1. Creating Simple React Components

Summary:
React components are the building blocks of any React application. A simple com

Details:
Functional components are the most basic way to define a component in React. Th

These components are reusable and can be nested inside other components to buil

Code Example:
```
const Welcome = () => {
  return <h1>Hello, world!</h1>;
};
```

2. Using JSX to Embed JavaScript in HTML

Summary:
JSX allows you to write HTML-like syntax directly in JavaScript, and you can emb

Details:
JSX makes it easier to visualize the UI structure while keeping logic close. You

This allows dynamic rendering of content based on variables or state.

Code Example:
```
const Greeting = ({ name }) => {
  return <h2>Hello, {name}!</h2>;
};
```

3. Rendering a List of Items Using .map()

Summary:
To dynamically render multiple elements, you can use JavaScript's .map() functi

Details:
The .map() function iterates over an array and returns a new array of elements.

Each item should have a unique key prop to help React manage updates efficientl

Code Example:
```
const ItemList = ({ items }) => {
  return (
    <ul>
      {items.map((item, index) => <li key={index}>{item}</li>)}
    </ul>
  );
};
```

4. Understanding Component Reuse (Instantiation)

Summary:
React components can be reused multiple times with different props.

Details:
Component reuse allows you to create modular and maintainable code. By passing

This is a key feature of React's declarative and component-based architecture.

Code Example:
```
<Greeting name="Gregory" />
<Greeting name="Alex" />
```