

JAVASCRIPT FUNDAMENTALS

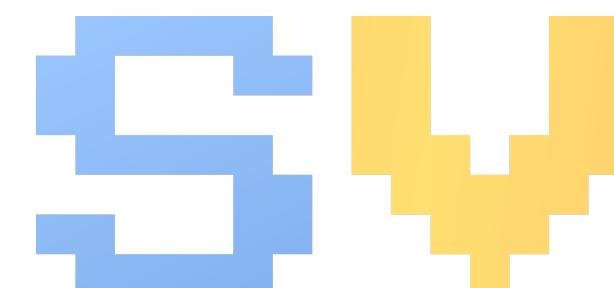
PART 2



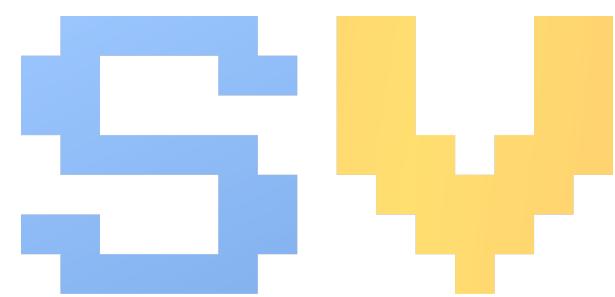
SV SILICON VALLEY

ЧТО МЫ ПРОЙДЕМ?

- Activating Strict Mode
- Functions
- Function Declarations vs. Expressions
- Arrow Functions
- Functions Calling Other Functions
- Introduction to Arrays
- Basic Array Operations (Methods)
- Introduction to Objects
- Dot vs. Bracket Notation
- Object Methods
- Iteration: The for Loop
- Looping Arrays, Breaking and Continuing
- Looping Backwards and Loops in Loops
- The while Loop



ACTIVATING STRICT MODE



FUNCTIONS

Функция - это блок кода, который может быть вызван из другого места в программе. Функция может принимать некоторые аргументы (также называемые параметрами) и возвращать результат.

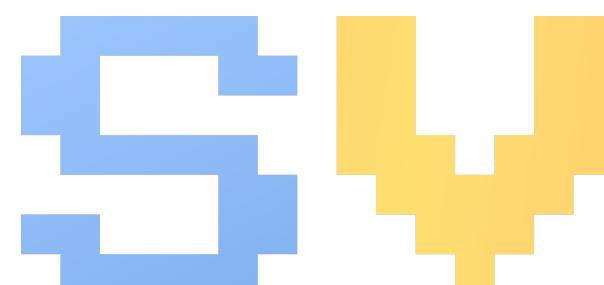
```
function myFunction(arg1, arg2) {  
    // тело функции  
    return 'Hello, World!';  
}
```

Аргумент – это значение, которое передаётся функции при её вызове.

Параметр – это переменная, указанная в круглых скобках в объявлении функции.

Примитивные типы данных, такие как числа, строки и булевые значения, всегда передаются по значению.

Объекты, массивы и функции всегда передаются по ссылке.



FUNCTION DECLARATIONS VS. EXPRESSIONS

Функциональное объявление используется для создания функции с именем.

```
function myFunction() {  
    // тело функции  
}
```

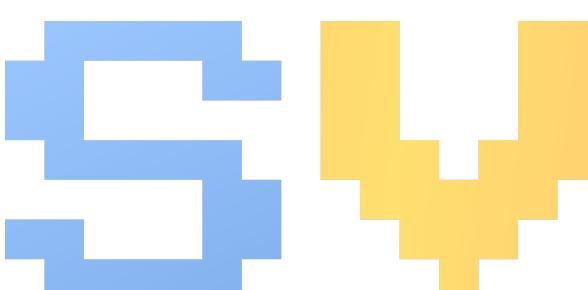
Функциональное объявление позволяет создавать функции, которые могут быть вызваны до того, как они определены. Это называется "холодным" (hoisted) объявлением функции.

Функциональное выражение может быть записано так:

```
const myFunction = function() {  
    // тело функции  
    return 'Hello, World!';  
}
```

Функциональное выражение не может быть "холодным" (hoisted), то есть оно должно быть определено до того, как будет вызвано.

Анонимная функция - это функция без имени.



