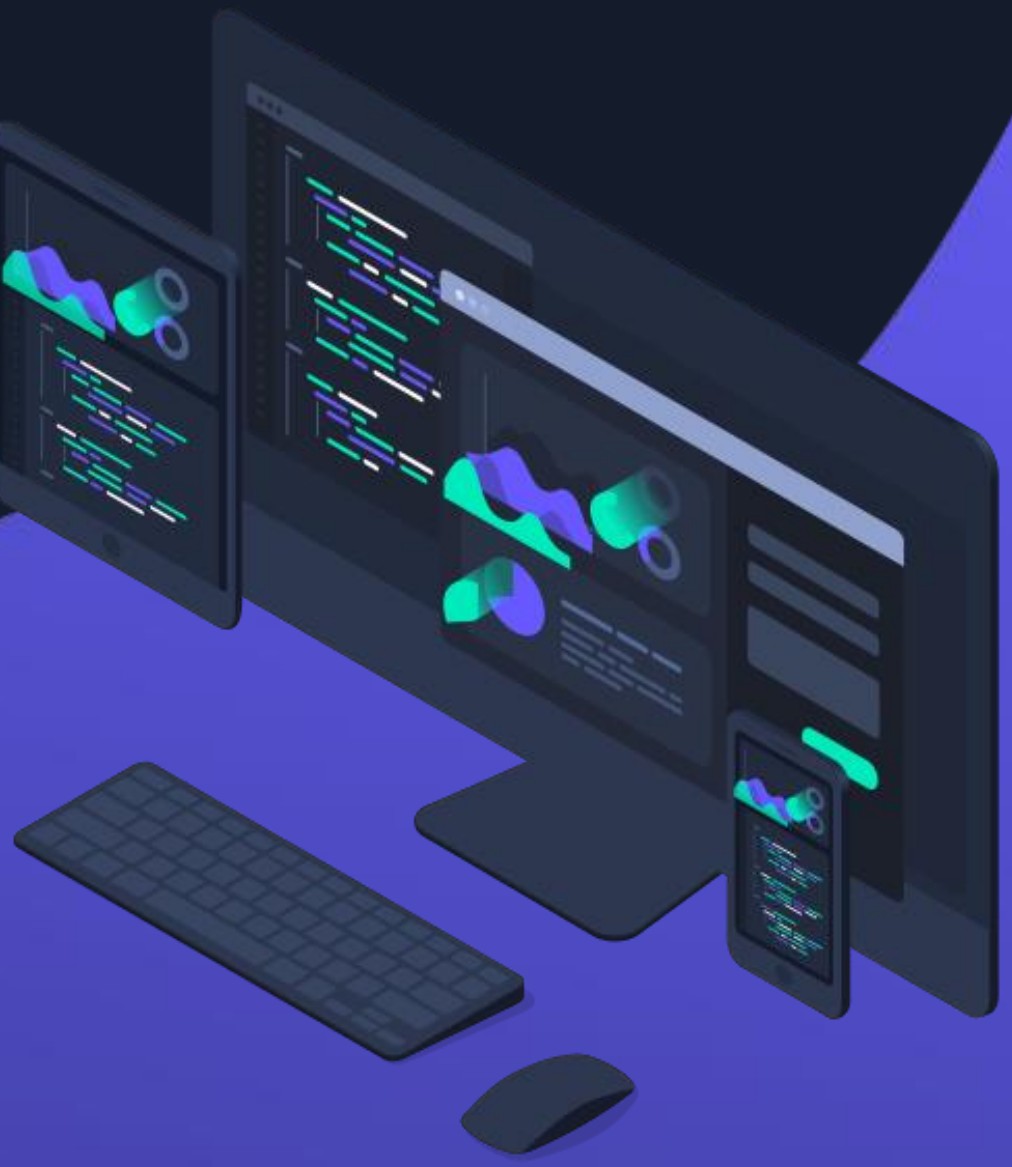
ITEA

ONLINE



I T E A
ONLINE

На сегодняшнем занятии:

