React

ANIL JOSEPH

Agenda

- Introduction to React
- Creating React Applications
- Understanding Components
- JSX
- Components Deep Dive
- Debugging
- Virtual DOM
- Server-side communication with HTTP Requests
- Single Page Applications with Routing
- Forms
- Redux
- Introduction to Testing
- Deployment

Presentation By Anil Joseph(anil.jos@gmail.com)

Software

Node.js and NPM

- Version 10 or higher

Yarn(Optional)

JavaScript Editor(Any one)

- **Visual Studio Code**
- Sublime Text
- Atom

Browsers

- Chrome, Firefox

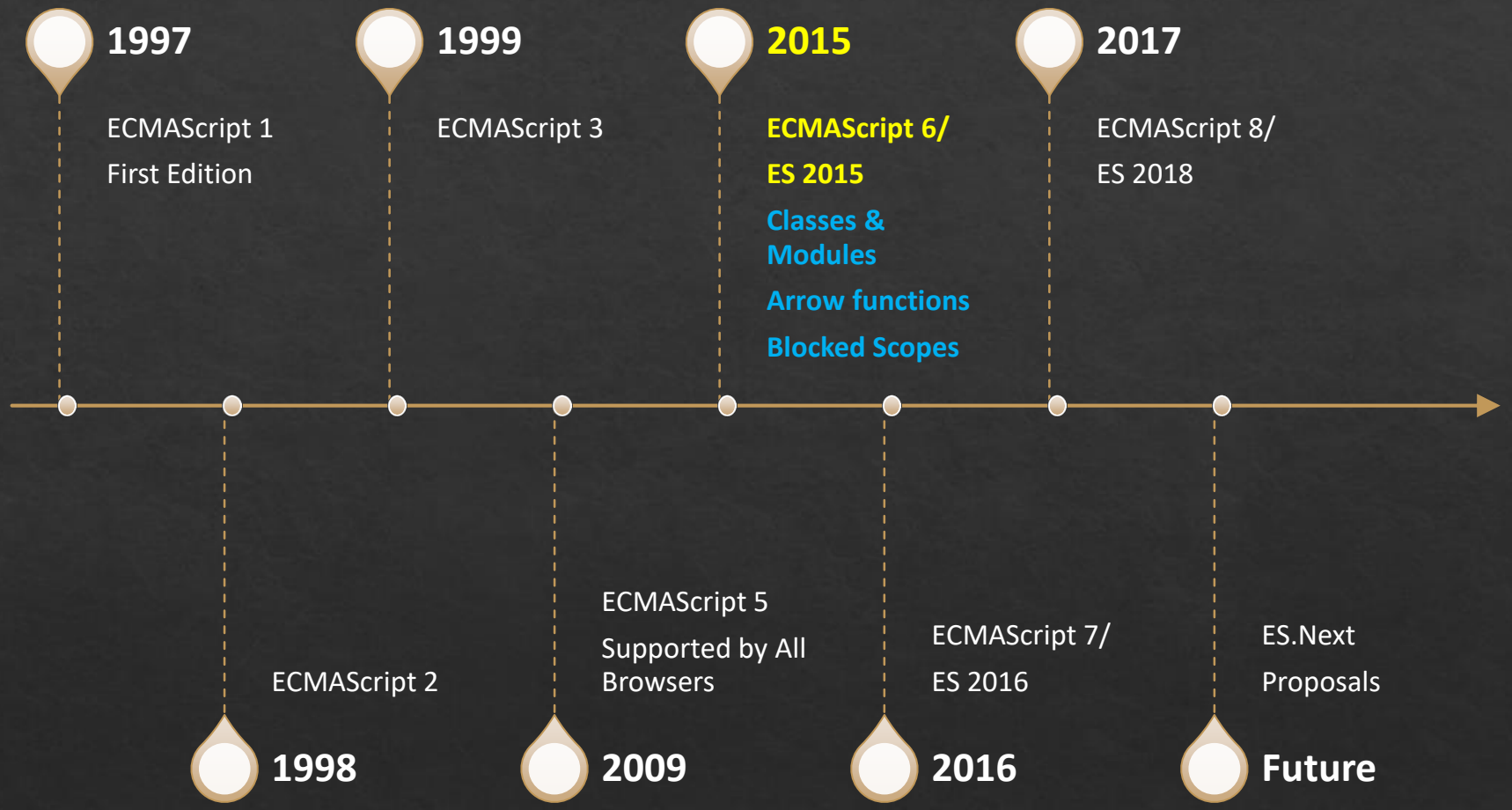
Presentation By Anil Joseph(anil.jos@gmail.com)

JavaScript

- JavaScript (JS) is an interpreted programming language.
- It’s a dynamic language.
- Supports object-oriented programming.
- As part of web browsers, implementations allow
  - Client-side scripts to interact with the user.
  - Alter the document content that is displayed.
  - Control the browser.
  - Communicate asynchronously.
- JavaScript was developed by Brendan Eich at Netscape.
- Released in September 1995

Presentation By Anil Joseph(anil.jos@gmail.com)

ECMAScript Versions



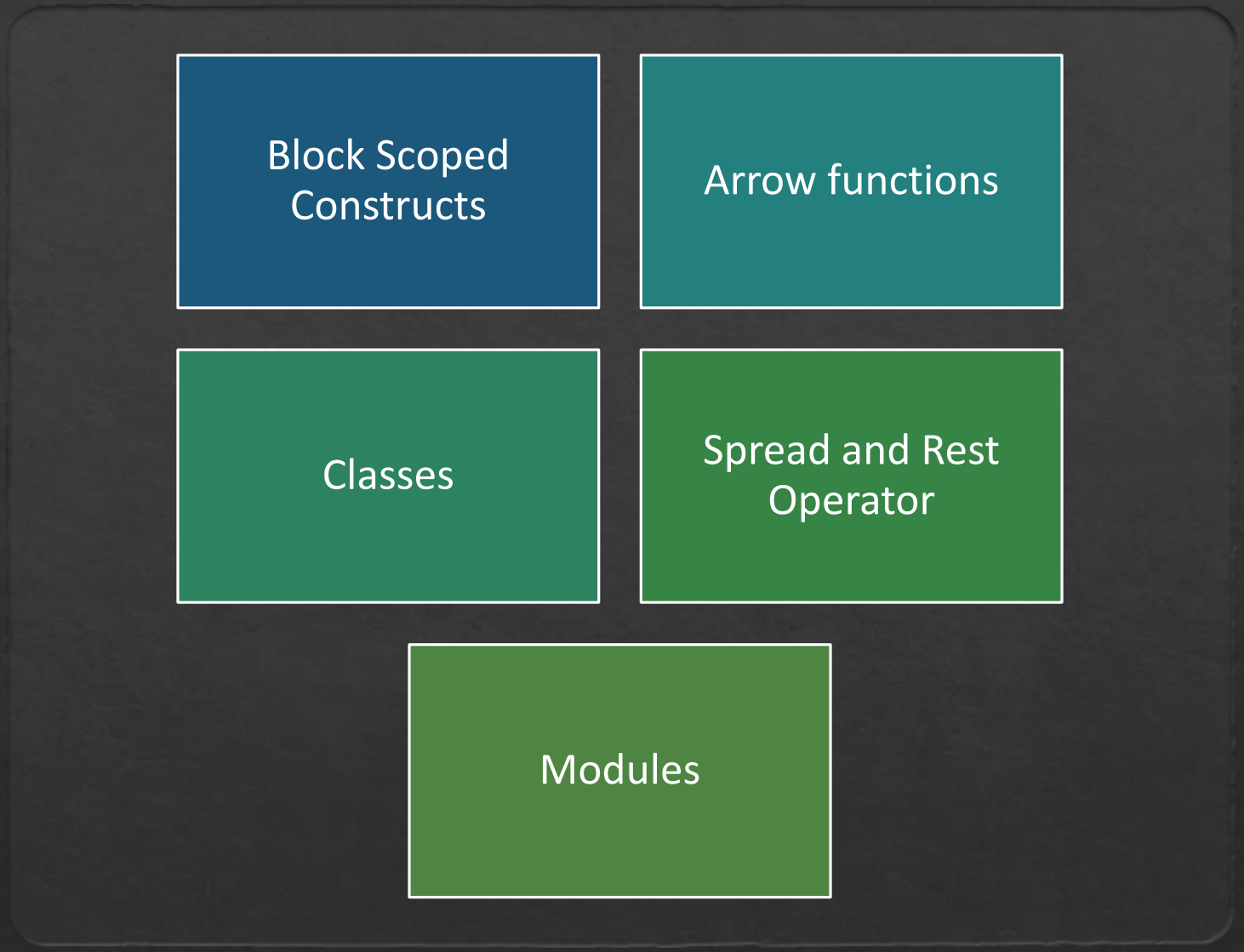
Presentation By Anil Joseph(anil.jos@gmail.com)

