Introduction to react

Teaching Faculty: Umur INAN

Prepared by Muhyieddin **AL-TARAWNEH**, Umur **INAN**

REACT

- It is library for building components and other features for building a single page application.

- ReactDom:
  - is a package that provides DOM specific methods that can be used at the top level of a web app to enable an efficient way of managing DOM elements of the web page.

REACT

- One of the most popular libraries, with over 100,000 stars on GitHub.
- React is not a framework (unlike Angular).
- React is an open-source project created by Facebook.
- React is used to build user interfaces (UI) on the front end.

CREATE-REACT-APP

- It sets up the development environment so that we can use the latest JavaScript features and optimizes your app for production.
- Node >= 8.10 and npm >= 5.6
- it uses Babel and webpack

```
npm i -g create-react-app
```

Virtual DOM

- The virtual DOM (VDOM) is a programming concept where an ideal, or “virtual”, representation of a UI is kept in memory and synced with the “real” DOM by a library such as ReactDOM.
- This process is called reconciliation.

Virtual DOM

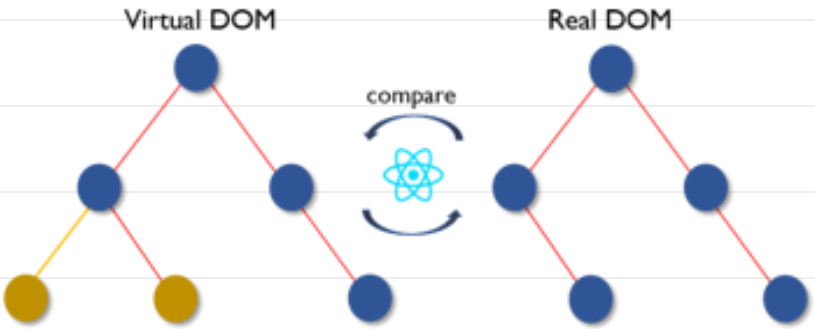
- When you use React, at a single point in time you can think of the render() function as creating a tree of React elements.
- On the next state or props update, that render() function will return a different tree of React elements.
- React then needs to figure out how to efficiently update the UI to match the most recent tree.

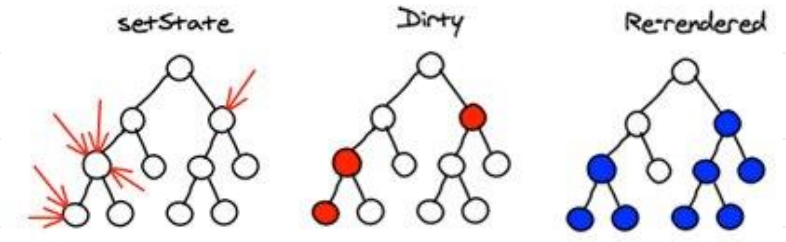
Virtual DOM update

- 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, React then compares the current version of the virtual DOM with the previous version of the virtual DOM. This process is called "diffing".

Virtual DOM update





React.createElement()

- It needs at least 3 arguments (component, props, ...children)
  - The element we want to render to DOM
  - Properties or an object for configuration
  - Children
- Configuration – Use camelCase naming standard:
  - Id
  - ClassName
  - style

React.createElement()

```
React.createElement('div', null, 'Hello World');
```

React elements

- React elements are **immutable**. Once you create an element, you can't change its children or attributes.
- The way to update the UI is to create a new element and pass it to *ReactDOM.render(element, root DOM)*.
- React Only Updates What's Necessary - React DOM compares the element and its children to the previous one, and only applies the DOM updates necessary to bring the DOM to the desired state.
- Unlike browser DOM elements, React elements are plain objects, and are cheap to create. React DOM takes care of updating the DOM to match the React elements.

JSX

- JSX just provides syntactic sugar for the React.createElement function.
- It is NOT a HTML. It is JavaScript!
- It is recommend using it with React to describe what the UI should look like.
- JSX may remind you of a template language, but it comes with the full power of JavaScript.

```
<p>  Hello World. This is my first React App.  </p>
```

JSX

- User-Defined Components Must Be Capitalized.
- When an element type starts with a lowercase letter, it refers to a built-in component like or and results in a string 'div' or 'span' passed to React.createElement.
- Must return one parent item. Not more than one.
- JSX Prevents Injection Attacks - Everything is converted to a string before being rendered. This helps prevent XSS (cross-site-scripting) attacks.

React components

- Building blocks of react app.
- React separates concerns with loosely coupled units called “components” that contain both the markup (HTML) and logic (JS).
- Components let you split the UI into independent, reusable pieces.
- Components are “made of” elements.
- There are 2 types of components:
  - Functional – Stateless, dumb, presentational. Preferred.
  - Class – Stateful, smart, containers. Should override render() method.

Stateful vs Stateless components

- When would you use a stateless component?
  - When you just need to present the props.
  - When you don't need a state, or any internal variables.
  - When creating element does not need to be interactive.
  - When you want reusable code.

Stateful vs Stateless components

- When would you use a stateful component?
  - When building element that accepts user input.
  - or element that is interactive on page .
  - When dependent on state for rendering, such as, fetching data before rendering.
  - When dependent on any data that cannot be passed down as props.

Functional Components

- 90% cleaner code than class components.
- Class components are verbose.
- Class components get compiled. The compiled code could be messy.
- More consistent and easier to test.
- Class components are more complex.

Functional Components

- Purely presentational.
- Represented by a function.
- Returns React element.
- Aka stateless, dumb, presentational

Functional Components

```
function Welcome(props) {
return (
        <p>
          Hello World. This is my first React App.
        </p>
        );
}
```

Class-Based Component

- Inherits from React.Component
- Should override render() method
- Aka containers, smart, stateful

Class-Based Component

```
class Table extends React.Component {
    render() {
      return (
          <p> Hello World. This is my first React App. </p>
      );
    }
}
```

PROPS

- Props are arguments passed to components.
- Can be used to pass data from the parent component to child components. (One way, parent -> child)
- They are passed via HTML attributes.
- Props are read-only!

PROPS

```
const hello= (props)=>{
 return (<h1> Hello {props.name} </h1>)
}
```

```
function App() {

return (
    <Hello name='Umur'></Hello>
 );
}
```

STATE

- State is used to change the component.
- Changes to state trigger an UI update.
- The state object is where you store property values that belongs to the component.
- You can store as many properties as needed.

USESTATE

- Returns a stateful value, and a function to update it.
- During the initial render, the returned state (state) is the same as the value passed as the first argument (initialState).

```
const [state, setState] = useState(initialState)
```

- The setState function is used to update the state. It accepts a new state value and enqueues a re-render of the component.

USESTATE

- If the new state is computed using the previous state, you can pass a function to setState.
- The function will receive the previous value and return an updated value.

```
const [count, setCount] = useState(initialState)
<button onClick={()=>setCount(initialCount)}> Reset </button>
<button onClick={()=>setCount(prevCount=> prevCount-1)}> - </button>
```

Main points

- A good design reflects re-usability and adaptability and most importantly traceability of requirements. A good web design promotes stability in the business application and flexibility in the presentation layer. The Field of Pure Creative Intelligence is characterized by the qualities of stability and flexibility.
- Transcendental consciousness is the experience of pure consciousness, the unified field of physics. Just by having this experience one gains this wholeness of knowledge and actions will be accord with all the laws of nature.