React Questions

* **Whats the difference between Real DOM & Virtual DOM?** 
  + Real DOM updates slow, can directly update HTML, Creates new DOM if element updates, DOM manipulation is very expensive, & has too much memory wastage
  + Virtual DOM updates faster, can’t directly update HTML, Updates the JSX if element updates, DOM manipulation is very easy, & no memory wastage
* **What is JSX?** 
  + JSX is a shorthand for Javascript XML. This is a type of file used by React which utilizes the expressiveness of Javascript along with HTML like template syntax
* **What is Virtual DOM & how does it work?** 
  + Virtual DOM is a lightweight Javascript object which originally is just the copy of the real DOM.
  + It is a node tree that lists the elements, their attributes & content as objects & their properties
  + React’s render function creates a node tree out of the React components, it then updates the tree in response to the mutations in the data model which is caused by various actions done by the user or by the system.
  + The process of comparing the Virtual DOM to Real DOM is called diffing
* **Why can’t browsers read JSX?** 
  + Browsers can only read Javascript objects but JSX is not a regular Javascript object
  + So we need to use a complier like Babel to transform JSX to a Javascript Object for the browser to read
* **How different is Reacts ES6 syntax compared to ES5?** 
  + Syntax has changed from ES5 to ES6 in following

1. Require vs import & export vs exports

2. The way component & functions are written

3. The way props & state are written

* **How is React different from Angular?** 
  + Architecture, React only the View of MVC; Angular Complete MVC
  + Rendering, React is server-side rendering; Angular Client-side rendering
  + DOM, React uses virtual DOM; Angular uses real DOM
  + Data binding, React one-way data binding; Angular two-way data binding
  + Debugging, React compile time debugging; Angular runtime debugging
* **What are Props & how are they used in react?** 
  + Props is the shorthand for Properties in React. They are read-only components which must be kept pure i.e immutable
  + They are passed down from parent to child components throughout an application
  + A child component can never send props back to a parent component because react uses unidirectional data flow & are generally used to render the dynamically generated data
* **What is state in React & how is it used?** 
  + States are the objects which determine components rendering & behavior, they are mutable unlike the props & create dynamic & interactive components
* **How can you update the state of a component?** 
  + States of a component can be updated using this.setState()
* **What is arrow function in React & how is it used?** 
  + Arrow functions are more of brief syntax for writing the function expression.
  + These functions allow to bind the context of the components properly since in ES6 auto bending is not available by default
  + Arrow functions are mostly useful while working with the higher order functions
* **What is an event in React?** 
  + In React, event are the triggered reactions to specific actions like mouse hover, mouse click, key press, etc
  + Handling these events are similar to handling events in DOM elements but with some syntactical difference likes

1. Events are named using camel case instead of just using the lowercase

2. Event are passed as functions instead of strings

* **What are synthetic event in React?** 
  + Synthetic events are the objects which act as a cross-browser wrapper around the browsers native events
  + They combine the behavior of different browser into one API
  + This is done to make sure that even show consistent properties across different browsers
* **What are the different phases of React component’s lifecycle?** 
  + The different phases of React are

1. **Mounting Phase**, this is the phase when the component is about to start its life journey & make its way to the DOM

2. **Update Phase**, once the component gets added to the DOM, it can potentially update & re-render when a prop or state change occurs

3. **Unmounting Phase**, this is the final phase of a component’s lifecycle in which the component is destroyed & removed from the DOM

4. **Error Handling**, sometimes code doesn’t run or there’s a bug somewhere

* **What are the lifecycle methods of React components?** 
  + Different lifecycle methods are available in different phases of a react component
  + In the **mounting phase** lifecycle method’s available are

1. **Constructor()**, this is the very first method call before the component is mounted to the DOM

2. **Static getDerivedStateFromProps()**, this method runs before the render & allows a component to update its internal state in response to a change in props

3. **Render()**, is used to render element to the DOM its where JSX is returned & it should be a pure function

4. **ComponentDidMount()**, which runs after the render is called its good place to make network request as soon as the component is mounted to the DOM

* + In the **updating phase** lifecycle method’s available are

1. **Static getDerivedStateFromProps(),** its the first method invoked andhas the same purpose as in the mounting phase

2. **ShouldComponentUpdate()**, is the next method called in this lifecycle method you can return a boolean & control weather the component gets re-rendered or not

3. **Render()**, is called immediately afterwards depending on the return call from shouldComponentUpdate() which defaults to true

4. **getSnapshotBeforeUpdate()**, is called after the render method its used if you need to grab some information from the DOM & potentially change it just after an update is made

5. **ComponentDidUpdate(),** this lifestyle method is invoked after the getSnapshotBeforeUpdate is invoked. As with the getSnapshotBeforeUpdate it receives the previous props & state as arguments and its used in conjunction with getSnapshotBeforeUpdate

* + In the **unmounting phase** lifecycle method’s available are

1. **componentWillUnmount(),** this lifecycle method is invoked immediately before a component is unmounted & destroyed, it ideal place to perform any necessary cleanup such as clearing up timers or cancelling any network request

* + In the **error handling** lifecycle method’s available are

1. **Static getDerivedStateFromError()**, whenever an error is thrown in a descendant component, this method is called first & the error thrown passed as an argument

2. **ComponentDidCatch()**, used to send error to external log

* **What do you understand by refs in React?**
  + Refs as in reference are attributes which help store reference to a particular react element or component, which will be returned by the components render configuration function
  + Its used to return references to a particular element or component returned by render()
  + They come in handy when we need DOM measurements or to add methods to components
* **List some of the cases when you should use Refs?**
  + Some of the use cases for refs are

1. When you need manage focus, select text or media playback

2. To trigger imperative animations

3. Integrate with third-party DOM libraries

* **How do you modularize code in React?**
  + We can modularize code by using the export & import properties, they help in writing the components separately in different files
* **How are forms created in React?**
  + React forms are similar to HTML forms, but in React the state is contained in the state property of the component & is only updated via setState()
  + The element can’t directly update their states & their submission is handled by a Javascript function
* **What the difference between controlled & uncontrolled components?**
  + Controlled component don’t maintain their own state, data is controlled by the parent component & they take in the current values through props & then notify the changes via callbacks
  + Uncontrolled components maintain their own state, data is controlled by the DOM, & Refs are used to get their current values
* **What are Higher Order Components(HOC)?**
  + Higher Order Component is an advanced way reusing the component logic, basically its a pattern that is derived from Reacts compositional nature
  + HOC are custom components which wrap another component within it, they can dynamic provide child component but they won’t modify or copy any behavior from their input component
* **What are Pure Components?**
  + Pure components are the simplest & fastest components which can be written
  + They can replace any component which only has a render(), these components enhance the simplicity of the code & performance of the application
* **What is the significance of Keys in React?**
  + Keys are used to identifying unique Virtual DOM Elements with their corresponding data driving the UI
  + They help React to optimize the rendering by recycling all the existing elements in the DOM
  + These keys must be a unique number of string, using which React just reorders the elements instead of re-rendering them this improves application performance