Challenges with JavaScript

- Inconsistent browser implementations to access the DOM, XML parsing etc.
- JavaScript supports ***global variables***, which could cause conflicts when using multiple libraries.
- Creations of ***namespaces*** is not straight forward.
- Implementation of ***object-oriented code*** differs from conventional object-oriented languages.
- Some language features are complex to understand

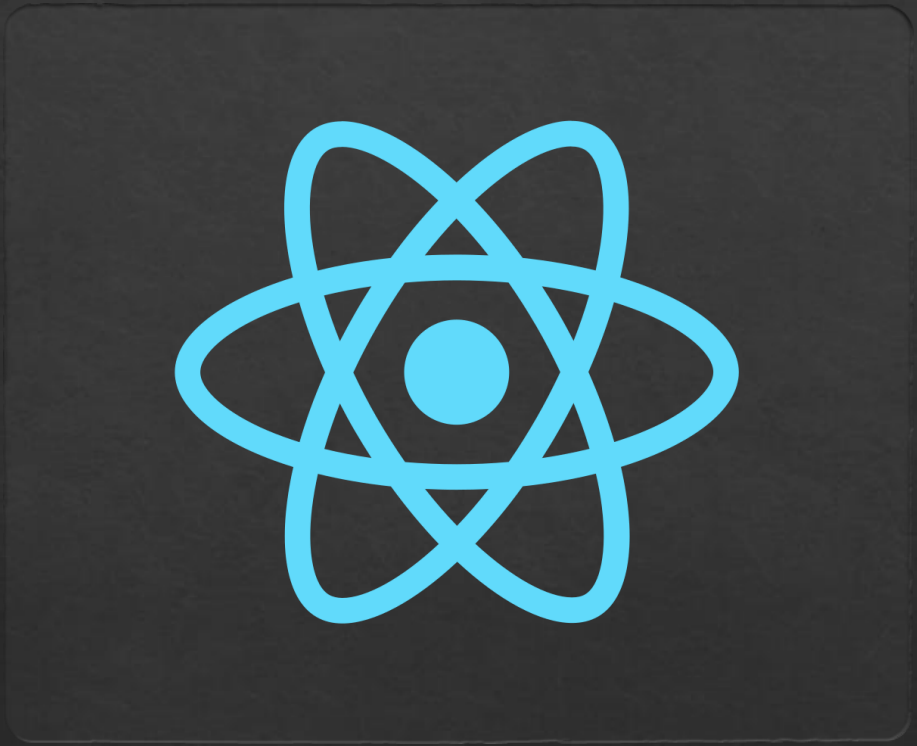
Presentation By Anil Joseph(anil.jos@gmail.com)

ES6/7 Features (Used in React)



Presentation By Anil Joseph(anil.jos@gmail.com)

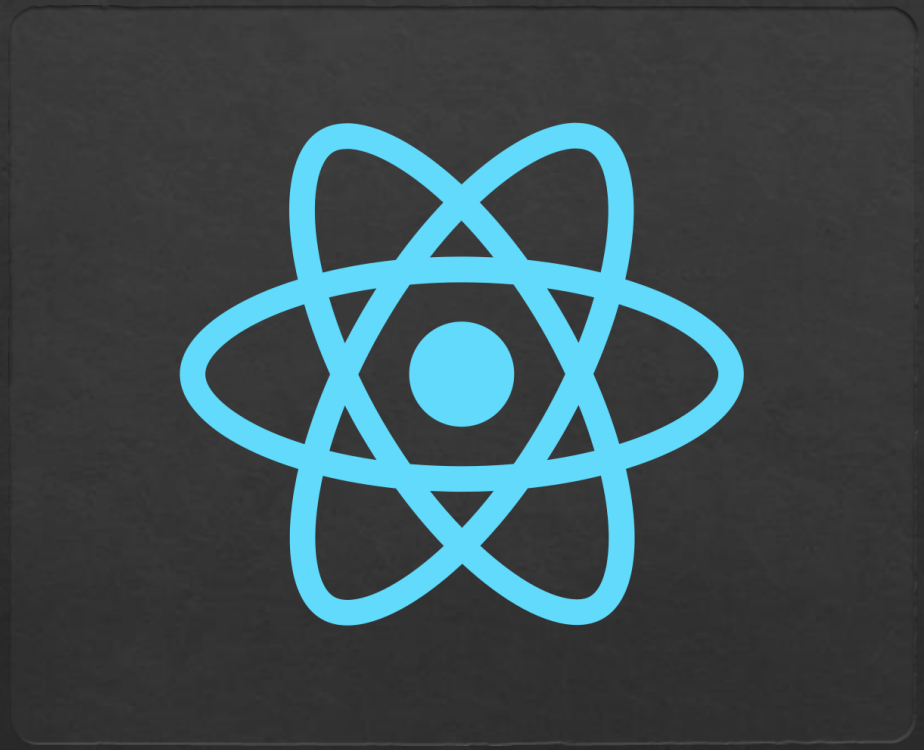
What is React?



- A JavaScript library to build ***User Interfaces***.
- Building applications for the ***browser***.
- React allows to create custom HTML elements called ***components***.
- Components are used to build ***complex User Interface***
- Reacts promotes writing ***maintainable, manageable and reusable code***.

Presentation By Anil Joseph(anil.jos@gmail.com)

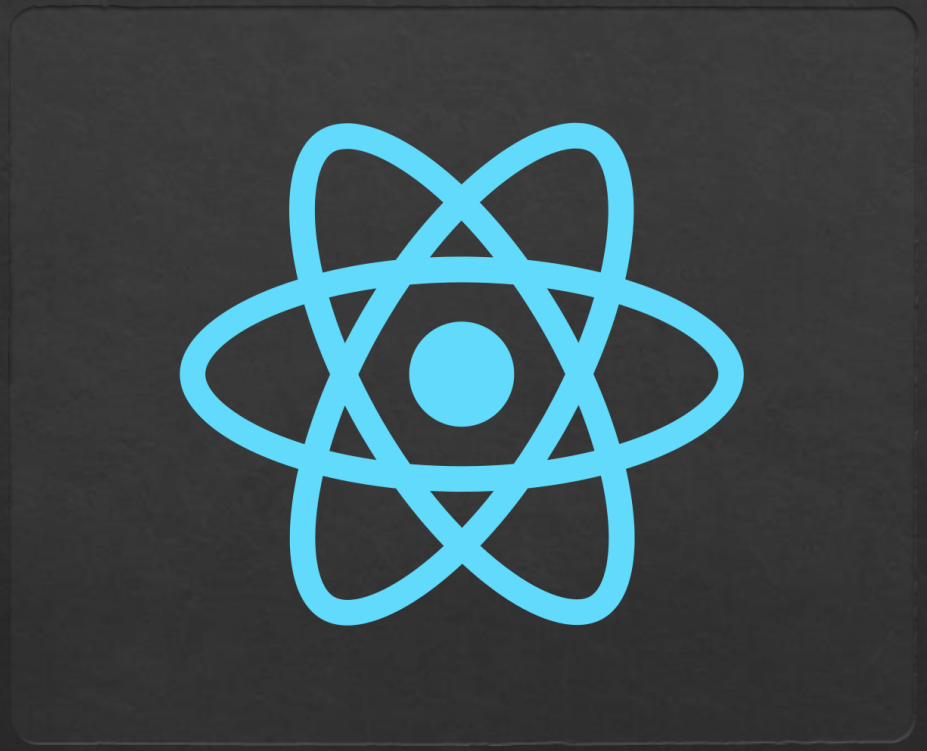
Why React?



- React handles *State*
  - UI State is difficult to handle with JavaScript
- React focuses on business logic
  - Focus on business logic
- React is *fast*.
  - Apps made in React can handle complex updates and still feel quick and responsive.

Presentation By Anil Joseph(anil.jos@gmail.com)

Why React?



- React is *modular*
  - Instead of writing large, dense files of code, you can write many smaller, reusable files.
- React is *scalable*.
  - Large programs that display a lot of changing data are where React performs best.
- React has a Huge ecosystem, community

Presentation By Anil Joseph(anil.jos@gmail.com)

History

React was created by **Jordan Walke**, a software engineer at Facebook.

