1. List some of the cases when you should use Refs.

Following are the cases when refs should be used:

- **When you need to manage focus, select text or media playback**
- **To trigger imperative animations**
- **Integrate with third-party DOM libraries**

2. How do you modularize code in React?

We can modularize code by using the export and import properties. They help in writing the components separately in different files.

```
1     //ChildComponent.jsx
2     export default class ChildComponent extends React.Component {
3        render() {
4           return(
5      <div>
6      <h1>This is a child component</h1>
7        </div>
8           );
9        }
10    }
11
12    //ParentComponent.jsx
13    import ChildComponent from './childcomponent.js';
14    class ParentComponent extends React.Component {
15       render() {
16          return(
17          <div>
18            <App />
19              </div>
20          );
21       }
22    }
```

3. 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 and is only updated via setState(). Thus the elements can't directly update their state and their submission is handled by a JavaScript function. This function has full access to the data that is entered by the user into a form.

```
handleSubmit(event) {
    alert('A name was submitted: ' + this.state.value);
    event.preventDefault();
}
 render() {
    return (
    <form onSubmit={this.handleSubmit}>
          <label>
             Name:
 <input type="text" value={this.state.value} onChange={this.handleSubmit}
/>
          </label>
           <input type="submit" value="Submit" />
       </form>
 );
}
```

4. What do you know about controlled and uncontrolled components?

Controlled Components	Uncontrolled Components
1. They do not maintain their own state	1. They maintain their own state
2. Data is controlled by the parent component	2. Data is controlled by the DOM
3. They take in the current values through props and then notify the changes via callbacks	3. Refs are used to get their current values

5. What are Higher Order Components(HOC)?

Higher Order Component is an advanced way of reusing the component logic. Basically, it's a pattern that is derived from React's compositional nature. HOC are custom components which wrap another component within it. They can accept any dynamically provided child component but they won't modify or copy any behavior from their input components. You can say that HOC are 'pure' components.

6. What can you do with HOC?

HOC can be used for many tasks like:

- **Code reuse, logic and bootstrap abstraction**
- **Render High jacking**
- **State abstraction and manipulation**
- **Props manipulation**

7. What are Pure Components?

***Pure* components are the simplest and fastest components which can be written. They can replace any component which only has a render(). These components enhance the simplicity of the code and performance of the application.**

8. What is the significance of keys in React?

Keys are used for 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 or string, using which React just reorders the elements instead of re-rendering them. This leads to increase in application's performance.

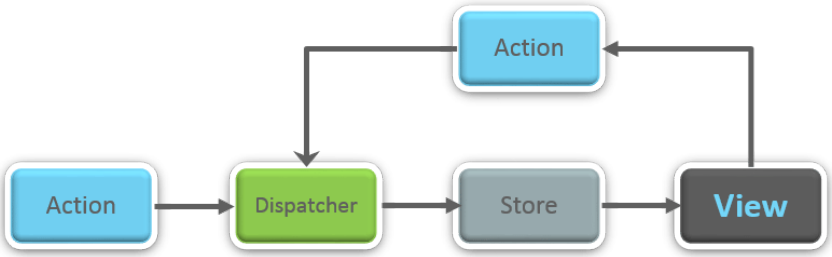
9. What were the major problems with MVC framework?

Following are some of the major problems with MVC framework:

- **DOM manipulation was very expensive**
- **Applications were slow and inefficient**
- **There was huge memory wastage**
- **Because of circular dependencies, a complicated model was created around models and views**

10. Explain Flux.

Flux is an architectural pattern which enforces the uni-directional data flow. It controls derived data and enables communication between multiple components using a central Store which has authority for all data. Any update in data throughout the application must occur here only. Flux provides stability to the application and reduces run-time errors.

