React JS

- Vishal Avalani

Why?

- Very complex to manage DOM manipulations manually
- Well defined architecture
    - MVC
    - MVVM etc.

Library Vs Framework?

Library: Collection of functions which are useful when writing web apps

Ex: jQuery, React JS

Framework: Implementation of web app where code fill in details/gaps.

Ex: Angular, Ember, Meteor etc.

Popular framework/libraries

- React
- Angular
- Backbone
- Ember
- Meteor
- Knockout
- Vue and many more...

Web

- SPA
  - Rich Internet Apps
- Scaleable
- Reusable
- Maintainable JS Code

What?

- Very popular JS library for building User Interfaces
- Declarative
- Component based
- Technology agnostic - focus on problem solving

History

- Designed by Jordan Walke
- 2011 - deployed on FB newsfeed
- 2013 - Open sourced

Ref: https://en.wikipedia.org/wiki/React_(web_framework)#History

Terms

- JSX
- Components
- State
- Props
- Hooks

Installation

- Node JS
- Create React App: https://reactjs.org/docs/create-a-new-react-app.html

  Basic React Application ready for use.

Understand basic structure of application

JSX

Templating Engine

const element = <h1>Welcome to React Training</h1>;

(WOW! Can we really write HTML in Javascript?)

What?

- Templating Engine
- Syntactic extension to JavaScript
- Separation of concern (SOC) - UI and logic remains separate

Behind the scenes

```
const v1 = (
	<h1 className=’title’>Welcome to React Training!</h1>
);
```

acts as

```
const v1 = React.createElement(
			‘h1’,
			{className: ‘title’},
			‘Welcome to React Training!’);
```

Contd..

```
const v1 = {
    type: ‘h1’,
    props: {
        className: ‘title’,
        children: ‘Welcome to React Training!’
    }
};
```

Embedding Expressions in JSX

Integrate trainees array into JSX

Components

- Returns set of React Elements
- Enable to split UI into independent reusable pieces
- It also accepts inputs
- Two types primarily
  - Stateful/Class based/Container/Smart
  - Stateless/Functional/Dumb/Presentational
- **Convention**: User Defined - starts with capital letter
- Lowercase are Built in components (DOM tags)

Stateless/Dumb Components

- Mainly concerned with rendering view
- Only access props and render accordingly
- Can’t access any lifecycle methods
    - No state access

Stateful/Smart Components

- Extend React.Component
- Need to implement render method
- Bring life to component
    - Data Fetching
    - State updates
- Access to lifecycle methods
- Provide data to dumb components for rendering
- Maintain state and communicate with data sources

State

- Limited to component
- Fully controlled by component
- Can be passed as props to children components
- Only class based can have local state

Embedding State in App

Trainees array into state

setState()

Can we do this.state.employees = xyz?

Props

- JSX attributes are passed into a component as single object
    - Available in the component as props
    - Can pass multiple attributes
    - Cannot modify props
- Example of trainees and pass to component

Handling Events

- Similar to how we handle on DOM elements
  - Use camelCase
  - Pass function as event handler
- Example

Lifting State Up

- Several components may share same data
- Changes in one component needs to reflect in another
- Best to move it up so that both can share

Keys

Keys should be given to elements inside array

Helps identify internally what has changed and in re-rendering

Demo Session

Lifecycle Methods

Stages:

- Mounting
- Updating
- Unmounting

Mounting

Called when instance of a component is being created and inserted in DOM.

1. constructor()
2. getDerivedStateFromProps()
3. render()
4. componentDidMount()

Version 16.3.0 or higher

Updating

Called when component is re-rendered.

1. getDerivedStateFromProps()
2. shouldComponentUpdate()
3. render()
4. getSnapshotBeforeUpdate()
5. componentDidUpdate()

