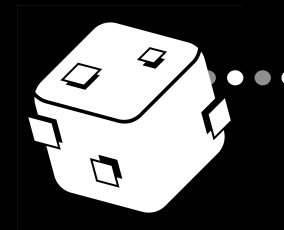




# Стилизация React: от CSS до Tailwind CSS





01

Зачем стилизовать React

02

Классические подходы

03

TailwindCSS как новая норма

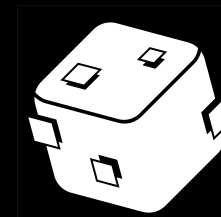
04

Продвинутый Tailwind

05

Практика



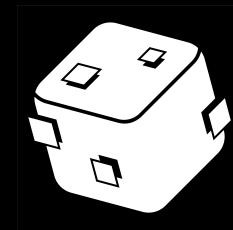


# 01

---

## Зачем стилизовать React





# Проблема глобальных стилей

---

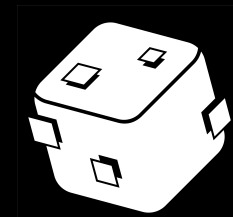
В React каждый экран — это компонент. Главная задача стилизации — **управлять** стилями внутри множества компонент, чтобы они не конфликтовали.

## Проблема новичков:

- ✗ Создаёшь два компонента
- ✗ Оба используют класс `'title'`
- ✗ Стили пересекаются - всё ломается

## Решение:

Использовать подходы, которые изолируют стили, делая их локальными для каждого компонента.



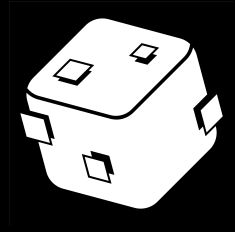
# 02

---

Классические  
ПОДХОДЫ



# Подход №1: Обычный CSS



Самый базовый способ, но со скрытыми ловушками.

App.css

```
.title { color: red; font-weight: bold; }
```



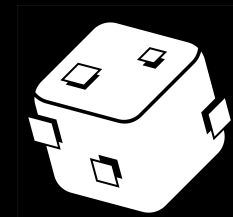
App.jsx

Hello

**Но главный недостаток: все стили глобальные.**

Если другой компонент тоже назовёт класс '.title', они пересекутся. Такой подход подходит только для маленьких проектов.

# Подход №2: CSS Modules



Рекомендуется для начинающих. Обеспечивает полную изоляцию стилей.

## Button.module.css

```
.btn { background: #2563eb; padding: 12px 20px; border-radius: 8px; color: white; }
```

## Button.jsx

```
import styles from './Button.module.css';  
Click
```

.btn →

.btn\_ajd93kds

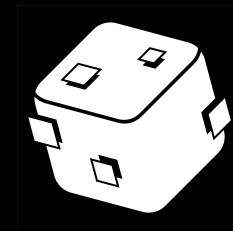
React сам превращает классы, гарантируя их уникальность.

✓ Стили не пересекаются

✓ Легко для новичков

✓ Похоже на обычный CSS

✓ Не требует доп. инструментов



# Подход №3: CSS-in-JS

CSS пишется прямо внутри JS, используя библиотеки.

## Пример (Styled Components)

```
import styled from 'styled-components'; const Box = styled.div`  
background: #f0f0f0; padding: 20px; border-radius: 12px;`; Hello!;
```

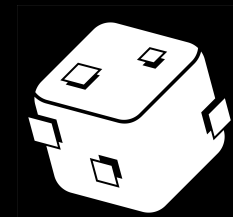
## Преимущества

- ✓ Компонент содержит и логику, и стиль
- ✓ Легко передавать пропсы
- ✓ Подходит для SSR (Next.js)

## Недостатки

- ✗ Хуже по производительности
- ✗ Может быть громоздким





# 03

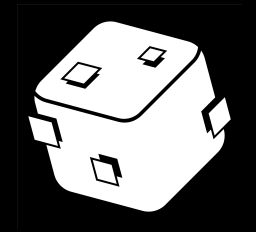
---

TailwindCSS как  
Новая норма



# TailwindCSS: Лидер индустрии

Utility-first CSS framework, который полностью меняет подход.



## CSS способ:

```
.button { background: #2563eb; color: white; padding: 8px 16px; border-radius: 6px; }
```

Создание класса, написание CSS, применение класса.

✓ Быстрее

✓ Встроенная адаптивность

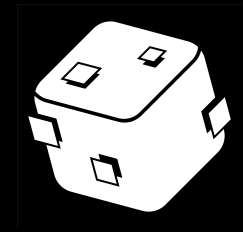
## Tailwind способ:

Click

Использование готовых утилит напрямую в JSX.