It was first deployed on **Facebook's** newsfeed in 2011

Later on **Instagram** in 2012.

It was open-sourced in 2013.

Presentation By Anil Joseph(anil.jos@gmail.com)

Alternatives

Angular

- Open-source front-end web application platform from Google.

Vue

- Open-source progressive JavaScript framework for building user interfaces.

Presentation By Anil Joseph(anil.jos@gmail.com)

Getting Started

The Two React libraries

- React
- ReactDOM

Babel

- The compiler for writing next generation JavaScript.

JSX

- A JavaScript extension syntax allowing quoting of HTML

Presentation By Anil Joseph(anil.jos@gmail.com)

Getting Started Playgrounds

CodePen

- https://codepen.io/pen

CodeSandbox

- https://codesandbox.io/s/new

Presentation By Anil Joseph(anil.jos@gmail.com)

Creating a React Project

Build Process: Why?

Manage Dependency

- Use multiple libraries

Optimize Code

- Small in size

Next Generation Features using ES6 and ES7

- Leaner code, Easier to read, Faster

More Productive

- CSS Auto Prefixes, Linting

Presentation By Anil Joseph(anil.jos@gmail.com)

Build Process: How?

Dependency Management Tool

- npm and yarn

Bundler

- WebPack

Complier

- Complies ES6 to ES5/ES3
- Babel + Presets

Development Server

- Web Server for development

Presentation By Anil Joseph(anil.jos@gmail.com)

Build Process: Tool

- The React team provides a tool called **Create-React-App**.
- Use to quickly start development on React with zero configuration
- Integrates the various tools and libraries required for the build process
  - Webpack, NPM, Yarn, Babel
- Uses the ***react-scripts*** package

Presentation By Anil Joseph(anil.jos@gmail.com)

Create Project

1

Install NodeJs

- Runtime to run/execute JavaScript on the machine/server
- This installation also install npm(Node Package Manager)

2

Install “create react app”

- **Tool to create react applications**
- **npm install –g create-react-app**

3

Create Application/Project

- **create-react-app reactapp**

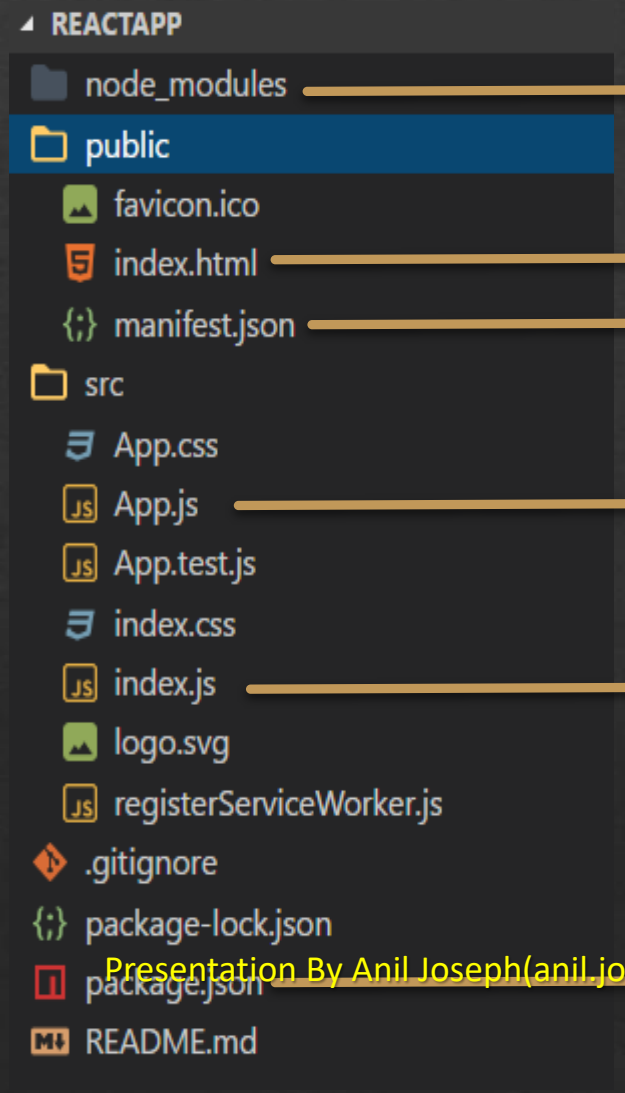
4

Start the Application

- **cd reactapp**
- **npm start**

Presentation By Anil Joseph(anil.jos@gmail.com)

Project Structure



Folder with all downloaded dependencies

The page displayed in the browser

The Web Application Manifest (for installable web applications)

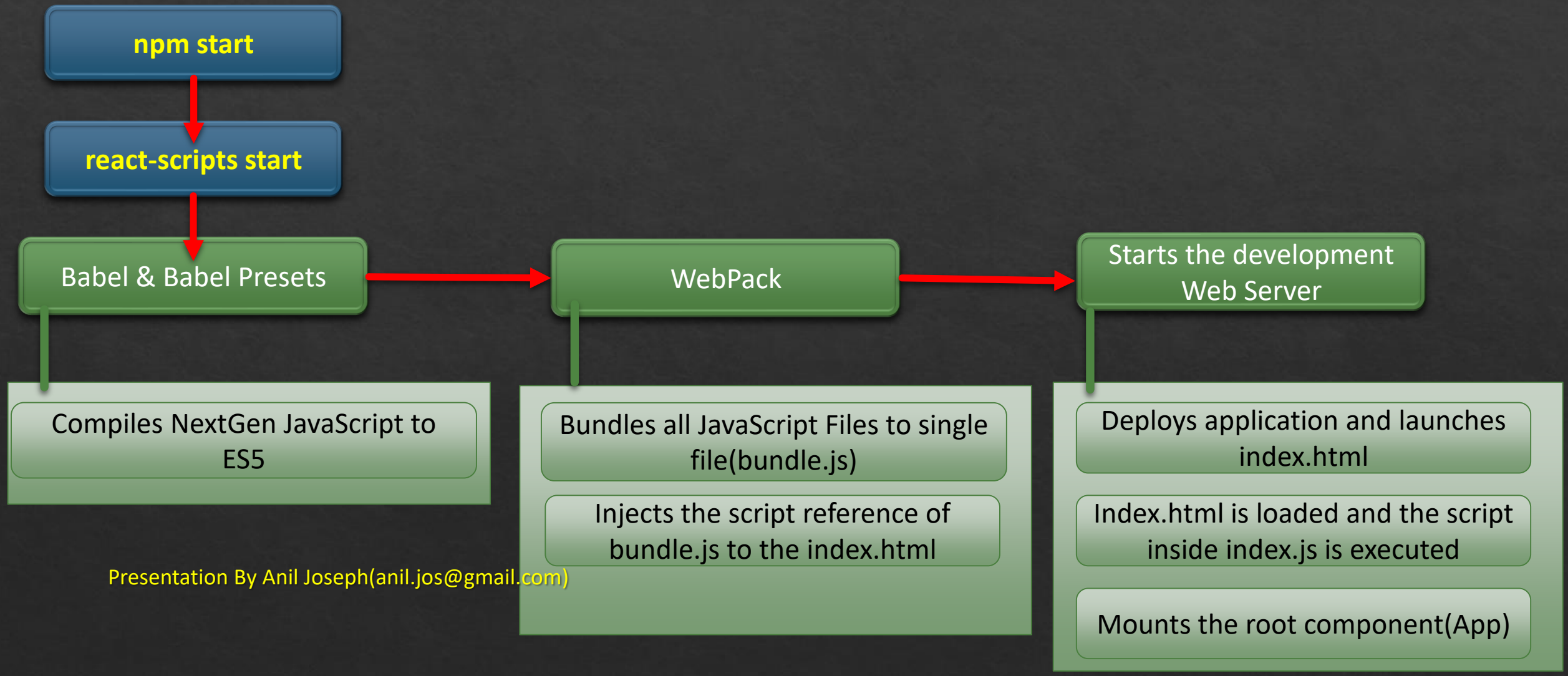
The root component

Initial Script executed on page load. Renders the root component

NPM configuration file, contains the dependencies and scripts

Presentation By Anil Joseph(anil.jos@gmail.com)