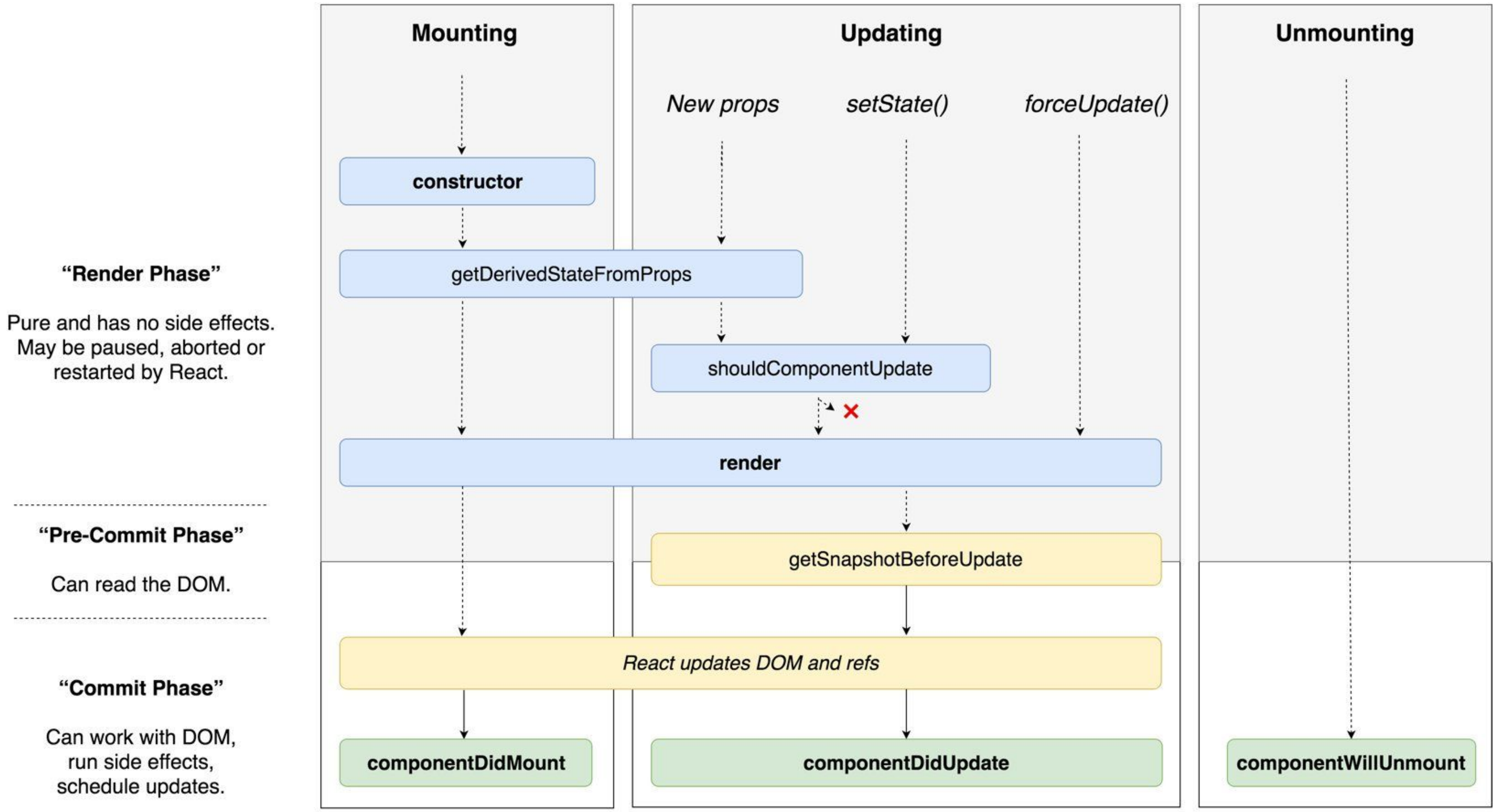
Unmounting

Component is removed from DOM.

componentWillUnmount()

Error Handling: called when there is

- error in rendering
- In lifecycle method
- constructor of any child component
    - componentDidCatch()



"Render Phase"
Pure and has no side effects. May be paused, aborted or restarted by React.
"Pre-Commit Phase"
Can read the DOM.
"Commit Phase"
Can work with DOM, run side effects, schedule updates.
Mounting
Updating
Unmounting
New props
setState()
forceUpdate()
constructor
getDerivedStateFromProps
shouldComponentUpdate
render
getSnapshotBeforeUpdate
React updates DOM and refs
componentDidMount
componentDidUpdate
componentWillUnmount

Hooks

- Why?
- React 16.8 or higher
- Very Basics
    - useState
    - useEffect etc...
- Demo

CLEAR THE MESS AND CONFUSION!

useState

- Can be used to create state variables
- Setters help in setting values
- We can also grab previous values

```
const [employees, setEmployees] = React.useState([]);
```

useEffect

- Called after every render
- Can track dependencies and be called accordingly

```
useEffect(() => {

   console.log('UseEffect called');

 });
```

How React works?

