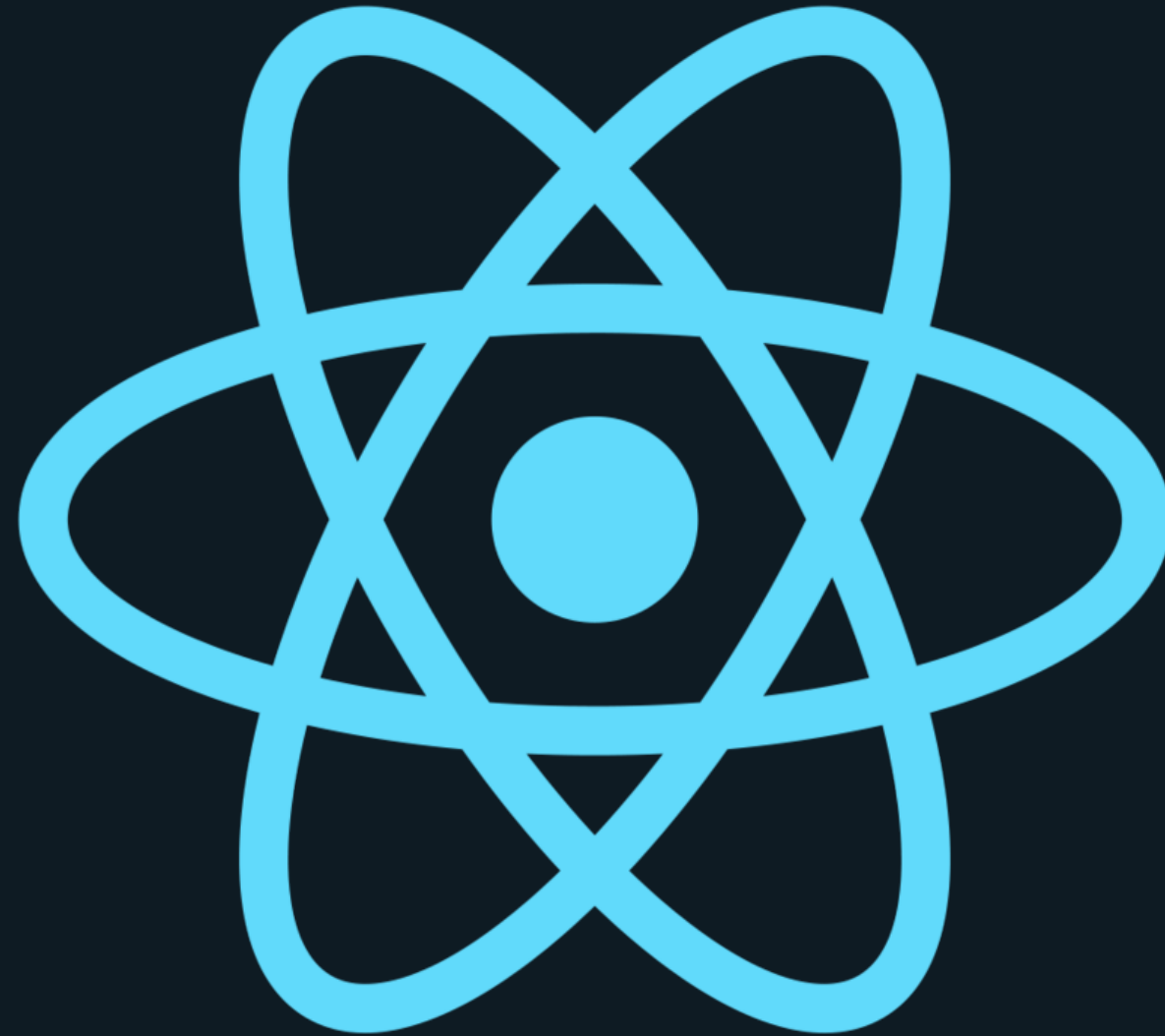


Learn to Code - React!



Objectives

- Describe Single Page Applications (SPA)
- Name the key concepts in React
- Explain components, props, and JSX
- Identify functional and class components
- Know when to use state vs setState
- Lift state up to the closest common ancestor

Describe Single Page Applications (SPA)

In a single-page application, all of the code needed is included in a single-page load. This includes all of the CSS, HTML, and JavaScript. The page does not reload, and it requires dynamic interaction with the Web server.

Name the key concepts in React

- Web Components
- 1 way data binding
- Virtual DOM
- JSX (which requires a "build" step)
- State management (Flux or Redux)

Components

- Components let you split the UI into independent, reusable pieces, and think about each piece in isolation.