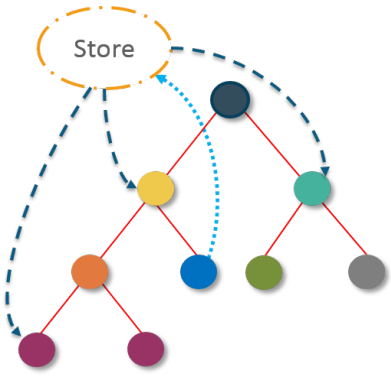


11. What is Redux?

Redux is one of the most trending libraries for front-end development in today's marketplace. It is a predictable state container for JavaScript applications and is used for the entire applications state management. Applications developed with Redux are easy to test and can run in different environments showing consistent behavior.

12. What are the three principles that Redux follows?

1. ***Single source of truth:*** **The state of the entire application is stored in an object/ state tree within a single store. The single state tree makes it easier to keep track of changes over time and debug or inspect the application.**
2. ***State is read-only:*** **The only way to change the state is to trigger an action. An action is a plain JS object describing the change. Just like state is the minimal representation of data, the action is the minimal representation of the change to that data.**
3. ***Changes are made with pure functions:*** **In order to specify how the state tree is transformed by actions, you need pure functions. Pure functions are those whose return value depends solely on the values of their arguments.**



13. What do you understand by "Single source of truth"?

Redux uses 'Store' for storing the application's entire state at one place. So all the component's state are stored in the Store and they receive updates from the Store itself. The single state tree makes it easier to keep track of changes over time and debug or inspect the application.

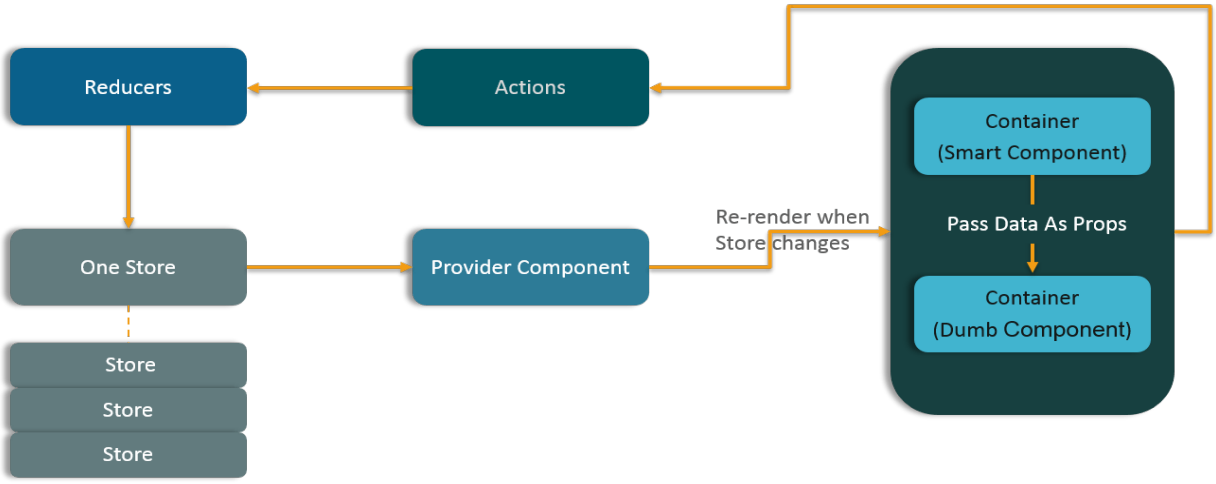
14. List down the components of Redux.

Redux is composed of the following components:

- **Action – It's an object that describes what happened.**
- **Reducer – It is a place to determine how the state will change.**
- **Store – State/ Object tree of the entire application is saved in the Store.**
- **View – Simply displays the data provided by the Store.**

In case you are facing any challenges with these React interview questions, please comment on your problems in the section below.