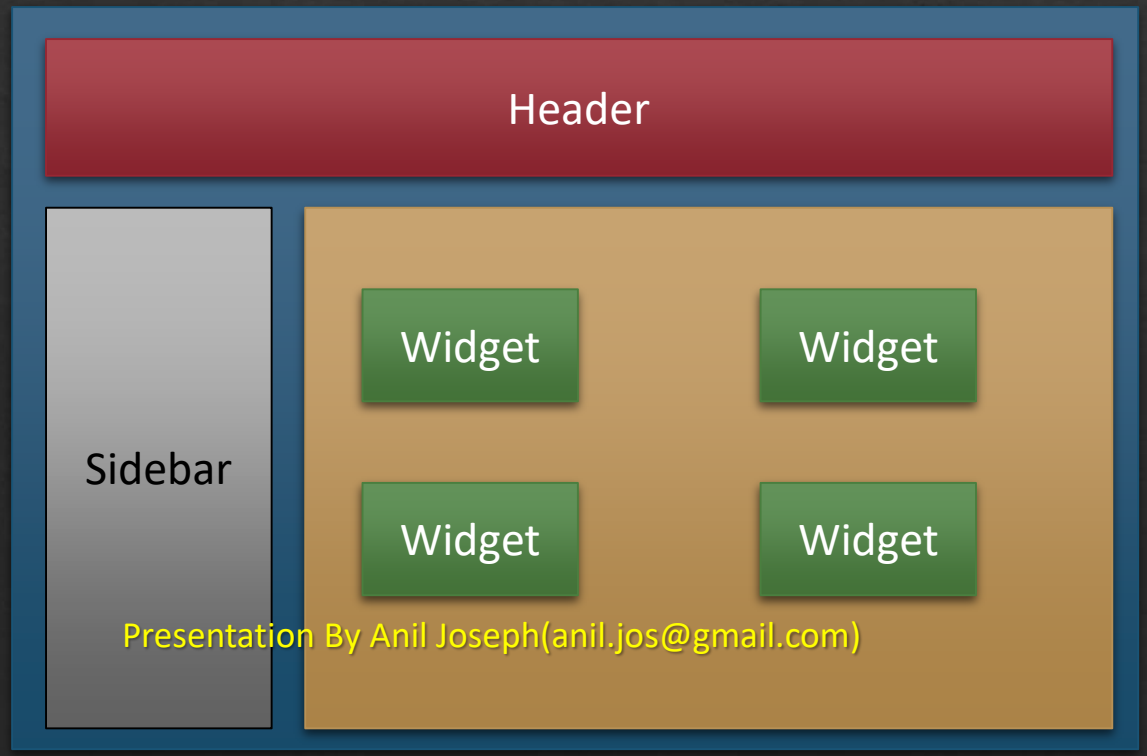
Project Execution

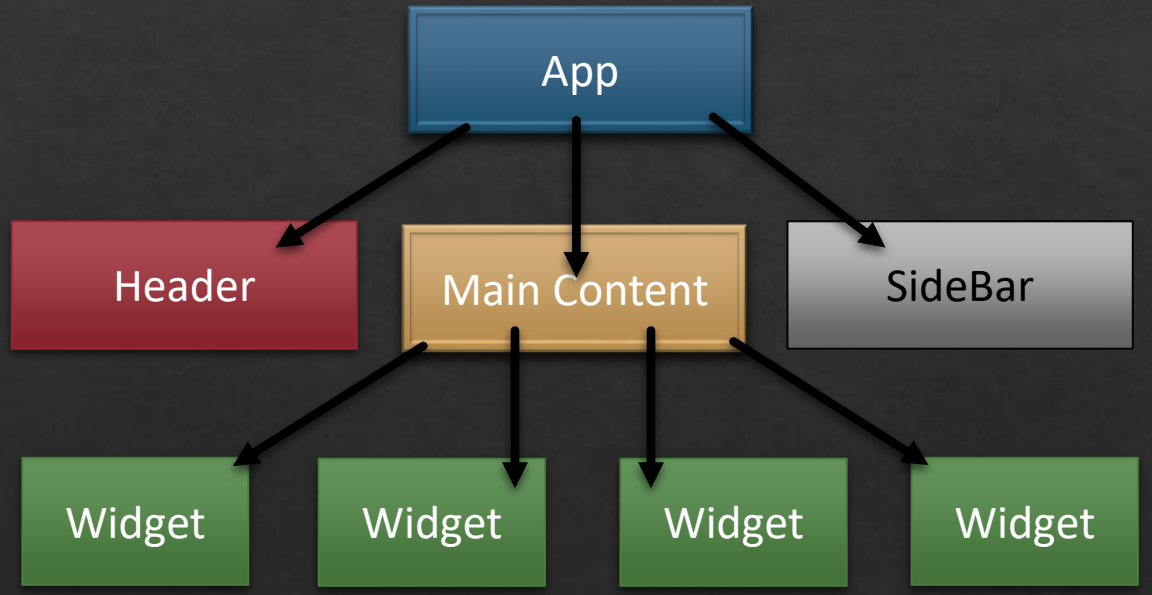


Presentation By Anil Joseph(anil.jos@gmail.com)

React Components

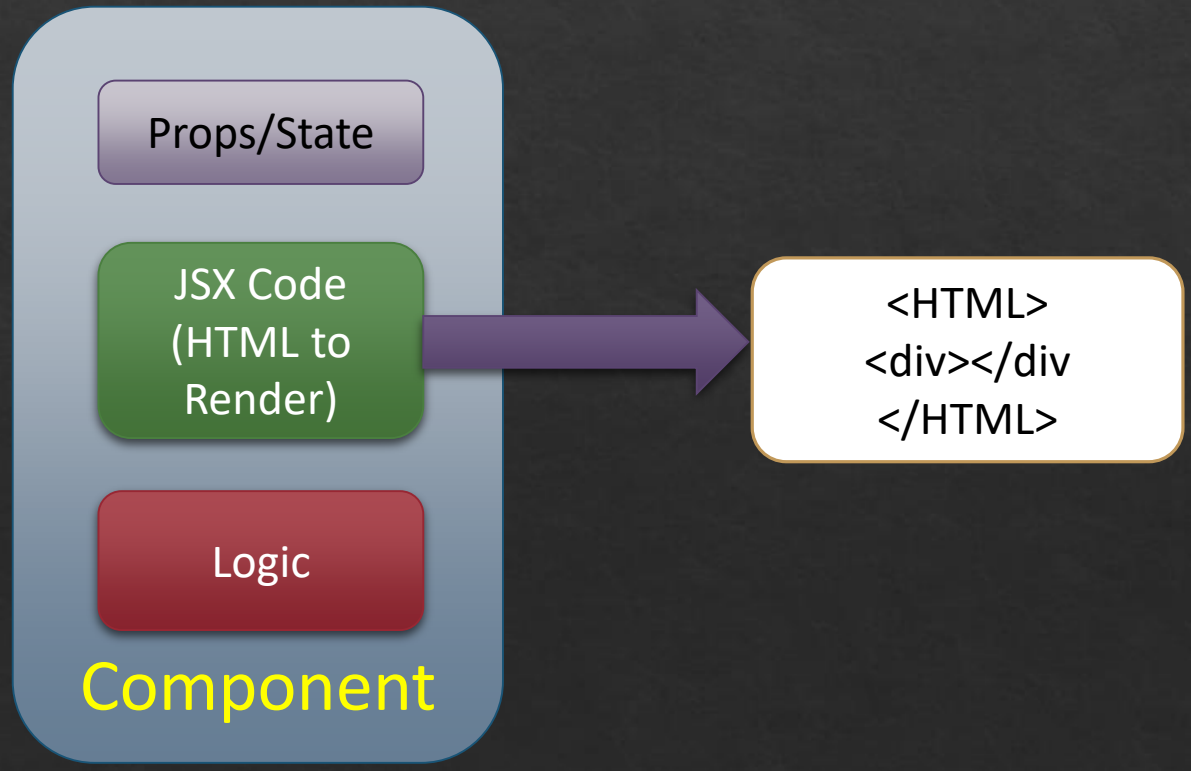
- Components are the **core building blocks of React applications**
- Creating a React Application is all about creating components.
- A React app can be depicted as a **component tree**





React Components

- The UI in a React Application is composed of ***components,*** the building blocks.
- Components are designed to be reusable.