1. var, let, const
2. Строгий режим «use strict»
3. Отладка в браузере Chrome
4. Советы по стилю кода
5. Массивы
6. Создание массива
7. Одномерные и многомерные массивы
8. Длина массива
9. Методы для работы с массивами

ITEA ONLINE

var, let, const

`var` – область видимости переменных `var` ограничивается либо функцией, либо, если переменная глобальная, то скриптом. Такие переменные доступны за пределами блока. Объявление переменной обрабатывается в начале выполнения функции («всплывает»), однако присвоение значения всегда происходит в той строке кода, где оно указано.

```
if (true) {
   var a = "test";
}
console.log("a: " + a); // "a: test"
```

```
function test() {
   var a = "test";
}
console.log("a: " + a); // ошибка
```

```
function sayHi() {
  console.log(phrase);    // undefined
  var phrase = "Привет";
}
sayHi();
```

```
function sayHi() {
  var phrase; // объявление переменной срабатывает вначале...
  console.log(phrase); // undefined
  phrase = "Привет"; // ...присвоение - в момент, когда
исполнится данная строка кода.
}
sayHi();
```

ITEA ONLINE

var, let, const

`let` – область видимости ограничивается блоком где она объявлена. Данный способ объявления переменной считается наиболее актуальным в текущий момент.

```
if (true) {
   let a = "test";
}
console.log("a: " + a); // ошибка, переменная «a» необъявлена (та, что в if'е - недоступна)
```

```
function test() {
   let a = "test";
}
console.log("a: " + a); // ошибка, переменная «a» необъявлена (та, что в функции - недоступна)
```

```
function sayHi() {
  console.log(phrase);    // ошибка, попытка использование переменной, которая не объявлена (нет всплытия)
  let phrase = "Привет";
}
sayHi();
```

ITEA ONLINE

var, let, const

`const` – используется чтобы объявить константную, то есть, неизменяемую переменную. Переменные, объявленные с помощью `const`, называются «константами». Их нельзя изменить. Попытка сделать это приведёт к ошибке

```
const myBirthday = '18.04.1982';
myBirthday = '01.01.2001'; // ошибка, константу нельзя перезаписать
```

ITEA ONLINE

Строгий режим «use strict»

Режим strict (строгий режим), введенный в ECMAScript 5, позволяет использовать более строгий вариант JavaScript.

```
// Синтаксис переключения в строгий режим всего скрипта
"use strict";
var v = "Привет! Я скрипт в строгом режиме!";

"use strict";
        // Предполагая, что не существует глобальной переменной
t = 17;  // t, эта строка выбросит ReferenceError
        // из-за опечатки в имени переменной
```

ITEA ONLINE

Строгий режим «use strict»

```
"use strict";

// Присваивание значения глобальной переменной, защищенной от записи
var undefined = 5; // выдаст TypeError
var Infinity = 5; // выдаст TypeError
```

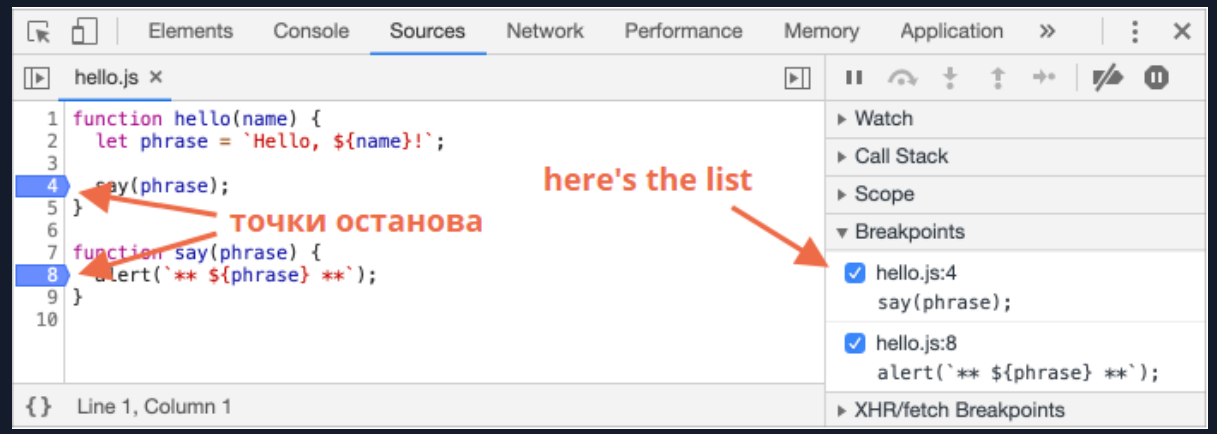
Строгий режим требует, чтобы все свойства, перечисленные в сериализованном объекте, встречались только один раз. В обычном коде имена свойств могут дублироваться, а значение свойства определяется последним объявлением

```
"use strict";
var o = { p: 1, p: 2 }; // !!! синтаксическая ошибка
```

