**Core React**

**What are props in React?**

A: props are properties that are passed into a child component from its parent, and are readonly.

**What's the difference between component props and state?**

A: Props are readonly and passed in from parent to child component; state is an internal object for a particular react component and can change, as it determines the state of the component. It is not visible to other components.

**What is a pure component?**

A: A pure (functional) component in React is stateless (aka a dumb component) and always renders the same given the same set of input props.

**How do we pass a property from a parent component props to a child component?**

A: For example: <ChildComponent someProp={props.someProperty} />

**What is JSX?**

A: JavaScript with extensions. Useful for creating React elements with the full power of JavaScript. Not required for a React app, but typically it's used.

**What are some lifecycle methods in a React component?**

A: componentDidMount, componentWillMount, componentWillUpdate, componentDidUpdate, shouldComponentUpdate, componentWilReceiveProps.

**What's the difference between a 'smart' component and a 'dumb' component?**

A: Smart components manage their state or in a Redux environment are connected to the Redux store. Dumb components are driven completely by their props passed in from their parent and maintain no state of their own.

**What would be a good lifecycle method to make a remote call to fetch data for a component?**

A: componentDidMount

**What does it mean for a component to be mounted in React?**

A: It has a corresponding element created in the DOM and is connected to that.

**What's a pure functional component in React?**

A: A component that has no internal state of its own, nor any side effects, and thus is often written as a function as opposed to an ES6 class.

**What are some recent changes in the React library (e.g. in version 14, 15)?**

A: Splitting out of ReactDOM into react-dom, introduction of refs, improved warnings + error messages...

**What is the render method for?**

A: To determine what should be rendered for a particular component. Could be a complex nested object of other child React components, or it could be as simple as a primitive JS object.

**What are some limitations of things you shouldn't do in the component's render method?**

A: You cannot modify the component's state (with setState), nor interact with the browser (do that in componentDidMount). render should be a pure function.

**Why would you need to bind event handlers to this?**

A: You need to do this in order for 'this' to refer to the object instance of the React component class in your callback code, otherwise 'this' will be undefined. An alternative is to use arrow functions in your event handlers and 'this' will be initialized as expected.

**What's an alternative way to avoid having to bind to this in event callback methods?**

A: Use arrow functions in your event handlers.

**How do you prevent the default behavior in an event callback in React?**

A: You call e.preventDefault(); on the event e passed into the callback.

**How do you embed inline expressions in JSX?**

A: Wrap them in curly braces like so: <img src={user.avatarUrl}></img>

**How would you prevent a component from rendering in React?**

A: Return null from the render method.

**What is the point of using keys in React?**

A: It allows for more efficient rendering of lists, so that React can reuse DOM elements without having to destroy + recreate them when lists change (slightly) in the UI.

**What's the typical pattern for rendering a list of components from an array of data?**

A: Call map on an array with an arrow function that executes for each array element, possibly outputing a React component for each.

**What's the difference between a controlled component and an uncontrolled one in React?**

A: A controlled component has its state completely driven by React, whereas uncontrolled components can maintain their own internal state. E.g., a textarea's value.

**What are some basic form elements that could be controlled or uncontrolled?**

A: textarea, input, select

**What's different about a React textarea element?**

A: Uses a value attribute to define its text.

**What happens when you call setState?**

A: The state property is updated in a React component with the object passed into setState, and this is done asynchronously. It tells React that this component and its children need to be re-rendered, but React may not do this immediately (it may batch these state update requests for better performance).

**What is reconciliation in React?**

A: It's React's process of re-rendering its tree of UI components.

**What's the difference between an Element and a Component in React?**

A: Elements are the fundamental building blocks of React, and describe what you want to see on the screen. They are just simple JS objects with props, key, ref, and type properties, whereas Components have a render method and optionally accept inputs.

**What are refs in React and what are they used for?**

A: Refs are React's "escape hatch" mechanism for a component to reference another component outside of the typical data flow. This could be in order to correctly integrate with third party libraries, change focus on another component in the UI, triggering animations, etc.

**Why does React use SyntheticEvents?**

A: For consistent, cross-browser event handling (SyntheticEvent is a wrapper around the native browser event object).

**What's the difference between createElement and cloneElement?**

A: As the names imply, createElement creates a new React Element, and cloneElement clones an existing one (and is faster - useful if you don't actually have to change anything in the original Element).

**What's a higher order component in React?**

A: It's an advanced technique / pattern in React for reusing component logic.

**Why would you use forceUpdate in a React component?**

A: In order to force a re-render if there is some condition React is not detecting that requires an update to the UI. Typically this should not be necessary to call.

**What is the point of shouldComponentUpdate() method?**

A: It's used for performance reasons, for example if the implementor of a component knows for sure that a particular property change does not necessitate a re-render, they could return false from this method and skip the re-render.

**What are PropTypes in React?**

A: They help indicate to React what data types a React component's properties are and should accept.

