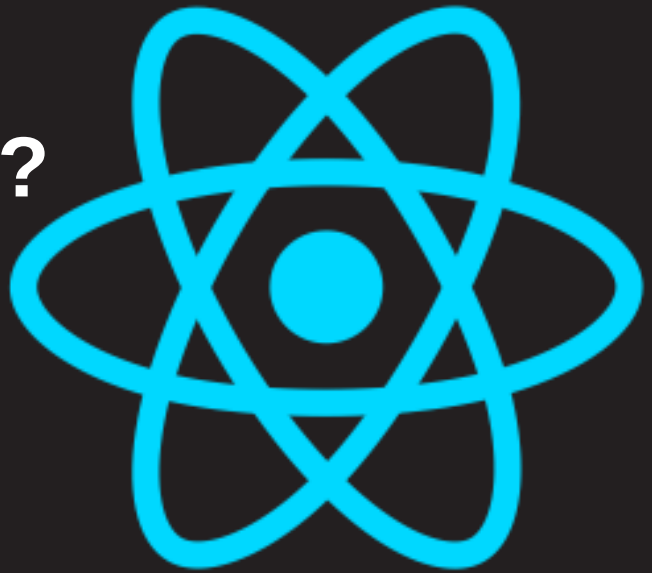
Components

What are Components?



LEARNING OBJECTIVES

- Gain insight into components
- Learn about the types of components
- Get a conceptual look at state and event listeners
- Discover JSX

Components of React

What are components?

- The core building blocks of a React application

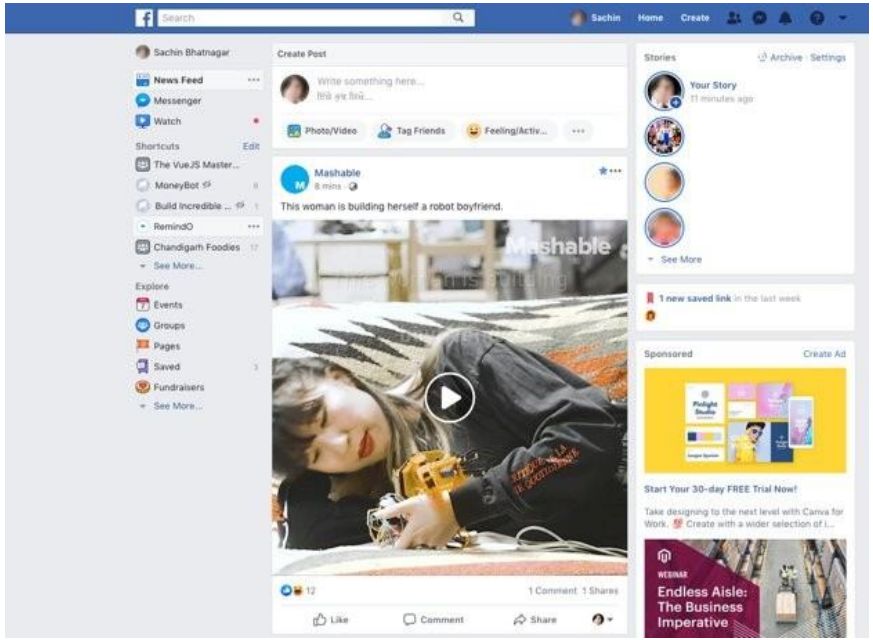
- Helps break down large interface into small units that can be built and maintained independently.

What are components?

Example:

- Social card on websites
- Components of a social card:

- Images
- Video
- Text
- Like and emotion buttons

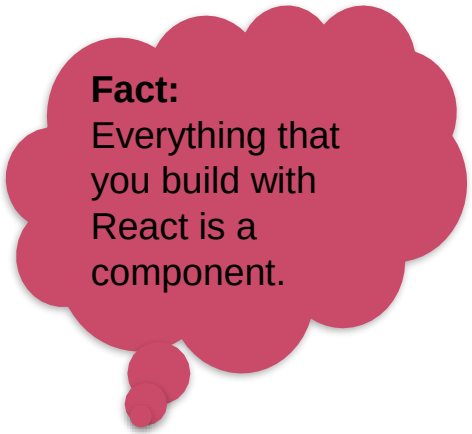


What are components?

Facebook uses more than 50,000 components & they're all built using React.

Why Components?

