

React Native

React components Part 2



React components Part 2

- ☐ Conditional Rendering
- ☐ Components
- ☐ Functional Component
- ☐ Reusable Components

Objektif sesi

- Memahami dan mengimplementasikan Conditional Rendering
- Memahami dan mengimplementasikan Components
- Memahami dan mengimplementasikan Functional Components
- Memahami dan mengimplementasikan Reusable Components

Conditional Rendering


Conditional rendering dalam React Native components mengacu pada teknik pengaturan atau penentuan apakah suatu elemen atau komponen akan dirender atau tidak, berdasarkan suatu kondisi tertentu. Ini memungkinkan kita untuk membuat antarmuka yang dinamis dan berubah berdasarkan logika atau data tertentu.

Contoh penggunaan conditional rendering dalam React Native bisa melibatkan penggunaan statement if, else, atau operator ternary (? :)

Conditional Rendering

Menggunakan if statement:

jsx

 Copy code

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


const MyComponent = ({ isLoggedIn }) => {
  if (isLoggedIn) {
    return <Text>Welcome user!</Text>;
  } else {
    return <Text>Please log in.</Text>;
  }
};

export default MyComponent;
```

Conditional Rendering

Menggunakan operator ternary:

jsx

 Copy code

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

const MyComponent = ({ isLoggedIn }) => (
  <Text>{isLoggedIn ? 'Welcome user!' : 'Please log in.'}</Text>
);

export default MyComponent;
```

Conditional Rendering

Menyembunyikan atau menampilkan elemen berdasarkan kondisi:

jsx

Copy code

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

const MyComponent = ({ showElement }) => (
  <View>
    {showElement && <Text>This element is visible.</Text>}
  </View>
);

