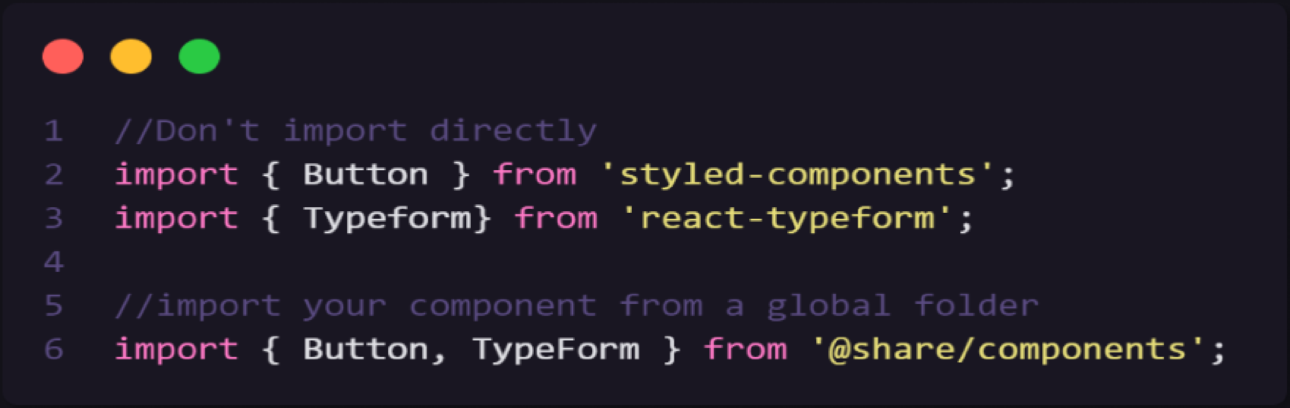
SNIPPET #4

# REACTJS BEST PRACTICE

@máriobatalhasilva

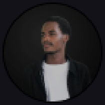
SNIPPET #4

# CENTRALIZE EXTERNAL COMPONENTS



```
//Don't import directly
import { Button } from 'styled-components';
import { Typeform} from 'react-typeform';

//import your component from a global folder
import { Button, TypeForm } from '@share/components';
```

in case you use  components from others packages or library, is a best practise centralize all the components in shared folder, because when you import in your local component, you'll not need to know  from where it comes.

