Podstawy React Native

Zajęcia Rozwijające Kreatywność

Cele zajęć

1. Zapoznanie z podstawowymi elementami React Native
2. Nauka struktury aplikacji i stylizowania komponentów

Wprowadzenie

React Native to framework do tworzenia aplikacji mobilnych (Android i iOS) przy użyciu języka JavaScript i biblioteki React. Pozwala pisać kod raz, a następnie uruchamiać go na różnych platformach. React Native został stworzony przez firmę Meta (Facebook) w 2015 roku, projekt jest open-source, do jego rozwoju przyczyniają się również inne firmy technologiczne.

Dlaczego warto używać React Native?

- **Jedna baza kodu na iOS i Androida** – pozwala na tworzenie aplikacji mobilnych dla obu systemów operacyjnych z wykorzystaniem jednego kodu.
- **Szybszy rozwój aplikacji** – dzięki Hot Reloading zmiany w kodzie są natychmiast widoczne bez potrzeby ponownego uruchamiania aplikacji.
- **Wydajność zbliżona do natywnych aplikacji** – korzysta z natywnych komponentów UI, co zapewnia lepszą wydajność niż inne technologie hybrydowe.
- **Używany przez duże firmy** – popularne aplikacje takie jak Facebook, Instagram, Airbnb czy Uber Eats wykorzystują React Native ze względu na jego stabilność i wydajność.

Struktura Projektu

Najważniejsze pliki w projekcie React Native:

- App.js - Główny plik aplikacji, w którym umieszczony zostanie kod interfejsu.
- assets/ - Folder na zasoby statyczne (pliki multimedialne takie jak obrazy, ikony, dźwięki itp.)
- app.json - (nie mylić z App.js) Jest to plik konfiguracyjny Expo, zawierający ustawienia projektu.
- package.json - Zawiera listę zależności (bibliotek) projektu oraz ustawienia
- node_modules/ - Folder z paczkami NPM (bibliotekami)

Kod w React Native

Otwórz plik App.js.

Na początku pliku znajdują się importy, czyli pobranie potrzebnych elementów np. StatusBra (pasek stanu w telefonie), StyleSheet (do stylizowania aplikacji, jak CSS w stronach internetowych), Text (do wyświetlania tekstu) oraz View.

View (widok) to jeden z podstawowych komponentów w React Native, który pełni rolę kontenera dla innych elementów aplikacji. Można go porównać do w HTML, ale działa w sposób natywny dla systemów Android i iOS. View grupuje inne elementy interfejsu (np. Text, Button, Image), obsługując przy tym stylizację (np. Ustawienia koloru, rozmiary, marginesy, pozycjonowanie, itp.).

Funkcja App to główny element aplikacji, który zwraca widok ekranu. React Native wymaga, aby funkcja zwracała widok View.

```
export default function App() {
  return (
    <View style={styles.container}>
      <Text>Hello World!</Text>
      <StatusBar style="auto" />
    </View>
  );
}
```

React Native nie używa standardowego CSSa do stylizacji tak jak w stronach internetowych. Zamiast tego, wykorzystywany zostaje obiektowy system stylizacji, który przypomina CSS, ale zapisuje style jako obiekty JavaScript. React Native korzysta z komponentu StyleSheet, który pomaga organizować style w wydajny sposób.

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
    alignItems: 'center',
    justifyContent: 'center',
  },
});
```

- StyleSheet.create({...}) – tworzy obiekt stylów.
- Każdy klucz (container) to osobny zestaw stylów, jak w CSS klasach.
- Wartości zapisane są w formacie obiektowym.

Tworzenie własnych stylów

Dodaj nowy zestaw stylów o nazwie „nowytext”:

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#f0f0f0', // Szare tło
    alignItems: 'center',
    justifyContent: 'center',
  },
  nowytext: {
    fontSize: 24, // Zwiększamy rozmiar czcionki
    color: 'blue', // Kolor tekstu
    fontWeight: 'bold', // Pogrubienie tekstu
  },
});
```

Następnie użyj utworzonego stylu w elemencie Text w View funkcji App:

```
<View style={styles.container}>
  <Text>Hello World!</Text>
  <Text style={styles.nowytext}>Hello, React Native!</Text>
  <StatusBar style="auto" />
</View>
```

Najważniejsze style w Real Native

Kolory i tło

backgroundColor: 'red' – zmiana koloru tła.
color: 'blue' – zmiana koloru tekstu.
opacity: 0.5 – przezroczystość (1 - pełna widoczność, 0 - całkowicie przezroczysty).

Tekst i czcionki

fontSize: 20 – rozmiar czcionki.
fontWeight: 'bold' – pogrubienie (normal, bold, 100-900).
fontStyle: 'italic' – kursywa (normal, italic).
textAlign: 'center' – wyrównanie tekstu (left, center, right).
textDecorationLine: 'underline' – podkreślenie (underline, line-through).
letterSpacing: 2 – odstęp między literami.
lineHeight: 30 – wysokość linii tekstu.

Rozmiary i pozycjonowanie

width: 100 – szerokość elementu.
height: 50 – wysokość elementu.
maxWidth: 200 – maksymalna szerokość.
maxHeight: 400 – maksymalna wysokość.
minWidth: 100 – minimalna szerokość.
minHeight: 200 – minimalna wysokość.

