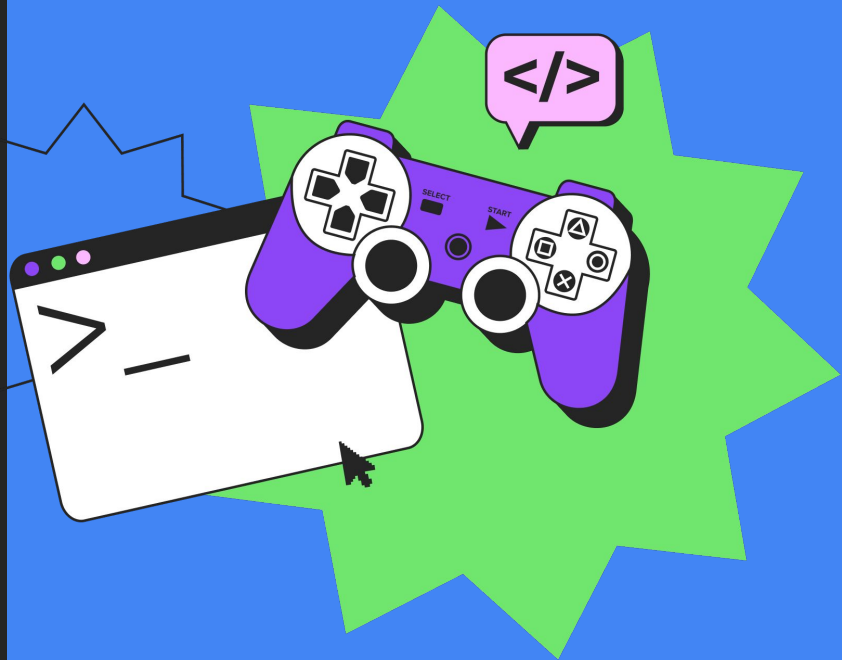


Знакомство с JS

Week 1
JavaScript





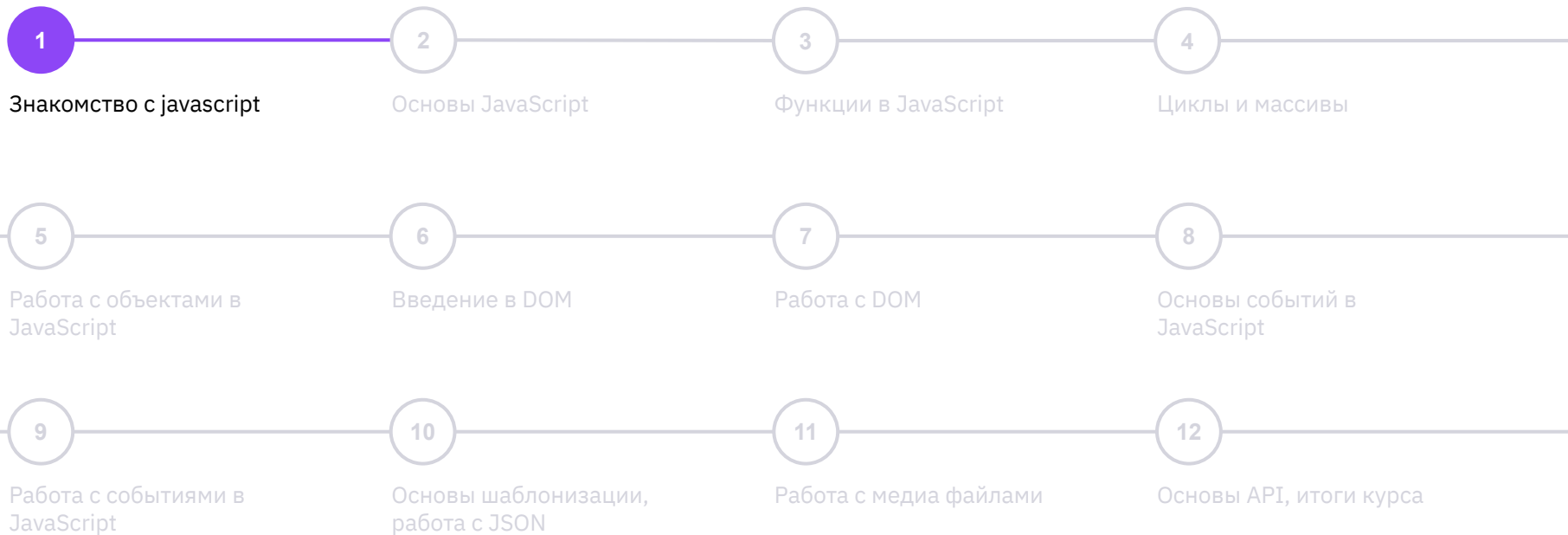
Кадочников Алексей

Frontend-разработчик

- 💥 Веб-разработчик со стажем более 9 лет
- 💥 Преподаватель GeekBrains с 2015 года
- 💥 Автор курсов по Frontend на портале Geekbrains
- 💥 Работал в таких компаниях, как VK и Wizard-C







План курса





Что будет на уроке сегодня

-  Знакомство с JavaScript(JS)
-  Стандарты JavaScript
-  Переменные и область видимости
-  Узнаем, какие типы данных есть в JS.