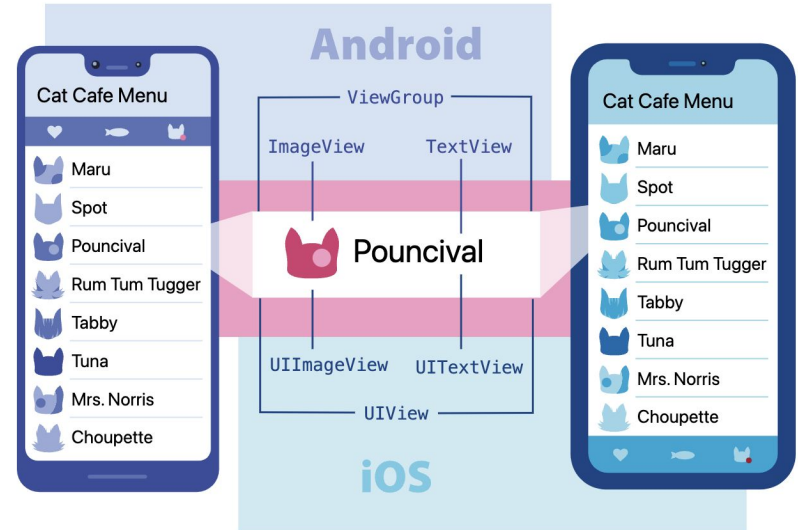
export default MyComponent;
```

Pada contoh-contoh di atas, kita menggunakan properti seperti `isLoggedIn` atau `showElement` untuk mengontrol apakah suatu elemen atau teks tertentu harus di render atau tidak.

Jika kondisinya benar, elemen akan dirender; jika tidak, elemen tersebut tidak akan ditampilkan di antarmuka pengguna. Ini memungkinkan kita untuk membuat UI yang responsif terhadap perubahan dalam data atau status aplikasi.

Components

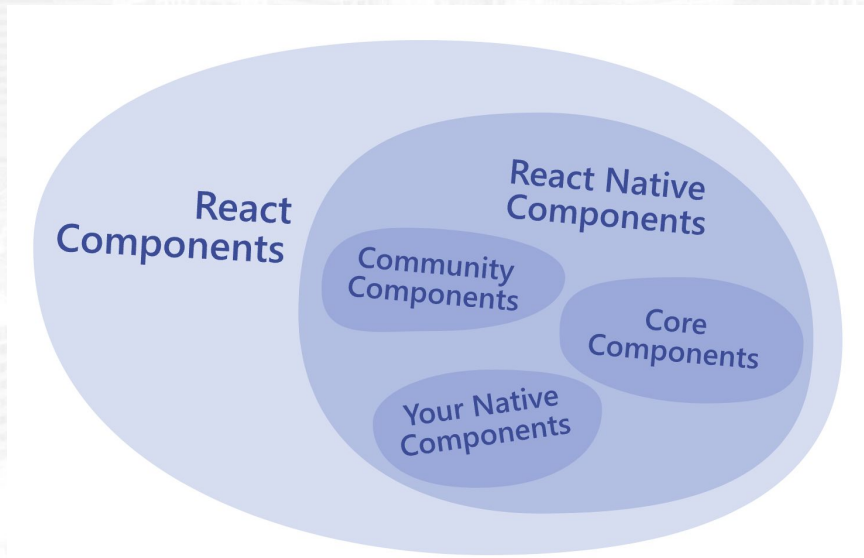
Dalam React Native, komponen (components) adalah blok bangunan dasar dari antarmuka pengguna (UI). Komponen adalah bagian dari UI yang dapat digunakan kembali, dan mereka membantu dalam menyusun aplikasi React Native menjadi struktur yang terorganisir dan mudah dikelola.



Just a sampling of the many views used in Android and iOS apps.

Native Components

Dengan React Native, kamu dapat menjalankan tampilan ini dengan JavaScript menggunakan komponen React. Saat runtime, React Native membuat tampilan Android dan iOS yang sesuai untuk komponen tersebut. Karena komponen React Native didukung oleh tampilan yang sama seperti Android dan iOS, aplikasi React Native terlihat, terasa, dan berfungsi seperti aplikasi lainnya.



Core Components

REACT NATIVE UI COMPONENT	ANDROID VIEW	IOS VIEW	WEB ANALOG	DESCRIPTION
<code><View></code>	<code><ViewGroup></code>	<code><UIView></code>	A non-scrolling <code><div></code>	A container that supports layout with flexbox, style, some touch handling, and accessibility controls
<code><Text></code>	<code><TextView></code>	<code><UITextView></code>	<code><p></code>	Displays, styles, and nests strings of text and even handles touch events
<code><Image></code>	<code><ImageView></code>	<code><UIImageView></code>	<code></code>	Displays different types of images
<code><ScrollView></code>	<code><ScrollView></code>	<code><UIScrollView></code>	<code><div></code>	A generic scrolling container that can contain multiple components and views
<code><TextInput></code>	<code><EditText></code>	<code><UITextField></code>	<code><input type="text"></code>	Allows the user to enter text

Basic Components

View

The most fundamental component for building a UI.

Text

A component for displaying text.

Image

A component for displaying images.

TextInput

A component for inputting text into the app via a keyboard.

ScrollView

Provides a scrolling container that can host multiple components and views.

StyleSheet

Provides an abstraction layer similar to CSS stylesheets.

User Interface

Button

A basic button component for handling touches that should render nicely on any platform.

Switch

Renders a boolean input.

List Views

FlatList

A component for rendering performant scrollable lists.

SectionList

Like FlatList, but for sectioned lists.

Berbeda dengan ScrollView yang lebih umum, komponen tampilan daftar berikut hanya menampilkan elemen yang sedang ditampilkan di layar. Hal ini menjadikannya pilihan yang baik untuk menampilkan daftar data yang panjang.

Android & iOS Components and APIs

Android :

BackHandler

Detect hardware button presses for back navigation.

DrawerLayoutAndroid

Renders a DrawerLayout on Android.

PermissionsAndroid

Provides access to the permissions model introduced in Android M.

ToastAndroid

Create an Android Toast alert.

iOS :

ActionSheetIOS

API to display an iOS action sheet or share sheet.

Others

ActivityIndicator

Displays a circular loading indicator.

Alert

Launches an alert dialog with the specified title and message.

Animated

A library for creating fluid, powerful animations that are easy to build and maintain.

Dimensions

Provides an interface for getting device dimensions.

KeyboardAvoidingView

Provides a view that moves out of the way of the virtual keyboard automatically.

Linking

Provides a general interface to interact with both incoming and outgoing app links.

Modal

Provides a simple way to present content above an enclosing view.

PixelRatio

Provides access to the device pixel density.

RefreshControl

This component is used inside a `ScrollView` to add pull to refresh functionality.

StatusBar

Component to control the app status bar.

Functional Components

Functional Component adalah salah satu bentuk komponen dalam React yang ditulis menggunakan sintaks fungsi (function). Functional Component merupakan bentuk yang lebih sederhana dibandingkan dengan Class Component, dan sering digunakan untuk tugas-tugas yang tidak memerlukan keadaan (state) atau lifecycle method