- Components help breakdown big monolithic codebase to small units
- They is easy to maintain
- Can be developed independently as composable units
- It helps avoid logistical challenges during errors and debugging
- It brings scalability to the codebase



Components: Build Great UIs and Apps

Composable units of components when added with declarative rendering along with state and virtual DOM offers the perfect toolkit for building great UI’s and Apps

Visualizing a Component

Red Hot Coffee Company

We've got a brand-new estate in our fold. Try the Namakkai Estate medium roast Arabica. Now available on our website. Order your pack now. http://bit.ly/293Ced

Type a comment…

SocialCard Component

Red Hot Coffee Company

Profile Component

We've got a brand-new estate in our fold. Try the Namakkai Estate medium roast Arabica. Now available on our website. Order your pack now. http://bit.ly/293Ced

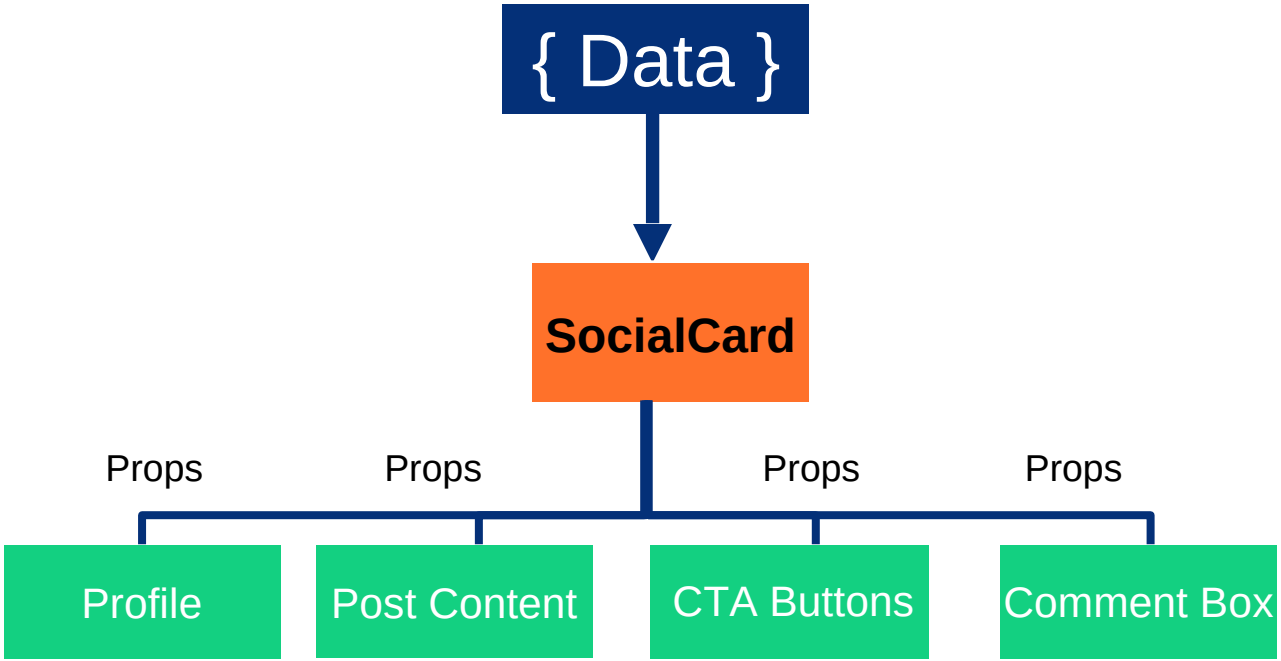
PostContent Component

CTAButtons Component

Type a comment…

CommentBox Component

Hierarchy of Components



One-way data flow

Advantages of unidirectional data flow

- Easy to understand, implement and debug common problems
- Easy recognition of data sources and the flow

Features of Components

- Components are reusable features that can be compose together to create a user interface.
- Reusability of components lets you design components that can be used across multiple projects, hence saving time.
- Modularizing features in components facilitates scalability & maintainability.

Open Source of React Community

Tip:
React's vibrant community has several component packages that can be downloaded and used in projects without the need of building them from scratch.

