

Student's Name

Instructor's Name

Course

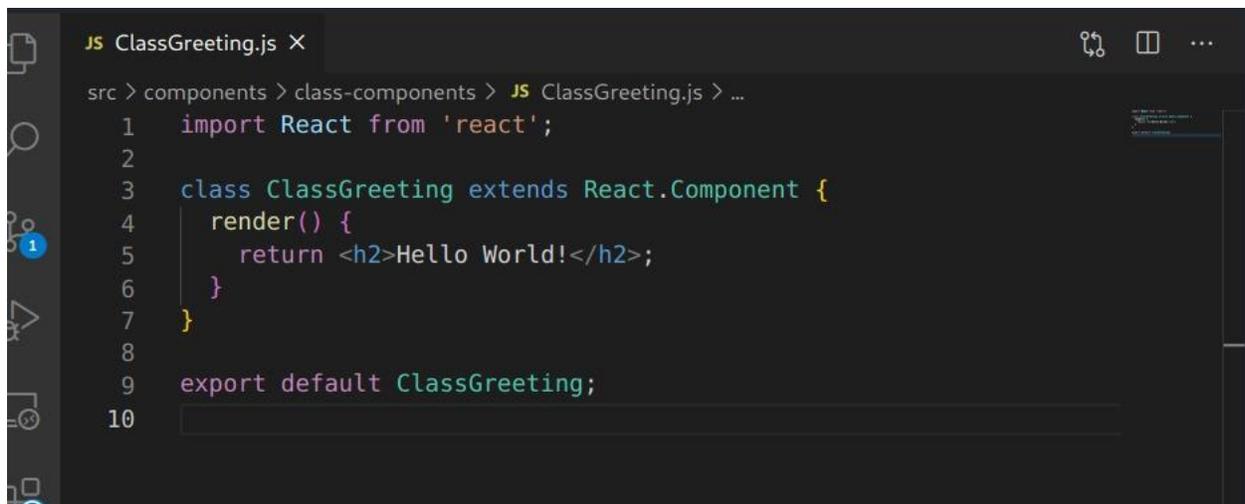
Date

## React Class and Functional Components

Components are building blocks for React applications. They are JavaScript functions that accept props (properties - arbitrary inputs) and return React elements describing the JSX rendered on the DOM (Azzam). Generally, React components define parts of the User Interface (UI) reusable. React provides two types of components: class components and functional components. Class components allow an imperative approach in coding, while functional components allow a declarative approach.

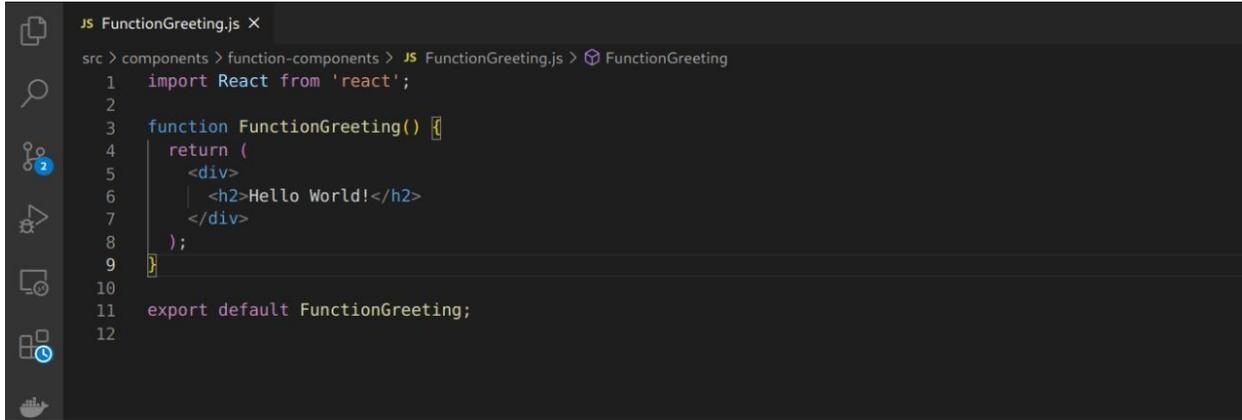
### Syntax

Class-based and functional components differ in their syntax. For instance, a class-based component is defined as shown below:



```
JS ClassGreeting.js ×
src > components > class-components > JS ClassGreeting.js > ...
1  import React from 'react';
2
3  class ClassGreeting extends React.Component {
4    render() {
5      return <h2>Hello World!</h2>;
6    }
7  }
8
9  export default ClassGreeting;
10
```

Functional components provide two ways to define the same component. One could define React functional components using either a classic function or an arrow function (Azzam). The code below shows an example of a functional component.

A screenshot of a code editor window titled 'JS FunctionGreeting.js'. The editor shows the following code:

```
1 import React from 'react';
2
3 function FunctionGreeting() {
4   return (
5     <div>
6       <h2>Hello World!</h2>
7     </div>
8   );
9 }
10
11 export default FunctionGreeting;
12
```

The code is displayed in a dark-themed editor with syntax highlighting. The file path in the breadcrumb is 'src > components > function-components > JS FunctionGreeting.js > FunctionGreeting'.

## React Props

React props pass variables to an application's components. They are used in the same way for both class-based and functional components (Azzam). However, these props are defined differently for each type. The code below shows how props are used in class-based components.

```
import React from "react";

class ClassGreeting extends React.Component {

  render() {

    return <h1> Hello, {this.props.name}</h1>

  }

}

export default ClassGreeting;
```

In functional components, props are defined as an argument of a function. The keyword *this* is not used. The following code shows how props are used in functional components.

```
import React from "react";

const FunctionGreeting = props => <h1> Hello,

{props.name}</h1>;

export default FunctionGreeting;
```

## State Management

There is a significant difference in the internal state management between class-based and functional components. In class-based components, the state is defined as an object within the class's constructor (Azzam). The state value is accessed through the command *this.state.name*. The component state cannot be mutated directly. Therefore, the component state is changed using *this.setState*.

Functional components use hooks for internal state management. For instance, the same case is implemented using the *useState* hook to define the variable *name* and a function *setName*. The default value is passed to *useState* as an argument (Azzam). The state cannot be directly accessed from outside the component. However, it is passed to other components as a prop. Generally, functional components make code segments shorter than class-based components. Besides, they increase an application's general performance.

Works Cited

Azzam, Anna. "React Functional vs. Class Components." *Medium*, 21 Jan. 2021,  
<https://medium.com/swlh/react-functional-vs-class-components-517be1d9ff13>. Accessed  
4 Jan. 2022.