Отладка в браузере Chrome

Отладка – это процесс поиска и исправления ошибок в скрипте. Все современные браузеры и большинство других сред разработки поддерживают инструменты для отладки – специальный графический интерфейс, который сильно упрощает отладку.

Точки останова (breakpoints)

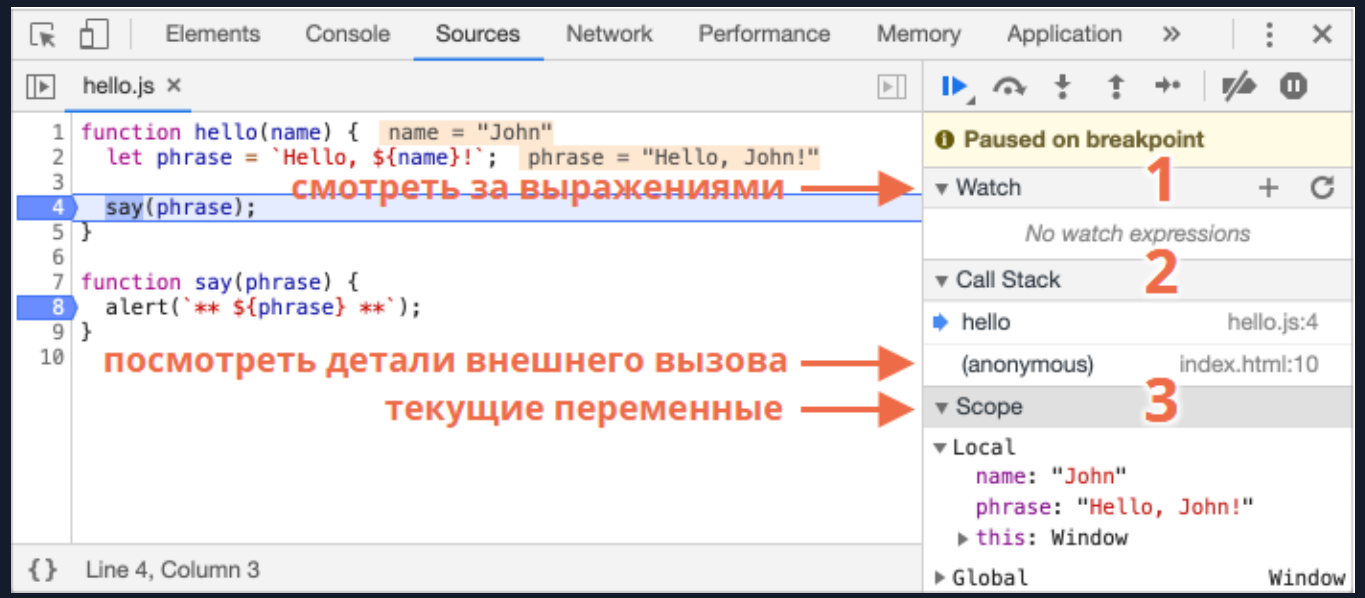


Точка останова – это участок кода, где отладчик автоматически приостановит исполнение JavaScript.

