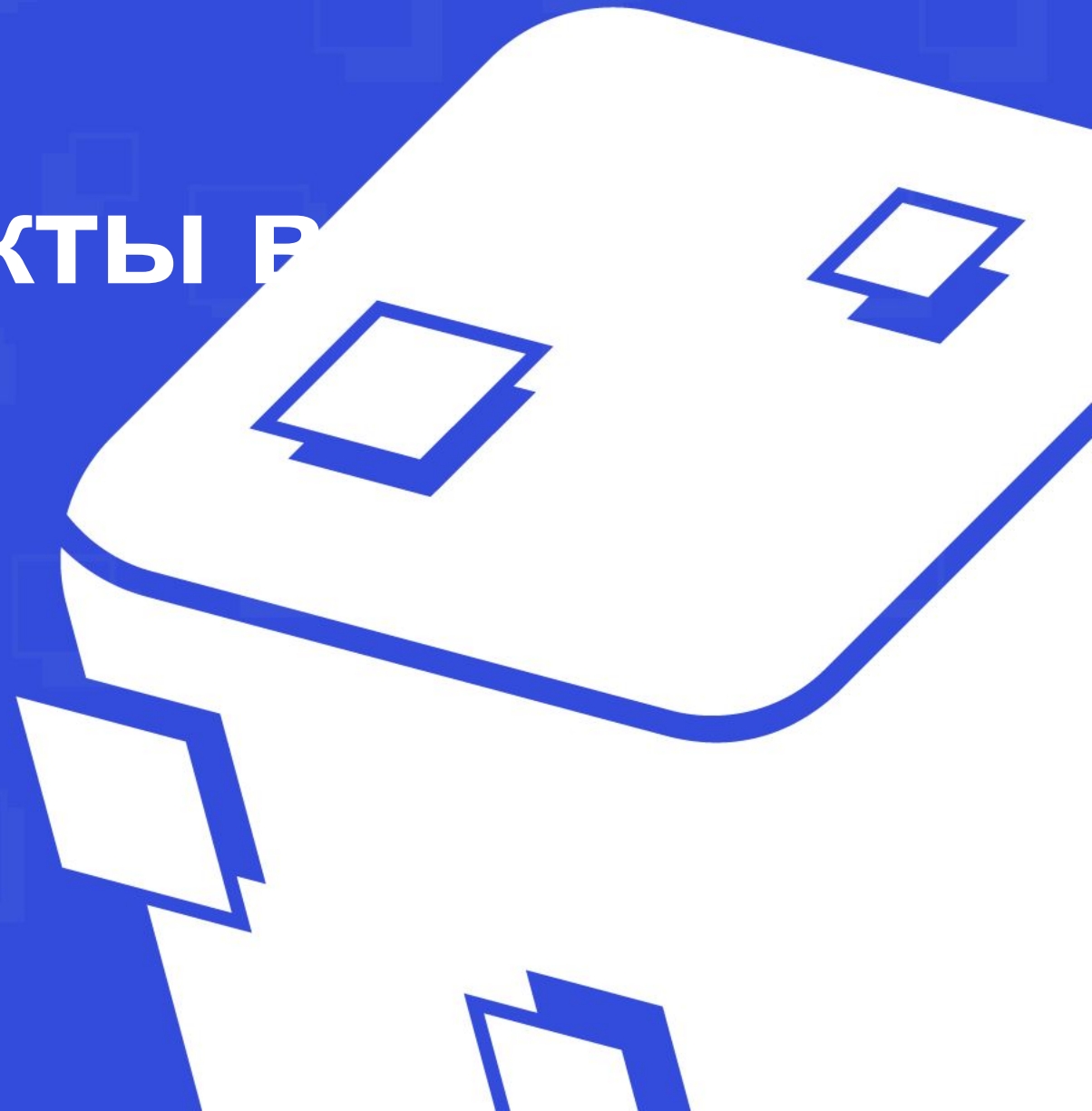


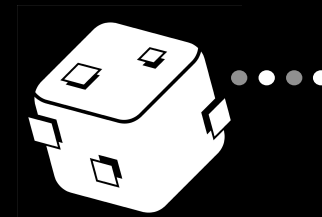


# Массивы и объекты в JavaScript



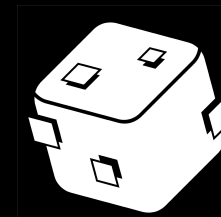
# Contents

”



- 01 | Введение и цели
- 02 | Массивы: базовые операции
- 03 | Методы перебора и преобразования
- 04 | Объекты: структурированные данные
- 05 | Массив объектов и практика



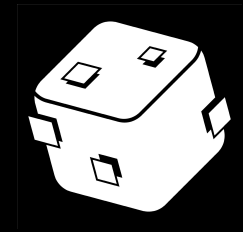


# 01

---

## Введение и цели

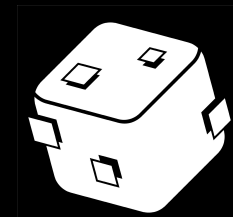




# Массивы и объекты в JavaScript

---

Изучение массивов и объектов как основных структур данных. Эти инструменты позволяют хранить и обрабатывать наборы информации в одной переменной, что критично для построения любых приложений.