Presentation By Anil Joseph(anil.jos@gmail.com)

Types of Components

Functional

- Presentational
- Stateless

Class Based

- Containers
- Stateful

Presentation By Anil Joseph(anil.jos@gmail.com)

JSX

- *JSX* is a syntax extension for JavaScript.
- It was written to be used with React.
- JSX code looks a lot like HTML.
  - **It's actually JavaScript**
- A *JSX compiler* will translate any JSX into regular JavaScript.
- JSX elements are treated as JavaScript *expressions*.
  - They can go anywhere that JavaScript expressions can go.
- That means that a JSX element can be
  - Saved in a variable
  - Passed to a function
  - Stored in an object or array

Presentation By Anil Joseph(anil.jos@gmail.com)

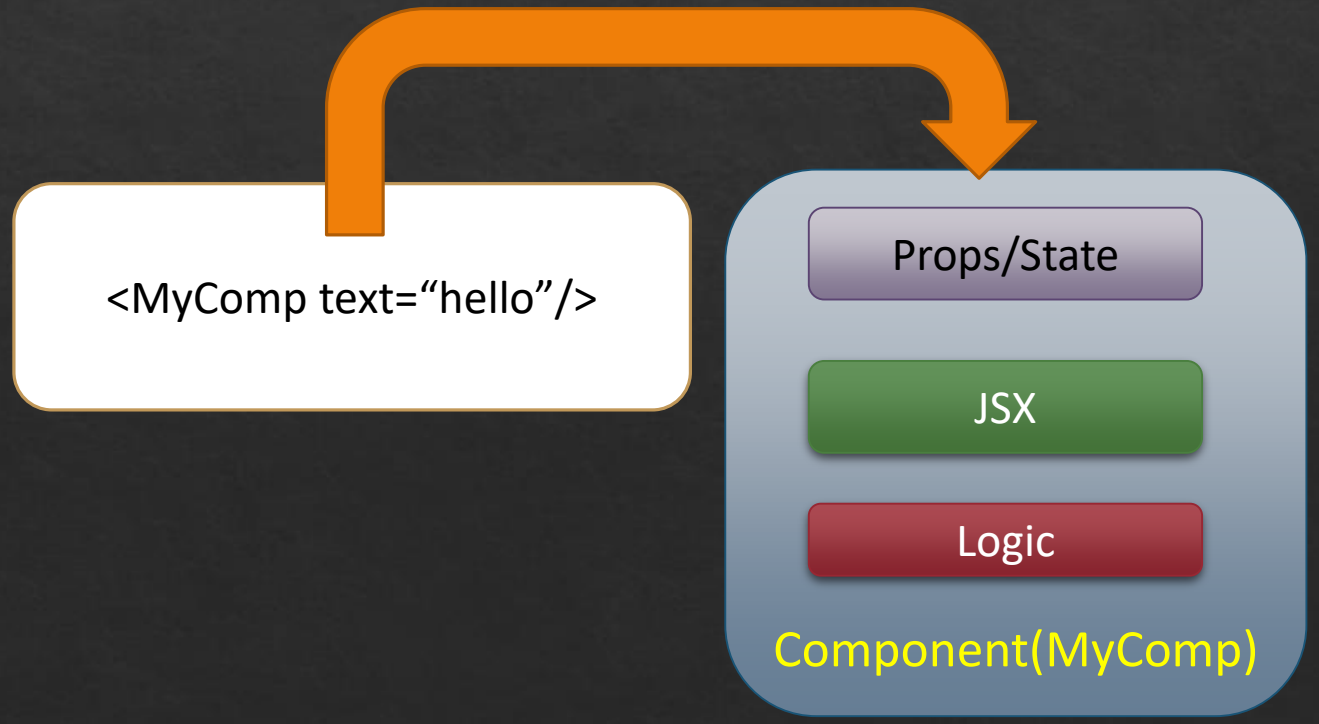
Components: Dynamic Content

- Dynamic content is outputted in the JSX using an expression.
- The syntax
  - { expression }
- This can be any one line expression
- Complex functionalities can be done by calling functions
  - { invokeSomeMethod() }

Presentation By Anil Joseph(anil.jos@gmail.com)

Components: Properties(props)

- Most components can be customized with different parameters when they are created.
- These creation parameters are called ***"props"***.
- ***Props*** are used similar to HTML attributes.
- ***Changes to props will automatically re-render the component.***



Presentation By Anil Joseph(anil.jos@gmail.com)

Props Types

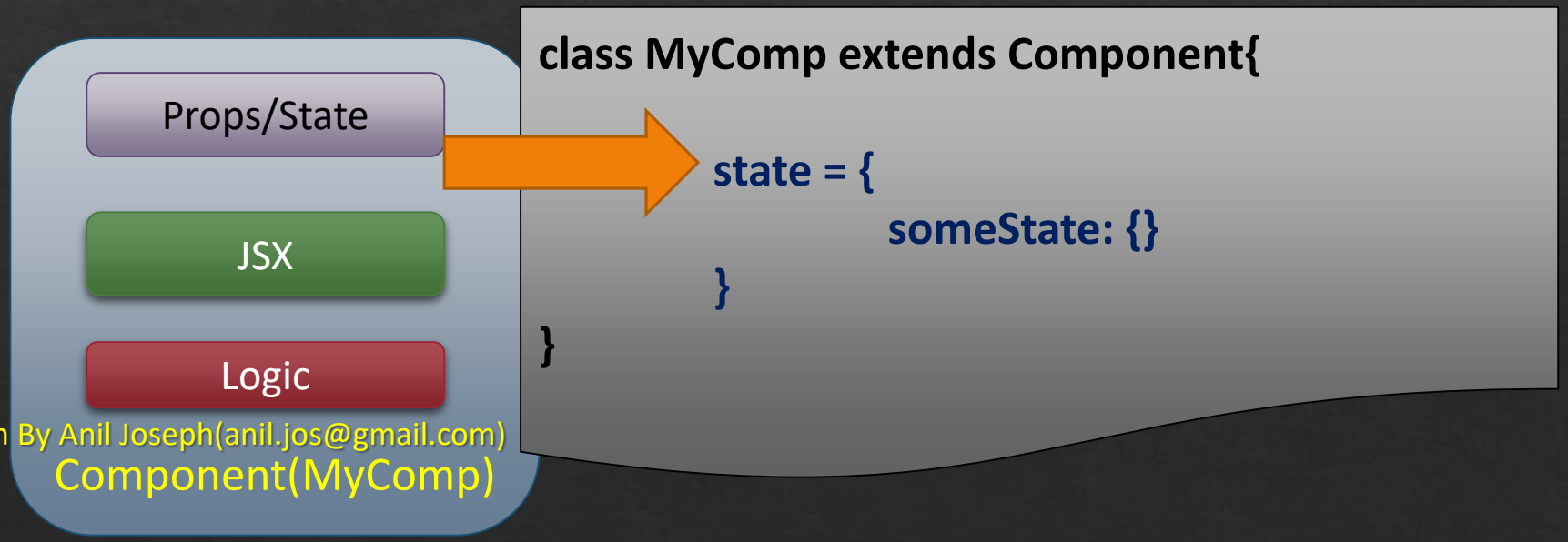
Use Prop-Types for Type Checking the Component Properties

npm install prop-types

Presentation By Anil Joseph(anil.jos@gmail.com)

Component State

- State holds information about the component
- State is used when a component needs to keep track of information between renderings.
- State is created and initialized in the component itself.
- State is available only with Stateful components(Class Based).
- **Component state should be always updated using the setState method**
- **State updates trigger a rerender of the component.**



Presentation By Anil Joseph(anil.jos@gmail.com)

Props and State

"props" and "state" are CORE concepts of React.

Only changes in "props" and/ or "state" trigger React to re-render the components and potentially update the DOM in the browser

props allow you to pass data from a parent (wrapping) component to a child (embedded) component.

State is used to change the component from within.

Presentation By Anil Joseph(anil.jos@gmail.com)

Event Handling

- Handling events with React elements is very similar to handling events on DOM elements with some syntactic differences.
- React events are named using camelCase, rather than lowercase.
- With JSX you pass a function as the event handler, rather than a string.
- Event handlers will be passed instances of ***SyntheticEvent***
  - A cross-browser wrapper around the browser's native event
- Details
  - https://reactjs.org/docs/events.html

Presentation By Anil Joseph(anil.jos@gmail.com)

Component Lifecycle

Presentation By Anil Joseph(anil.jos@gmail.com)

Component Lifecycle (Mount)

constructor

- Setup state
- Don’t Cause Side-Effects

componentWillMount

- Update State
- Don’t Cause Side-Effects
- Configuration

render

- Prepare and Struture JSX code

Render Child Components

ComponentDidMount

- Cause side effects
- **Initialize anything that relies on the DOM**

Presentation By Anil Joseph(anil.jos@gmail.com)

Component Lifecycle (Update)

componentWillReceiveProps(nextProps)

- Sync State to Props
- Don’t Cause Side-Effects

shouldComponentUpdate(nextProps,nextState)

- Decide whether to Continue or Not
- Don’t Cause Side-Effects

componentWillUpdate(nextProps, nextState)

- Sync State to Props
- Don’t Cause Side-Effects

render()

Update Child Component Props

componentDidUpdate()

- Cause Side-Effects
- Don’t Update State

Presentation By Anil Joseph(anil.jos@gmail.com)

Component Lifecycle (Unmount)

componentWillUnmount

- Remove Listeners
- Cancel Active network calls
- Invalidate Timers

Presentation By Anil Joseph(anil.jos@gmail.com)

AJAX and APIs

- React does not have any API's for AJAX calls.
- We have to use an AJAX Library for server communications
- Popular Libraries
  - Axios
  - jQuery AJAX
  - Fetch API
- In a component AJAX calls to fetch data from the server should be made in the ***componentDidMount*** lifecycle method.