Пока исполнение поставлено «на паузу», мы можем просмотреть текущие значения переменных, выполнить команды в консоли.

ITEA ONLINE

Отладка в браузере Chrome



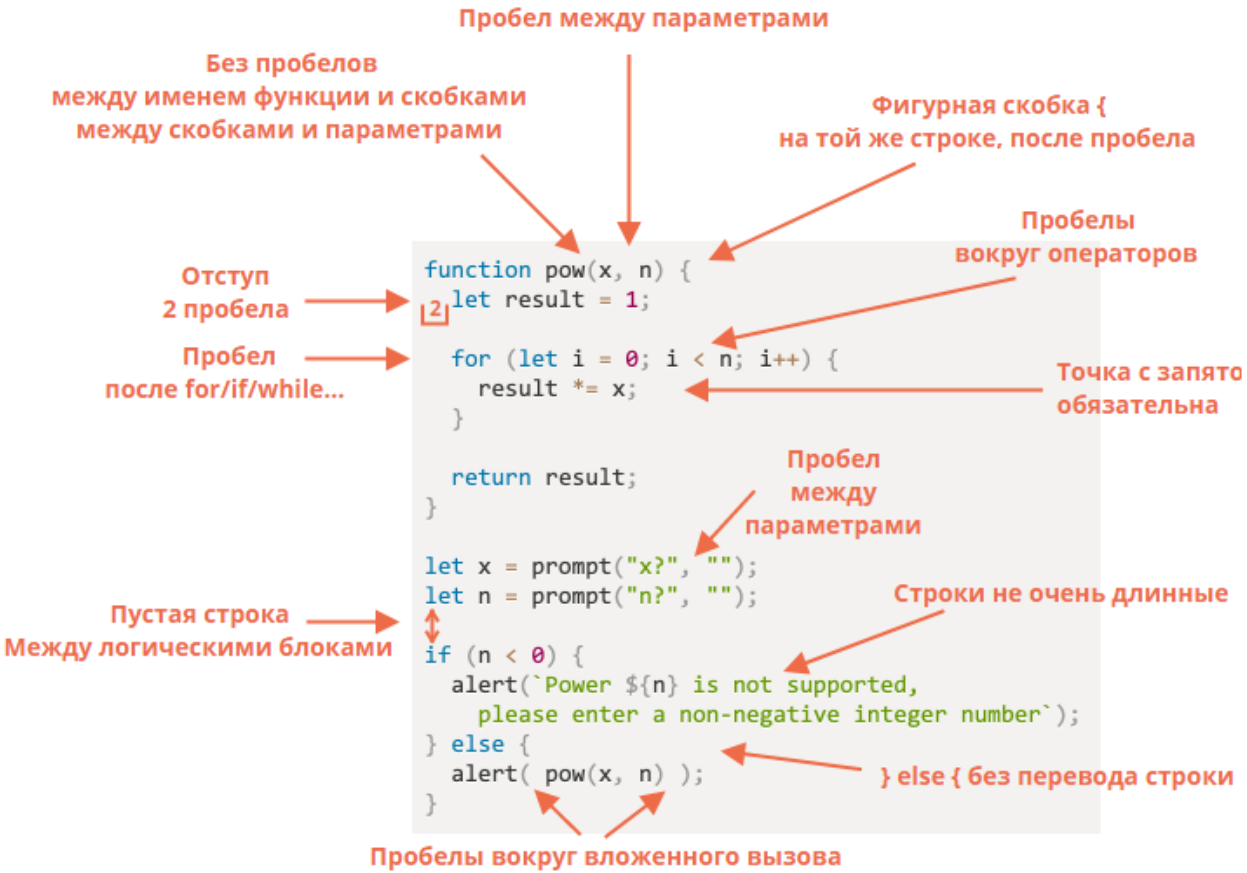
Watch показывает текущие значения выражений.

Call Stack показывает последовательность вызовов функций.