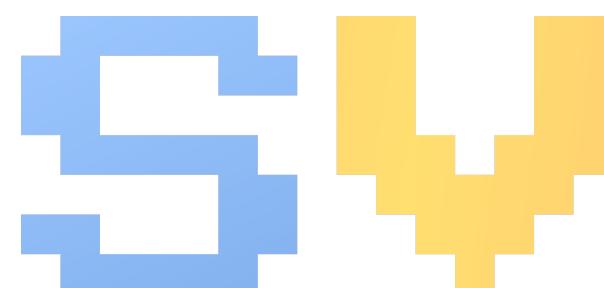
ARROW FUNCTIONS

Функция-стрелка (arrow function) - это сокращенная форма функционального выражения, которая использует символ "стрелка" (`=>`) в качестве связи между аргументами и телом функции.

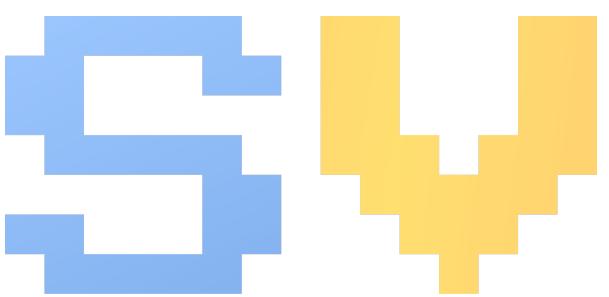
```
const myFunction = (arg1, arg2) => {  
    // тело функции  
    return 'Hello, World!';  
}
```

```
const myFunction = (arg1, arg2) => 'Hello, World!';
```

```
const myFunction = () => 'Hello, World!';
```



FUNCTIONS CALLING OTHER FUNCTIONS

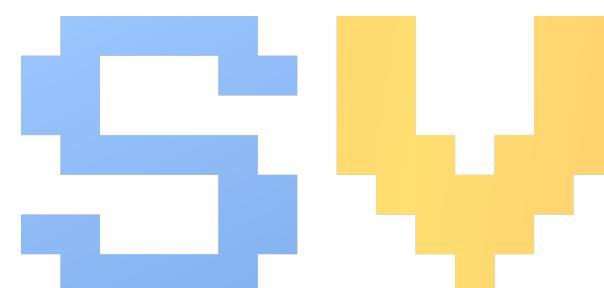
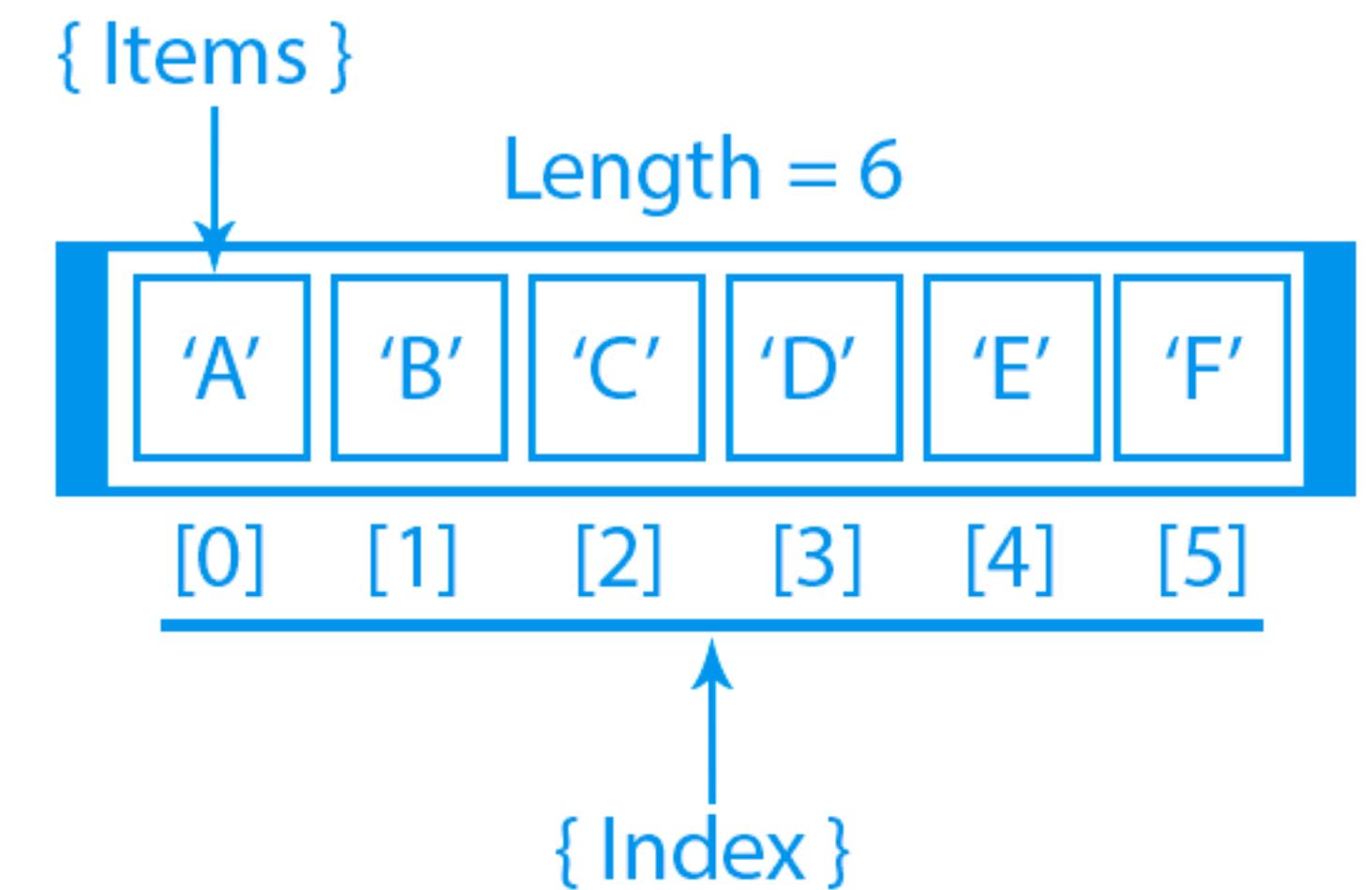


