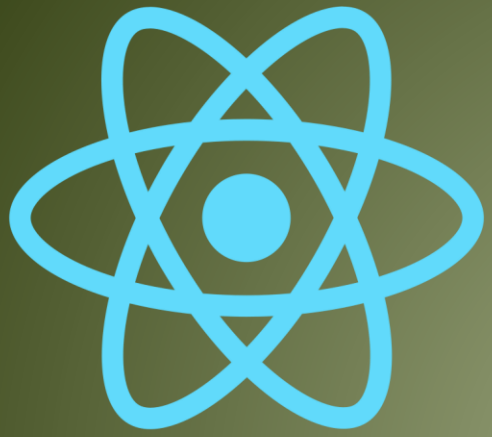


React

A JavaScript library for building user interfaces

Prof. Rachana V. Modi

What is React?

- React is a JavaScript front-end library.
- It is used for building user interfaces or reusable UI components.
- It is created and maintained by Facebook.
- React is specifically used in the development of single-page applications(SPA).

History

- Created by Jordan Walke, software engineer of Facebook
- First deployed on Facebook's News Feed in 2011
- Released for public in May 2013
- Users: Facebook, WhatsApp, Instagram, Netflix and yahoo mail

Why React?

- Fast
- Modular
- Scalable
- Flexible
- Open source
- Popular
- Large community

Features of ReactJS

- JSX
- Components
- Virtual DOM
- Javascript Expressions

Virtual DOM

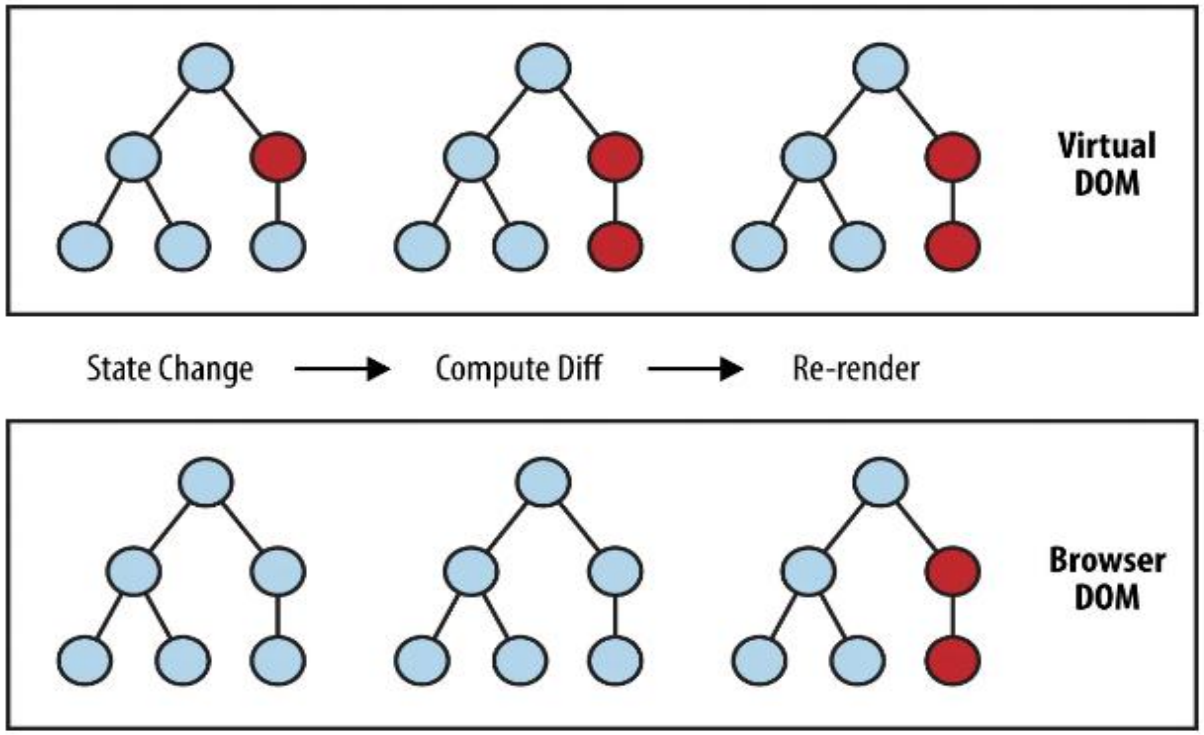
- **What is DOM?**

  Document Object Model

- **DOM issue**

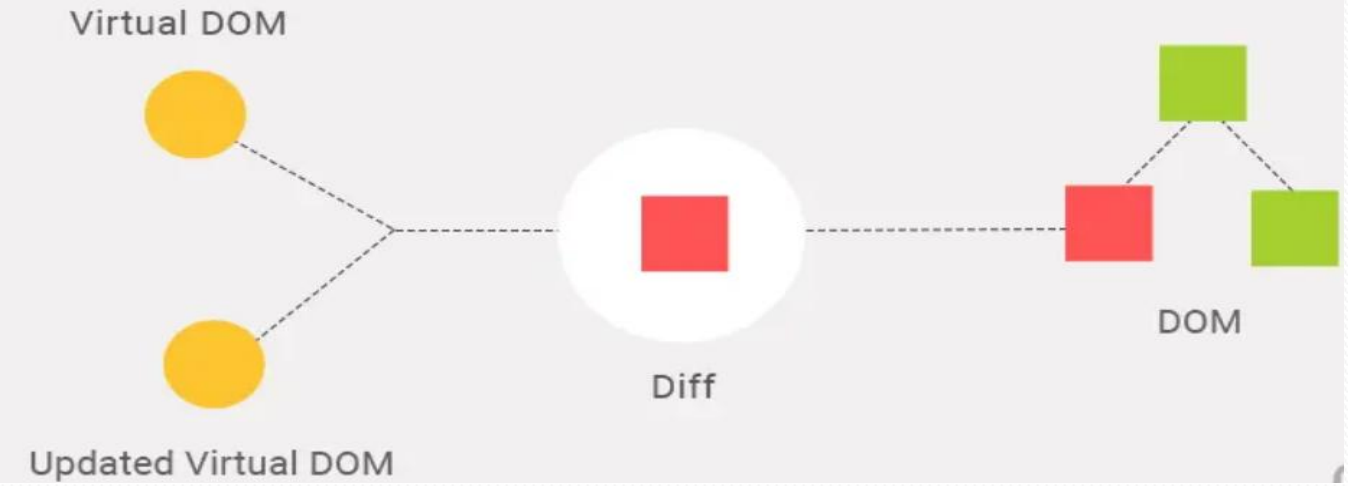
- **What is Virtual DOM?**
  - Virtual DOM is a virtual representation of DOM.
  - Every time the state of application changes, virtual DOM gets updated instead of real DOM.

How virtual DOM works?



How does React uses Virtual DOM?

- In React every UI piece is a component and each component has a state.
- React follows the observable pattern and listens for state changes.
- When the state of a component changes, React updates the virtual DOM tree.
- Once the virtual DOM has been updated, then React compares the current version of the virtual DOM with the previous version of the virtual DOM. This process is called “diffing”.
- Once React knows which virtual DOM objects have changed then React updates only those objects in the real DOM.



How does React updates real DOM?

Batch Update:

- React follows a batch update mechanism to update the real DOM. Hence, leading to increased performance. This means that updates to the real DOM are sent in batches, instead of sending updates for every single change in state.

- The repainting/rendering of the UI is the most expensive part, and React efficiently ensures that the real DOM receives only batched updates to repaint the UI.

Setup React Environment

- Node.js
- NPM
- Tool: create-react-app
- Command to install create-react-app:

C:\Users\Your Name>**npm install -g create-react-app**

Pre-requisite for ReactJS

- **NodeJS and NPM**
- **React and React DOM** - libraries. react is used for the components and react-dom is for rendering the components in the DOM
- **Webpack** used for module packaging, development, and production pipeline automation.
- **Babel** is a JavaScript compiler and transpiler used to convert one source code to others. It compiles React JSX and ES6 to ES5 JavaScript.

Create First React Application

- **Command to create a React application:**

  npx create-react-app project_name

- **Project Structure:**
  - node_modules
  - public
  - src
  - package-lock.json
  - package.json
  - README.md

How to run the React Application?

