React and Its High-Level View

Dr Harshad Prajapati
27 Nov 2023

From JavaScript to React

How we Add Elements at Runtime using JavaScript

- Suppose, we have an element in HTML:
  - H
- We can add an element/node dynamically using JavaScript DOM API:
  - const h1 = document.createElement("h1");
    const textNode = document.createTextNode("Hello World!");
    h1.appendChild(textNode);
    const mydiv = document.getElementById('mydiv');
    mydiv.appendChild(h1);

How we Add Elements at Runtime using JavaScript

- We get output as

The H in output is generated using this div

H

This is added dynamically to with id=myDiv.

How we Add Elements at Runtime using JavaScript

- On inspecting the elements in browser we find:

```
▼<div id="mydiv">
   "H"
   <h1>Hello World!</h1> == $0
 </div>
▶<script type="text/babel">…</script>
```

This is added dynamically.

We tell step by step how to do

JavaScript code that adds <h1> element dynamically is

Create new element <h1>

Append new element <h1> under existing

```
const h1 = document.createElement("h1");
const textNode = document.createTextNode("Hello World!");
h1.appendChild(textNode);

const mydiv = document.getElementById('mydiv');
mydiv.appendChild(h1);
```

Vanilla JavaScript Approach is Imperative

How React renders UI components?

- React simplifies this dynamic DOM manipulation.
- Suppose, we have an element in HTML.
  -
- Using React code, we can render a Hello UI component in this div using
  - ReactDOM.render(<Hello />, document.getElementById('mydiv'));

react-page-1.html

react-page-1.html

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>React Standalone</title>
7     <!-- Add these script tags for React and ReactDOM -->
8     <script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
9     <script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script
10    <!-- Add Babel standalone for JSX transpilation -->
11    <script crossorigin src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
12  </head>
13  <body>
14    <h1>React Standalone</h1>
15    <!-- Your React code will go here -->
16    <div id="root"></div>
17
```

react and react-dom

babel for JSX

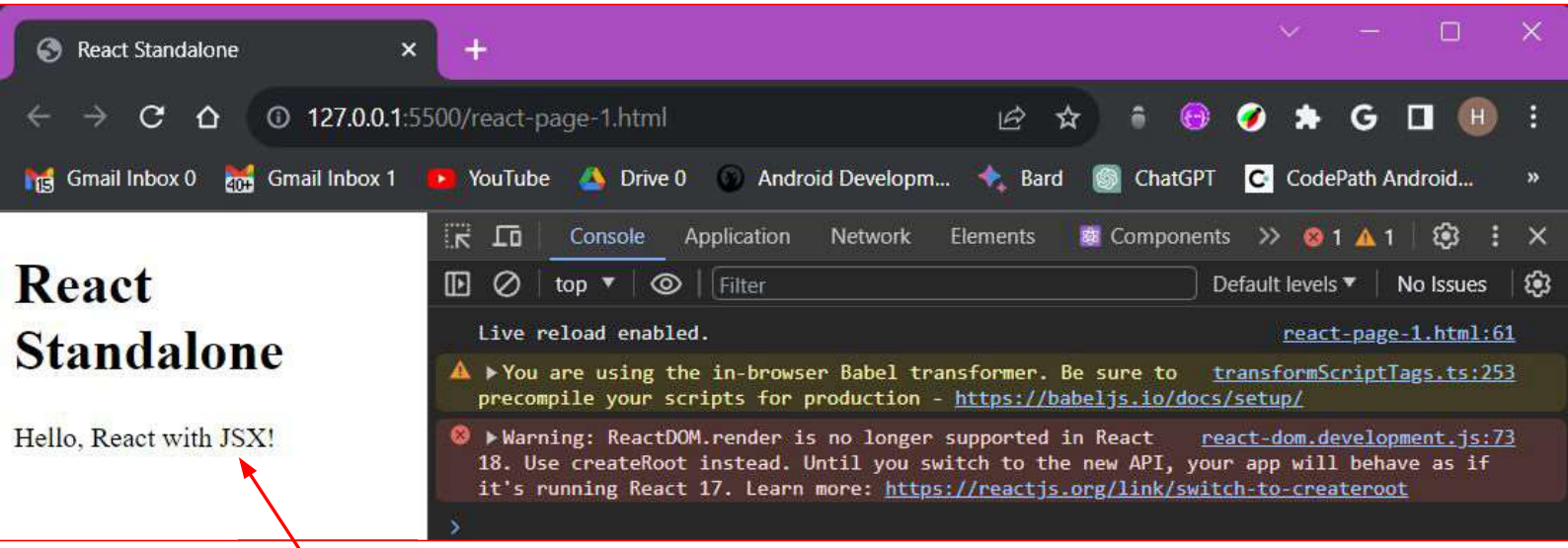
react-page-1.html

