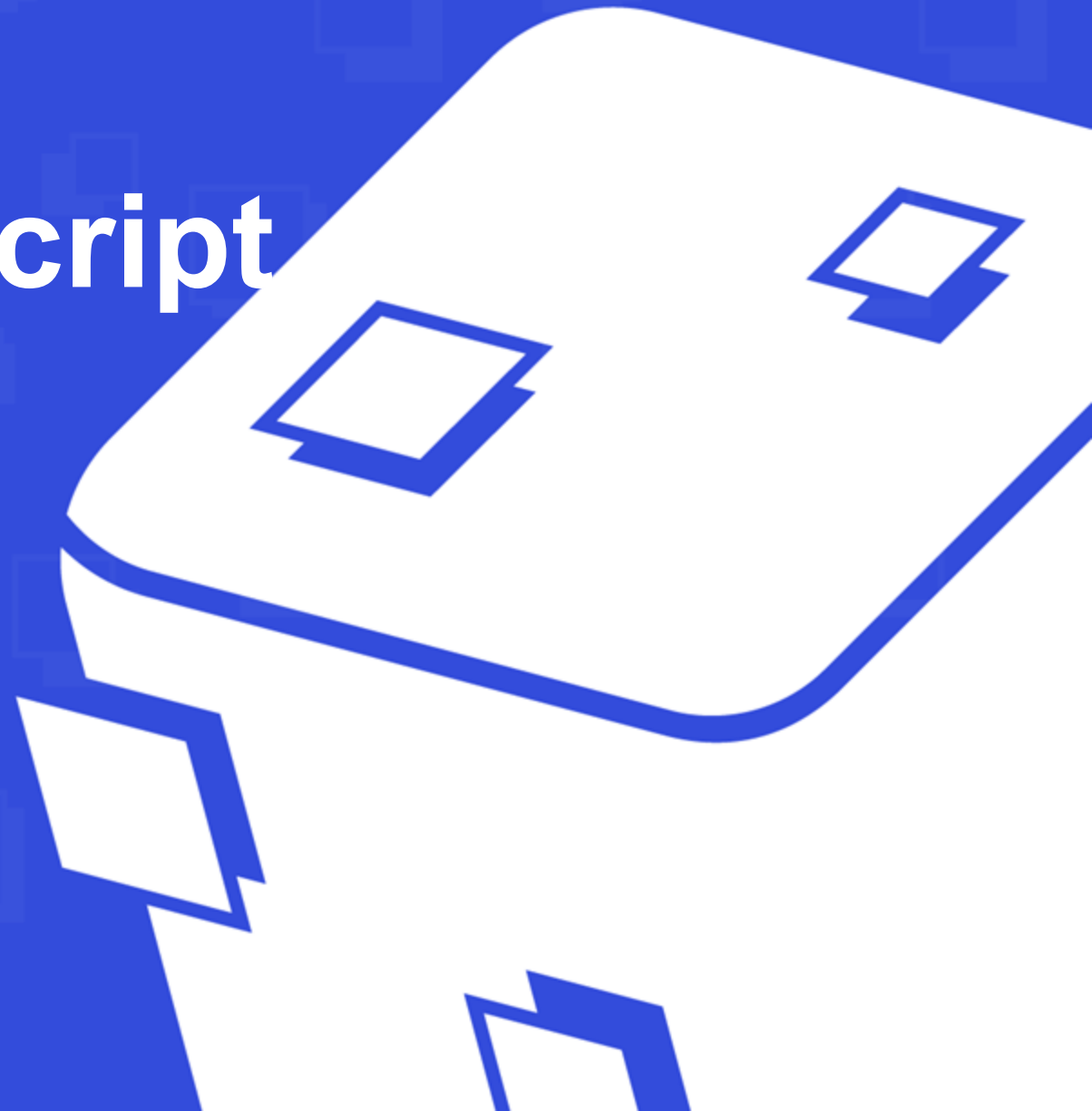


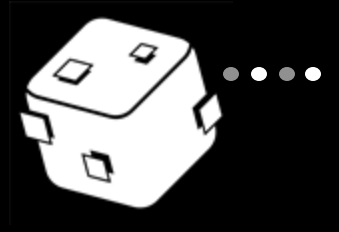


Функции в JavaScript



Contents

”



