

Pengenalan Reactjs

Imelda Doharta Aritonang



Roadmap

- Create React Project
- React Component
- JSX Template
- Props & State
- React Event Handlers
- Component Life Cycle
- CSS & Create Layout



Create React Project

- React sebagai Frontend
- React library yang mendukung antarmuka berbasis Javascript
- Menggunakan JSX (*Javascript Syntax Extension*, memungkinkan untuk menulis script HTML dalam Javascript)

```
const ele = <h1 className = {varName}>Hello!</h1>;
```
- Prinsipnya component dalam component
- Membuat SPA (Single Page Application)



Create React Project

Versi node terbaru

```
npx create-react-app my-app  
cd my-app  
npm start
```

Versi node lama

```
npm install -g create-react-app
```

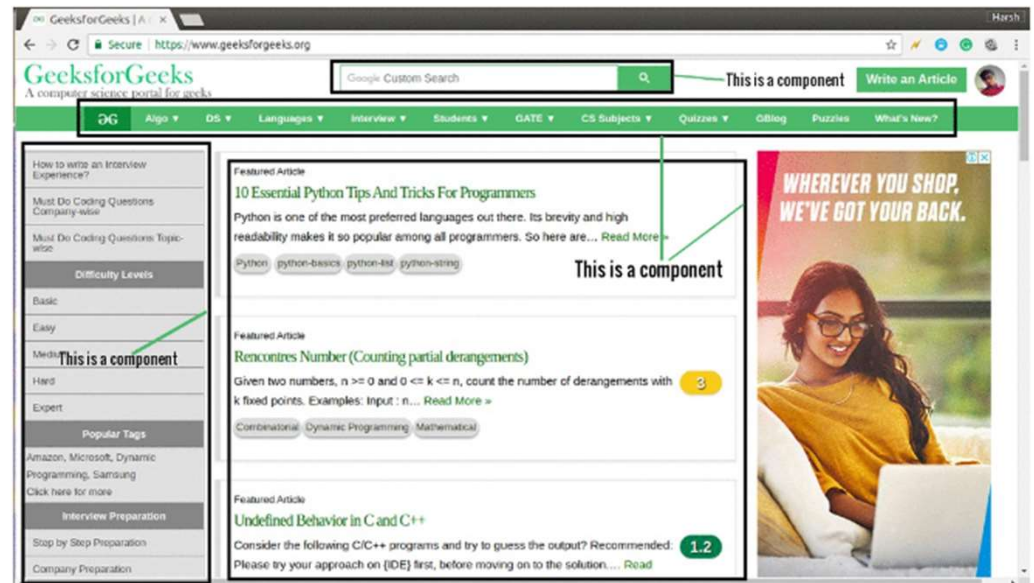
```
create-react-app myapp
```

React Component

React dibangun dari sekumpulan component

Ada 2 Jenis Component React, yaitu:

1. Class Components
2. Functional Components



Class Components

- Component yang pertama sekali ada sebelum publikasi functional component
- Lebih rumit
- Dalam class

```
import React, { Component } from 'react'
```

```
/* class component harus memiliki method  
render dan returnstatement  
*/
```

```
export default class MainComponent extends Component {  
  state = {counter : 0}  
  render() {  
    return (  
      <div>  
        <h1>Hello ReactJS</h1>  
      </div>  
    )  
  }  
}
```

Memakai method render

Harus memiliki return statement



Functional Components


- Pertama kali publish banyak kelemahan, dibuat fitur baru yaitu Hook
- Sehingga react yang menggunakan functional components disebut react hook
- Lebih sederhana daripada class components
- Dalam fungsi



Functional Components

```
import React from 'react'

// tidak perlu menggunakan render method
export default function FunctionComponent() {
  return (
    // my comment here..
    <div>
      <h1>Hello Function Component</h1>
      <h2>Hello Function Component</h2>
    </div>
  )
}
```



JSX, props dan state, React Event Handler

`const [products, setProduct] = useState(listOfProduct);` → Penggunaan state

```
{
  products.map(prod=>{
    return (
      <li>
        <CardProduct
          id={prod.id}
          name={prod.name}
          price={prod.price}
          likes={prod.likes}
          onLikes={onLikes}
        />
      )
    )
  })
}
```

↓
Mengirim data ke component lain melalui props

```
export default function CardProduct(props) {
  return (
    <div>
      <p>productid :{props.id} </p>
      <p>name : {props.name}</p>
      <p>price : {props.price}</p>
      <p>likes : {props.likes}</p>
      <button onClick={()=>props.onLikes()}>Likes</button>
      <button>Dislike</button>
    </div>
  )
}
```

↓
Membaca data yang dikirim component lain melalui props



Contoh

```
function App() {  
  return (  
    <div>  
      <Welcome name="Sara" />  
      <Welcome name="Cahal" />  
      <Welcome name="Edite" />  
    </div>  
  );  
}
```