```
18    <!-- Your JSX code -->
19    <script type="text/babel">
20      // Your React component with JSX syntax
21      const App = (
22        <div>
23          Hello, React with JSX!
24        </div>
25      );
26
27      // Render the React component
28      ReactDOM.render(
29        App,
30        document.getElementById('root')
31      );
32    </script>
33  </body>
34  </html>
```

Browser does not understand JSX, so babel is required.

App variable has JSX

Render the content of App variable



Created by App variable.

react-page-2.html

react-page-2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>React Standalone</title>
  <!-- Add these script tags for React and ReactDOM -->
  <script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
  <script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
  <!-- Add Babel standalone for JSX transpilation -->
  <script crossorigin src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
</head>
<body>
  <h1>React Standalone</h1>
  <!-- Your React code will go here -->
  <div id="root"></div>
```

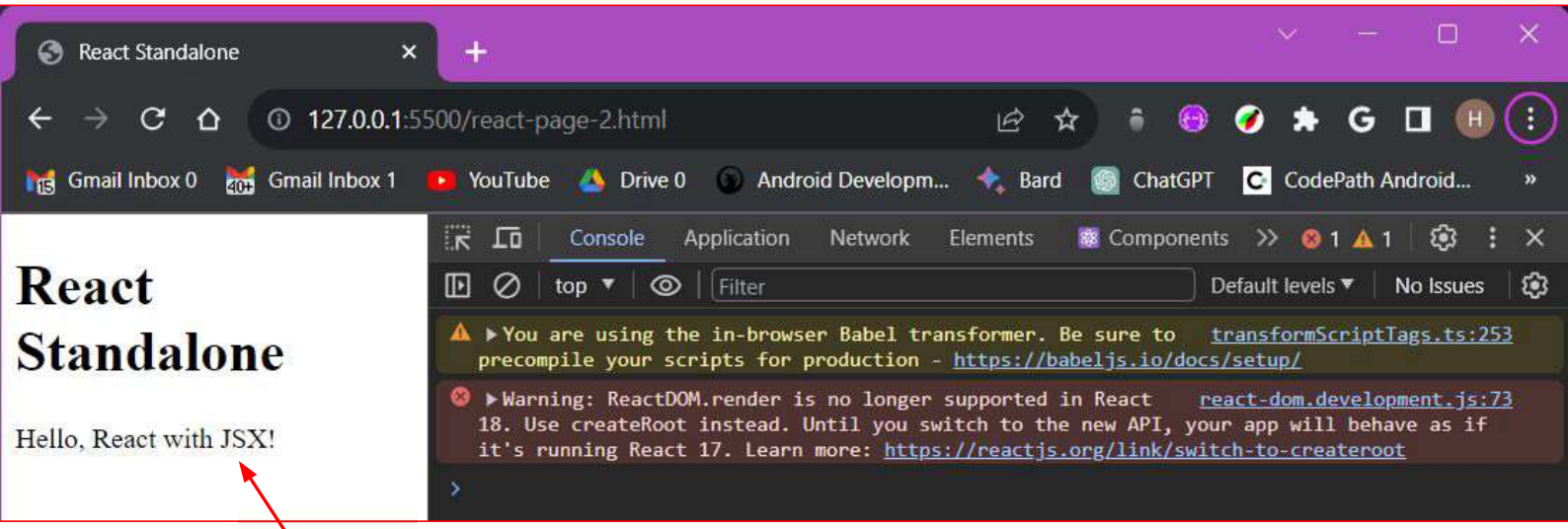

react-page-2.html

```
  <!-- Your JSX code -->
  <script type="text/babel">
    // Your React component with JSX syntax
    function App() {
      return (
        <div>
        Hello, React with JSX!
      </div>
      )
    }

    // Render the React component
    ReactDOM.render(
      App(),
      document.getElementById('root')
    );
  </script>
</body>
</html>
```

App is a function that returns JSX

Call App function.