Presentation By Anil Joseph(anil.jos@gmail.com)

axios

- Promise based HTTP client for the browser and node.js
- Installation
  - npm install axios
- Features
  - Make http requests
  - Supports the Promise API
  - Intercept request and response
  - Transform request and response data
  - Cancel requests
  - Automatic transforms for JSON data
  - Client side support for protecting against XSRF

Presentation By Anil Joseph(anil.jos@gmail.com)

axios methods

axios.request({config})

axios.get(url, {config})

axios.post(url,{data}, {config})

axios.delete(url, {config})

axios.put(url,{data}, {config})

Presentation By Anil Joseph(anil.jos@gmail.com)

axios global defaults

Base URL

- axios.defaults.baseURL = 'https://abc.com’;

Headers

- axios.defaults.headers.common['Authentication'] = AUTH_TOKEN;

Headers for specific methods

- axios.defaults.headers.post['Content-Type'] = 'application/json';

Presentation By Anil Joseph(anil.jos@gmail.com)

Debugging

- Error Messages
  - Messages generated by React during development mode
- Browser Developer Tools
- React Tools
  - Tools for Chrome and Firefox
- Error Boundaries
  - Error boundaries are React components that **catch JavaScript errors anywhere in their child component tree**
  - **Log errors**
  - **Display a fallback UI**

Presentation By Anil Joseph(anil.jos@gmail.com)

Immutability

- React components have the concept of *state*.
- Whenever state changes React re-renders the component.
- If state complicated (like with nested objects) it'd be hard to check whether your state changed or not.
  - Have to use *deep equality check*
- Immutability
  - *Whenever your object has to be mutated , don't mutate it*
  - *Create a copy*
- Immutability makes it easier to detect changes and update the UI

Presentation By Anil Joseph(anil.jos@gmail.com)

Virtual DOM

The Virtual DOM (VDOM)

- Where a “virtual”, representation of a UI is kept in memory
- This is synced with the “real” DOM.
- This process is called reconciliation.

Reconciliation

- When the render() function is called it creates a tree of React elements.
- On the next update render() will return a different tree
- React then figures out how to efficiently update the UI to match the most recent tree.
- Uses the Diff algorithm

Presentation By Anil Joseph(anil.jos@gmail.com)

Higher-order Components

- A Higher Order Component is just a React Component that wraps another one.
- Higher Order Components is a Pattern used extensively with React
- Uses
  - Code reuse, logic and bootstrap abstraction
  - Render Highjacking
  - State abstraction and manipulation
  - Props manipulation
- Its basically a functions that returns a class component with the Wrapped Component.

Presentation By Anil Joseph(anil.jos@gmail.com)

Single Page Applications(SPA)

Single-Page Applications are applications that load a **single** HTML **page** and dynamically update that **page** as the user interacts with the **app**.

SPAs use AJAX and HTML5 to create fluid and responsive Web **Applications**

Do not have constant **page** reloads.

Much of the work happens on the client side, in JavaScript.

Presentation By Anil Joseph(anil.jos@gmail.com)

Routing

- Routers allow to navigate between the different views in the application using JavaScript on the client-side.
- Navigations are updated to the browser history which allows to go back and forward
- Usage of path and search parameters are allowed
- React does not provide any API for routing.
- Many other Libraries available which integrates with React
  - React-router(The de-facto standard)
  - Director
  - Aviator
  - Finch

Presentation By Anil Joseph(anil.jos@gmail.com)

React Router v4

- React Router is a collection of **navigational components** that compose declaratively with your application.
- Installation
  - npm install react-router-dom

Presentation By Anil Joseph(anil.jos@gmail.com)

React Router Components

<BrowserRouter>

- A <Router> that uses the HTML5 history to keep your UI in sync with the URL.

<HashRouter>

- A <Router> that uses the hash portion of the to keep your UI in sync with the URL.

<Route>

- The Route component is perhaps the most important component in React Router

<Link>

- Provides declarative, accessible navigation around your application.

<NavLink>

- A special version of the <Link> that will add styling attributes to the rendered element when it matches the current URL.

Presentation By Anil Joseph(anil.jos@gmail.com)

State Management

React Context

- Available from React 16.3

React Redux

- Library to manage state

Presentation By Anil Joseph(anil.jos@gmail.com)

Types of State

Local UI State

- Show/Hide UI
- Handled By the Component

Persistent State

- Orders, Blogs
- Stored on Server, can be managed by Redux or React Context

Client State

- IsAuntenticated, Filter Information
- Managed by Redux or React Context

Presentation By Anil Joseph(anil.jos@gmail.com)

Redux

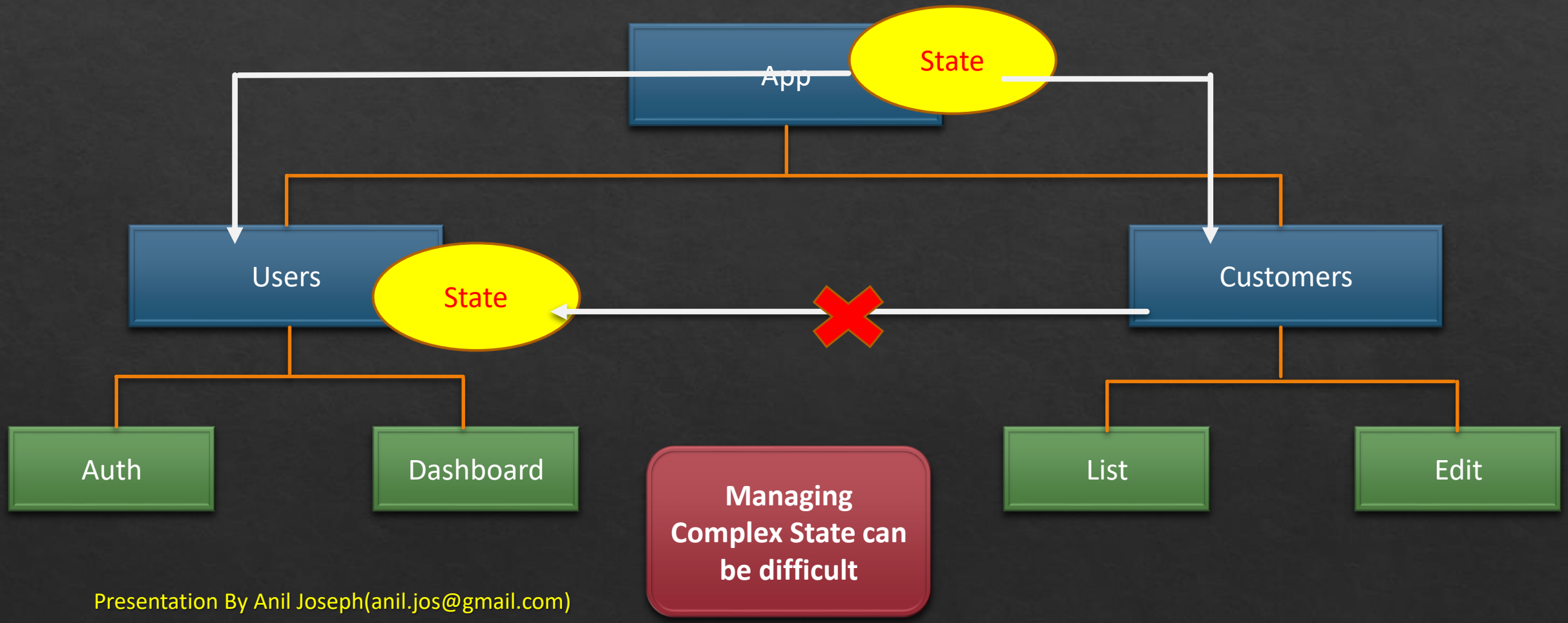
Redux is an open-source JavaScript library designed for managing application state.

