Redux

Interview Questions

Redux

1- WHAT IS REDUX ?

Redux is a predictable state container for JavaScript apps based on the Flux design pattern.
Redux can be used together with ReactJS, or with any other view library.

2- WHAT IS FLUX ?

Flux is an application design paradigm used as a replacement for the more traditional mvc pattern.
It is not a framework or a library but a new kind of architecture that complements React and the concept of Unidirectional Data Flow.
Facebook used this pattern internally when working with React
The workflow between dispatcher, stores and views components with distinct inputs and outputs.

Redux

3- WHAT IS REDUX DEVTOOLS ?

Redux DevTools is a live-editing time travel environment for redux with hot reloading, action replay, and customizable UI. If you don’t want to bother with installing Redux DevTools and integrating it into your project, consider using Redux DevTools Extension for Chrome and Firefox.

4- WHAT IS THE DIFFERENCE BETWEEN COMPONENT AND CONTAINER IN REDUX ?

- Component is part of the React API. A Component is a class or function that describes part of a React UI.
- Container is an informal term for a React component that is connected to a redux store. Containers receive Redux state updates and dispatch actions, and they usually don't render DOM elements; they delegate rendering to presentational child components.

5- WHAT ARE REDUCERS IN REDUX ?

Redux

The reducer is a pure function that takes the previous state and an action, and returns the next state.

(previousState, action) => newState

It's called a reducer because it's the type of function you would pass to Array.prototype.reduce(reducer, ?initialValue). It's very important that the reducer stays pure. Things you should never do inside a reducer:
Mutate its arguments;
Perform side effects like API calls and routing transitions;
Call non-pure functions, e.g. Date.now() or Math.random().

Redux

6- WHAT IS REDUX-SAGA ?

redux-saga is a library that aims to make side effects (i.e. asynchronous things like data fetching and impure things like accessing the browser cache) in React/Redux applications easier and better. It is available in NPM as

```
npm install --save redux-saga
```

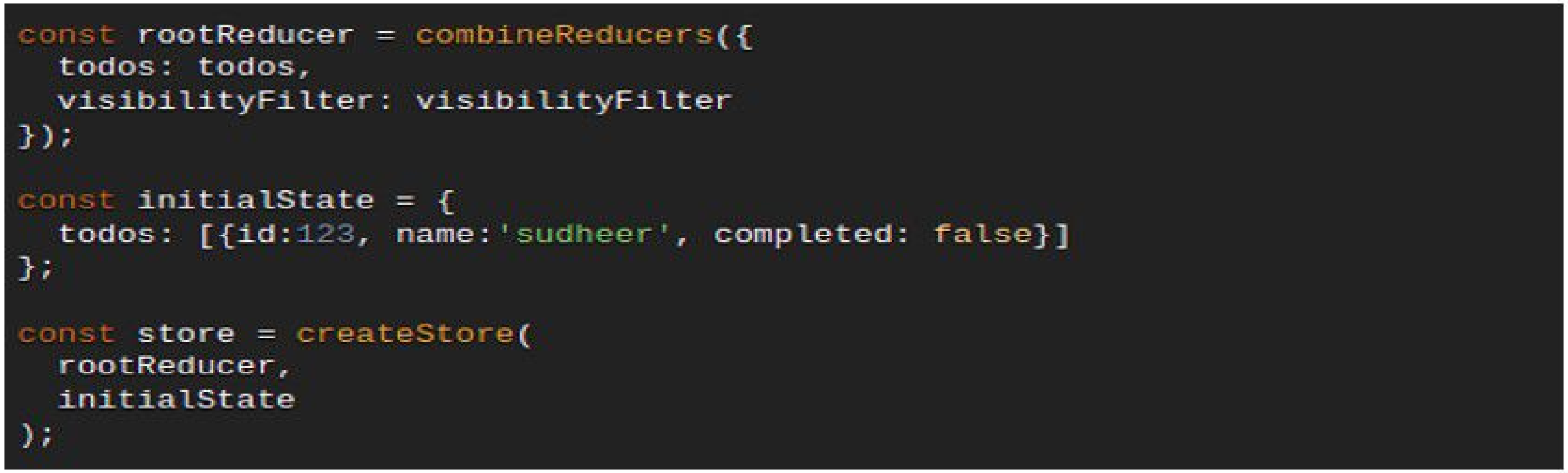
7- WHAT IS REDUX THUNK ?