Identify "container" (smart) components

The screenshot displays the 'Learn Galvanize' web application interface. The browser's address bar shows the URL `https://learn.galvanize.com/cohorts/201/units/2414/content_files/57681`. The application layout is divided into three main sections, each highlighted with a red box and a label:

- Header Component:** The top navigation bar, which includes the 'galvanize LEARN' logo, a user profile for 'JEFF DEAN', and a dropdown menu for 'COHORT: REACT'. Below this is a secondary navigation bar with links to 'DASHBOARD', 'UNITS', 'ACTIVITY', 'PERFORMANCES', 'STUDENTS', 'CURRICULUM', and 'COHORT SETUP'.
- Sidebar Component:** A vertical sidebar on the left side of the page. It features a 'NEXT: UNIT 4' button at the top, followed by 'UNIT 3 Components'. A list of items is shown, including 'Unit Overview', 'LESSON 01 React Components', 'EXERCISE 01 Shopping Cart: Header / Footer', 'LESSON 02 Props', 'EXERCISE 02 Shopping Cart: Copyright with Props', and 'LESSON 03 JSX Collections'. At the bottom, there is a 'HIDE SIDEBAR' button.
- Content Component:** The main content area on the right. It displays 'LESSON 11' and the title 'Identifying Components'. A 'Github' button is located in the top right corner of this section. Below the title, a horizontal line separates the header from the main text. The text states: 'By the end of this lesson you should be able to:' followed by a numbered list:
 1. Take a wireframe or design and identify possible components
 2. Identify list componentsAt the bottom of the content area, there are navigation links: '← RESOURCES' on the left and 'NEXT: JSX REFERENCE →' on the right.

Identify "presentation" (dumb) components

The screenshot shows the Galvanize Learn web application. Red boxes and arrows highlight several UI elements identified as presentation components:

- Navigation Bar:** Includes the "galvanize LEARN" logo, user profile "JEFF DEAN", and a menu with items like "DASHBOARD", "UNITS", "ACTIVITY", "PERFORMANCES", "STUDENTS", "CURRICULUM", and "COHORT SETUP".
- Unit Navigation:** A "NEXT: UNIT 4" button with left and right arrows.
- Unit Sidebar:** A list of units and exercises on the left side, including "UNIT 3 Components", "LESSON 01 React Components", "EXERCISE 01 Shopping Cart: Header / Footer", "LESSON 02 Props", "EXERCISE 02 Shopping Cart: Copyright with Props", and "LESSON 03 JSX Collections".
- Lesson Content:** The main area for "LESSON 11 Identifying Components", which includes a "Unit Overview" section and a list of tasks: "1. Take a wireframe or design and identify possible components" and "2. Identify list components".
- Footer/Bottom Bar:** Includes a "RESOURCES" link and a "NEXT: JSX REFERENCE" button.

Red arrows point from labels to the corresponding components:

- DropdownMenu Component:** Points to the "COHORT: REACT" dropdown and the "Github" dropdown.
- ForwardButton Component:** Points to the "NEXT: UNIT 4" button.
- LinkList Component:** Points to the list of units and exercises in the sidebar.
- Link Components:** Points to the "EXERCISE 01" and "EXERCISE 02" links in the sidebar.

Pass data as props and render dynamic values in JSX

— Props are Read-Only

```
class Welcome extends React.Component {  
  render() {  
    return <h1>Hello, {this.props.name}</h1>;  
  }  
}
```


Differentiate between functional and class components

```
function Welcome(props) {  
  return <h1>Hello, {props.name}</h1>;  
}
```

VS

```
class Welcome extends React.Component {  
  render() {  
    return <h1>Hello, {this.props.name}</h1>;  
  }  
}
```

Know when to use state vs setState

— Use state to assign initial state in a constructor

```
constructor(props) {  
  super(props)  
  this.state = {  
    cards: CardData  
  }  
}
```

Know when to use state vs setState

- Use setState to update state and re-render the component

```
onSubmit = (card) => {  
  this.setState({  
    cards: this.state.cards.concat(card)  
  })  
}
```

Lift state up to the closest common ancestor

- Single “source of truth” in React apps
- Rely on the top-down data flow
- Avoid duplicating lifted state in child components

Code Along

- Clone (or Download) [this repo](#)
- We will work together to wire up these components using props, state, and setState
- Here is [a video](#) of the code along that you can refer to afterwards

Objectives

- Describe Single Page Applications (SPA)
- Name the key concepts in React
- Explain components, props, and JSX
- Identify functional and class components
- Know when to use state vs setState
- Lift state up to the closest common ancestor