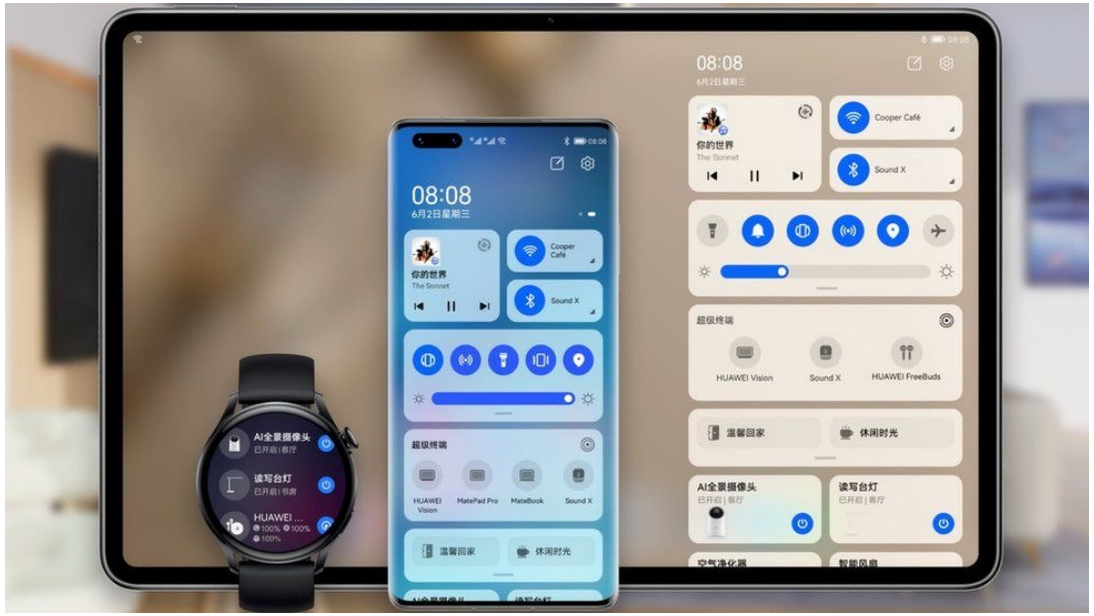
Лекция 12
Нативные приложения и адаптивность

Разработка интернет приложений

Канев Антон Игоревич

Виды нативных приложений

- **Нативное приложение** (native app) – прикладная программа, которая разработана для определенной платформы или для определенного устройства
- **Мобильное приложение** (mobile app) — прикладная программа, предназначенная для работы на смартфонах, планшетах и других мобильных (портативных, переносных, карманных) устройствах, собирается отдельно для iOS или Android
- **Десктопное приложение** (desktop app) — программа, которая устанавливается на компьютер пользователя и работает под управлением операционной системы, собирается отдельно для maxOS, Windows, Linux



Языки для мобильной разработки

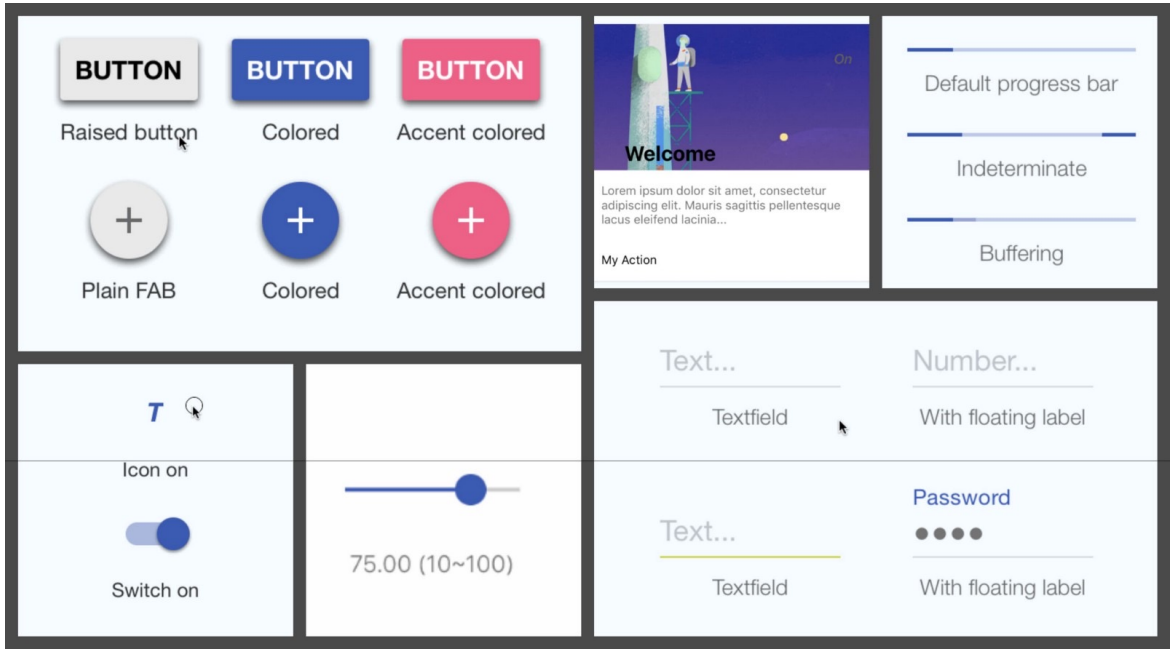
- iOS: Objective-C, Swift
- Android: Java, Kotlin