15. Show how the data flows through Redux?



16. How are Actions defined in Redux?

Actions in React must have a type property that indicates the type of ACTION being performed. They must be defined as a String constant and you can add more properties to it as well. In Redux, actions are created using the functions called Action Creators. Below is an example of Action and Action Creator:

```
1    function addTodo(text) {
2        return {
3            type: ADD_TODO,
4            text        }}
```

17. Explain the role of Reducer.

Reducers are pure functions which specify how the application's state changes in response to an ACTION. Reducers work by taking in the previous state and action, and then it returns a new state. It determines what sort of update needs to be done based on the type of the action, and then returns new values. It returns the previous state as it is, if no work needs to be done.

18. What is the significance of Store in Redux?

A store is a JavaScript object which can hold the application's state and provide a few helper methods to access the state, dispatch actions and register listeners. The entire state/ object tree of an application is saved in a single store. As a result of this, Redux is very simple and predictable. We can pass middleware to the store to handle the processing of data as well as to keep a log of various actions that change the state of stores. All the actions return a new state via reducers.

19. How is Redux different from Flux?

Flux	Redux
1. The Store contains state and change logic	1. Store and change logic are separate
2. There are multiple stores	2. There is only one store
3. All the stores are disconnected and flat	3. Single store with hierarchical reducers
4. Has singleton dispatcher	4. No concept of dispatcher
5. React components subscribe to the store	5. Container components utilize connect
6. State is mutable	6. State is immutable

In case you are facing any challenges with these React interview questions, please comment on your problems in the section below.

20. What are the advantages of Redux?

Advantages of Redux are listed below:

Predictability of outcome – Since there is always one source of truth, i.e. the store, there is no confusion about how to sync the current state with actions and other parts of the application.

Maintainability – The code becomes easier to maintain with a predictable outcome and strict structure.

Server-side rendering – You just need to pass the store created on the server, to the client side. This is very useful for initial render and provides a better user experience as it optimizes the application performance.

Developer tools – From actions to state changes, developers can track everything going on in the application in real time.

Community and ecosystem – Redux has a huge community behind it which makes it even more captivating to use. A large community of talented individuals contribute to the betterment of the library and develop various applications with it.

Ease of testing – Redux's code is mostly functions which are small, pure and isolated. This makes the code testable and independent.

Organization – Redux is precise about how code should be organized, this makes the code more consistent and easier when a team works with it.

21. What is React Router?

React Router is a powerful routing library built on top of React, which helps in adding new screens and flows to the application. This keeps the URL in sync with data that's being displayed on the web page. It maintains a standardized structure and behavior and is used for developing single page web applications. React Router has a simple API.

22. Why is switch keyword used in React Router v4?

Although a is used to encapsulate multiple routes inside the Router. The 'switch' keyword is used when you want to display only a single route to be rendered amongst the several defined routes. The <switch> tag when in use matches the typed URL with the defined routes in sequential order. When the first match is found, it renders the specified route. Thereby bypassing the remaining routes.

23. Why do we need a Router in React?

A Router is used to define multiple routes and when a user types a specific URL, if this URL matches the path of any 'route' defined inside the router, then the user is redirected to that particular route. So basically, we need to add a Router library to our app that allows creating multiple routes with each leading to us a unique view.

```
<switch>
    <route exact path='/' component={Home}/>
    <route path='/posts/:id' component={Newpost}/>
    <route path='/posts'  component={Post}/>
</switch>
```

24. List down the advantages of React Router.

Few advantages are:

- **Just like how React is based on components, in React Router v4, the API is *'All About Components'*. A Router can be visualized as a single root component (<BrowserRouter>) in which we enclose the specific child routes (<route>).**
- **No need to manually set History value: In React Router v4, all we need to do is wrap our routes within the <BrowserRouter> component.**
- **The packages are split: Three packages one each for Web, Native and Core. This supports the compact size of our application. It is easy to switch over based on a similar coding style.**

25. How is React Router different from conventional routing?

Topic	Conventional Routing	React Routing
PAGES INVOLVED	Each view corresponds to a new file	Only single HTML page is involved
URL CHANGES	A HTTP request is sent to a server and corresponding HTML page is received	Only the History attribute is changed
FEEL	User actually navigates across different pages for each view	User is duped thinking he is navigating across different pages