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?
[meaning: manipulating DOM elements using JS (e.g., createElement) vs directly updating the HTML file]

- **Better performance:** changing the DOM directly is more efficient in terms of performance, because if you change the HTML, the browser has to render the entire document again.
- **State preservation:** replacing HTML recreates the entire DOM structure from scratch, which can lead to a loss of user input or state that was previously in the DOM. Changing the DOM directly allows you to preserve the existing state of the page.
- **Accessibility**: when you replace HTML, the page may need to reprocess which could cause delays in providing accessibility features.
- **Smoother animation/transition effects**: making changes to the DOM directly results in smoother transitions/animations.

---

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

- **Event handling:** frameworks provide simpler event handling and erase the need for attaching complex event listeners and managing events.
- **Form handling**: frameworks provide form handling, eliminating the need for interacting with form elements.
- **animations/transitions:** frameworks provide APIs or abstractions to create animations, as this can be quite complex to set up and manage. APIs specify animation properties, durations, triggers etc.
- **Automatic code updating:** frameworks handle code updating, so developers don't have to worry about this and can focus on building functionality instead.

---

3. What essence (important aspects) do JavaScript frameworks elevate (highlight)?

- **Reusability**: frameworks promote reusability by providing component based architectures. Therefore they encourage developers to break down the application into smaller, self-contained modules/components that can be reused.
- **Abstraction**: frameworks abstract away low-level complexities, providing higher-level abstractions. APIs also simplify common tasks such as DOM manipulation, event handling, state management.
- **Community:** frameworks foster developer communities, as they provide a shared set of conventions that you should follow, allowing developers to collaborate effectively with each other.
- **Developer Experience (DX):** frameworks enhance developer experience by providing APIs, documentation, and powerful tools for developers to use, increasing their productivity and reducing development time.

---

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

Frameworks achieve abstraction by simplifying and hiding complex code through the use of higher-level APIs. APIs provide pre-built functionalities that developers may use without needing to understand the details of the abstraction, steering away from the low-level complexities. They are used for common tasks such as handling data, managing user interactions and rendering UI elements.

---

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

The most important part of learning a framework is understanding its core concepts/principles. This means learning about the frameworks architecture, data flow, component structure, and key features.

Once the developer understands these fundamental aspects, they can make use of the framework's capabilities.