Что мы уже знаем

HTML

Язык разметки
документа

Контент сайта

CSS

Каскадная таблица
стилей

Стили для контента

JavaScript

?



JavaScript

Javascript – язык программирования. С помощью которого мы можем добавить любой интерактив на страницу, любое взаимодействие с пользователем.

Проще говоря, это язык программирования который может реализовать любой функционал на вашем сайте в зависимости от того, что за действие вы выполняете на сайте



Программы, содержащие движок для выполнения JavaScript.



Веб-браузер



**Среды выполнения Node.js,
Deno, Electron**

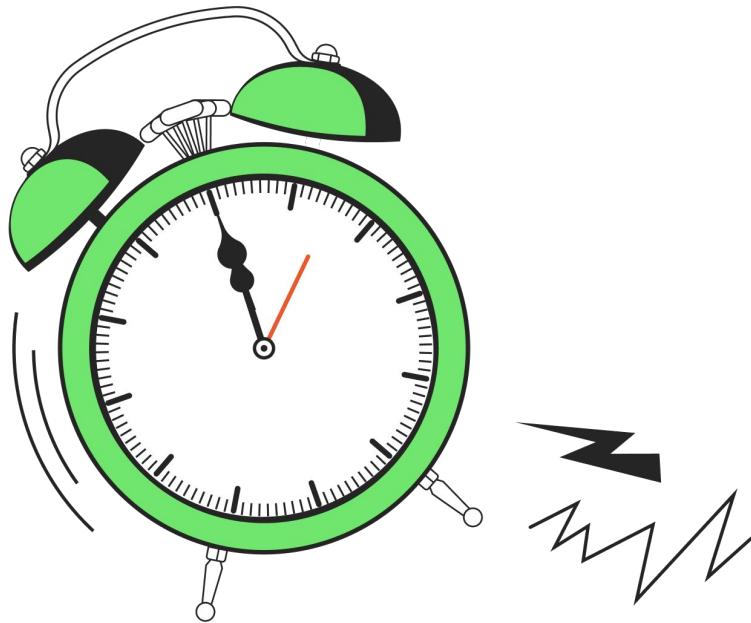


**Любое другое ПО, которое
можно установить, например,
на робот-пылесос, или
визуальная среда управления
ракетой Falcon 9**

Среда выполнения

Среда выполнения — это совокупность движка для исполнения JavaScript, программы, в которой этот движок работает, и операционной системы.

- Браузер (V8) + Windows 10
- Node.js + Ubuntu
- Electron (V8) + Windows





Стандарты

Первоначально язык зародился без стандартов, но позже был передан в организацию, занимающуюся разработкой стандартов, **Ecma International**. Именно там были выработаны первые и все последующие стандарты языка, которые включают все тонкости того, как должен быть реализован интерпретатор, какой функционал будет у языка, какие ключевые слова и поведение.



Переменные и область видимости



Переменные можно создать с помощью ключевых слов:



let

Новый способ объявления переменных



const

Если переменная объявлена через const, её значение изменить нельзя



var

Устаревший способ объявления переменных, который имеет пару недостатков (которые мы сейчас разберём)



Область видимости

Участок кода, который может обратиться к переменной и получить из неё значение.



Область видимости let и var

```
1 if (3 > 1) {  
2     let a = 5;  
3 }  
4  
5 console.log(a); // error a is not defined  
6  
7 if (3 > 1) {  
8     var b = 5;  
9 }  
10  
11 console.log(b); // 5
```



let

```
1 let result = 8;  
2 console.log(result); // 8  
3  
4 let updatedResult = result + 2;  
5 console.log(updatedResult); // 10
```



const

Если переменная объявлена через `const`, её значение изменить нельзя. Правильнее называть такие значения константами.

```
1 const GAP_SIZE = 5;  
2 GAP_SIZE = 3; // TypeError: Assignment to constant variable.
```



Имена переменных

1. Имена переменных чувствительны к регистру.
2. Должны быть легко читаемы.
3. Должны быть легко отличимы.
4. Должны рассказывать своим именем, что в них хранится.

```
1 let result = 5;  
2 let Result = 3;  
3 console.log(result); // 5  
4 console.log(Result); // 3
```




Типы данных

- строка (string);
- число (number);
- булево значение (boolean);
- undefined;
- объект (object);
- null;
- symbol;
- большое число (BigInt).





