# Glossary: Understanding Function Components with Array and DOM Manipulation

Welcome! This alphabetized glossary contains many of the terms you'll find within this course. This comprehensive glossary also includes additional industry-recognized terms not used in course videos. These terms are important for you to recognize when working in the industry, participating in user groups, and participating in other certificate programs.

| Term | Definition |
|---|---|
| **Abstraction principle** | A principle that provides a way to make reusable components that encapsulate UI features. |
| **Array** | A data structure in JavaScript used to store multiple values in a single variable. |
| **Array destructuring** | A feature that allows you to extract values from an array and assign them to individual variables. |
| **Array literal** | A syntax in JavaScript for creating arrays. It allows you to define an array by enclosing a comma-separated list of elements within square brackets. |
| **Attributes** | Provide additional information about elements. They can specify properties, styles, or behavior. |
| **Component composition** | A combination of multiple smaller components to create complex functionality. |
| **Document object model (DOM)** | An interface for web pages and documents used to represent an HTML structure as a tree-like structure. |
| **Elements** | Are the building blocks of HTML documents. |
| **forEach()** | A method that iterates over each element of an array and executes a callback function. |
| **Hierarchy principle** | A principle that allows you to arrange components in a hierarchy, with parent and child components. |
| **Higher-order component** | A function that allows you to add components' features, such as state management or logic, without modifying its implementation. |
| **Hook** | A function that enables you to reuse code logic across components without changing component hierarchy or introducing unnecessary nesting. |
| **Initialization** | A step in which React runs the function body of the functional component, setting up the initial structure and behavior of the component. |
| **map()** | A method commonly used to iterate over each element of an array and return a new array of React elements. |
| **Mounting phase** | A functional component lifecycle phase in which React initializes the functional component, preparing it for rendering on the DOM. |
| **Props Default** | Is default value for props using defaultProps. This ensures that if a prop is not provided, the component will still render with the default value, enhancing predictability and robustness in component behavior. |
| **Properties (props)** | A fundamental concept in React for passing data from parent to child components. They enable communication and customization between different parts of a React application. |
| **Rendering Array** | Involves dynamically generating and displaying elements based on the contents of the array. This is commonly done using JavaScript's map() function to iterate over the array and generate React elements for each item. |
| **Reuse principle** | A principle to reuse chunks of code, making it easier to organize and maintain. |
| **Side effects** | A step that includes data fetching, subscriptions, or DOM manipulation using the 'useEffect' hook with an empty dependency array. |
| **State** | The condition of a component at a specific time and it holds information that influences the object's behavior and renders it based on that information. |
| **State management** | Involves handling and updating the data over time, allowing components to dynamically change their appearance and behavior in response to user actions or other events. |
| **State initialization** | A step in which React utilizes the 'useState' hook to declare and initialize state variables within the component. |
| **Updating phase** | A functional component lifecycle phase in which React responds to changes in the component's state or props by re-invoking the function body of the component. |
| **Unmounting phase** | A functional component lifecycle phase in which React executes cleanup operations when removing a component from the DOM. |
| **useState hook** | A built-in React Hook that allows functional components to manage state. It provides a way to declare state variables and update them within a functional component. When the state changes, React automatically re-renders the component to reflect the updated state. |
| **Virtual DOM** | An abstraction of the actual DOM implemented in memory and is kept in sync with the real DOM by React's reconciliation process. |

Skills Network