Created by App function.

react-page-3.html

react-page-3.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>React Standalone</title>
  <!-- Add these script tags for React and ReactDOM -->
  <script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
  <script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
  <!-- Add Babel standalone for JSX transpilation -->
  <script crossorigin src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
</head>
<body>
  <h1>React Standalone</h1>
  <!-- Your React code will go here -->
  <div id="root"></div>

```

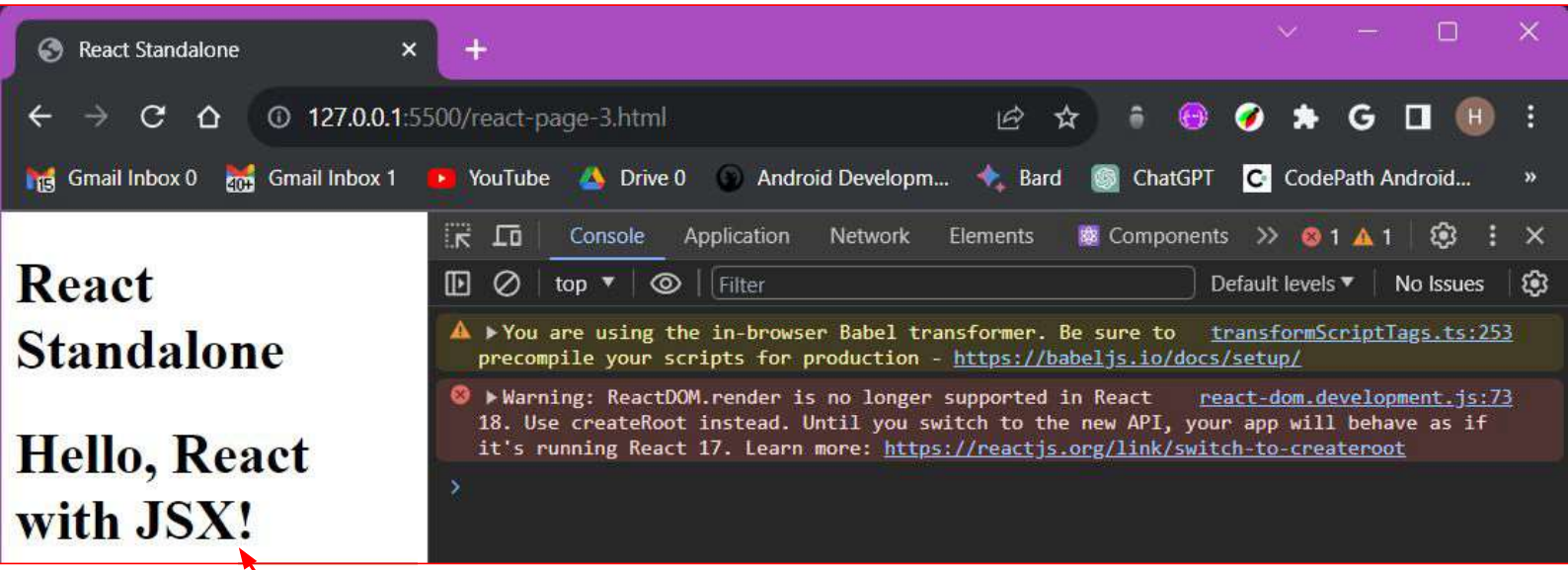
react-page-3.html