Как определить тип значения

Чтобы определить, к какому типу данных относится значение, мы можем применить оператор **typeof**

```
1 console.log(typeof 2) // "number"  
2 console.log(typeof "2") // "string"
```

Теперь ваша очередь!



Отличия типов

По-разному сравниваются

**Имеют разные встроенные
методы для работы со
значениями**



Строка

Текст, заключённый в кавычки или обратные кавычки (backticks `). Двойные и одинарные кавычки не имеют различий. Обратные кавычки позволяют передавать в строку переменную или вызов функции. Такие строки называются строковыми шаблонами.

```
1 let string = "Hello";  
2 string = 'Hello';  
3  
4 let result = 8;  
5 let literal = `Результат: ${result}`; // Результат: 8  
6 let emptyString = "";
```



Число

Тип данных «число» или `number` — это число, целое или десятичное. Также сюда относятся числа в двоичной, восьмеричной и шестнадцатеричной системах счисления. В JavaScript для типа `number` установлено ограничение на максимальные значения от $-(2^{53}-1)$ до $(2^{53}-1)$.

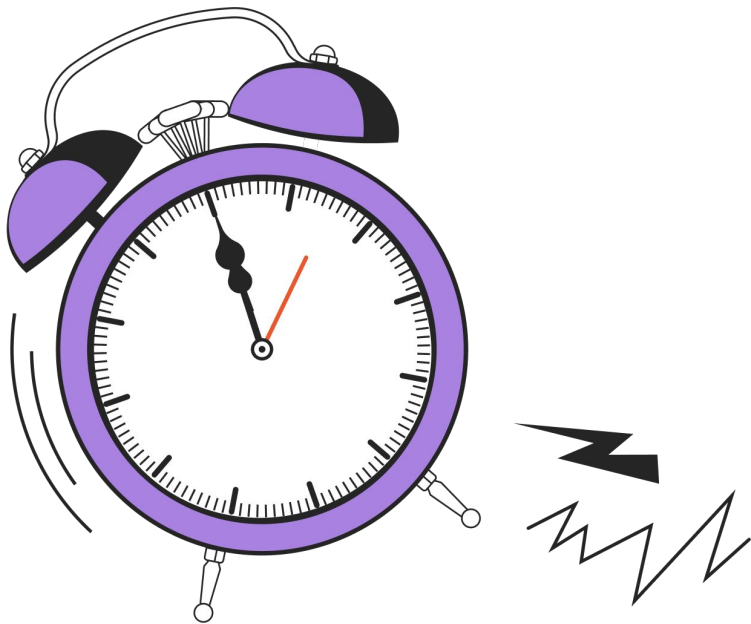




Специальные числовые значения

- Nan
- Infinity, - Infinity

Используй слайд, когда акцент нужно сделать сначала на картинке/фото, а текст поясняет картинку/фото.





NaN

NaN возникает при ошибке вычисления и «поглощает» все другие арифметические операции, которые мы пытаемся производить над переменной. Возникает, когда движок не может получить результат математической операции, например когда один из операторов не является числом.

```
1 let numberGradeValue = 5;  
2 let stringGradeValue = "A";  
3 const result = numberGradeValue - stringGradeValue; // NaN
```



Infinity, - Infinity

Infinity — бесконечность. Часто получается в результате деления на 0. При попытке деления на 0 программа не выдает ошибок, а просто присваивает результату значение **Infinity**. Как и с **NaN**, с этим значением арифметически ничего не сделать. Его очень удобно использовать для задания бесконечных значений вместо очень больших чисел.

-Infinity — это отрицательная бесконечность. Например, от деления отрицательного числа на 0.