Anatomy of a component

```
class ToggleButton extends Component {
  state = {
    isOn: false
  }
  toggle = () => {
    this.setState({
      isOn: !this.state.isOn
    });
  }
  render() {
    return (
      <div className={this.state.isOn ? "btn btn-on" : "btn"} onClick={this.toggle}>
          {this.props.label}
      </div>);
  }
}
```

Render Method

```
class ToggleButton extends Component {
  state = {
    isOn: false
  }
  toggle = () => {
    this.setState({
      isOn: !this.state.isOn
    });
  }
  render() {
    return (
      <div className={this.state.isOn ? "btn btn-on" : "btn"} onClick={this.toggle}>
          {this.props.label}
      </div>);
  }
}
```

The render method describes the visual elements of a component, typically using JSX

State of a Component

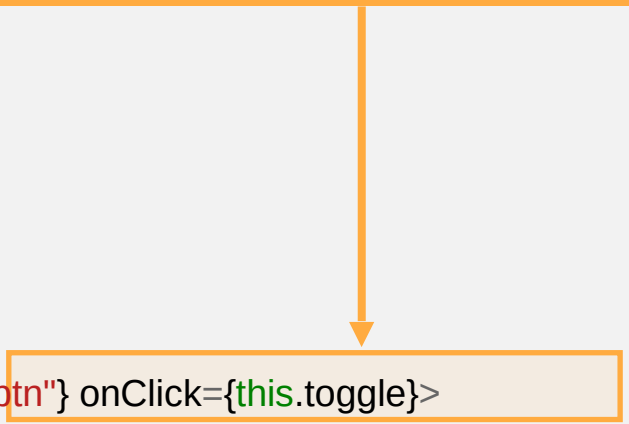
```
class ToggleButton extends Component {
  state = {
    isOn: false
  }
  toggle = () => {
    this.setState({
      isOn: !this.state.isOn
    });
  }
  render() {
    return (
      <div className={this.state.isOn ? "btn btn-on" : "btn"} onClick={this.toggle}>
          {this.props.label}
      </div>);
  }
}
```

State is used to describe visual elements using data. Updates to the state causes the component to re-render with visual updates.

Event Listeners

```
class ToggleButton extends Component {
  state = {
    isOn: false
  }
  toggle = () => {
    this.setState({
      isOn: !this.state.isOn
    });
  }
  render() {
    return (
      <div className={this.state.isOn ? "btn btn-on" : "btn"} onClick={this.toggle}>
          {this.props.label}
      </div>);
  }
}
```

Listening to events such as mouse clicks



Methods

```
class ToggleButton extends Component {
  state = {
    isOn: false
  }
  toggle = () => {
    this.setState({
      isOn: !this.state.isOn
    });
  }
  render() {
    return (
      <div className={this.state.isOn ? "btn btn-on" : "btn"} onClick={this.toggle}>
        {this.props.label}
      </div>);
  }
}
```

