DWA_12 Knowledge Check

To complete this Knowledge Check, ensure you have worked through all the lessons in **Module 12: Declarative Abstractions.**

To prepare for your session with your coach, please answer the following questions. Then download this document as a PDF and include it in the repository with your code.

---

1. What are the benefits of direct DOM mutations over replacing HTML?

Efficiency: Directly manipulating the DOM allows you to make specific changes to individual elements or attributes without having to recreate the entire HTML structure. This can be more efficient in terms of performance compared to replacing the entire HTML, especially when dealing with large or complex web pages.
Improved responsiveness: By directly manipulating the DOM, you can update the content dynamically in response to user interactions or other events without triggering a full page reload or refresh. This can result in a more seamless and responsive user experience, as the changes can be applied instantly without the need for server round trips.

---

2. What low-level noise do JavaScript frameworks abstract away?

Event Handling: Frameworks handle the cross-browser inconsistencies and boilerplate code associated with event handling. They provide standardized and streamlined approaches to attaching event listeners and handling user interactions, making it easier to respond to events like clicks, keystrokes, and form submissions.
State Management: Many frameworks offer built-in state management solutions that abstract away the complexity of managing application state. They provide mechanisms to store, update, and synchronize data across components or modules, reducing the need for manual state tracking and manipulation.

---

3. What essence do JavaScript frameworks elevate?

Scalability: JavaScript frameworks provide scalability through their modular and component-driven nature. They support the development of large-scale applications by enabling the separation of concerns, facilitating code reuse, and allowing for easy integration of additional features and modules. Frameworks also often include tools for code splitting, lazy loading, and asynchronous module loading, enabling efficient resource allocation and optimizing application performance as the project grows.
Maintainability: JavaScript frameworks emphasize code organization, modularization, and separation of concerns. They encourage the use of component-based architectures, where functionality is encapsulated within reusable and independent components. This promotes code reusability, simplifies code maintenance, and allows for easier collaboration among developers. Frameworks also often enforce coding standards and conventions, making codebases more consistent and maintainable.

4. Very broadly speaking, how do most JS frameworks achieve abstraction?

DOM Abstraction: Frameworks abstract away the low-level details of directly manipulating the Document Object Model (DOM). They provide APIs and methods that simplify the process of creating, updating, and deleting DOM elements, attributes, and event handlers. By providing a higher-level abstraction over the DOM, frameworks handle cross-browser inconsistencies, optimize performance, and provide a more intuitive programming interface.
Event Handling Abstraction: Frameworks abstract away the cross-browser inconsistencies and boilerplate code associated with event handling. They provide standardized and simplified approaches to attaching event listeners and handling user interactions. This abstraction simplifies event management and allows developers to focus on handling specific events rather than worrying about browser-specific quirks.

5. What is the most important part of learning a JS framework?

Is learning the Fundamentals: Start by grasping the fundamental concepts and principles of the framework. Understand how it handles components, state management, routing, and other core features. Familiarize yourself with the framework's syntax, architecture, and development patterns.