```
function Welcome(props) {  
  return <h1>Hello, {props.name}</h1>;  
}
```

The simplest way to define a component is to write a JavaScript function

Contoh Functional Components

```
jsx Copy code

import React from 'react';
import { View, Text } from 'react-native';

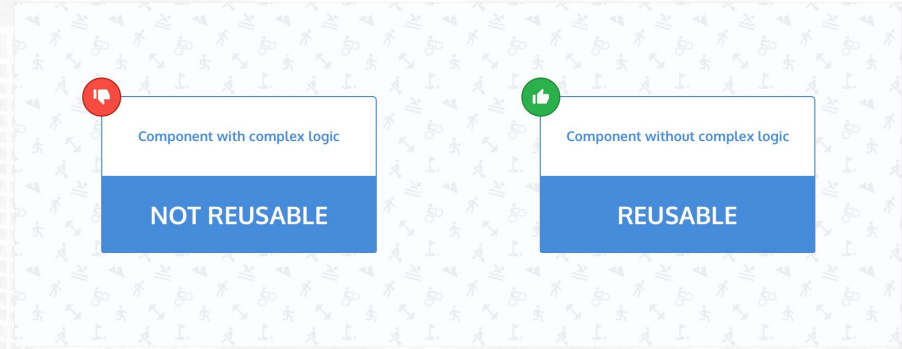
// Functional Component
const MyFunctionalComponent = () => {
  return (
    <View>
      <Text>Hello, I am a Functional Component!</Text>
    </View>
  );
};

export default MyFunctionalComponent;
```

- MyFunctionalComponent adalah sebuah Functional Component.
- `() => { ... }` adalah sintaks dari arrow function yang digunakan untuk mendefinisikan komponen fungsional.
- Dalam tubuh komponen fungsional tersebut, kita mengembalikan elemen JSX yang menggambarkan antarmuka pengguna. Pada contoh di atas, kita mengembalikan elemen View yang berisi elemen Text dengan pesan "Hello, I am a Functional Component!".
- `export default MyFunctionalComponent;` digunakan untuk mengekspor komponen agar dapat digunakan di file lain.

Reusable Components

Reusable Components merujuk pada komponen-komponen dalam pengembangan perangkat lunak yang dirancang untuk dapat digunakan kembali di berbagai bagian atau proyek. Dengan kata lain, Reusable Components dapat dengan mudah diintegrasikan ke dalam berbagai konteks tanpa perlu menulis ulang kode. Konsep ini mendukung prinsip-prinsip pengembangan perangkat lunak yang baik, seperti reusabilitas, keterbacaan, dan pemeliharaan yang mudah.



Contoh Reusable Components

Button Component:

Sebuah komponen tombol yang dapat digunakan di seluruh aplikasi untuk menciptakan konsistensi dalam desain antarmuka pengguna.

```
jsx
// Button.js
import React from 'react';
import { TouchableOpacity, Text } from 'react-native';

const Button = ({ onPress, label }) => {
  return (
    <TouchableOpacity onPress={onPress}>
      <Text>{label}</Text>
    </TouchableOpacity>
  );
};

export default Button;
```

```
jsx
// ExampleUsage.js
import React from 'react';
import { View } from 'react-native';
import Button from './Button';

const ExampleUsage = () => {
  const handleButtonPress = () => {
    console.log('Button Pressed!');
  };

  return (
    <View>
      <Button onPress={handleButtonPress} label="Click me" />
    </View>
  );
};

export default ExampleUsage;
```

Contoh Reusable Components

Card Component:

dapat digunakan untuk menampilkan informasi terstruktur di berbagai bagian aplikasi.

```
jsx

// Card.js
import React from 'react';
import { View, Text } from 'react-native';

const Card = ({ title, content }) => {
  return (
    <View>
      <Text>{title}</Text>
      <Text>{content}</Text>
    </View>
  );
};

export default Card;
```

```
jsx
Copy code

// AnotherComponent.js
import React from 'react';
import { View } from 'react-native';
import Card from './Card';

const AnotherComponent = () => {
  return (
    <View>
      <Card title="Sample Card" content="This is some content." />
    </View>
  );
};

export default AnotherComponent;
```



Referensi
<https://reactnative.dev/>



Thank you