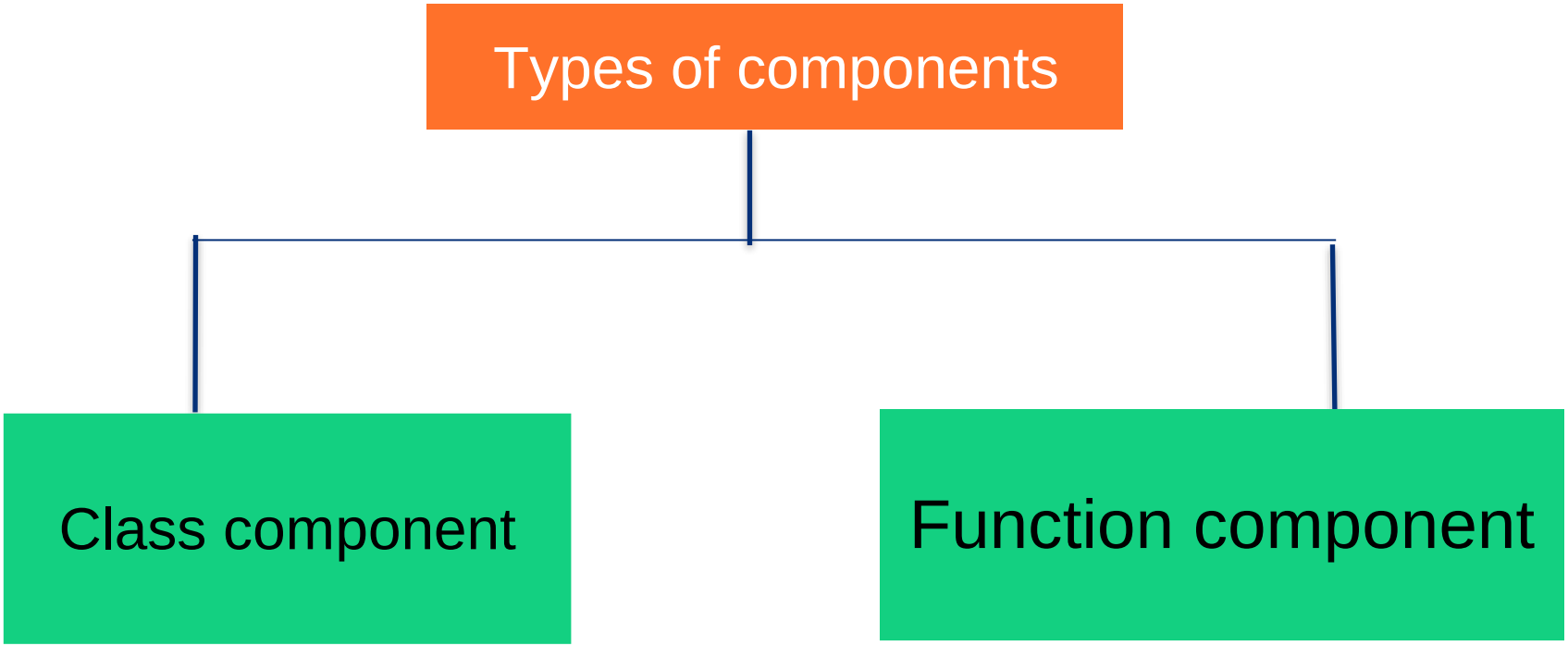
A method that updates the state

Components

Types of Components

knowledge hut

Types of Components



Features of Class Components

- Class components let you store local state.
- Allows you to create lifecycle hooks

Hands On

Class Components Offers

- Native support for state management
- Instance methods
- Lifecycle hooks

Features of Function Components

- Function Components are simple Java script function that return a React element.
- It can also bring in props object as an argument.
- It is a presentational component, which accepts data through a prop and renders it on the UI.
- Function Components are perfect for separating the visuals from container components.

Function Components vs Class Components

- Unlike Class Components, function components do not offer features like **built in state management**, **lifecycle methods** etc

- Function components with its syntax and simplicity is best-suited for elementary and simple use cases, unlike class components which are used for complex use cases

How do we incorporate state and Lifecycle methods in Function Components?

How do we incorporate state and Lifecycle methods in Function Components?

By using React's **Hooks API**

Hands On

How do you pick between class and function components?

- Function components coupled with Hooks API offers a cleaner and simpler syntax.
- Improves readability of the code and is also easier to test.

Having said that, there is nothing wrong with using class components as well and we'll be using them throughout this course, along with function components.

Components

JSX

What is JSX?

- JSX is a syntax that lets you describe what a component renders.
- Though it looks like HTML, it is an extension to regular JavaScript and is more powerful and expressive
- It is like a templating syntax which makes it easy to incorporate visual elements and dynamic data

Fact: React encourages to use standard JavaScript instead of using library or framework specific methods and directives.

Features of JSX?

- JSX differs a bit from HTML, for instance, instead of using class, we can use className and create event listeners like onClick
- JSX uses curly braces for rendering a dynamic value or for integrating a JavaScript expression.

```
class Notification extends Component {
  render() {
    return (
    <div className="notifications">
      <span id="nt-title">Notification</span>
      <Panel message={this.props.notificationText} />
      <Button type="dismiss" onClick={this.props.dismiss}/>
    </div>);
  }
}
```

How JSX Works?

- JSX lets us describe visual elements in an HTML like syntax but internally transforms the syntax into code that React actually uses.
- It is designed to produce React elements.
- JSX code is compiled into standard JavaScript functions containing nested invocations of the createElement function, which is what React uses to create renderable elements internally.

createElement Function

The createElement function takes in 3 arguments.

1. Type of element or the reference to a component.

2. Object where key-value pairs translate to props and their values.

3. The final argument is for child components or content that needs to go between the opening and closing tags of an element or a component.

Things to Know

Attribute names in JSX, unlike HTML must be in camelCase, just as they would be when using DOM APIs in JavaScript.

thank you!