INTRODUCTION TO ARRAYS

В JavaScript массив (array) - это объект, который используется для хранения набора значений в одной переменной. Массивы могут хранить значения любого типа, в том числе и другие массивы.

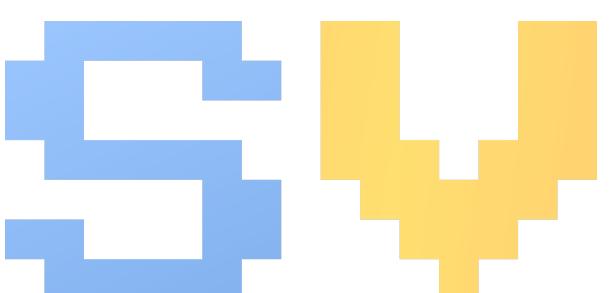
```
const numbers = [1, 2, 3, 4, 5];
const names = ['John', 'Jane', 'Mike'];
const mixed = [1, 'John', true, [2, 3]];

const numbers = [1, 2, 3, 4, 5];
console.log(numbers[0]); // 1
```



BASIC ARRAY OPERATIONS (METHODS)

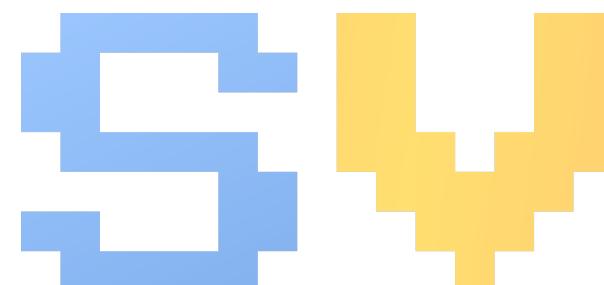
- push – добавляет элементы в конец.
- pop – извлекает элемент из конца.
- splice - этот метод принимает два аргумента - индекс элемента, который нужно удалить, и количество удаляемых элементов.
- filter - создает новый массив, который содержит только те элементы исходного массива, для которых указанная функция вернула true
- reduce - выполняет функцию, аккумулирующую обработку каждого элемента массива с возвращением одного результата
- sort - сортирует элементы массива в порядке возрастания
- reverse - разворачивает элементы массива в обратном порядке



INTRODUCTION TO OBJECTS

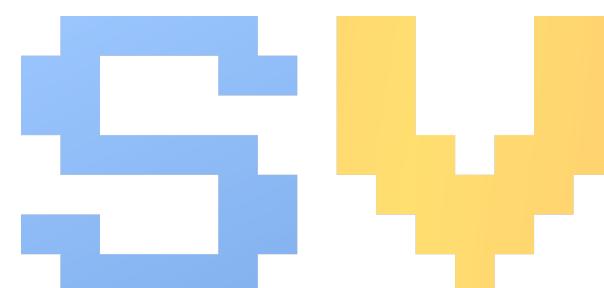
В JavaScript объекты представляют собой набор свойств, каждое из которых соответствует значению. Свойства объекта могут хранить любые типы данных, в том числе другие объекты и массивы.

```
const person = {  
    name: 'John',  
    age: 30,  
    hobbies: ['music', 'movies'],  
    address: {  
        street: 'Main Street',  
        city: 'New York',  
        state: 'NY'  
    }  
};  
  
console.log(person.name); // John  
console.log(person['name']); // John
```