- 01 | Введение
- 02 | Создание и вызов
- 03 | Параметры и возврат
- 04 | Стрелочные функции
- 05 | Практика и итог

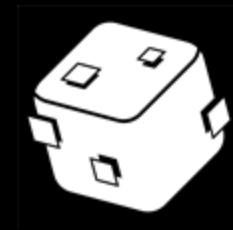




01

Введение





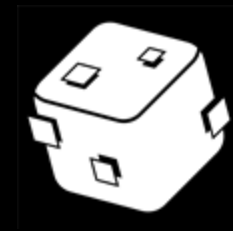
Функции в JavaScript



Понять, что такое функции и зачем они нужны.



Научиться **создавать** , **вызывать** и **передавать данные**.

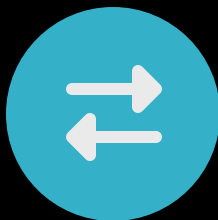


Что будет на уроке?



Объявление и вызов

Основы создания и запуска функций.



Параметры и return

Как передавать данные и получать результаты.



Стрелочные функции

Современный и удобный синтаксис.



Практика

Пишем свои собственные функции.

Что такое функция?

Функция — это **мини-программа** внутри программы.
Она выполняет определённое действие и может
вызываться много раз.



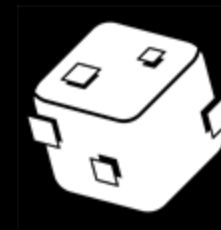
Сложить два числа



Показать приветствие



Проверить пароль



Зачем нужны функции?



Не повторять код

Пишите один раз, используйте много раз.



Делить программу

Сложная задача становится простой.



Сделать код понятнее

Легче читать, легче исправлять.



02

Создание и вызов

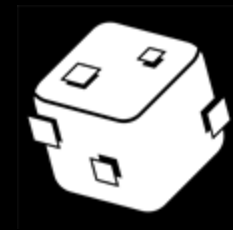




Объявление функции

Сначала мы **создаем** функцию. Пока она только определена, но не выполнена.

```
function sayHello()  
{ alert("Привет, мир!");  
}
```



Вызов функции

Чтобы функция выполнялась, её нужно **вызвать**, написав её имя со скобками.

```
sayHello(); // Выведет "Привет, мир!"
```

```
sayHello();
```

```
sayHello();
```

```
sayHello();
```

Одинаковый код можно использовать много раз!



03

Параметры и возврат





Функции с параметрами

Параметры — это данные, которые мы передаём в функцию, чтобы сделать её гибче.

```
function greet(name) {  
    alert("Привет, " + name + "!");  
}  
greet("Алмас");  
greet("Арман");
```

Вход: "Алмас"



Результат: "Привет, Алмас!"

Вход: »Арман"



Результат: "Привет, Арман!"



Несколько параметров

Функция может принимать два, три или более параметров.

```
function add(a, b)
{
    alert(a + b);
}
add(5, 3); // 8
add(10, 7); // 17
```