```
18    <!-- Your JSX code -->
19    <script type="text/babel">
20      // Your React component with JSX syntax
21      const App = () => {
22        return (
23          <h1>Hello, React with JSX!</h1>
24        );
25      };
26
27      // Render the React component
28      ReactDOM.render(
29        <App />,
30        document.getElementById('root')
31      );
32    </script>
33  </body>
34  </html>
```

App is a function that returns JSX.
That is React component

Use App (Component) with an HTML tag syntax.
React internally can call the App() function in place of <App />

React Approach is Declarative



Created by App (React) component.

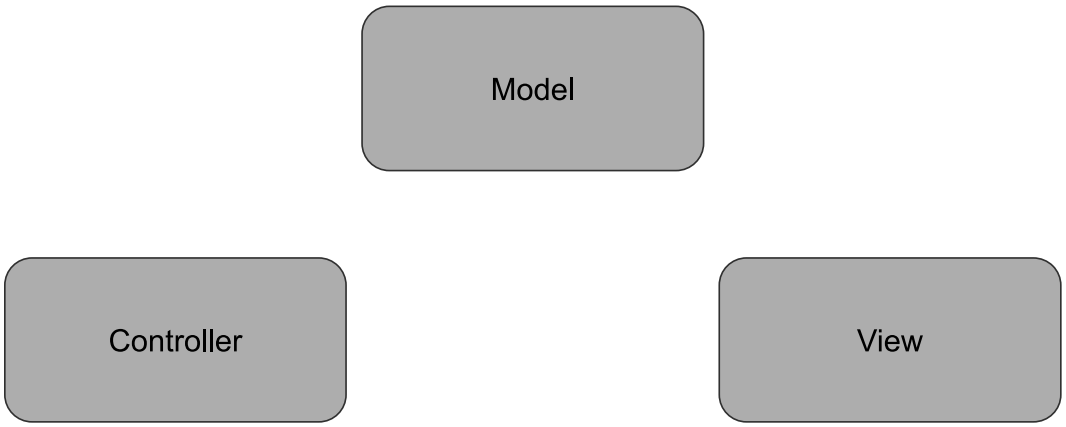
High-Level View of React

Is React a framework?

- No, React is a JavaScript **library** from **Facebook**.
  - **Initially** Facebook used **internally**, then made **publicly available**.

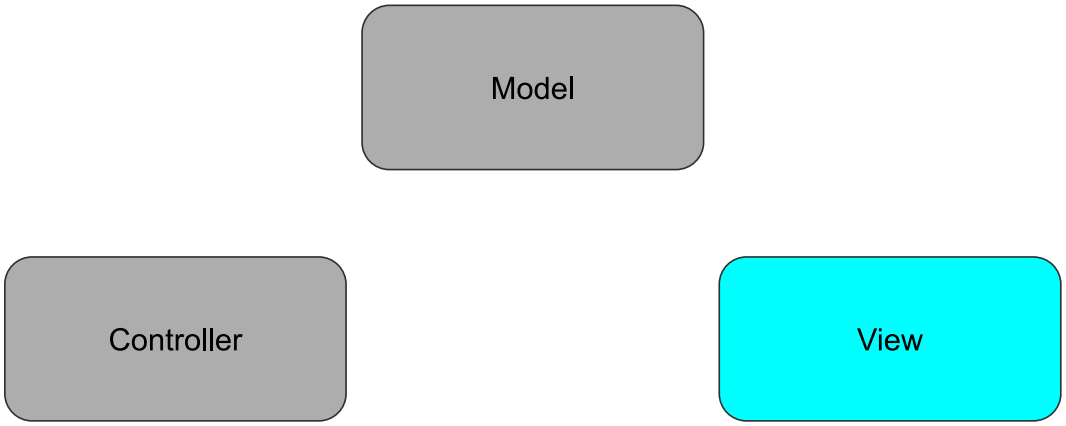
Source: https://react.dev/

React is NOT an MVC Framework



React is NOT an MVC Framework

- React is a JavaScript library used to create UI components.



React Library is