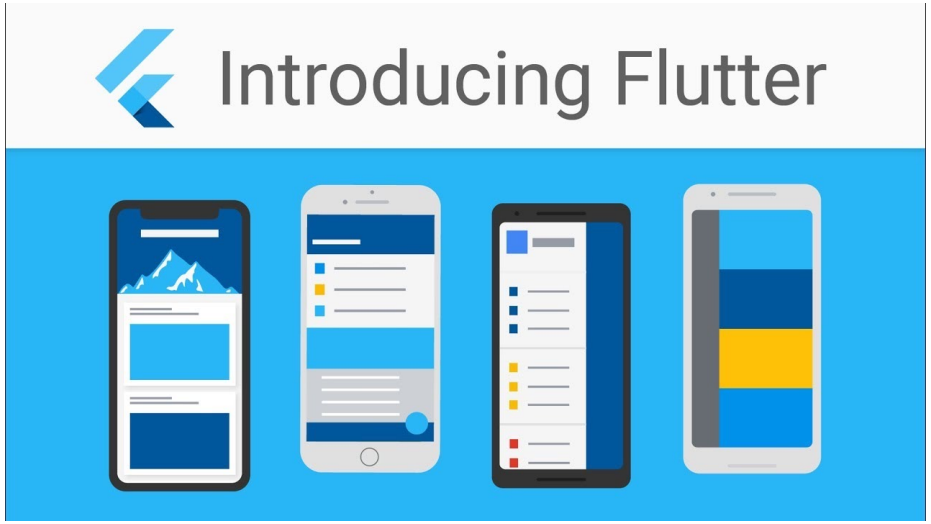
Java

Kotlin

Кроссплатформенная разработка

- Flutter, React Native, Kotlin multiplatform для кроссплатформенной мобильной разработки
- Flutter на Dart от Google
- React Native на JavaScript
- Kotlin multiplatform на Kotlin от JetBrains



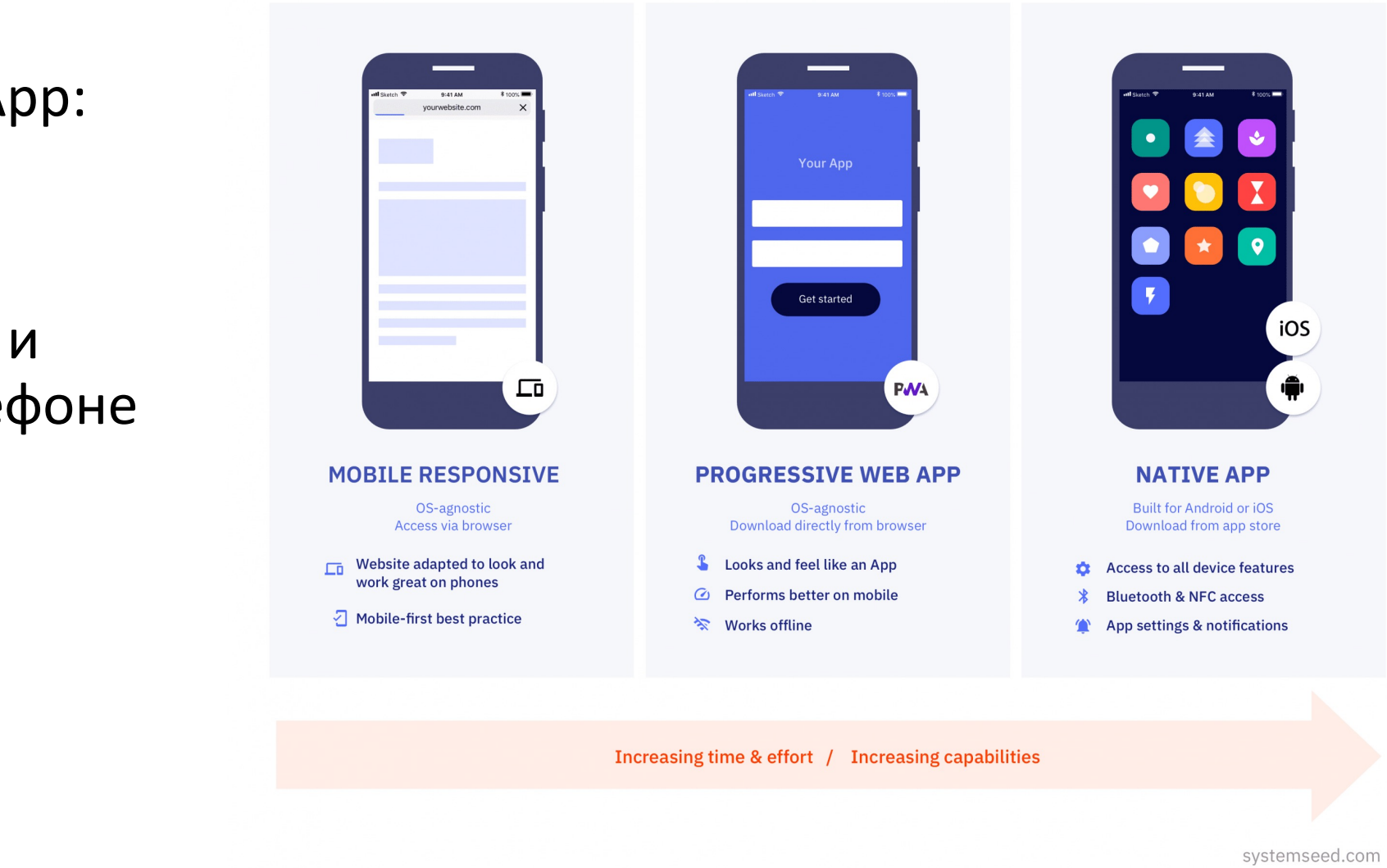


- Electron/Tauri и Qt для кроссплатформенных десктопных приложений
- Electron на JavaScript
- Tauri на JavaScript и Rust
- Qt на C++, PyQt на Python