Marginesy i paddingi

margin: 10 – margines dookoła elementu.
marginTop: 5 – margines od góry (marginBottom, marginLeft, marginRight).
padding: 15 – wewnętrzny odstęp (paddingTop, paddingBottom, paddingLeft, paddingRight).

Obramowanie i cienie

borderWidth: 2 – grubość obramowania.
borderColor: 'black' – kolor obramowania.
borderRadius: 10 – zaokrąglenie rogów.
borderTopLeftRadius: 10 – zaokrąglenie tylko jednego rogu.
elevation: 5 – cień (tylko dla Androida).
shadowColor: 'black' – cień elementu (iOS).
shadowOpacity: 0.5 – przezroczystość cienia.

shadowOffset: { width: 2, height: 2 } – przesunięcie cienia.
shadowRadius: 3 – rozmycie cienia.

Podstawowe komponenty w React Native

React Native oferuje zestaw komponentów natywnych, które działają podobnie do elementów HTML, ale są dostosowane do aplikacji mobilnych na Androida i iOS.

Text

Text – jest używany do wyświetlania tekstu (odpowiednik <p> lub <h1>). Obsługuje stylizację, np. kolor, wielkość czcionki.

```
<Text style={{ fontSize: 20, color: 'blue' }}>To jest tekst!</Text>
```

Każdy tekst musi być wewnątrz <Text> – nie można używać zwykłego napisu w View

```
<View>Hello World!</View> // ❌ BŁĄD!
```

Button

Button (przycisk) - służy do wywoływania akcji po kliknięciu. Ma bardzo ograniczone możliwości stylizacji - jeśli chcemy zastosować bardziej zaawansowany funkcje, należy użyć komponentu Pressable!

```
import { Button, Alert } from 'react-native';

<Button title="Kliknij mnie" onPress={() => Alert.alert("Cześć!")} />
```

Pressable

Pressable to przycisk z większą kontrolą (zamiennik Button). Pozwala zmieniać styl przy kliknięciu oraz obsługuje różne interakcje (np. onPressIn, onPressOut).

```
import { Pressable, Text } from 'react-native';

<Pressable onPress={() => alert('Kliknięto!')}>
  <Text style={{ padding: 10, backgroundColor: 'blue', color:
'white' }}>Kliknij mnie</Text>
</Pressable>
```

TextInput

TextInput - pozwala użytkownikowi wprowadzać dane. Obsługuje placeholdery (informacje wyświetlane przed wprowadzeniem tekstu), walidację wejścia, wprowadzanie za pomocą klawiatury numerycznej, itp.

```
import { TextInput } from 'react-native';

<TextInput placeholder="Wpisz swoje imię" />
```

Przykład z obsługą zmiany tekstu:

```
import { useState } from 'react';
import { TextInput, Text, View } from 'react-native';
```

```
export default function App() {
  const [text, setText] = useState('');

  return (
    <View>
      <TextInput
        placeholder="Wpisz coś..."
        onChangeText={setText}
      />
      <Text>Wpisałeś: {text}</Text>
    </View>
  );
}
```

Image

Image obsługuje wyświetlanie obrazów (lokalnych, zapisanych w plikach, lub zdalnych, pobranych z internetu). Obrazy wymagają podania szerokości i wysokości.

```
import { Image } from 'react-native';

<Image source={require('./logo.png')} style={{ width: 100, height: 100
}} />
```

Przykład ładowania obrazu z internetu:

```
<Image source={{ uri: 'https://example.com/logo.png' }}
style={{ width: 100, height: 100 }} />
```

ScrollView

ScrollView pozwala na przewijanie, gdy treść jest za duża.

```
import { ScrollView, Text } from 'react-native';

<ScrollView>
  <Text>Tekst 1</Text>
  <Text>Tekst 2</Text>
  <Text>Tekst 3</Text>
</ScrollView>
```

FlatList

FlatList to efektywna lista elementów. Sprawdza się lepiej niż ScrollView w przypadku dużej ilości elementów, ponieważ renderuje tylko elementy widoczne na ekranie.

Przykład z tablicą danych:

```
import { FlatList, Text } from 'react-native';

const data = [
  { id: '1', name: 'Element 1' },
  { id: '2', name: 'Element 2' },
  { id: '3', name: 'Element 3' },
];
```

```
<FlatList
  data={data}
  keyExtractor={(item) => item.id}
  renderItem={({ item }) => <Text>{item.name}</Text>}
/>
```

Modal (popup)

Modal (okno dialogowe) pozwala na wyświetlenie okienka na ekranie (popup). Modal jest przydatny do powiadomień i formularzy.

```
import { useState } from 'react';
import { Modal, View, Text, Button } from 'react-native';

export default function App() {
  const [visible, setVisible] = useState(false);

  return (
    <View>
      <Button title="Pokaż modal" onPress={() => setVisible(true)} />
      <Modal visible={visible} animationType="slide">
        <View>
          <Text>To jest modal!</Text>
          <Button title="Zamknij" onPress={() => setVisible(false)} />
        </View>
      </Modal>
    </View>
  );
}
```