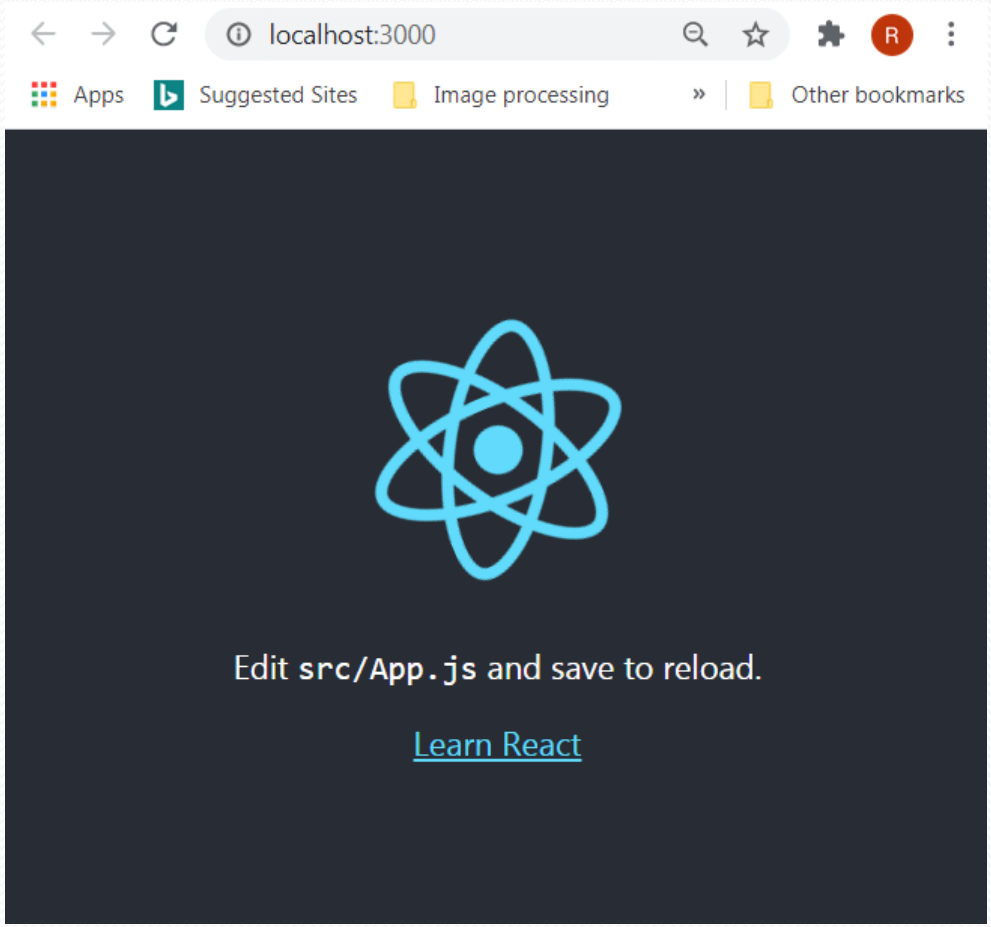
- Command to move to the project directory:

  C:\Users\Your Name>cd project_name

- Command to run the React application:

  C:\Users\Your Name\ project_name >npm start

Output


localhost:3000
Apps
Suggested Sites
Image processing
Other bookmarks
Edit src/App.js and save to reload.
Learn React

React components

- Component - Application is divided into a small logical group of code
- Component - Core building blocks of a React application
- Each component exists in the same space, but they work independently from one another and merge all in a parent component, which will be the final UI of an application.

- Two types of components:
  - Function Components (stateless components)
  - Class Components (stateful components)

React State

- React components has a built-in state object.
- The state object is used to store a property value of component.
- When the state object changes, the component re-renders.

Creating the state Object:

- The state object is initialized in the constructor.
- Example:

```
class Student extends React.Component {
    constructor(props) {
      super(props);
      this.state = {branch: "CE"};
    }
```

React State

Changing the state Object:

- To change a value in the state object, use the this.setState() method.
- setState() method change the state object, it will ensure that the component knows its been updated and calls the render() method and all the other lifecycle methods.

- Note: Never call setState() inside constructor()

React State

Example: Changing the state Object

```
class Student extends
React.Component {
    constructor(props)
    {
      super(props);
      this.state = {branch: "CE”};
    }

    UpdatedBranch = () => {
      this.setState({branch: "IT"});
    }
```

```
 render() {
      return (
<div> <p>Student is in
{this.state.branch} branch.  </p>

<button type="button"
onClick={this.UpdatedBranch}>
Update Branch</button>
        </div>
      );
    }
}
```

React Props

- Props are read-only components.
- It is an object which stores the value of attributes of a tag and work similar to the HTML attributes.
- It allows passing data from one component to other components. It is similar to function arguments and can be passed to the component the same way as arguments passed in a function.
- Props are immutable so we cannot modify the props from inside the component.

State Vs Props

Props	State
Props are read-only.	State changes can be asynchronous.
Props are immutable.	State is mutable.
Props allow you to pass data from one component to other components as an argument.	State holds information about the components.
Props can be accessed by the child component.	State cannot be accessed by child components.