PWA

Progressive Web App:

- Выглядит как приложение
- Работает лучше и быстрее на телефоне
- Работает offline



PWA. manifest.json

```
{
  "name": "Tile Notes",
  "short_name": "Tile Notes",
  "start_url": "/",
  "display": "standalone",
  "background_color": "#fdfdfd",
  "theme_color": "#db4938",
  "orientation": "portrait-primary",
  "icons": [
    {
      "src": "/logo192.png",
      "type": "image/png", "sizes": "192x192"
    },
    {
      "src": "/logo512.png",
      "type": "image/png", "sizes": "512x512"
    }
  ]
}
```

PWA. Service Worker

Регистрируем service worker, делаем это в файле `index.js` после рендера корневого компонента:

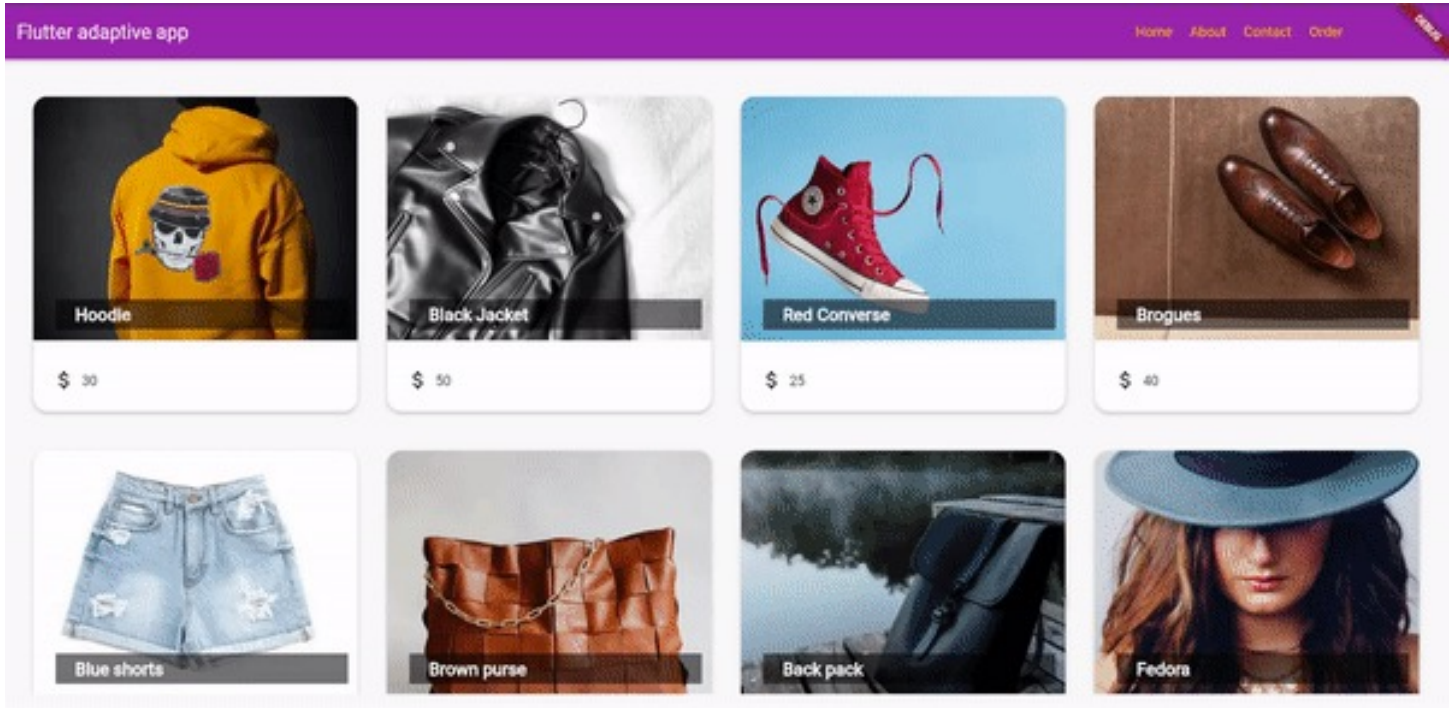
```
if ("serviceWorker" in navigator) {
  window.addEventListener("load", function() {
    navigator.serviceWorker
      .register("/serviceWorker.js")
      .then(res => console.log("service worker registered"))
      .catch(err => console.log("service worker not registered", err))
  })
}
```

Создаем файл serviceWorker.js и кладем его в директорию public:

```
self.addEventListener('fetch',() => console.log("fetch"));
```

Адаптивность

- свойства обертки, которые позволяют переносить элементы на новую строчку, если предыдущая заполнилась (**flex-wrap**), а так же задают отступы между соседними элементами сверху и снизу (**gap**)
- на свойства самой карточки: в данном случае нас интересует первое свойство **flex**, а точнее последнее значение в нем. Это значение определяет, в какой момент элементы переносятся на новую строчку, а именно если размер элемента становится равным **300px**