Redux Thunk middleware allows you to write action creators that return a function instead of an action. The thunk can be used to delay the dispatch of an action, or to dispatch only if a certain condition is met. The inner function receives the store methods dispatch and getState() as parameters.

Redux

8- HOW TO SET INITIAL STATE IN REDUX ?

You need to pass initial state as second argument to createStore



```
const rootReducer = combineReducers({
  todos: todos,
  visibilityFilter: visibilityFilter
});

const initialState = {
  todos: [{id:123, name:'sudheer', completed: false}]
};

const store = createStore(
  rootReducer,
  initialState
);
```

9- WHAT ARE REDUX SELECTORS AND WHY TO USE THEM ?

Selectors are functions that take Redux state as an argument and return some data to pass to the component. For example, to get user details from the state:

const getUserData = state => state.user.data;

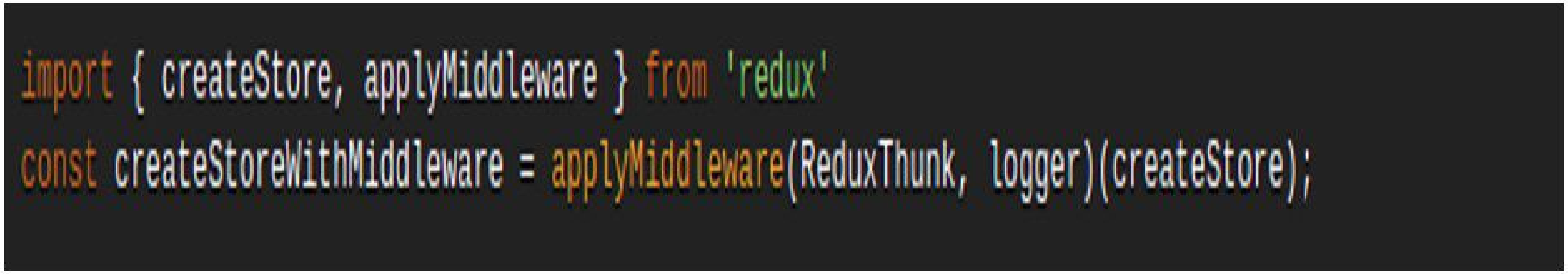
Redux

10- WHAT IS THE DIFFERENCE BETWEEN REACT CONTEXT AND REACT REDUX ?

You can use Context in your application directly and is going to be great for passing down data to deeply nested components which what it was designed for. Whereas Redux is much more powerful and provides a large number of features that the Context Api doesn't provide.
Also, React Redux uses context internally but it doesn’t expose this fact in the public API. So you should feel much safer using context via React Redux than directly because if it changes, the burden of updating the code will be on React Redux instead developer responsibility.

11- HOW TO ADD MULTIPLE MIDDLEWARES TO REDUX ?

You can use applyMiddleware where you can pass each piece of middleware as a new argument. So you just need to pass each piece of middleware you'd like. For example, you can add Redux Thunk and logger middlewares as an argument as below,



```
import { createStore, applyMiddleware } from 'redux'
const createStoreWithMiddleware = applyMiddleware(ReduxThunk, logger)(createStore);
```

Redux

12- WHAT ARE THE FEATURES OF REDUX DEVTOOLS ?

Below are the major features of Redux devTools

1. Lets you inspect every state and action payload
2. Lets you go back in time by “cancelling” actions
3. If you change the reducer code, each “staged” action will be re-evaluated
4. If the reducers throw, you will see during which action this happened, and what the error was
5. With persistState() store enhancer, you can persist debug sessions across page reloads

Redux

13- HOW TO STRUCTURE REDUX TOP LEVEL DIRECTORIES ?

Most of the applications has several top-level directories as below

1. Components Used for “dumb” React components unaware of Redux
2. Containers Used for “smart” React components connected to Redux
3. Actions Used for all action creators, where file name corresponds to part of the app 4. Reducers Used for all reducers, where file name corresponds to state key 5. Store Used for store initialization This structure works well for small and mid-level size apps.

Redux

14- WHAT ARE THE CORE PRINCIPLES OF REDUX ?

Redux follows three fundamental principles:

1. Single source of truth:
The state of your whole application is stored in an object 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 ready only:
The only way to change the state is to emit an action, an object describing what happened. This ensures that neither the views nor the network callbacks will ever write directly to the state.
3. Changes are made with pure functions:
To specify how the state tree is transformed by actions, you write pure reducers(Reducers are just pure functions that take the previous state and an action, and return the next state).