Scope показывает текущие переменные.

I T E A
ONLINE

Советы по стилю кода



⚠ Ни одно правило не является жёстко обязательным

ITEA ONLINE

Массивы

Массив в JavaScript — набор разнотипных компонентов (элементов), доступ к которым осуществляется по индексу (индексам).

Индекс — элемент перечислимого множества, который указывает на конкретный элемент массива, обычно является неотрицательным целым числом. Через индекс происходит обращение к конкретному элементу массива.


```
…
let fruits = ["Яблоко", "Апельсин", "Слива"];
console.log(fruits[1]); // Апельсин
…
```

Создание массива

Массивы в JavaScript можно создать двумя способами:

- через блок-инициализатор [];
- через конструктор Array.

Второй способ используется реже, так как первый короче.

Однако, в тех случаях, когда нужно задать массив на определенное количество элементов с пустыми значениями, первый способ не подходит.

```
let fruits = ["Яблоко", "Апельсин"];
```

Имя массива

Элементы массива

```
let array = new Array(123, true, "yo");
```

Обращение к элементам массива:

- `fruits[0];` `// Яблоко`
- `array[2];` `// yo`
- `fruits[3];` `// undefined`

ITEA ONLINE

Одномерные и многомерные массивы

Массивы могут содержать элементы, которые тоже являются массивами. Это можно использовать для создания многомерных массивов.

Одномерный

```
let a = [1, 2, 3];

alert(a[1]); // 2,
центральный элемент
```

Многомерный

```
let matrix =
[
  [1, 2, 3], // 0
  [4, 5, 6], // 1
  [7, 8, 9]  // 2
];

alert(matrix[2][0]); // 7,
центральный элемент
```

ITEA ONLINE

Длина массива

Свойство length автоматически обновляется при изменении массива. Если быть точными, это не количество элементов массива, а наибольший цифровой индекс плюс один.

```
var a = new Array();        // a.length => 0      // Пустой массив
var a = new Array(10);      // a.length => 10     // Пустые элементы 0-9
var a = new Array(1,2,3);   // a.length => 3      // Элементы с 0 по 2 индекс

var a = [4, 5];             // a.length => 2      // Элементы с 0 по 1 индекс
a[5] = -1;                  // a.length => 6      // Элементы на 0, 1 и 5 индексе
a[49] = 0;                  // a.length => 50     // Элементы на 0, 1, 5 и 50
```

ITEA ONLINE

Длина массива

Свойство length можно как считывать так и переопределять. Если изменить длину массива в меньшую сторону – то будут потеряны значения, которые не попадут в «новую» длину массива.

```
var a = [4, 5, 3, 10, 11]; // a.length => 5      // Элементы с 0 по 4 индекс
a.length = 3;              // a.length => 3      // Изменяем длину массива
console.log(a[3]);         // undefined          // Были потеряны все значения
                                                 // после 2го индекса
a.length = 5;              // undefined          // Вернули длину массива к
                                                 // изначальной, но это не вернет
                                                 // потерянные значения
console.log(a[4]);         // undefined          // Значение потеряно и в данный
                                                 // момент неопределено
```

I T E A
ONLINE

Методы для работы с массивами

- `join()` — преобразует элементы массива в строки и объединяет их.
- `reverse()` — меняет порядок следования элементов на противоположный.
- `sort()` — сортировка элементов массива в алфавитном порядке.
- `concat()` — возвращает новый массив, содержащий элементы исходного массива, дополненного параметрами метода.
- `slice()` — возвращает фрагмент, или подмассив, указанного массива.
- `splice()` — универсальный метод для вставки/удаления элементов массива.
- `push()` — добавляет элементы в конец массива и возвращает его новую длину.
- `pop()` — удаляет последний элемент из массива и возвращает удаленное значение.
- `unshift()` — добавляет элементы в начало массива и возвращает его новую длину.
- `shift()` — удаляет первый элемент из массива и возвращает удаленное значение.

ITEA ONLINE

Q&A

ITEA

ONLINE