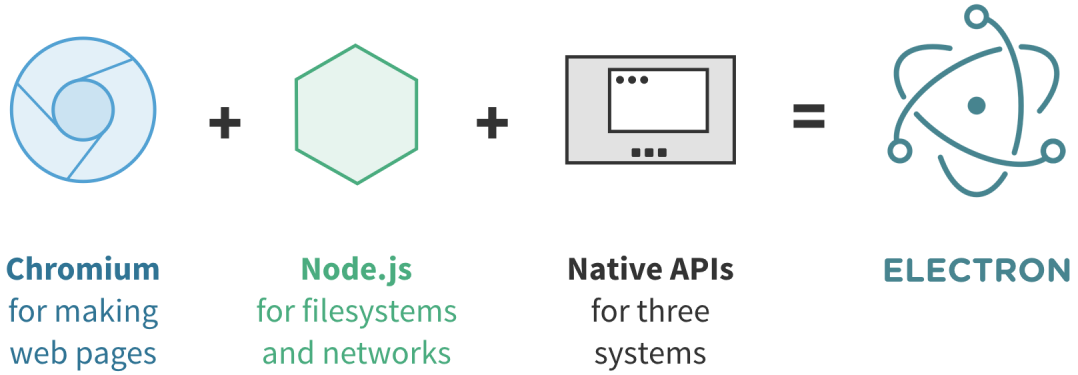


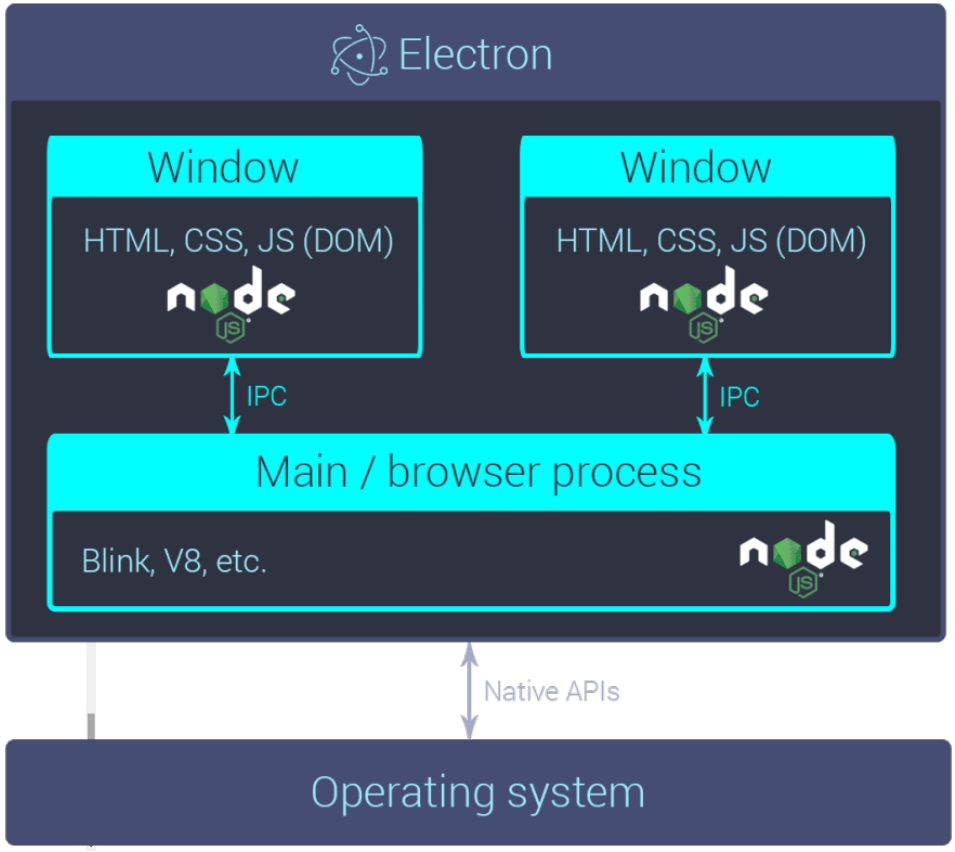

```html
<div class="cards__wrapper">
    <div class="card__item">
        <div class="card__img">
            <img src="./images/book.jpg" alt="book">
        </div>
        <div class="card__info">
            <div class="card__text">
                <div class="card__title">Название книги</div>
                <div class="card__description">Описание книги</div>
            </div>
            <button class="card__btn">Приобрести</button>
        </div>
    </div>
</div>
```

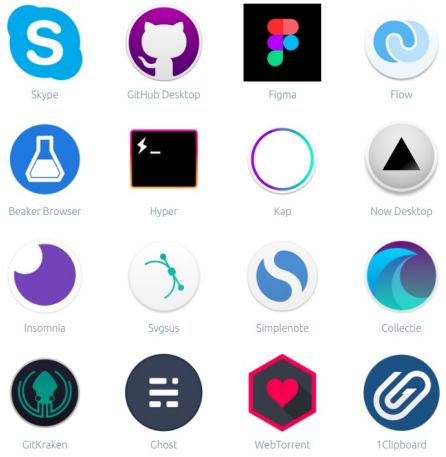
```css
.cards__wrapper {
    display: flex;
    justify-content: space-between;
    align-items: center;
    flex-wrap: wrap;
    gap: 20px;
}

.card__item {
    flex: 1 1 300px;
    padding: 15px;
    background-color: bisque;
    border-radius: 10px;
    display: flex;
    justify-content: space-between;
}
```

Electron

- **Electron** — фреймворк, разработанный GitHub.
- Позволяет разрабатывать нативные графические приложения для операционных систем с помощью веб-технологий, комбинируя возможности Node.js для работы с back-end и браузера Chromium

