
Press Esc to exit full screen
SLIDE
React Core Concepts

1. JavaScript XML (JSX)

JSX is a syntax extension for JavaScript that lets you write HTML-like markup inside a JavaScript file.

Rules you should know:

- To return multiple elements from a component, wrap them with a single parent tag like or <></>
- "class" attribute becomes "className"
- Properties are written in camelCase. Example, onclick → onClick, tabindex → tabIndex
- Tags must be closed: <h1></h1>, <img/>

2. Component?

- Components are the foundation upon which you build user interfaces (UI)
- In a React app, every piece of UI is a component.
- React components are regular JavaScript functions except:

  1) Their names always begin with a capital letter.
  2) They return JSX markup.

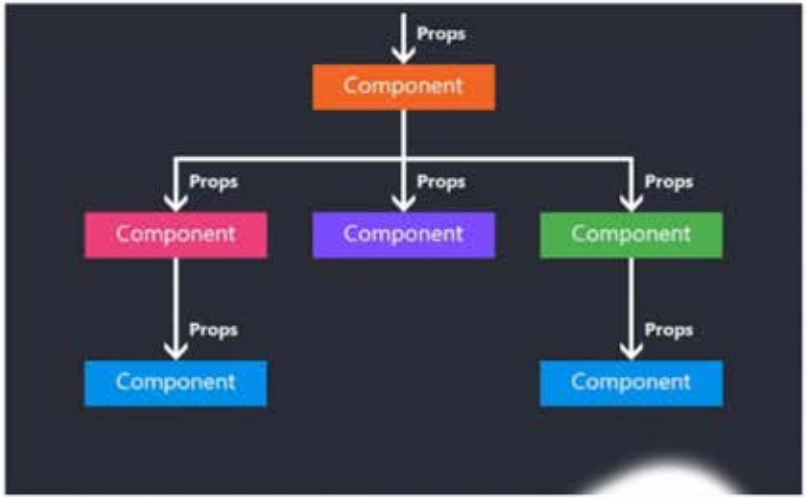
How to build a component?

- Define and export the function
- Add markup. For example:

```
return (
  <div>
    <img src="https://i.imgur.com/MK3eW3As.jpg" alt="Katherine Johnson" />
  </div>
);
```

3. Props

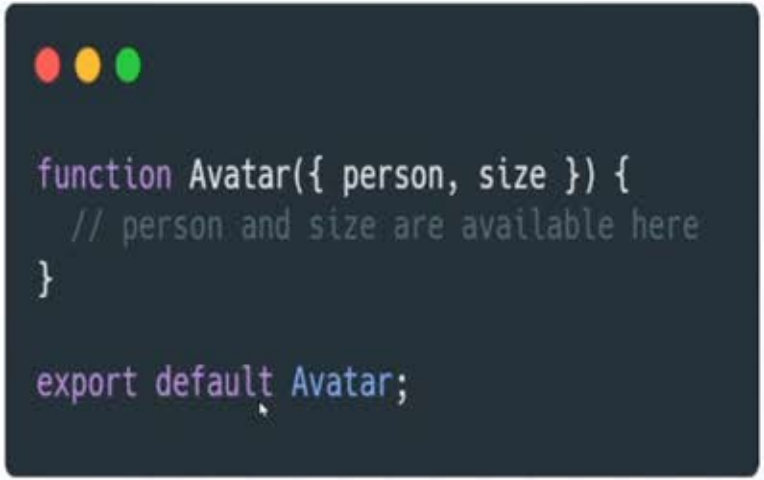
- React components use props to communicate with each other.
- Every parent component can pass some information to its child components by giving them props.
- You can pass any JavaScript value through props, including objects, arrays, and functions.



Rules of Props

```
function Profile() {

  return (

    <Avatar
      person={{ name: 'Lin Lanying', imageId: '1bX5QH6' }}
      size={100}
    />
  );

}

export default Profile;
```

Pass props to the child component



```
function Avatar({ person, size }) {
  // person and size are available here
}

export default Avatar;
```

Read props inside the child com

4. useState

useState is a React Hook that lets you add a state variable to your component.

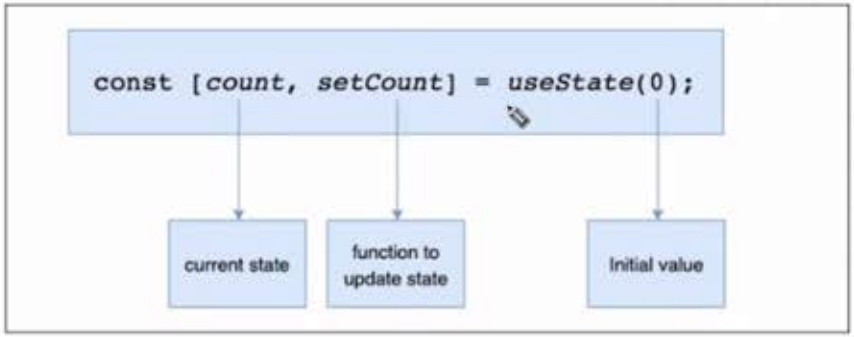
Syntax:

```
const [state, setState] = useState(initialState)
```

useState

useState returns an array with exactly two values:

- The current state. During the first render, it will match the initialState you have passed.
- The set function that lets you update the state to a different value and trigger a re-render.



5. Event Handlers

In React, when an event handler updates the state of a component, it triggers a re-render of the component and its children.

```jsx
import React, { useState } from 'react';

function MyComponent() {
  const [count, setCount] = useState(0);

  const handleClick = () => {
    setCount(count + 1);
  };

  return (
    <div>
      <p>Count: {count}</p>
      <button onClick={handleClick}>Increment</button>
    </div>
  );
}
```

The handleClick function updates the count state by calling the setCount function with a new value. This triggers a re-render of the component and updates the DOM to display the new count.

6. useEffect

- The useEffect Hook allows you to perform side effects in your components. One example of side effects are: fetching data

useEffect

- callback function: This function can perform any side effects, such as fetching data from an API
- dependencies: Dependencies is an optional array of values that the callback function relies on


SLIDE
We will learn more about React
This is just the beginning