**Explain the Virtual DOM concept in React.**

A: React builds up its own "virtual DOM" which is a lightweight representation of the DOM optimized for React's diffing algorithms and reconciliation process. Virtual DOM changes eventually propagate to the actual DOM at the end of the reconciliation process.

**How would you write an inline style in React?**

A: For example: <div style={{ height: 10 }}>

**What is ReactDOM?**

A: It's a top-level React API to render a React element into the DOM, via the ReactDOM.render method.

**If you need to access the underlying DOM node for a React component, what's the typical way to do this in React?**

A: ReactDOM.findDOMNode(component)

**What is the React context?**

A: It's an experimental API that allows you to pass data down through a tree of components without having to use props.

**What is React Fiber?**

A: A rewrite under the covers of React to make it as responsive as possible, while maintaining backwards compatibility.

**What are some new features in react 16?**

A: fragments, error boundaries, portals, improved server-side rendering and reduced file size.

**What does ReactDOM.hydrate do?**

A: Hydrates a container whose contents were rendered by ReactDOMServer, and warns about differences between client + server side rendering.

**What are some approaches to avoid "callback hell" when making server calls?**

A: Use of generator functions, redux-saga, effects.

**How would you write "Hello World" in React?**

A: ReactDOM.render( <h1>Hello, world!</h1>, document.getElementById('root') );

**How does React prevent injection attacks?**

A: React DOM escapes any values embedded in JSX before rendering them to help prevent cross site scripting attacks.

**What is the "class" attribute name when used by a React component?**

A: className

**Can a React component modify its props?**

A: No, or at least it's not supposed to.

**Why is it not a good idea to modify a component's state directly, but rather call setState()?**

A: There are some optimizations around the use of setState, for example multiple calls to setState may be batched / executed more efficiently.

**How do you prevent the default behavior when handling an action from a React component?**

A: You would call e.preventDefault() on the synthetic event passed into a component's callback.

**What is one common way a component can effectively hide itself?**

A: It can return null from its render method.

**Redux and Related Concepts**

**What are typical middleware choices for handling asynchronous calls in Redux?**

A: Redux Thunk, Redux Promise, Redux Saga

**What is the point of Redux?**

A: Application state management that is easy to reason about, maintain and manage in an asynchronous web application environment.

**Where is the state kept in a React + Redux application?**

A: In the store.

**What are the 3 fundamental principles of Redux?**

A: Immutability of the application state, no side-effects in reducers, ?

**What's the typical flow of data like in a React + Redux app?**

A: Callback from UI component dispatches an action with a payload, which then is intercepted in a reducer, possibly producing a new application state, which is then propagated down through the tree of components in the application from the Redux store.

**What is mapStateToProps and mapDispatchToProps?**

A: Functions typically seen in Redux applications that provide functions to Redux on how to map state / dispatch function to a set of props.

**What are thunks?**

A: Functions (that return other functions) which are useful for handling asynchronous / delayed dispatch of actions in the context of a Redux application.

**What is Flux?**

A: Unidrectional application flow paradigm popular a few years back in React; mostly superceded by Redux these days.

**What is Mobx?**

A: Simple application state management revolving around the use of observables / reactive programming.

**Other Popular React Libraries / Ecosystem**

**What is Yarn and how is it useful?**

A: Package manager that guarantees dependencies will be installed in the exact same way on any machine. It caches previously downloaded packages and does most actions concurrently for better performance.

**What is Babel and how is it often used in the context of a React application?**

A: Babel is a JS compiler, and is often used for things like translating JSX into JS, or translating ES6 JS code into ES5 for broader browser support.

**what is GraphQL?**

A: A library that includes a query language + type system for fetching data efficiently (in one request vs multiple requests in a typical RESTful API) for applications from an API.

**What is Flow?**

A: Static type checker for JavaScript, often used in React projects for improved code quality and maintainability.

**what is Enzyme?**

A: A popular JS testing library for React, from the folks at Airbnb.

**What is Material UI?**

A: An open source project of React components that implement Google's Material design.

**What is Storybook?**

A: A development environment for UI components. Allows interactive development of components, component library browsing, etc.

**What does a JS toolchain typically consist of for a complex React application?**

A: a package manager (e.g. Yarn), a bundler (e.g. webpack), and a compiler (e.g. Babel).

**What would you use redux-saga for?**

A: Easier to read asynchronous code handling, avoiding "callback hell" (i.e. deeply nested callbacks).

**What is reselect?**

A: A library that supports memoizing selectors in the context of a React / Redux application. Basically this means higher performing code by avoiding unnecessary recomputation of values that have ultimately not changed.

**What is normalizr?**

A: A library which normalizes nested JSON data according to some schema. The point is to reduce redundant data sent to the client.

**What is Relay?**

A: It's a framework for data fetching functionality using GraphQL, efficiently fetching specified data for a component.