a = 5, b = 3

8

a = 10, b = 7

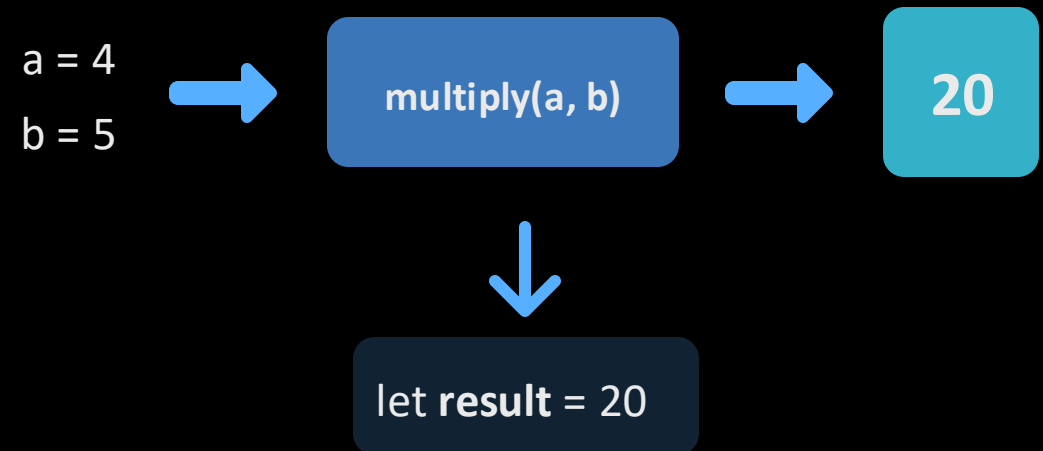
17

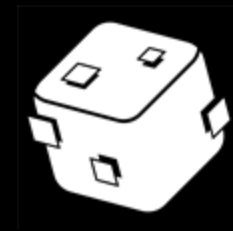


Возврат значения (return)

Ключевое слово `return` позволяет функции вернуть результат для дальнейшего использования в коде.

```
function multiply(a, b) {  
  return a * b;  
} // Результат сохраняется в переменной  
let result = multiply(4, 5);  
alert(result); // 20
```





Разница между alert и return



alert()

Просто **показывает** результат на экране.



return

Возвращает значение в код для
дальнейшего использования.



04

Стрелочные функции



Стрелочные функции

Короткий и современный способ записи функции.

```
// Обычная функция
function sayHi() {
  alert("Привет!"); } //
Стрелочная функция
const sayHi = () => {
  alert("Привет!"); };
```



Быстрее, короче, современнее!



Стрелочная функция с параметром

```
const greet = (name) => {  
  alert("Привет, " + name);  
};
```

Вход: "Мария"



(name) => { ... }



"Привет, Мария"



Если параметр один

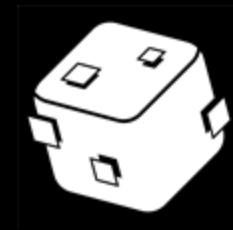
Можно писать без скобок. Если тело состоит из одной строки, можно опустить фигурные скобки и `return` .

```
// Полная запись  
const square = (n) => { return n *  
n; } // Сокращенная запись  
const square = n => n * n;
```

`alert(square(5));`



25



Преимущества стрелочных функций



Меньше кода

Компактная запись, особенно для простых действий.



Удобство

Идеально подходят для коротких операций.



Современность

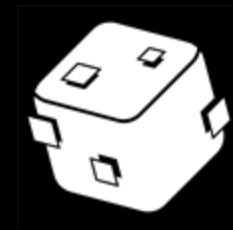
Широко используются в современных проектах и фреймворках.



05

Практика и итог

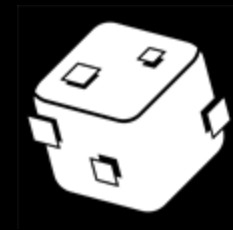




Практика 1

Создайте функцию `showMessage()` , которая выводит сообщение.

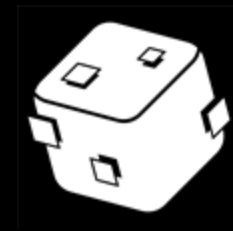
```
function showMessage() {  
  alert("Добро пожаловать на сайт!"); }  
// Не забудьте вызвать!  
showMessage();
```



Практика 2

Создайте функцию `sayHi(name)` , которая приветствует пользователя.

```
function sayHi(name) {  
  alert("Привет, " + name + "!");  
  sayHi("Алия");  
}
```



Практика 3

Создайте функцию `sum(a, b)` и выведите результат.

```
function sum(a, b) { return a + b; }  
let result = sum(10, 20);  
alert(result); // 30
```

10 + 20



30

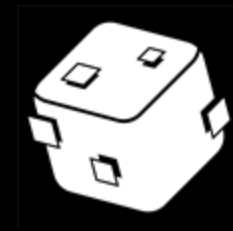


Практика 4

Напишите стрелочную функцию, которая считает квадрат числа.

```
// Ваш код здесь  
const square = ...  
alert(square(6)); // 36
```

6^2
↓
36



Итоги урока



Что такое функции и зачем они нужны.



Как объявлять и вызывать функции.



Как передавать параметры и возвращать значения.



Что такое стрелочные функции и как их применять.