## Цель урока



### Создавать

Научимся создавать коллекции данных:  
от простого списка до  
структурированных записей.



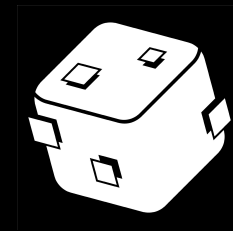
### Изменять

Узнаем, как модифицировать данные:  
добавлять, удалять и обновлять  
элементы.



### Обрабатывать

Освоим методы для фильтрации,  
преобразования и анализа данных.



# Что будет на уроке





1=  
2=

## Массивы

Что это такое и как с ними работать.



## Методы массивов

Основные инструменты: , , ,  
.



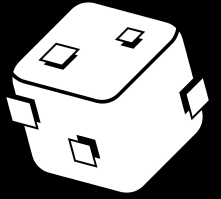
## Объекты

Что это такое и как использовать ключи.



## Практика

Создание и обработка списка пользователей.

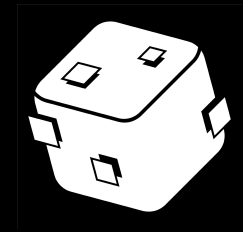


# 02

---

## Массивы: базовые операции





# Что такое массив?

Массив — это упорядоченный список данных.  
Например, список имён, чисел или товаров.

```
let fruits = ["яблоко", "банан", "груша"];
```

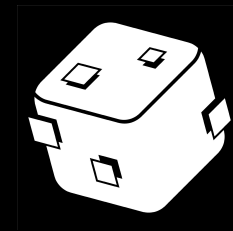
📦 В одной переменной хранится сразу несколько значений.

**яблоко**  
fruits[0]

**банан**  
fruits[1]

**груша**  
fruits[2]





# Доступ к элементу по индексу

Нумерация в массиве начинается с **0**.

```
let fruits = ["яблоко", "банан", "груша"];
```

fruits[0]

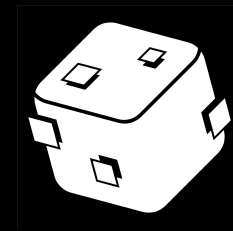


яблок  
о

fruits[2]



груш  
а



# Изменение элементов массива

```
let fruits = ["яблоко", "банан", "груша"];
```



```
fruits[1] = "апельсин";
```



```
alert(fruits); // яблоко, апельсин, груша
```

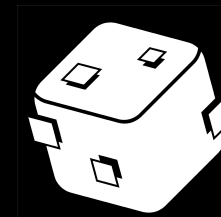
# Метод push()

Добавляет один или несколько элементов в конец массива и возвращает новую длину.

```
let fruits = ["яблоко", "банан"];
```

```
fruits.push("киви");
```

```
// ["яблоко", "банан", "киви"]
```



яблоко

банан



КИВИ

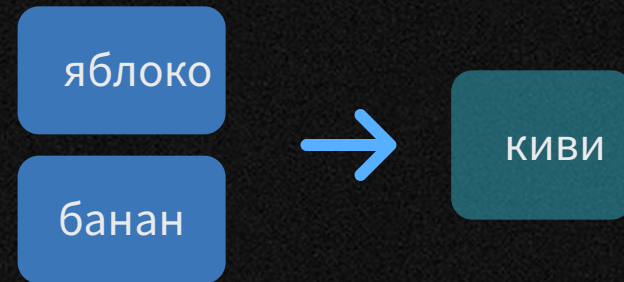
# Метод pop()

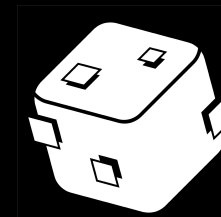
Удаляет последний элемент из массива и возвращает его. Массив изменяется на месте.

```
let fruits = ["яблоко", "банан", "киви"];
```

```
fruits.pop(); // Возвращает "киви"
```

```
// ["яблоко", "банан"]
```



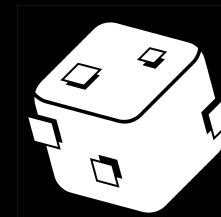


# 03

---

## Методы перебора и преобразования





# Метод `map()`

Создаёт **новый массив**, применяя функцию к каждому элементу исходного.

Исходный массив

[1, 2, 3]

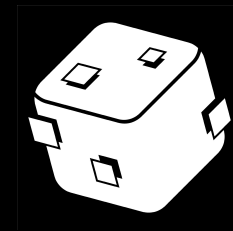


`.map(n => n * 2)`

Новый массив

[2, 4, 6]

```
let doubled = numbers.map(n => n * 2);
```



# Метод filter()

Создаёт **новый массив**, содержащий только элементы, прошедшие проверку.