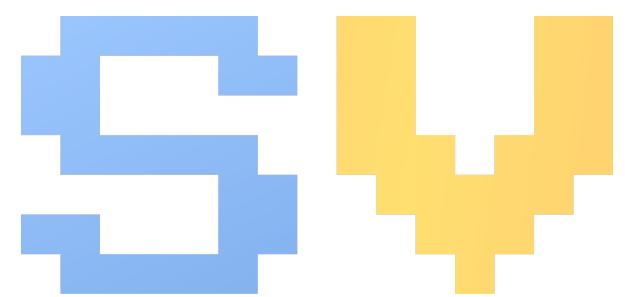


ОБЪЕСТ МЕТОДЫ

- `Object.assign(target, ...sources)` - копирует свойства из одного или нескольких объектов в целевой объект.
Возвращает целевой объект.
- `Object.create(prototype, properties)` - создает новый объект с указанным прототипом и свойствами.
- `Object.keys(obj)` – возвращает массив ключей.
- `Object.values(obj)` – возвращает массив значений.
- `Object.entries(obj)` - возвращает массив пар [ключ, значение].
- `Object.freeze(obj)` - замораживает объект, не позволяя добавлять, удалять или изменять его свойства.



CONST ARRAY AND OBJECTS

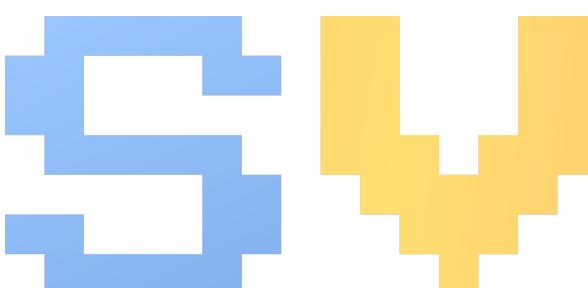


ITERATION: THE FOR LOOP

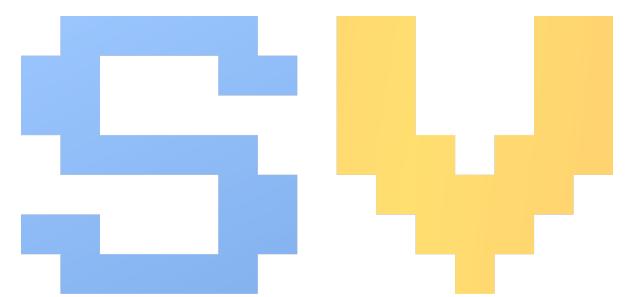
Цикл for - это один из способов итерации (перебора) в JavaScript. Он позволяет выполнять одну и ту же операцию несколько раз.

```
for (initialization; condition; final-expression) {  
    statement  
}
```

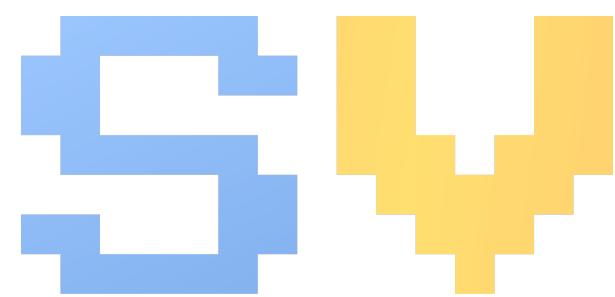
- initialization - выражение, которое выполняется один раз перед началом цикла. Обычно используется для инициализации счетчика.
- condition - логическое выражение, которое выполняется перед каждой итерацией цикла. Если оно возвращает true, то тело цикла выполняется. Если возвращает false, то цикл завершается.
- final-expression - выражение, которое выполняется после каждой итерации цикла. Обычно используется для изменения счетчика.
- statement - операторы, которые выполняются в теле цикла.