Булево значение

Булево (логическое) значение может быть только правдивым (true) или ложным (false). Оно применяется в тех местах, где нам нужно именно логическое значение, например «включено/выключено», «содержит/не содержит» и т. п.

```
1 let isPassExam = true;  
2 if (isPassExam) console.log('Поздравляем, Вы сдали экзамен!'); // Поздравляем, вы сдали экзамен!
```



Undefined

Undefined — это специализированный тип данных, который имеет одно-единственное значение `undefined` и автоматически присваивается вновь созданному, но не инициализированному другим значением переменным.

```
1 let isPassExam;  
2 console.log(isPassExam); // undefined, т.к. переменной не задано значение
```



Объект

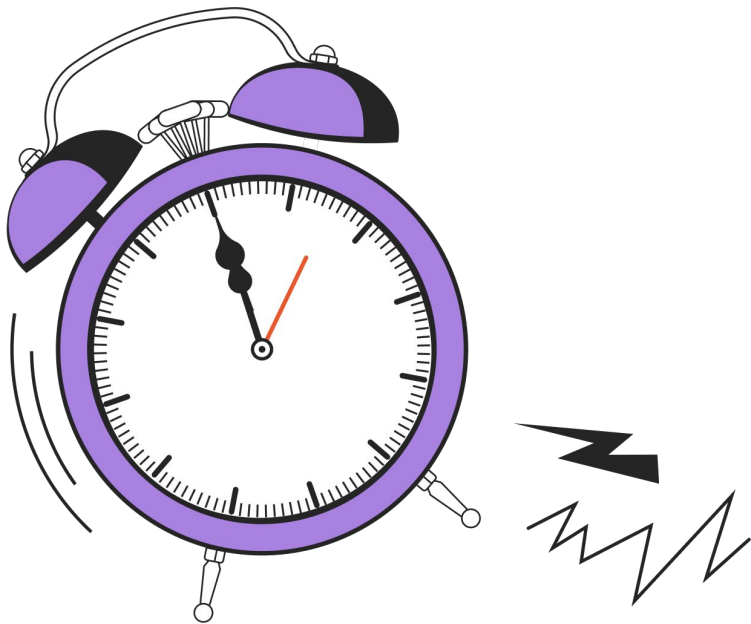
Для хранения набора данных используется переменная типа объект. Обращаться к полям (свойствам) объекта можно через знак “.” или указывая имя ключа (свойства) в квадратных скобках.

```
1 let student = {  
2   firstName: "Ivan",  
3   lastName: "Petrov",  
4   age: 33,  
5   faculty: "Information Technologies"  
6 };  
7  
8 console.log(`${student.firstName} ${student.lastName} is ${student.age} years old`); // Ivan Petrov  
   is 33 years old.
```



Два специальных объекта:

- Array (массив)
- Function (функция)





Array

Массив — упорядоченный список элементов. У каждого элемента свой индекс. Индексы начинаются с 0. Также массив имеет свойство длина (length). Обращение к элементам массива идёт по индексу, указанному в квадратных скобках после имени массива.

```
1 let listOfBooks = ['Стив Макконел "Совершенный код"', 'Роберт Мартин "Чистый код"'];  
2 console.log(listOfBooks[0]); // Стив Макконнелл "Совершенный код"  
3 console.log(listOfBooks.length); // 2 - длина массива (2 элемента).
```



Function

Функция — это объект, имеет свои свойства и методы. Она может быть сохранена в переменную, передана как аргумент в другую функцию.

```
1 const add2= function(x) {  
2   return x + 2;  
3 }  
4  
5 console.log(add2.toString()); // function(x) { return x + 2; }  
6 console.log(add2(10)); // 12 - выводится результат выполнения функции.
```



Null

Null — это специальное значение, которое используется с переменными объектного типа для указания, что в переменной нет объекта. Null — примитивный тип. В операциях сравнения и булевой логике он выступает как ложное (false) значение.



Symbol

Новый тип данных, чтобы создавать уникальные ключи для свойств объектов или каких-либо других целей. Для создания символа необходимо вызвать функцию (конструктор) этого типа `Symbol()`. Также в качестве аргумента в эту функцию можно передать строку, что позволит создать символ на основе вашей строки.

```
1 const uniqKey= Symbol();
2 console.log(uniqKey.toString()); // Symbol()
3 const uniqKey2 = Symbol('test');
4 const uniqKey3 = Symbol('test');
5 console.log(uniqKey2.toString()); // Symbol('test')
6 console.log(uniqKey2 == uniqKey3) // false - символы всегда создаются уникальными.
```




BigInt





При сложных расчётах или при использовании меток времени с микросекундами можно использовать супербольшие значения чисел (только целые числа).


Такие числа записываются как обычное целое число, только в конце добавляется литера n.

```
1 let bigNumber = 63374851375010000n;  
2 console.log(bigNumber); // 63374851375010000n
```



Итоги урока

-  Познакомились с JavaScript(JS)
-  Разобрали стандарты JavaScript
-  Рассмотрели переменные и область видимости
-  Узнали, какие типы данных есть в JS.

Спасибо 
за внимание

