительные циклы статей на разные темы.

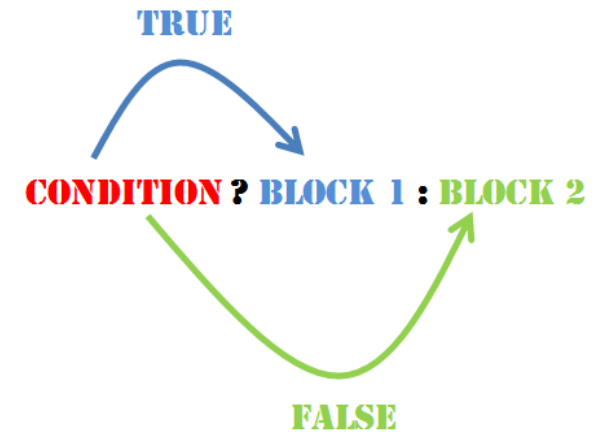
**Предварительные знания – лучший помощник** в обучении, поэтому к следующему занятию жду, что **пройдёте разделы 1.1-1.4, 2.1-2.12, 3.4, 4.1, 5.2-5.5 первой части справочника**

<http://learn.javascript.ru/>



# Что делает этот код?

```
22  
23   let age = +prompt('Enter your age (in years)');  
24  
25   let message = age >= 18 ? 'Allowed' : 'Denied';  
26  
27   console.log(message);  
28
```



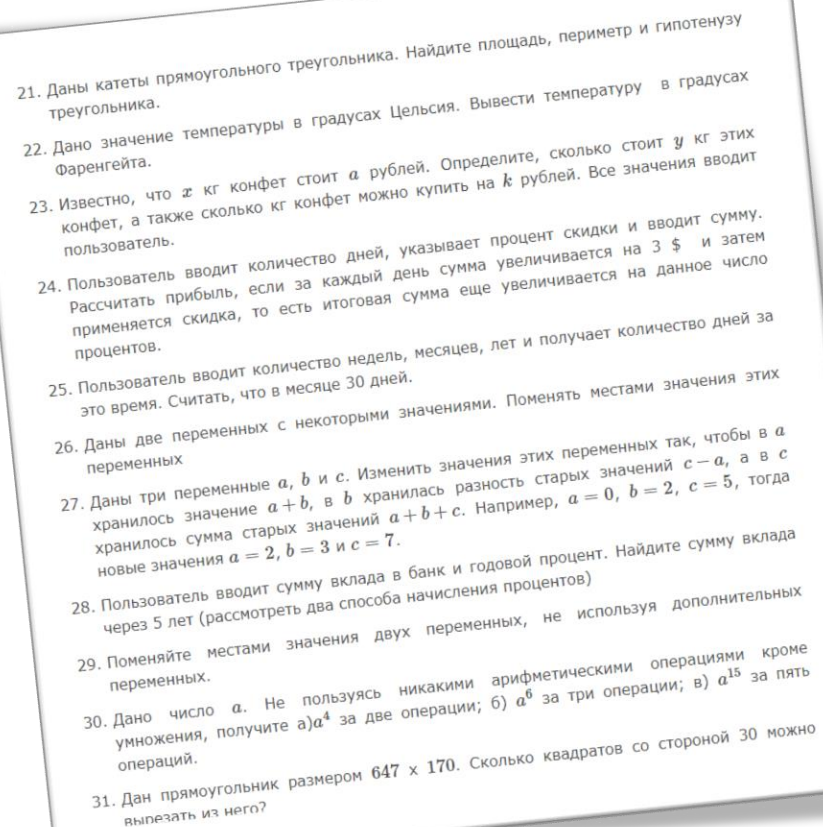
Разберитесь как работает приведенная конструкция (25 строка), чем она может быть полезна. По возможности применяйте её при решении домашнего задания.

Скучное но необходимое  
домашнее задание

# Домашнее задание

Первым делом необходимо **натренироваться применять базовые конструкции**. Поэтому напишите код который решит задачи (№7-№74) из раздела «Простейшая арифметика» и «Условный оператор и арифметика».

Решения этих задач **загружать на GitHub не нужно** (правильность их работы можно легко проверить с ответами, или калькулятором). Но если возникнут проблемы, не стесняйтесь задавать вопросы. Для этого **загрузите ваш код** (в котором возникли проблемы) **на GitHub**, и ссылку на него **с описанием проблемы** сбросьте в канал **#trouble** в Slack'е.

- 
21. Даны катеты прямоугольного треугольника. Найдите площадь, периметр и гипотенузу треугольника.
  22. Дано значение температуры в градусах Цельсия. Вывести температуру в градусах Фаренгейта.
  23. Известно, что  $x$  кг конфет стоит  $a$  рублей. Определите, сколько стоит  $y$  кг этих конфет, а также сколько кг конфет можно купить на  $k$  рублей. Все значения вводит пользователь.
  24. Пользователь вводит количество дней, указывает процент скидки и вводит сумму. Рассчитать прибыль, если за каждый день сумма увеличивается на 3 \$ и затем применяется скидка, то есть итоговая сумма еще увеличивается на данное число процентов.
  25. Пользователь вводит количество недель, месяцев, лет и получает количество дней за это время. Считать, что в месяце 30 дней.
  26. Даны две переменных с некоторыми значениями. Поменять местами значения этих переменных
  27. Даны три переменные  $a$ ,  $b$  и  $c$ . Изменить значения этих переменных так, чтобы в  $a$  хранилось значение  $a + b$ , в  $b$  хранилась разность старых значений  $c - a$ , а в  $c$  хранилось сумма старых значений  $a + b + c$ . Например,  $a = 0$ ,  $b = 2$ ,  $c = 5$ , тогда новые значения  $a = 2$ ,  $b = 3$  и  $c = 7$ .
  28. Пользователь вводит сумму вклада в банк и годовой процент. Найдите сумму вклада через 5 лет (рассмотреть два способа начисления процентов)
  29. Поменяйте местами значения двух переменных, не используя дополнительных переменных.
  30. Дано число  $a$ . Не пользуясь никакими арифметическими операциями кроме умножения, получите а)  $a^4$  за две операции; б)  $a^6$  за три операции; в)  $a^{15}$  за пять операций.
  31. Дан прямоугольник размером  $647 \times 170$ . Сколько квадратов со стороной 30 можно выпилить из него?

<http://www.itmathrepetitor.ru/prog/zadachi-na-vychisleniya/>