State Vs Props

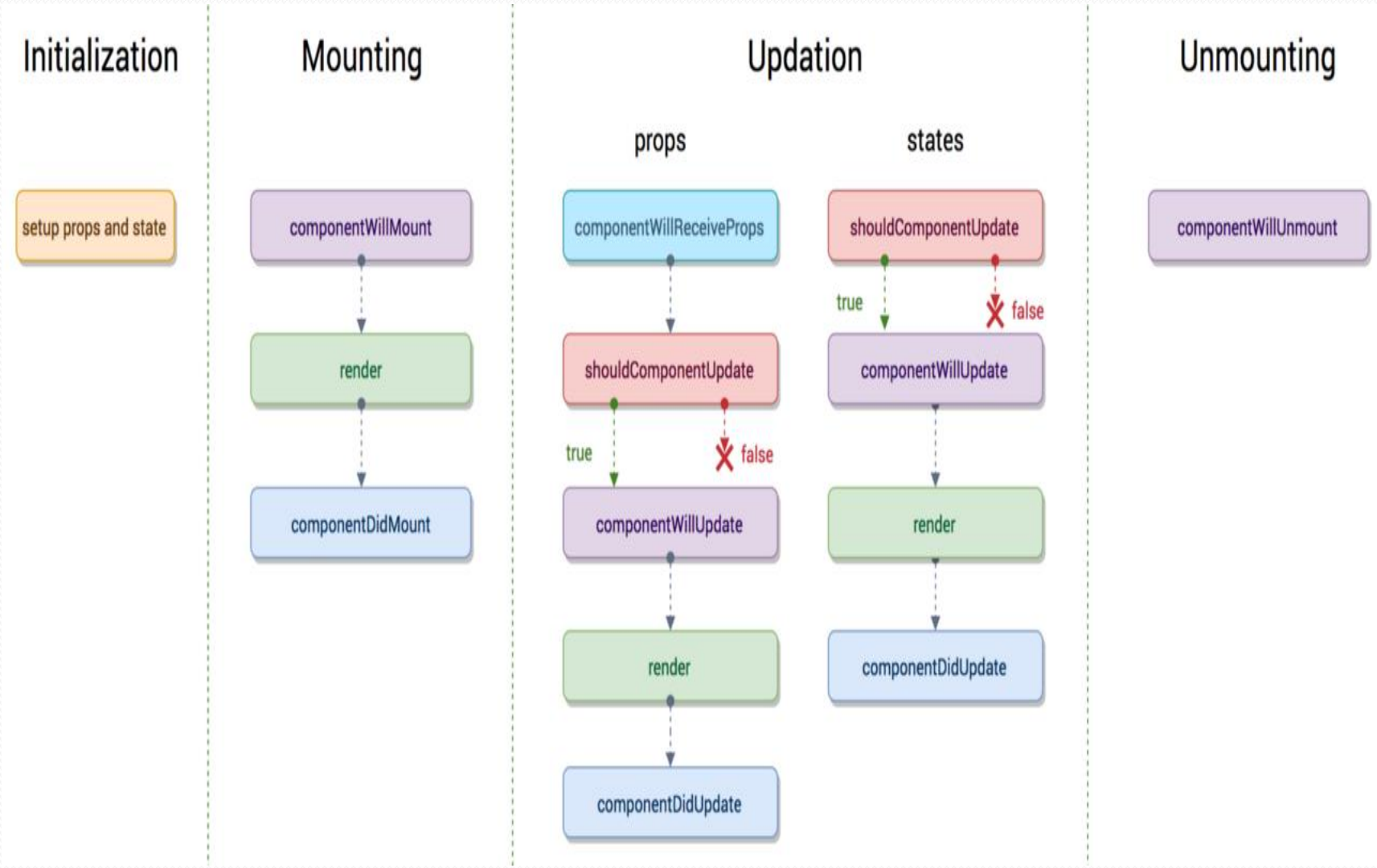
Props	State
Props are used to communicate between components.	States can be used for rendering dynamic changes with the component.
Stateless component can have Props.	Stateless components cannot have State.
Props make components reusable.	State cannot make components reusable.
Props are external and controlled by whatever renders the component.	The State is internal and controlled by the React Component itself.

React Component Life-Cycle

Every component creation process involves various lifecycle methods.

The lifecycle of the component is divided into four phases:

- **Initial Phase** - getDefaultProps(), getInitialState()
- **Mounting Phase(Birth of component)** - componentWillMount(), componentDidMount(), render()
- **Updating Phase (Growth of component)** - componentWillRecieveProps(), shouldComponentUpdate(), componentWillUpdate(), render(), componentDidUpdate()
- **Unmounting Phase (Death of component)** - componentWillUnmount()


Initialization
setup props and state
Mounting
componentWillMount
render
componentDidMount
Updation
props
componentWillReceiveProps
shouldComponentUpdate
true
false
componentWillUpdate
render
componentDidUpdate
states
shouldComponentUpdate
true
false
componentWillUpdate
render
componentDidUpdate
Unmounting
componentWillUnmount

Any Query?