It is primarily used together with React or Angular for building user interfaces.

Redux was built on top of functional programming concepts.

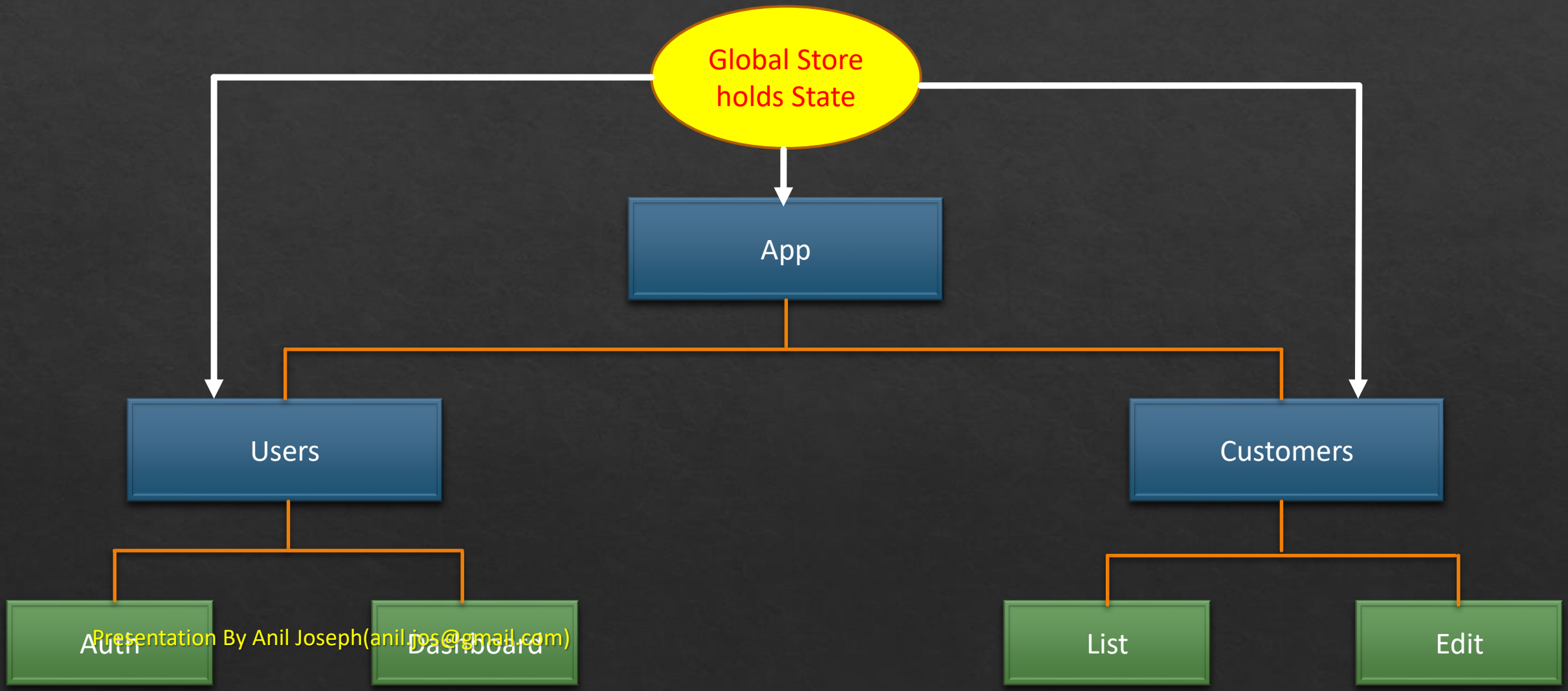
Presentation By Anil Joseph(anil.jos@gmail.com)

Why Redux?



Presentation By Anil Joseph(anil.jos@gmail.com)

Redux State Management



Presentation By Anil Joseph(anil.jos@gmail.com)

Redux Parts

Store

- Store is the object that holds the application state
- The entire state is represented by a single store.

Actions

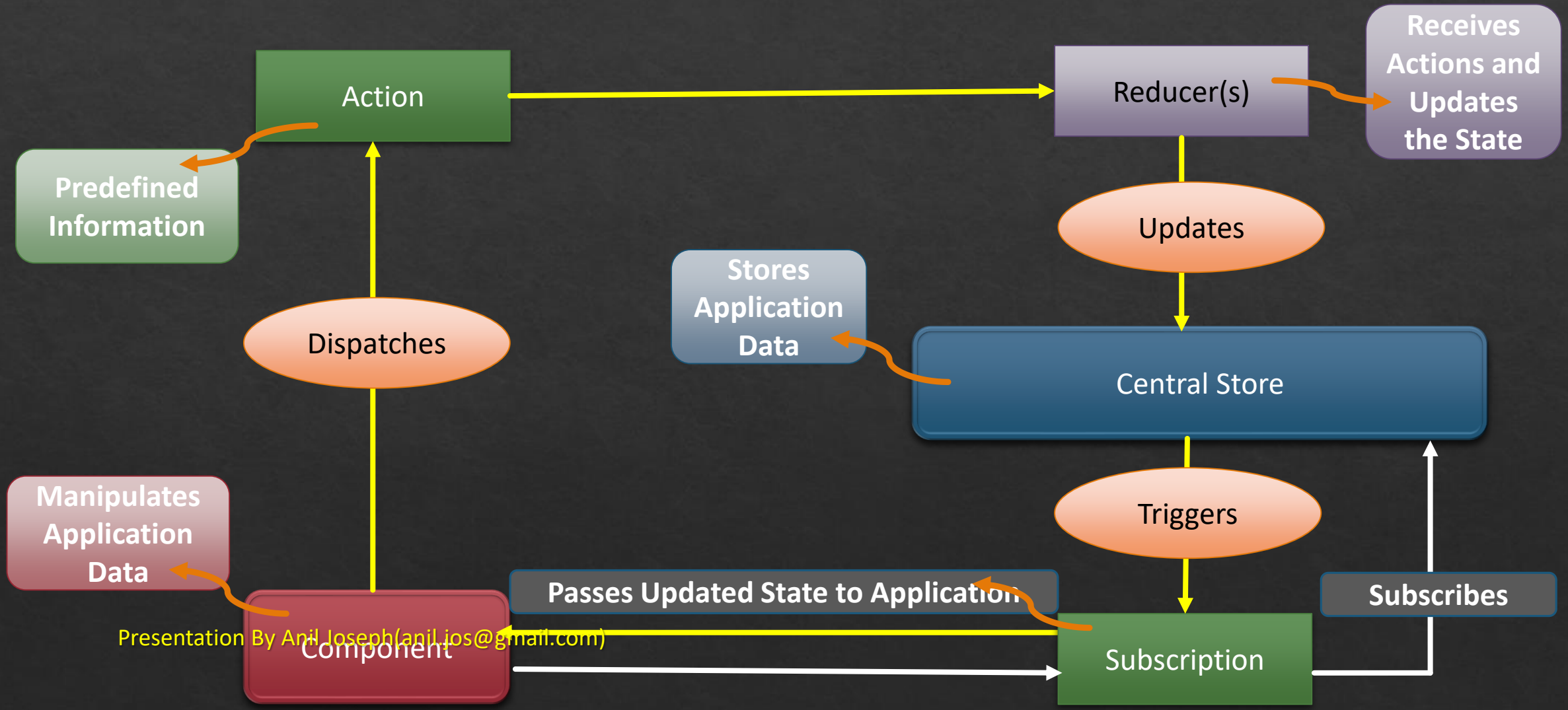
- Actions are events.
- Actions send data from the application (user interactions, internal events such as API calls, and form submissions) to the store.

Reducers

- In Redux, reducers are functions (pure)
- Its takes the current state of the application and an action and then return a new state.