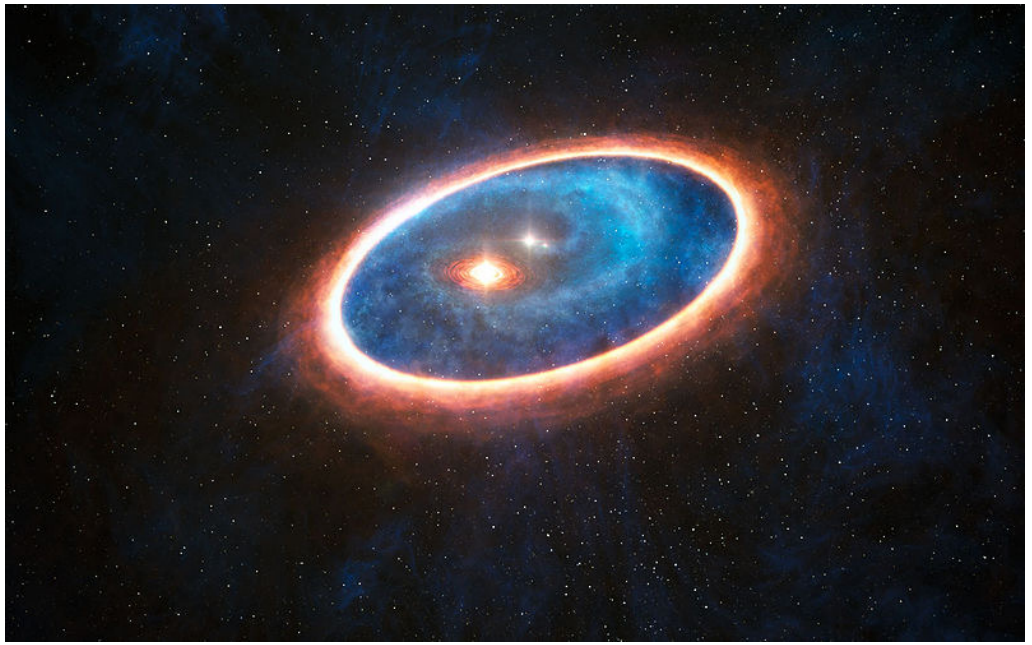






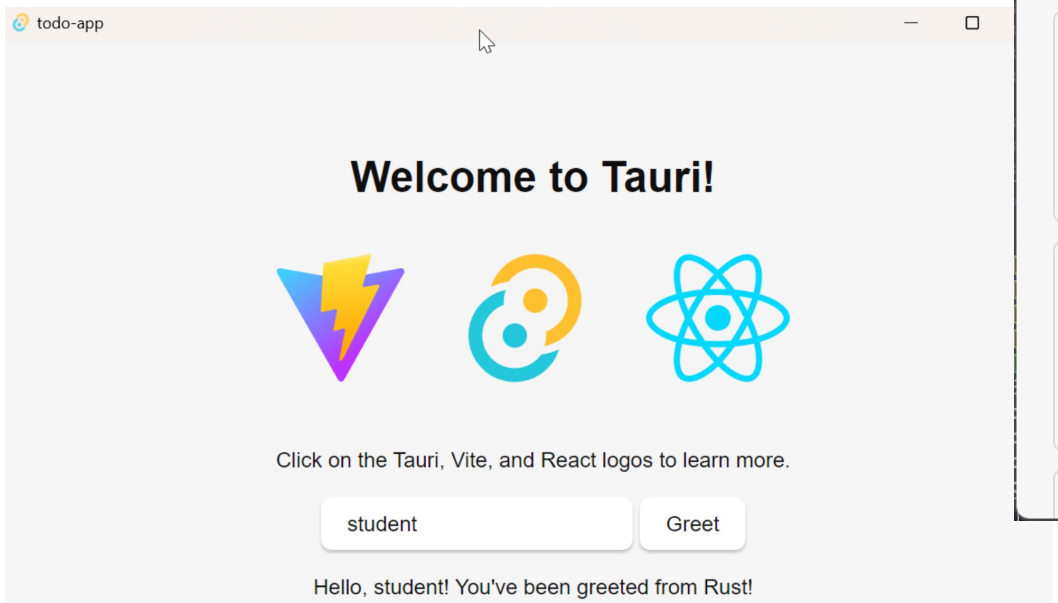
Tauri

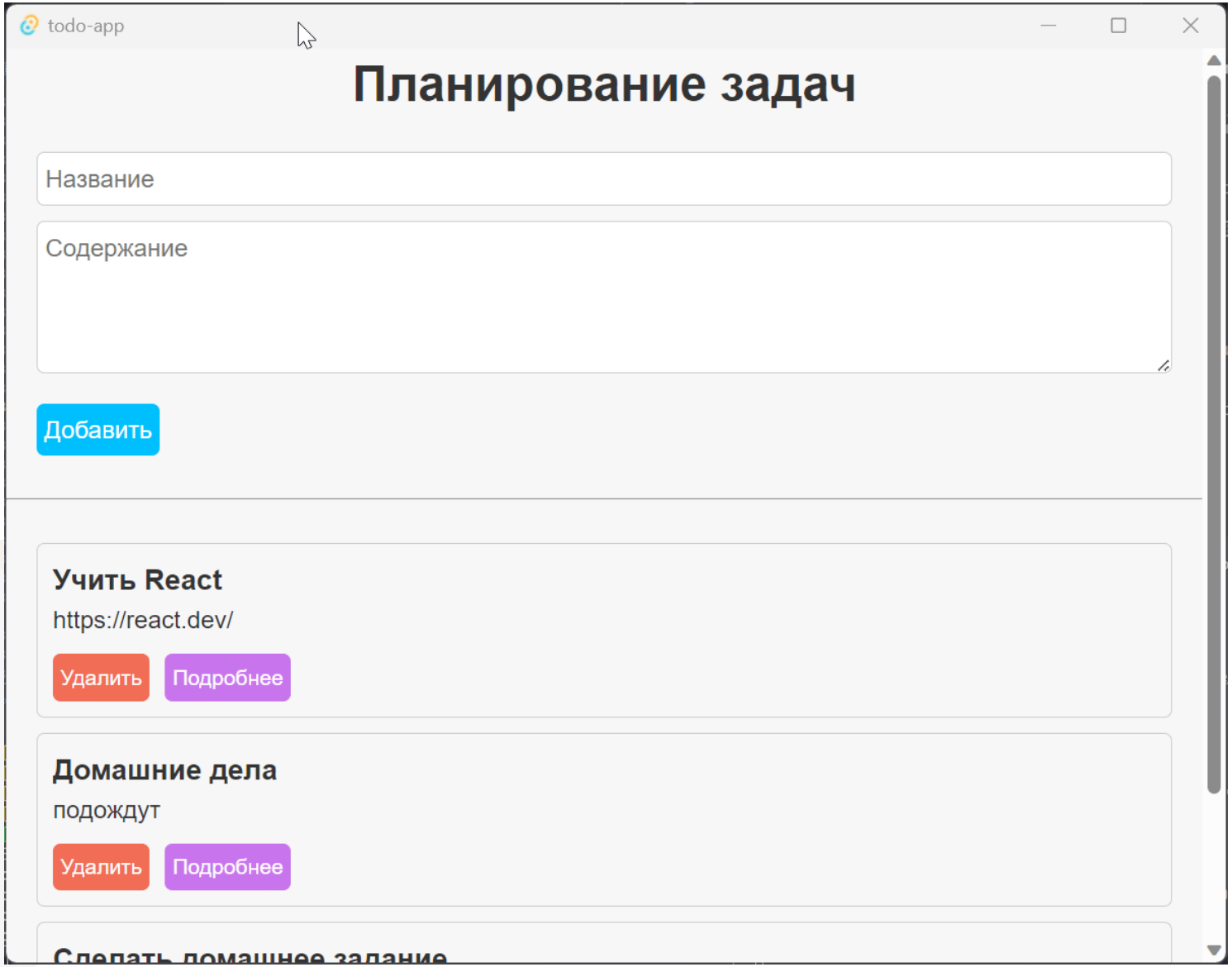
- Фреймворк для создания десктопных приложений, похожий на Electron, но позволяющий использовать Rust вместо Node.js, например, для взаимодействия с файловой системой.



Tauri

- Мы разрабатываем с помощью React приложение, которое имеет доступ к файловой системе, нативному меню, диалоговым окнам и к нашему API





Tauri

- Создание интерфейса Tauri приложения на React включает использование известных нам компонентов, обработчиков событий и хуков