- Browser DOM is browser Object
- Virtual DOM is React Object
    - Lightweight representation of Browser DOM
    - In memory tree structure
    - Fast manipulations compared to browser DOM
    - Created completely from scratch by setState()

Algorithms

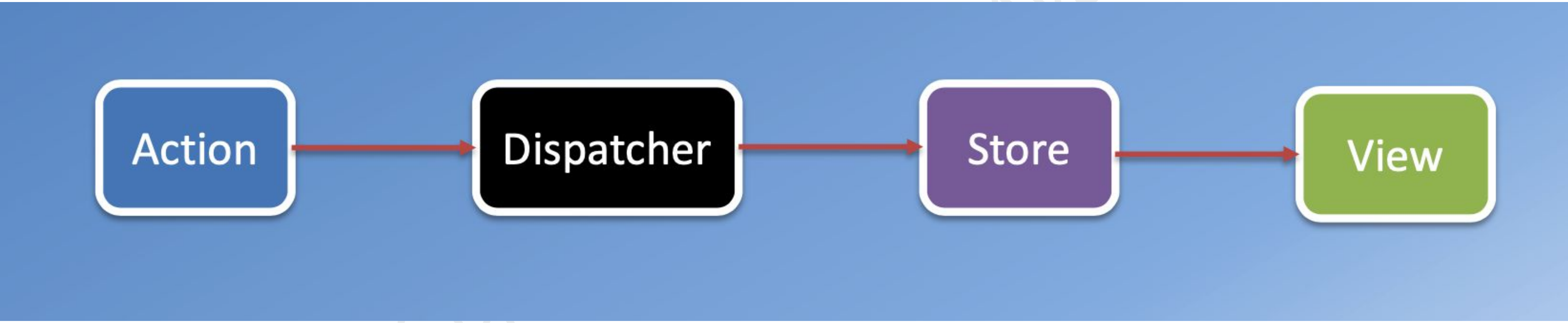
- Difing
  - Updates entire subtree if diffing detects that two elements are of different types
  - Using **key** you can hint child elements as stable
  - **Must read:** https://www.cronj.com/blog/diff-algorithm-implemented-reactjs/
- React Fiber
  - New algorithm in React 16
  - **Must read:** https://raphamorim.io/understanding-react-fiber-incremental-rendering-feature/

Flux Architecture

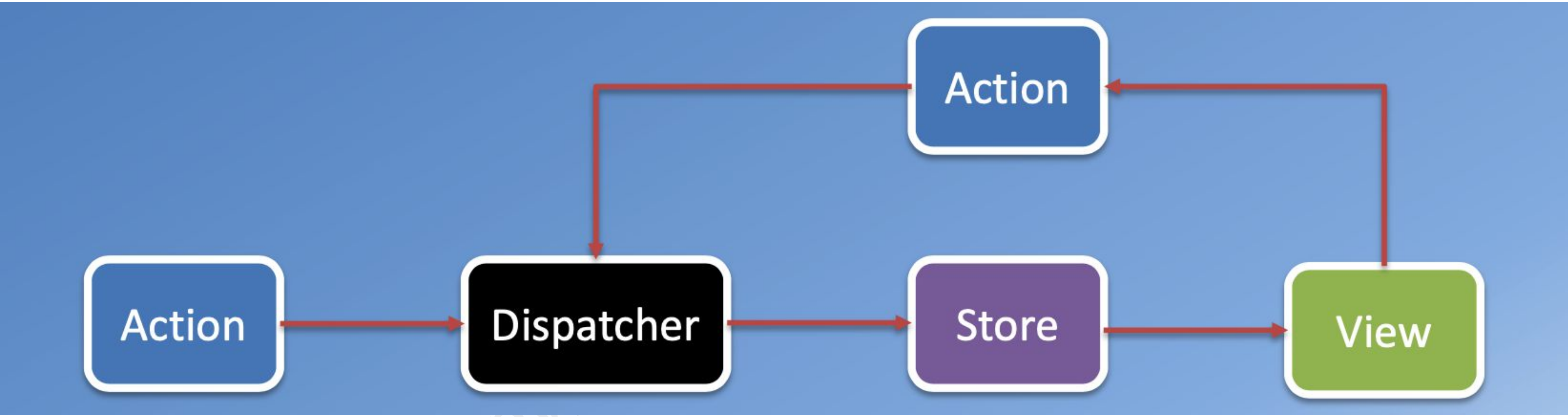
- Why?
- What? - Flux is an architecture that Facebook uses internally when working with React
- MVC pattern - React takes care of V
- Flux takes care of M
- Problems
  - Cascading updates
  - Race conditions etc.
- Reference:
  - https://facebook.github.io/flux/
  - https://www.cabotsolutions.com/2017/01/detailed-study-flux-react-js-application-architecture

Architecture

- Unidirectional data flow



Handling User Actions



Store can’t be updated directly. It has to go via actions.