Исходный массив

[5, 12, 8, 20]

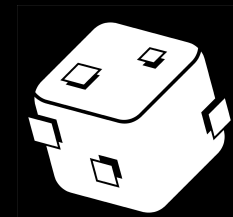


`.filter(n => n > 10)`

Отфильтрованный

[12, 20]

```
let big = numbers.filter(n => n > 10);
```



# Пример: Список имён

```
let names = ["Алия", "Олжас", "Айгерим"];
```



```
names.push("Бекзат");
```



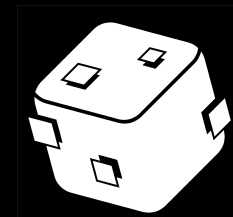
```
names.pop(); // Удалит "Бекзат"
```



```
alert(names);
```

```
// ["Алия", "Олжас", "Айгерим"]
```



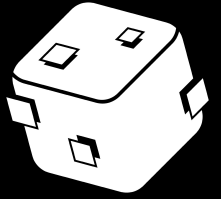


# 04

---

Объекты:  
структурированные  
данные



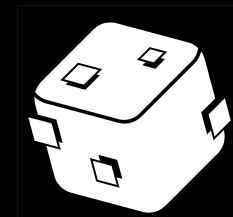


# Что такое объект?

Объект — это набор свойств, где каждое свойство имеет **ключ** и **значение**.

```
let user = {  
  name: "Алия",  
  age: 25,  
  city: "Астана"  
};
```





# Доступ к свойствам объекта

```
let user = { name: "Алия", age: 25, city: "Астана" };
```

## Точечная нотация

user.name



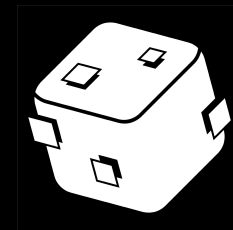
**"Алия"**

## Скобочная нотация

user["city"]



**"Астана"**



# Изменение данных в объекте

```
let user = { name: "Алия", age: 25, city: "Астана" };
```

**Было**

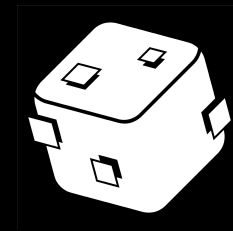
age: 25  
city: "Астана"



```
user.age = 26;  
user.city = "Алматы";
```

**Стало**

age: 26  
city: "Алматы"



# Добавление свойств

Объекты в JavaScript динамически расширяемы. Вы можете добавить новое свойство в любой момент.

```
let user = { name: "Алия" };
```

+

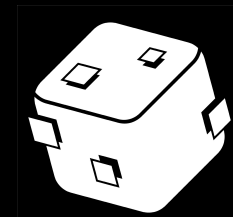
```
user.email = "aliya@example.com";
```

name: "Алия"



name: "Алия"

email:  
"aliya@example.com"



# Удаление свойств

Оператор **delete** полностью удаляет свойство из объекта.

```
let user = { name: "Алия", city: "Астана" };
```



```
delete user.city;
```

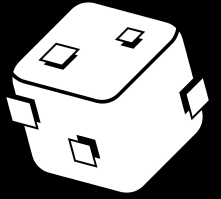
name:  
"Алия"

city:  
"Астана"



name:  
"Алия"

city:  
"Астана"

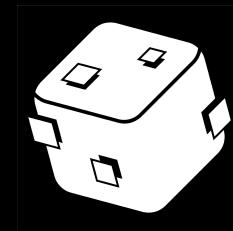


# 05

---

## Массив объектов и практика





# Массив объектов

Часто нужно хранить список однотипных сущностей, например, пользователей.

```
let users = [
```

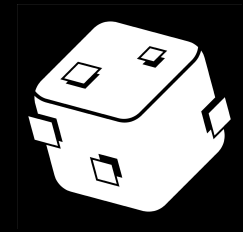
```
{ name: "Алия",  
  age: 25 },
```

```
{ name: "Олжас",  
  age: 28 },
```

```
{ name: "Айгерим",  
  age: 22 }
```

```
];
```



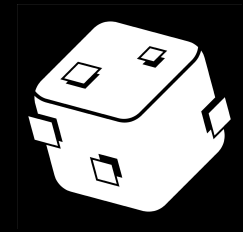


# Перебор массива объектов

Метод **forEach()** позволяет последовательно обработать каждый элемент.

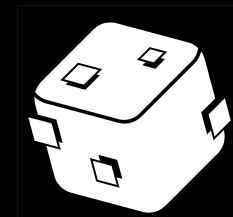
```
users.forEach(user => {  
  console.log(user.name + " — " + user.age + " лет");  
});
```

```
> Алия — 25 лет  
> Олжас — 28 лет  
> Айгерим — 22 лет
```



# Практика: Список пользователей

1. Создайте массив из 3 объектов пользователей.
2. Каждый объект должен содержать **имя** и **возраст**.
3. С помощью  создайте новый массив только с именами.
4. С помощью  оставьте только тех, кто старше 23 лет.
5. Выведите результаты в консоль.



## Итоги урока



Что такое массивы и как с ними работать.



Как добавлять, удалять и фильтровать данные.



Что такое объекты и ключи.



Как создать список пользователей и обработать его.



Как применять методы **push**, **pop**, **map**, **filter** на практике.