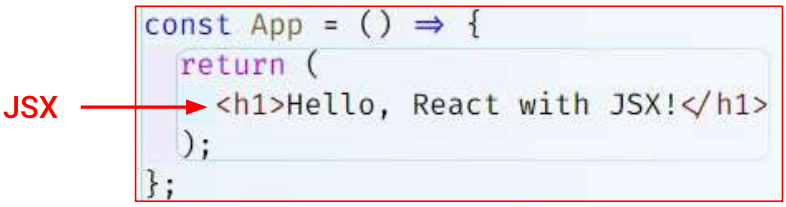
- Component-based
- Declarative
- Efficient
- Flexible

What React allows us to do?

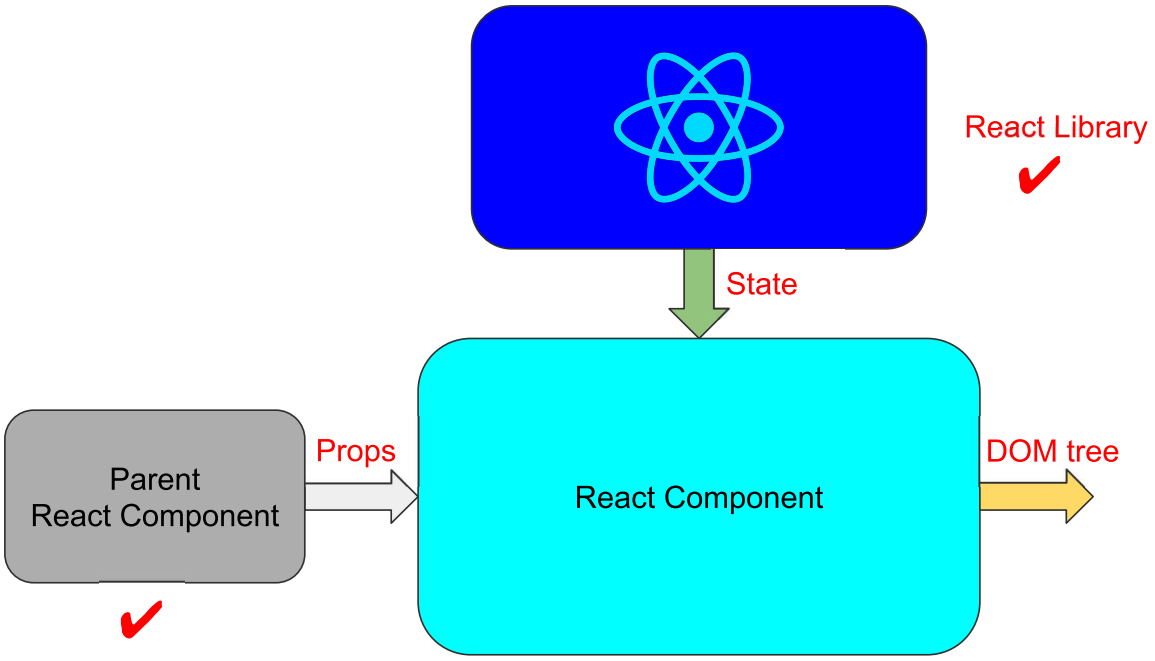
- React allows us to create small, reusable, isolated UI components.
- These smaller components can be composed to make higher level components.
- React apps are made using such components.
  - A component is a piece of UI that has its own:
    - Logic (JavaScript, Event Handling, DOM manipulation)
    - Appearance (HTML and styles)

React Components at Present

- Now, React components are JavaScript functions that return Markup.



React Component and its Control



Each component's state is maintained by React

What is JSX?

- JSX: JSX is an extension created by Facebook that adds XML syntax to JavaScript.
- Browser does not understand, JSX.
  - JSX can be converted into JavaScript using Babel translator.
  - Babel: Babel is also used to convert advanced JavaScript concepts into what browser can understand.

React Components

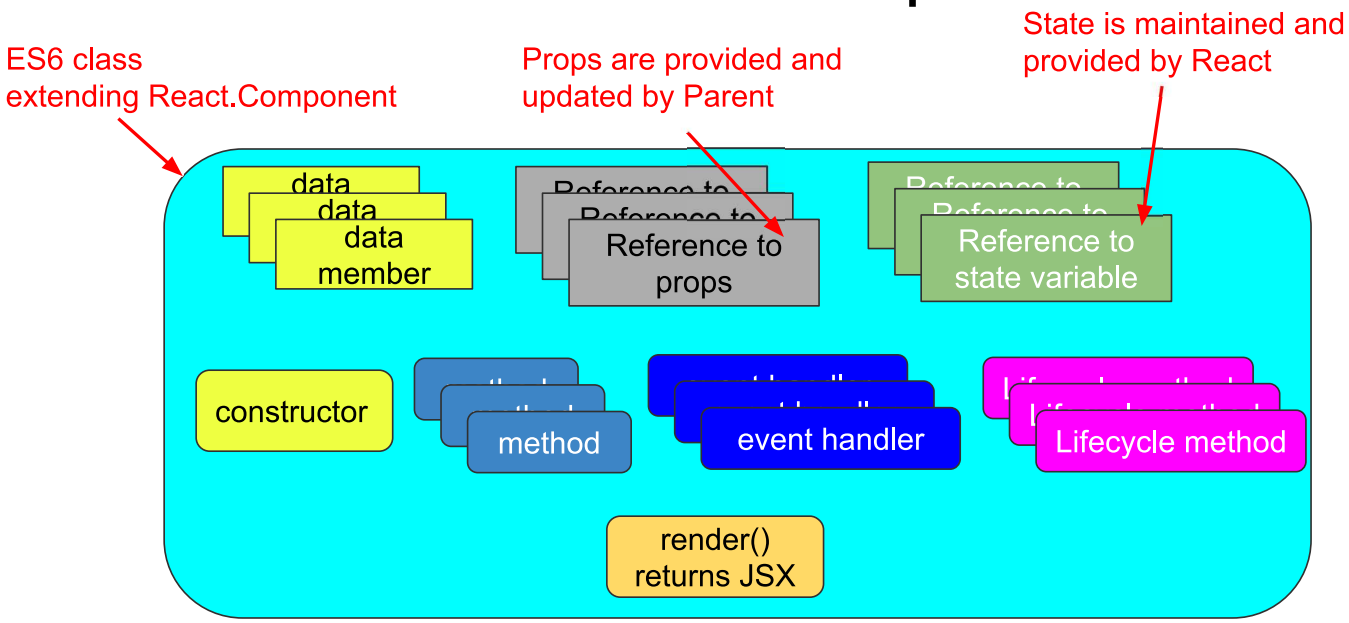
How to create React Component

- There are two ways:
  - Class based (Using ES6 class).