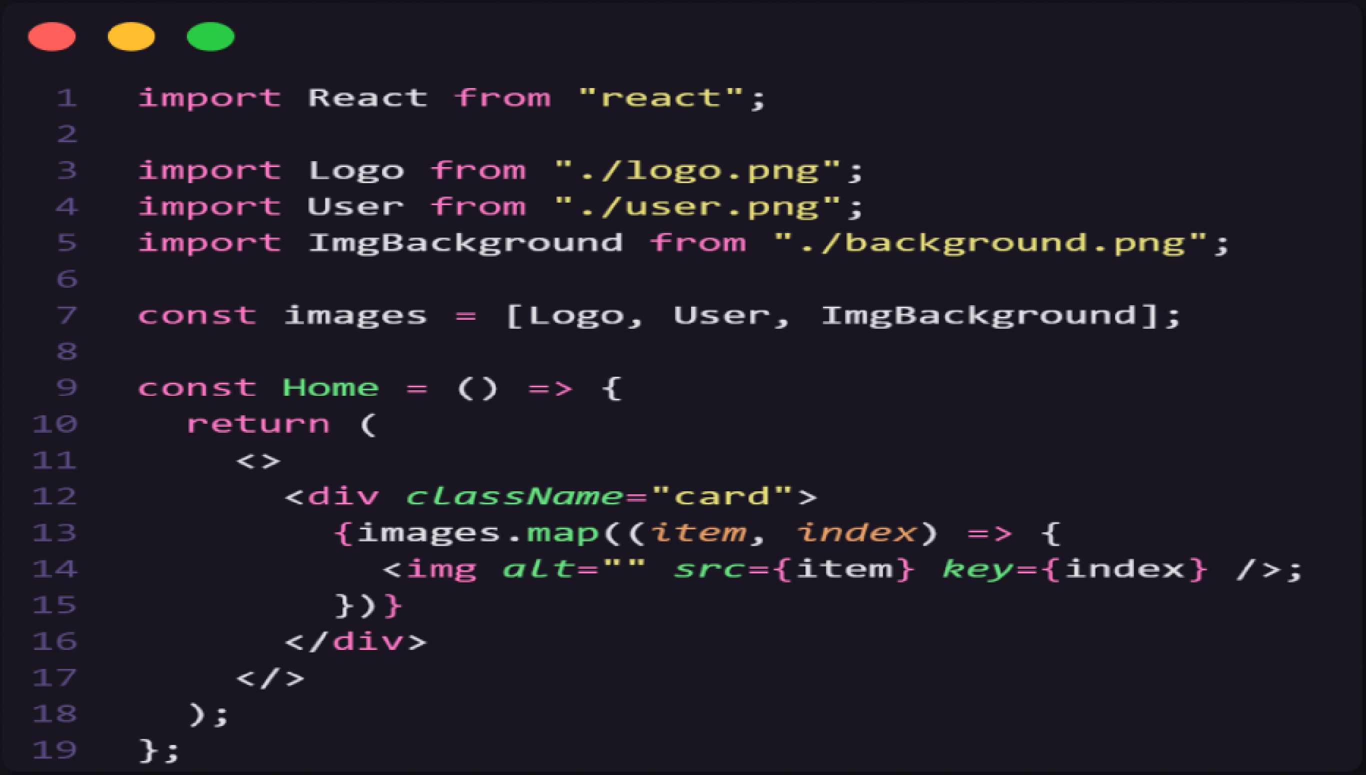


@máriobatalhasilva

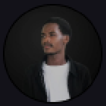
SNIPPET #4

# IMAGE ITERATION

iterate images is the most common problem when you’re import images from isolated file or folder. the best practise is import images from a shared folder or create an array of imported images.



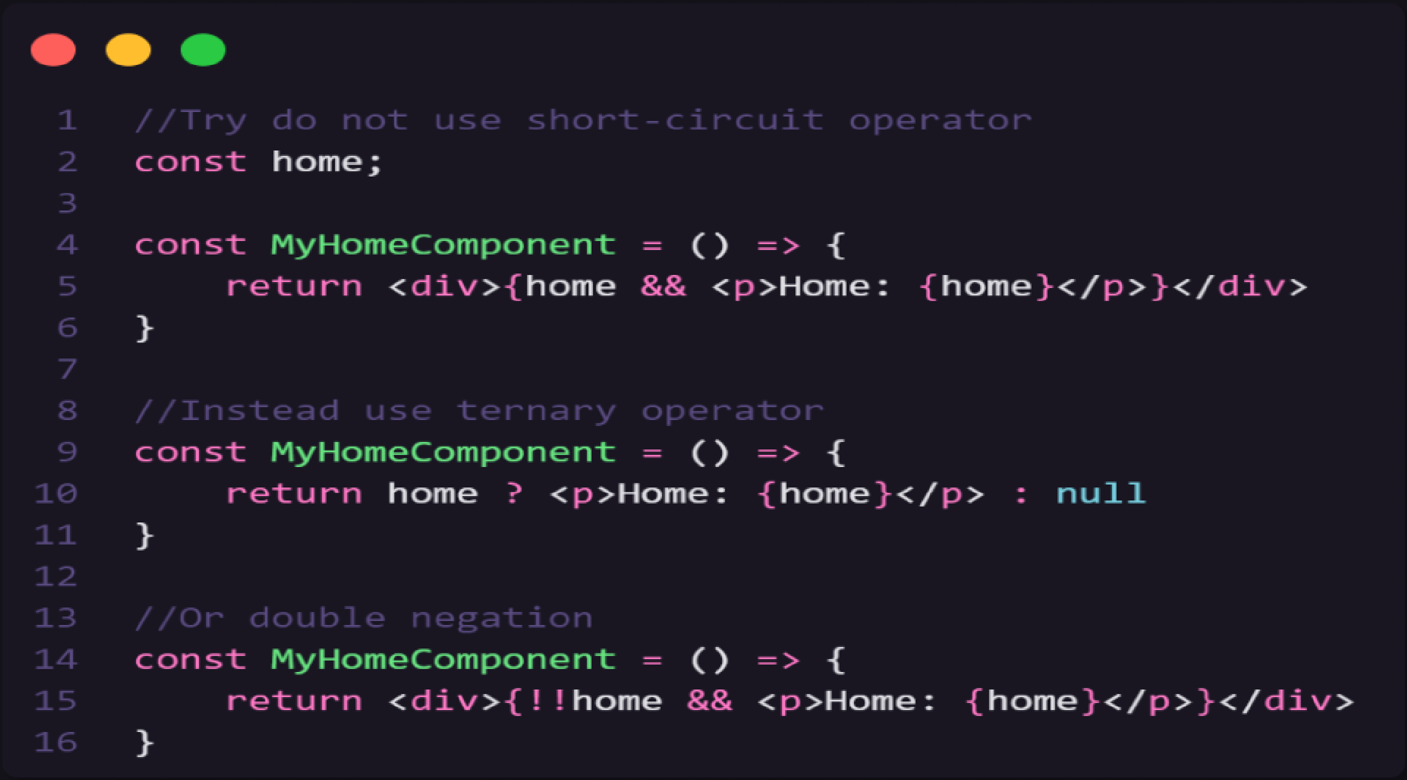
```
import React from "react";

import Logo from "./logo.png";
import User from "./user.png";
import ImgBackground from "./background.png";

const images = [Logo, User, ImgBackground];

const Home = () => {
  return (
    <>
      <div className="card">
        {images.map((item, index) => {
          <img alt="" src={item} key={index} />;
        })}
      </div>
    </>
  );
};
```



