# Career Services Assignment 9 – ReactJS Flash Cards

**Points possible:** 50

|  |  |  |
| --- | --- | --- |
| Category | Criteria | % of Grade |
| Completeness | All requirements of the assignment are complete. | 100 |

**Instructions:** Research common ReactJS interview questions online and create 20 flash cards from the information you find. Study your flash cards regularly to better prepare for interviews. Fill out the table below with the information you put on each of your flash cards.

|  |  |
| --- | --- |
| **Front of Card** | **Back of Card** |
| What are the main features of React? | 1. JSX-A syntax extension to JavaScript. It is used with React to describe what the user interface should look like. By using JSX, we can write HTML structures in the same file that contains JavaScript code. 2. Components-The building blocks of any React application, and a single app usually consists of multiple components. It splits the user interface into independent, reusable parts that can be processed separately 3. Virtual DOM-React keeps a lightweight representation of the real DOM in the memory, and that is known as the virtual DOM. When the state of an object changes, virtual DOM changes only that object in the real DOM, rather than updating all the objects. 4. One-way data binding-Keeps everything modular and fast. A unidirectional data flow means that when designing a React app, you often nest child components within parent components. 5. High performance-React updates only those components that have changed, rather than updating all the components at once. This results in much faster web applications.1 |
| Can web browsers read JSX directly? | Web browsers cannot read JSX directly. They are built to only read regular JS objects and JSX is not a regular JavaScript object. The file needs to be transformed into a regular JavaScript object. Babel is a tool for this.1 |
| What are the **DOM** and the **virtual DOM**? | DOM stands for Document Object Model. The DOM represents an HTML document with a logical tree structure. Each branch of the tree ends in a node, and each node contains objects. React keeps a lightweight representation of the real DOM in the memory, and that is known as the virtual DOM. When the state of an object changes, the virtual DOM changes only that object in the real DOM, rather than updating all the objects.1 |
| How is React different from Angular? | **React:** Made by Facebook, Architecture: View layer of MVC, Virtual DOM, Uni-directional databinding, server-side rendering.  **Angular:** Made by Google, Architecture: Complete MVC, Real DOM, Bi-directional databinding, client-side rendering. |
| What are the benefits of React over Angular? | Easy creation of dynamic applications, improved performance, reusable components, unidirectional data flow, and dedicated tools for easy debugging.1 |
| How do you create a React app? | 1. Install **NodeJS** on the computer because we need **npm** to install the React library. Npm is the node package manager that contains many JavaScript libraries, including React. 2. Install the create-react-app package using command prompt or terminal. 3. Open in text editor and start coding off of the included JS, HTML and CSS files.1 |
| What is an **event** in React? | An event is an action that a user or system may trigger, such as pressing a key, a mouse click, etc. React events are named using camelCase, rather than lowercase in HTML. With JSX, you pass a function as the event handler, rather than a string in HTML.1 |
| What are **synthetic events** in React? | Synthetic events combine the response of different browsers’ native events into one API, ensuring that the events are consistent across different browsers. The application is consistent regardless of the browser it is running in. An example is **preventDefault();**.1 |
| How do you write comments in React? | Single-line comments have a hash **#** in front. Multi-line comments use **/\***…**\*/** like CSS.1 |
| How do you traverse **lists** in React? | The traversal of lists is done using the **map()** function.1 |
| Why do **lists** needs **keys** in React? | A key is a unique identifier. It identifies which items have changed, been updated or deleted from the lists. It helps determine which components need to be re-rendered, instead of re-rendering all components every time, which increases performance.1 |
| What are **forms** in React? | Forms to enable users to interact with web applications and enter the required information needed. Forms contain certain elements, such as text fields, buttons, checkboxes, radio buttons, etc. They are also used for other tasks like user authentication, searching, filtering, indexing, etc.1 |
| What is an **arrow function** and how is it used in React? | An arrow function is a shorter way of writing a function in ES6. It does not require you to bind **‘this’** inside the constructor. They prevents bugs caused by the use of **‘this’** in callbacks.1 |
| How is React different from React Native? | React: Web only, uses HTML and CSS.  React Native: Mobile (Android & iOS) only, doesn’t use HTML or CSS.1 |
| What are the types of components in React? | There are two types. Stateless Functional components have no state of their own and only contain render methods. They may derive data from other components as props. Stateful Class components can hold and manage their own state and have a separate render method to return JSX on the screen.1 |
| What does **render()** do in React? | Each component needs **render()** to return HTML on the page. If you need to render more than one element, all of the elements must be inside one parent tag like <div> or <form>.1 |
| What is a **state** in React? | **State** is a built-in React object used to contain data or info about the component. It determines the behavior of the component and how it will render. **State** can change over time as a response to user action or system-generated events, and when it changes, the component re-renders.1 |
| What are **props** in React? | **Props**, short for Properties, is a built-in object that stores the value of attributes of a tag and works similarly to HTML attributes. **Props** provide a way to pass data from one component to another. They are passed to the component in the same way as arguments are passed in a function.1 |
| What are the differences between **state** and **props**? | **State**: Holds info about components, is mutable, can be changed, child components cannot access, and stateless components cannot have **state**.  **Props**: Allows you to pass data from one component to another as an argument, are immutable, cannot be changed, child components can access, and stateless components can have **props**.1 |
| What is a higher-order component in React? | A higher-order component acts as a container for other components. This helps to keep components simple and enables re-usability. They are generally used when multiple components have to use a common logic.1 |

**Sources**

1 <https://www.simplilearn.com/tutorials/reactjs-tutorial/reactjs-interview-questions>