✓ Компактный код

✓ Огромное комьюнити



# Установка TailwindCSS (кратко)



## 1. Установка

```
npm install -D tailwindcss  
postcss autoprefixer
```



## 2. Инициализация

```
npx tailwindcss init -p
```



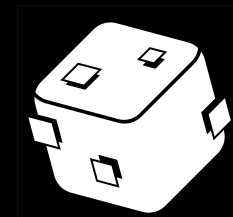
## 3. Настройка

Указать пути в 'content'.



## 4. Подключение

Добавить директивы  
в 'index.css'.



# TailwindCSS: Палитра и сетка за одну строку

## Цвета

text-red-500  
bg-green-600

## Типографика

text-xl  
font-bold

## Отступы

p-4 (padding)  
mt-8 (margin-top)

## Скругления

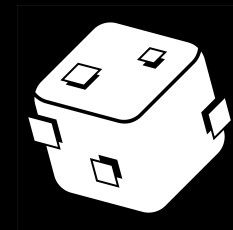
rounded  
rounded-full

## Flexbox

flex  
items-center

## Тени

shadow  
shadow-lg



# Пример: Карточка на TailwindCSS

```
p-6 bg-white shadow-lg rounded-xl ">  
text-xl font-bold mb-2 "> Заголовок  
text-gray-600 leading-relaxed "> Это пример карточки...
```

**p-6 / bg-white / shadow-lg / rounded-xl**

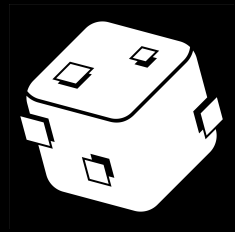
Общий padding, белый фон, большая тень, сильное скругление.

**text-xl / font-bold / mb-2**

Размер шрифта, жирный, отступ снизу.

**text-gray-600 / leading-relaxed**

Цвет текста, высота строки.

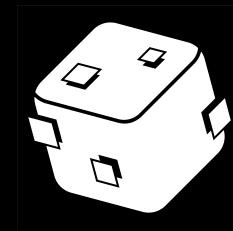


# 04

---

## Продвинутый Tailwind





# Адаптивность с префиксами

Tailwind создаёт брейкпойнты, которые применяют стили только на определённой ширине экрана.

**sm:**  
≥ 640px

**md:**  
≥ 768px

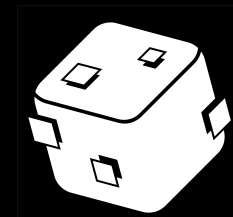
**lg:**  
≥ 1024px

**xl:**  
≥ 1280px

## Пример:

```
sm:bg-green-400 md:bg-yellow-400 lg:bg-red-400 "> Adaptive
```

Цвет фона будет меняться в зависимости от ширины экрана.



# Почему TailwindCSS ускоряет разработку



## Не нужно переключаться между JS и CSS

Весь UI живёт в JSX, что экономит время.



## Библиотека заранее продуманных утилит

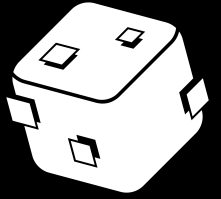
Не нужно вспоминать синтаксис `'margin-top: 16px;'`, достаточно `'mt-4'`.



## Стойкость к росту проекта

В больших кодовых базах CSS становится хаосом. Tailwind всё упрощает.



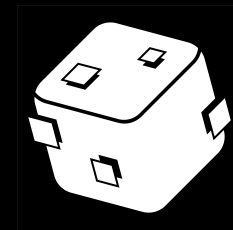


# 05

---

Практика





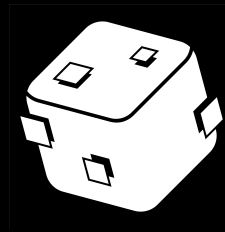
# Практическое задание

## Создайте компонент 'UserCard'

Используя **TailwindCSS**, создайте карточку пользователя, которая содержит:

- 👤 Аватар (круглый)
- 👤 Имя и должность
- ➕ Кнопку "Follow"
- 🔄 Тень и скругления
- 📱 Адаптивный layout

# Пример решения



**Arman Maksutov**

Frontend Developer

Follow

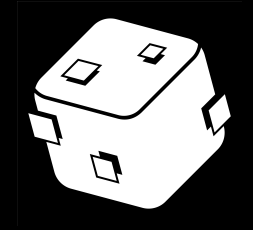
# Продвинутый пример

## Карточка товара

Сложный макет, собранный целиком из утилит Tailwind. Демонстрирует работу с изображениями, flex-раскладкой, отступами и адаптивом.

### Ключевые моменты:

- ✓ Комбинация текста, кнопок, отступов
- ✓ Адаптивные элементы
- ✓ Код остаётся читаемым



MacBook Pro 14

Apple M3 Pro, 16GB RAM