LOOPING ARRAYS, BREAKING AND CONTINUING



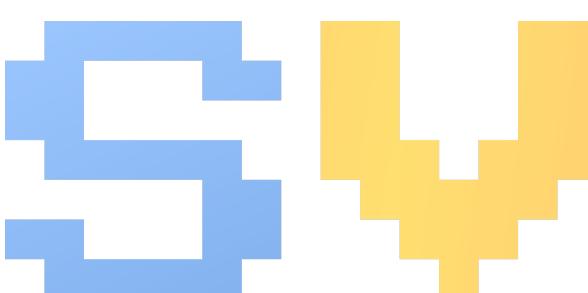
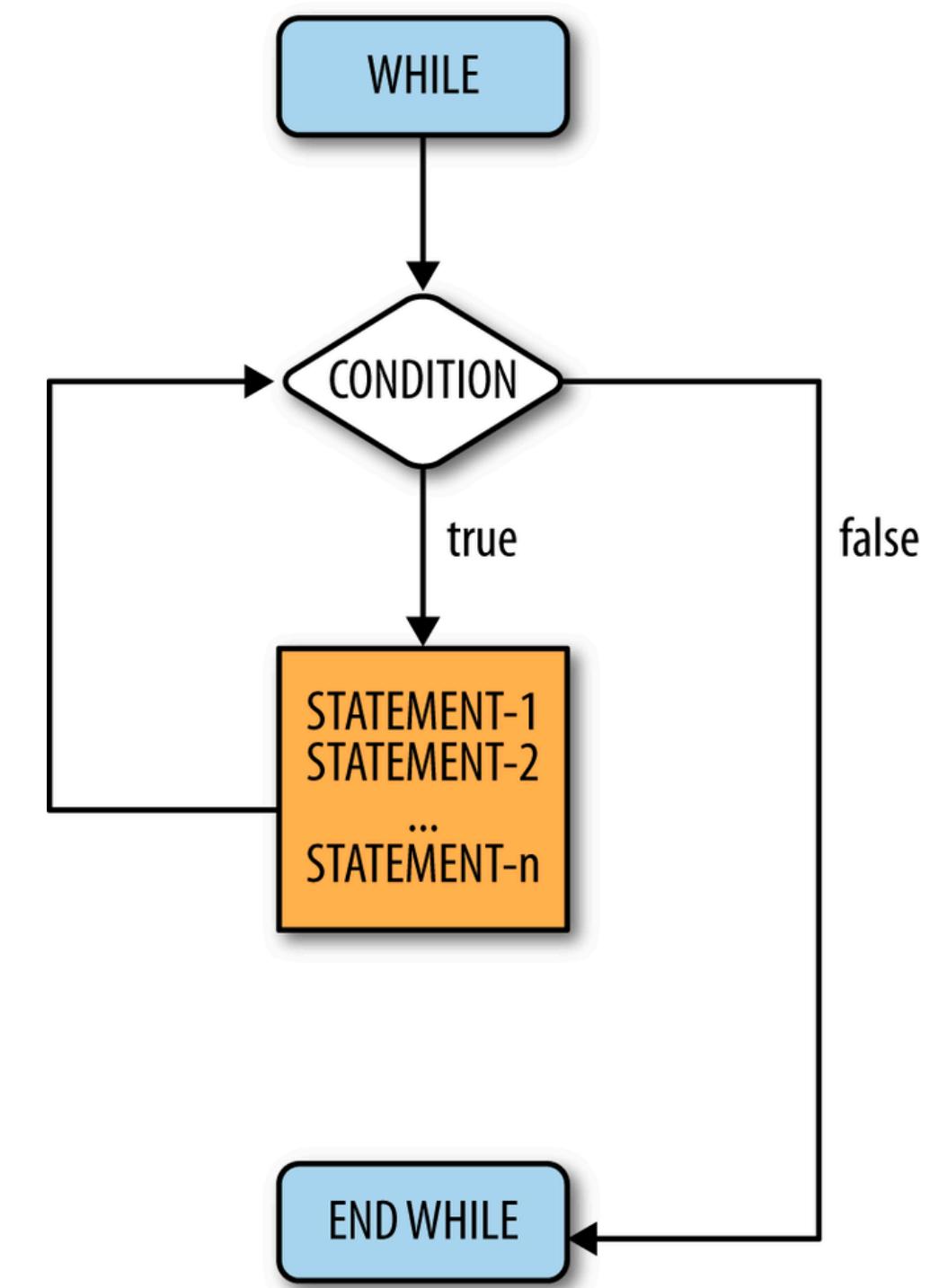
LOOPING BACKWARDS AND LOOPS IN LOOPS



ITERATION: THE FOR LOOP

В JavaScript также существует цикл while, который выполняется, пока указанное логическое условие истинно.

```
while (condition) {  
    statement  
}
```



ДОМАШНЕЕ ЗАДАНИЕ

1. Написать рекурсивную функцию, которая вычисляет факториал числа.
2. Написать рекурсивную функцию, которая проверяет, является ли строка палиндромом (т.е. читается с обоих сторон одинаково).
3. Реализуйте собственную версию функции map для массивов. Функция должна принимать массив и функцию-обработчик, а возвращать новый массив с результатами применения функции-обработчика к каждому элементу исходного массива.