  - Functional component (Using function and closure).

How to create Class-based React Component

- Class based component has
  - Lifecycle methods.
  - constructor, accepts props (properties) (this.props).
  - render() method.
  - state object (this.state).
- Class based component cannot use Hooks.

What does a class based React Component can contain?

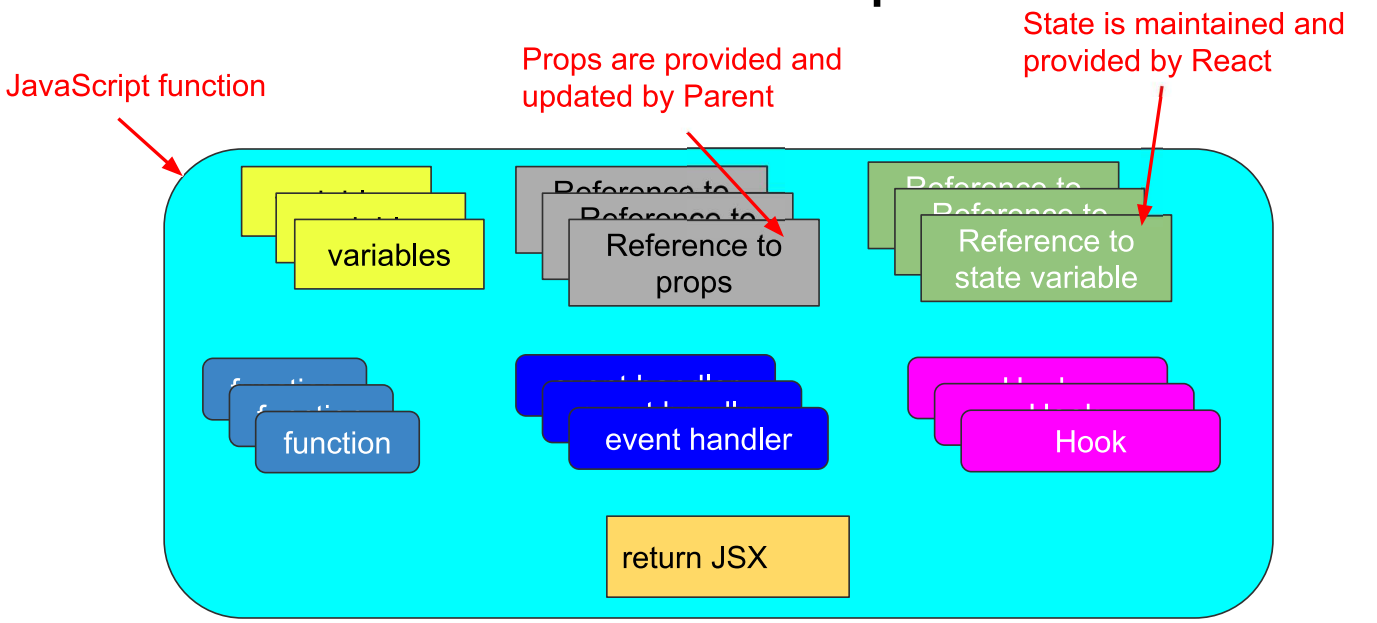


React Component is a standalone unit, and can respond to events

How to create Functional React Component

- Functional component is a **JavaScript function**. (Generally a **closure**)
- It can **create** and **use**:
  - variables.
  - functions **within itself** (function).
  - event handlers **within itself** (function).
- It can **use**:
  - **hooks**.
  - **props**.
  - **state**.

What does a Functional React Component can Contain?



React Component is a standalone unit, and can respond to events

References

- https://legacy.reactjs.org/docs/add-react-to-a-website.html
- https://legacy.reactjs.org/docs/hello-world.html
- https://legacy.reactjs.org/docs/introducing-jsx.html
- https://react.dev/