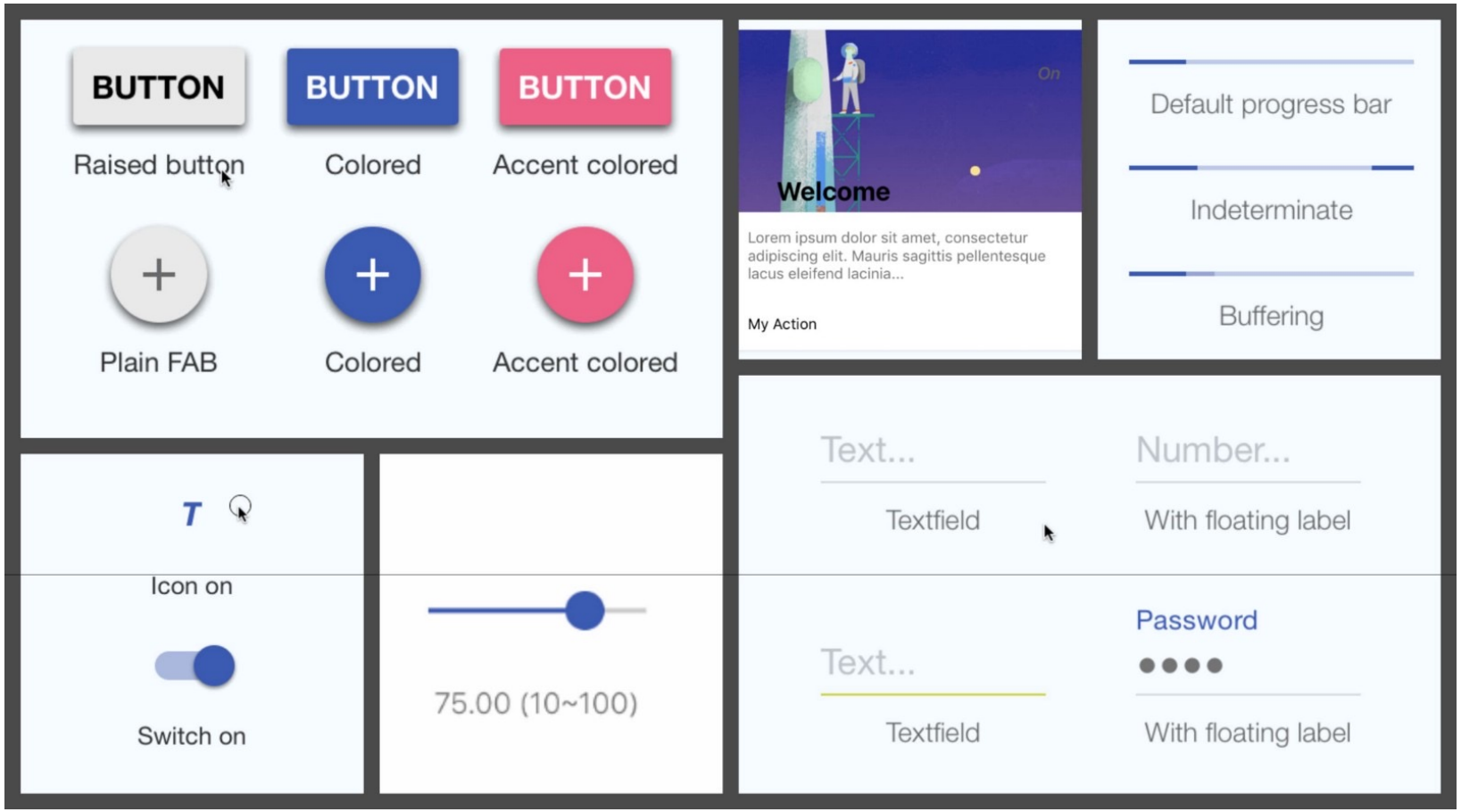
```
const TodoListPage = () => {
    // список всех задач
    const [todos, setTodos] = useState([]);

    // новая задача
    const [newTodo, setNewTodo] = useState({ title: '', content: '' });

    // добавление новой задачи
    const handleAddTodo = () => {
        if (!newTodo.title || !newTodo.content) {
            console.error('Поля не должны быть пустыми');
            return;
        };
        const newTodoWithId = { ...newTodo, id: Date.now() };
        setTodos([...todos, newTodoWithId]);
        setNewTodo({ title: '', content: '' });
    };

    // удаление задачи
    const handleDeleteTodo = (id) => {
        const updatedTodos = todos.filter((todo) => todo.id !== id);
        setTodos(updatedTodos);
    };

    {/*return ( ... )*/}
}
```

```
export function TodoListPage() {
    // список всех задач
    const [todos, setTodos] = useState([]);

    // новая задача
    const [newTodo, setNewTodo] = useState({ title: '', content: '' });

    return (
        <div>
            <h1>Планирование задач</h1>
            <div className='container'>
                <input
                    className='input-title'
                    type='text'
                    placeholder='Название'
                    value={newTodo.title}
                />
                <textarea
                    className='input-content'
                    placeholder='Содержание'
                    value={newTodo.content}
                />
                <button className='button button-success text-lg'>
                  Добавить
                </button>
            </div>
            <hr />
            <div className='container'>
                {todos.map((todo) => (
                    <div className='todo' key={todo.id}>
                        <h3 className='todo-title'>
                            {todo.title}
                        </h3>
                        <p className='todo-content'>
                            {todo.content}
                        </p>
                        <button className='button button-danger text-md'>
                            Удалить
                        </button>
                    </div>
                ))}
            </div>
        </div>
    );
```

```
<button
    className='button button-success text-lg'
    onClick={handleAddTodo}
>
    Добавить
</button>
```

React Native

- React Native – фреймворк для кроссплатформенной разработки на JavaScript
- Позволяет создавать нативные приложения с помощью известных нам технологий: axios, redux-toolkit и тд