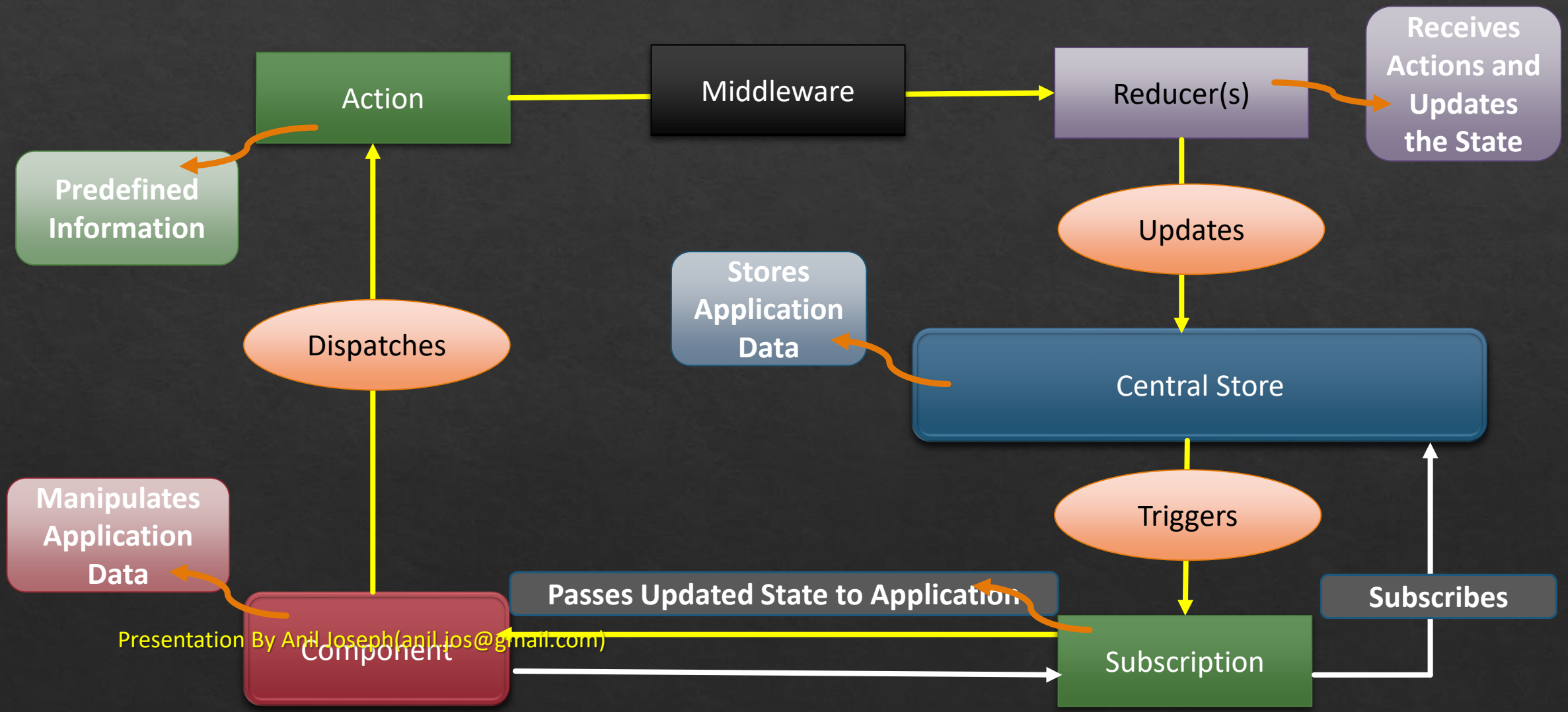
Presentation By Anil Joseph(anil.jos@gmail.com)

Redux Flow



Presentation By Anil Joseph(anil.jos@gmail.com)

Redux Flow



Presentation By Anil Joseph(anil.jos@gmail.com)

Redux React

Redux react is a library that integrates Redux to a React Application

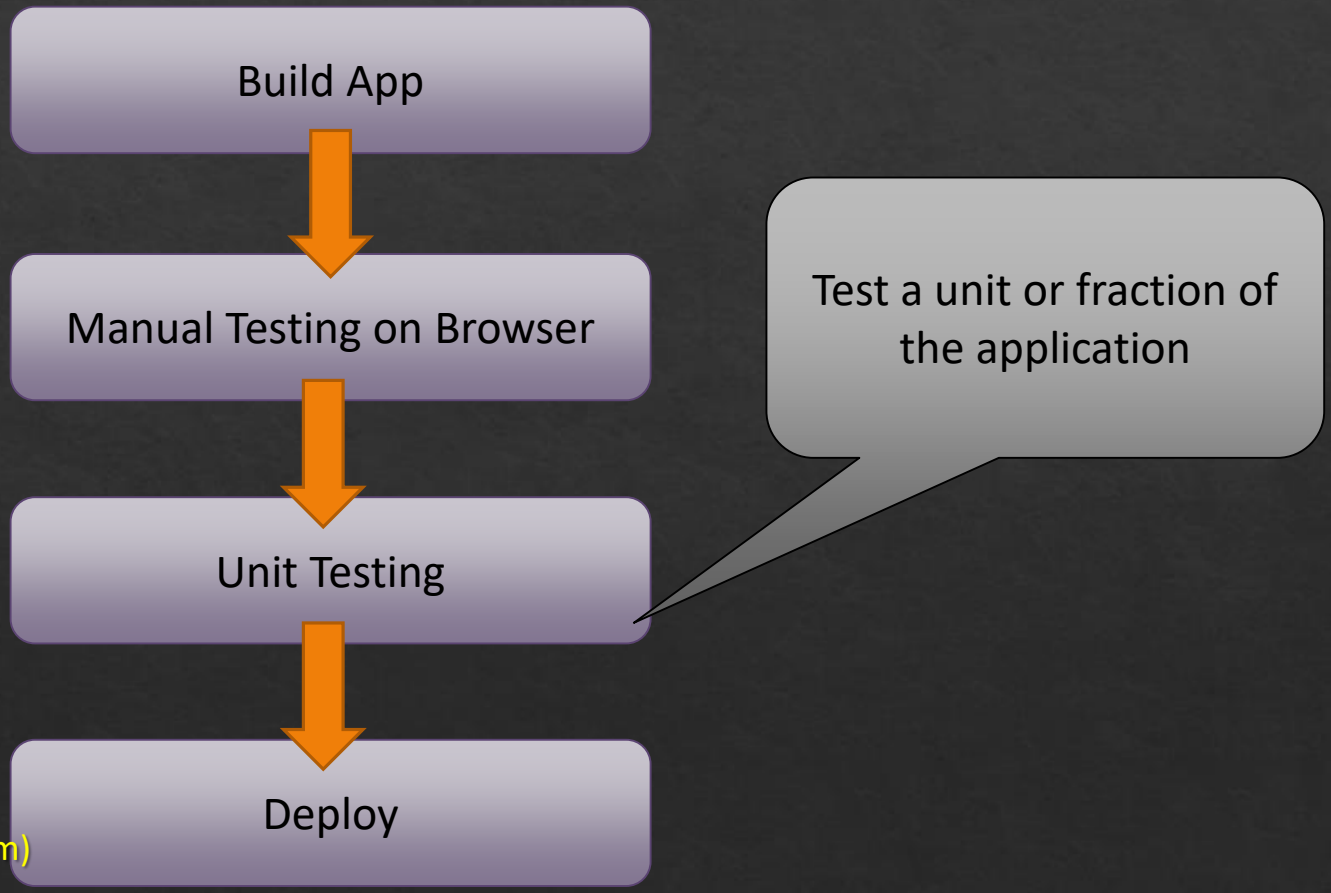
Comprises of Components & Functions

Installation

npm install react-redux

Presentation By Anil Joseph(anil.jos@gmail.com)

Testing



Presentation By Anil Joseph(anil.jos@gmail.com)

Testing Tools

Testing API
&
Test Runner

Write the Unit Test.
Executes the Unit Tests.

Jest
(Built over Jasmine)

Test Utilities

Simulate the React App

React Test Utils

Enzyme

Presentation By Anil Joseph(anil.jos@gmail.com)

What to Test?

Test Isolated Components

Test Conditional Outputs

Don’t Test Library Functions

Don’t Test Complex Connection

Presentation By Anil Joseph(anil.jos@gmail.com)

Installation

```
npm install enzyme react-test-renderer enzyme-adapter-react-16
```

Presentation By Anil Joseph(anil.jos@gmail.com)

React UI Components

- There are many UI Libraries available in the React Eco space
- Libraries
  - React-bootstrap
  - Material UI
  - Grommet
  - Ant Design React
  - React Desktop
  - React Belle

Presentation By Anil Joseph(anil.jos@gmail.com)

Deployment Steps

Set basePath of Router

<BrowserRouter basename="/app/">

Set homepage in package.json

"homepage": "/app"

Build and Optimize

npm run build

Server must Always serve index.html

Deploy build artifacts to server

Presentation By Anil Joseph(anil.jos@gmail.com)

Resources

React: https://reactjs.org/

Awesome React: https://github.com/enaqx/awesome-react

Tutorial: https://egghead.io/courses/start-learning-react

Presentation By Anil Joseph(anil.jos@gmail.com)