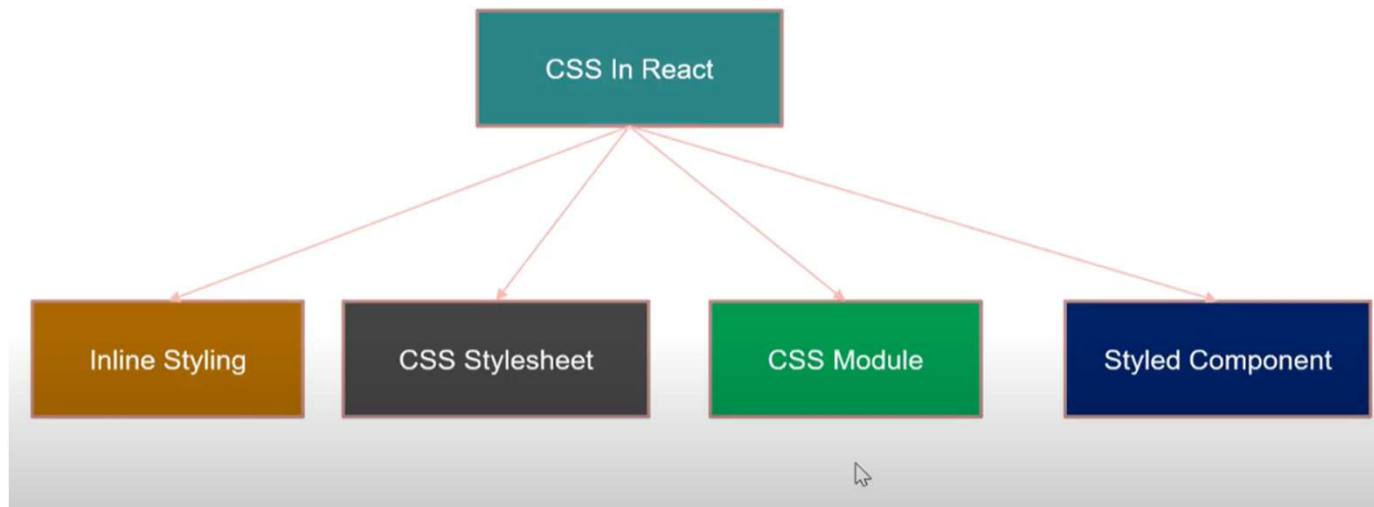
```
function Welcome(props) {  
  return <h1>Hello, {props.name}</h1>;  
}
```



Component Life Cycle

- Inisialisasi : phase inisialisasi state atau props dalam component
- Mounting : phase render output yang dibuat dalam return, lalu memproses method `componentDidMount` (class components) atau `useEffect` (functional components)
- Updating : phase memproses method `componentDidUpdate` atau `useEffect` jika ada perubahan pada state atau props
- Unmounting : phase memproses method `componentWillUnmount` atau `useEffect` jika props atau state component direset ke nilai awal

CSS dan Membuat Layout



CSS Stylesheet - inline stylesheet

```
<body>
  <h2>Udacity CSS: How to use the CSS inline style</h2>
  <hr>
  <p style="color: purple; font-weight: bold">I am sample paragraph element.</p>
</body>
</html>
```

The "style" property is added within the opening tag of the paragraph element

CSS Stylesheet - internal stylesheet

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>CSS Grid Layout</title>
    <style>
      p{
        font-size:25px;
        font-weight:bold;
        color: ■whitesmoke;
        background-color:□deeppink;
      }
    </style>
  </head>
  <body>
    <h2>Udacity CSS: How to use the CSS internal style</h2>
    <hr>
    <p>I am sample paragraph element.</p>
  </body>
</html>
```

Various CSS properties
have been set targeting the
HTML paragraph element
on the page

CSS Stylesheet - eksternal stylesheet

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>CSS Grid Layout</title>
    <link rel="stylesheet" href="CSSStylesheet.CSS">
  </head>
  <body>
    <h2>Udacity CSS: How to use the CSS External style</h2>
    <hr>
    <p>I am sample paragraph element.</p>
  </body>
</html>
```

The link tag is setup with the "stylesheet" usage.

The href points to the location of the stylesheet. As this stylesheet is local to the machine, it is referenced by name and extension alone.



CSS Module

```
import Styles from "./Style.css";

class Button extends React.Component {
  render() {
    return <span className={Styles.link}>{this.props.children}</span>
  }
}
```




Styled Component

```
import React from 'react';
import styled from 'styled-components';

const Button = styled.button`
  background-color: #008489;
  border: 1px solid #008489;
  color: #fff;
  padding: 10px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  cursor: pointer;
  margin-right: 10px;
`;

export default () =>(
  <Button> Button</Button>
);
```



Sass vs Scss

Perbedaan yg paling terlihat Sass tidak pakai semi colon. Scss akan memiliki eketension .scss, sedangkan sass .sass

Sass SYNTAX

```
$font-color: #fff  
$bg-color: #00f  
  
#box  
  color: $font-color  
  background: $bg-color
```

SCSS SYNTAX

```
$font-color: #fff;  
$bg-color: #00f;  
  
#box{  
  color: $font-color;  
  background: $bg-color;  
}
```

Dalam kedua kasus, baik itu Sass atau SCSS, kode CSS yang dikompilasi akan sama

```
#box {  
  color: #fff;  
  background: #00f;
```