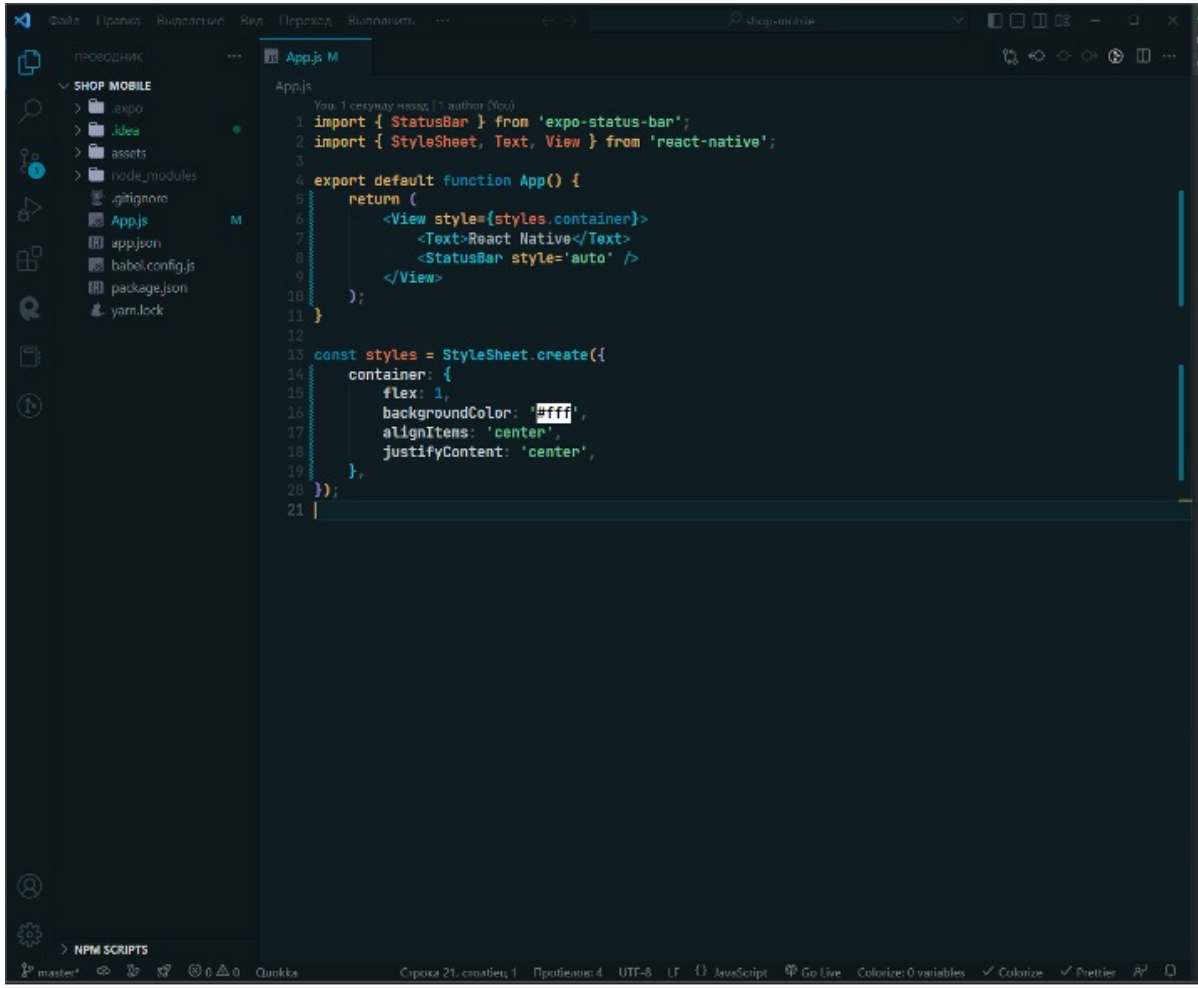


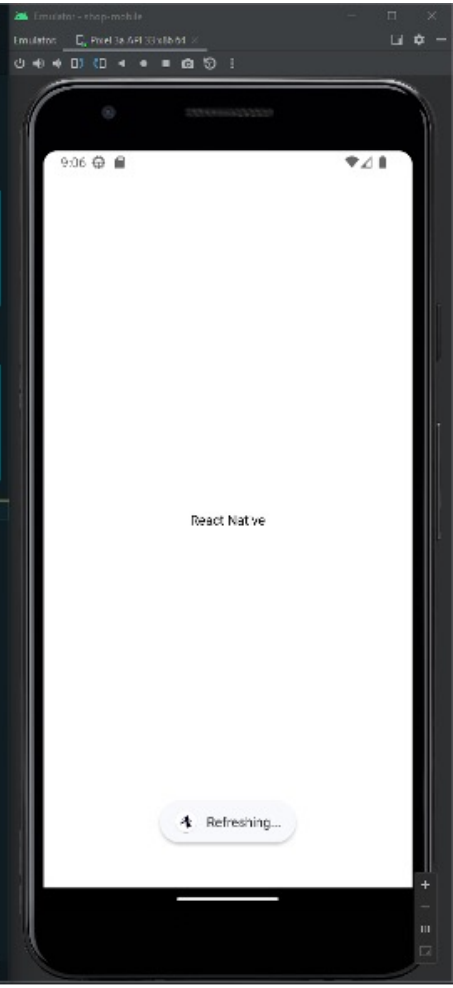
React Native

- Мы можем вести разработку в VS Code и смотреть изменения в телефоне через QR
- Или в Android Studio и эмуляторе

```
Starting project at C:\Users\mi\Desktop\1\shop-mobile
Starting Metro Bundler
```

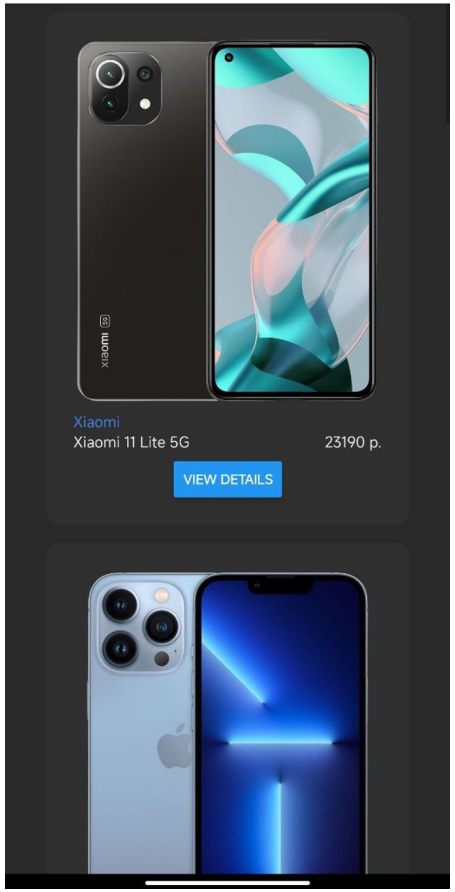
```
› Metro waiting on exp://192.168.0.167:19000
```





React Native

- Создадим карточки и заполним их данными из API




```
export default function ShopScreen({ navigation }) {
    const dispatch = useDispatch();
    const { devices } = useSelector((store) => store.device);

    useEffect(() => {
        async function getAllDevices() {
            await axiosInstance.get('/device').then((response) => dispatch(setDevices(response?.data)));
        }
        getAllDevices();
    }, [dispatch]);

    return (
        <ScrollView>
            <View style={styles.page}>
                {!!devices &&
                    devices.map((device) => <DeviceCard key={device.id} {...device} navigation={navigation} />)}
            </View>
        </ScrollView>
    );
}

const styles = StyleSheet.create({
    page: {
        display: 'flex',
        width: '100%',
        justifyContent: 'center',
        alignItems: 'center',
        backgroundColor: '#2a2a2a',
    },
});
```