@máriobatalhasilva

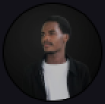
SNIPPET #4

# EVITE SHORT-CIRCUIT OPERATOR



```
//Try do not use short-circuit operator
const home;

const MyHomeComponent = () => {
    return <div>{home && <p>Home: {home}</p>}</div>
}

//Instead use ternary operator
const MyHomeComponent = () => {
    return home ? <p>Home: {home}</p> : null
}

//Or double negation
const MyHomeComponent = () => {
    return <div>{!!home && <p>Home: {home}</p>}</div>
}
```

short-circuit operator &&, sometimes can return 0 as value. it’s happen because react doesn’t treat 0 as false, for that we have to use ternary operator or double negation to solve our problem.



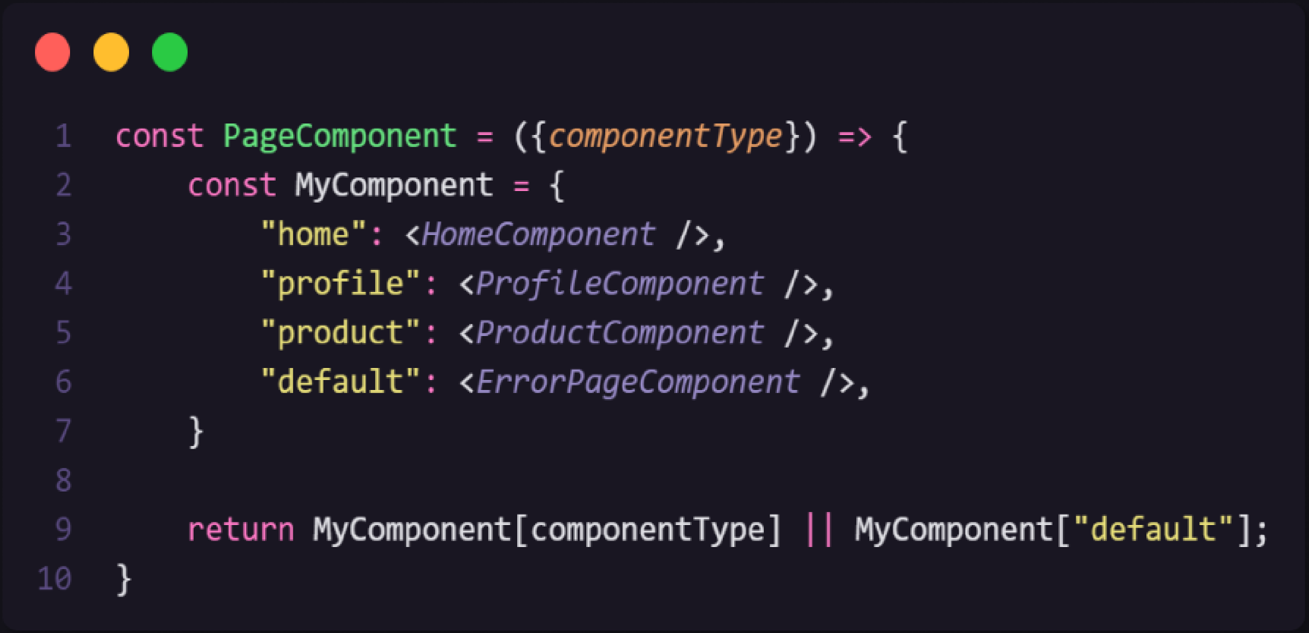
@máriobatalhasilva

SNIPPET #4

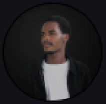
# OBJECT LITERALS INSTEAD OF SWITCH

most case when we need to use conditional statement  we prefer to use if/else or switch/case. the problem with switch/case is that  it’s doesn’t perform so well.

object literals is more simple, and we can use for example to instantiate a component.



```
const PageComponent = ({componentType}) => {
    const MyComponent = {
        "home": <HomeComponent />,
        "profile": <ProfileComponent />,
        "product": <ProductComponent />,
        "default": <ErrorPageComponent />,
    }

    return MyComponent[componentType] || MyComponent["default"];
}
```



@máriobatalhasilva

# SHARE AND LEAVE A COMMENT