Event Handling

HyperionDev

Muhammad Zahir Junejo

Lecture – Housekeeping

- ❑ The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.
    - ❑ Please review Code of Conduct (in Student Undertaking Agreement) if unsure
- ❑ No question is daft or silly - **ask them!**
- ❑ Q&A session at the end of the lesson, should you wish to ask any follow-up questions.
- ❑ Should you have any questions after the lecture, please schedule a mentor session.
- ❑ For all non-academic questions, please submit a query: www.hyperiondev.com/support

Lecture Objectives

1. Understanding Event-Driven Programming
2. Anatomy of an Event
3. Event Listeners and Event Handlers
4. Event Propagation
5. Bubbling vs. Capturing

Understanding Event-Driven Programming

- ❑ Event-Driven Programming is a paradigm in which the flow of the program is determined by events, actions, or messages that occur asynchronously. These events can include user interactions, data arrivals, or system notifications.
- ❑ Event-driven programming responds to events or occurrences asynchronously.
- ❑ Events can be triggered by various sources and drive the flow of the program.
- ❑ The event loop manages event dispatching and execution of event handlers.
- ❑ Event-driven programming is well-suited for building interactive and responsive applications.

Anatomy of an Event

- ❑ Event Type: Every event has a specific type or name that defines what kind of event it is. Common event types include "click," "keydown," "submit," "mouseover," and many more. These event types determine when and how an event occurs.
- ❑ Event Target: The event target is the HTML element on which the event occurred or was triggered. It represents the element that is the source of the event. For example, if a user clicks a button, the button element becomes the event target.
- ❑ Event Object: The event object contains detailed information about the event itself. It acts as a container for various properties and methods related to the event. Developers can access this object within event handlers to gather information about the event, such as its type, target, and additional data.
- ❑ Event handlers are functions that execute in response to events. They can access the event object to retrieve event-specific information and perform actions accordingly.

Event Listeners and Event Handlers

- Event Listener: A function or method that "listens" for a specific event on a particular element.

```
// Event listener
element.addEventListener('click', eventHandler);

// Event handler
function eventHandler(event) {
    let target = event.target;
    // Handle the event here
}
```

Event Propagation

- ❑ Events propagate through the DOM tree from the target element to the root or vice versa.
- ❑ Two phases: Capturing phase and Bubbling phase.
- ❑ Bubbling: Events start from the target element and "bubble" up to the root.
- ❑ Capturing: Events start from the root and "capture" down to the target element.
- ❑ Event flow can be controlled using the addEventListener method.

Event Loop

- The Event Loop is a core component of event-driven programming, responsible for managing the execution of asynchronous tasks and events.
- How the Event Loop Works:
  - Call Stack: The call stack is where function calls are queued and executed in a synchronous program. When a function is called, it's pushed onto the stack; when it completes, it's popped off.
  - Web APIs: Asynchronous operations, like fetching data or handling events, are delegated to the browser's Web APIs. These APIs work independently and don't block the main thread.
  - Callback Queue: When an asynchronous task is completed, its associated callback function is placed in the callback queue.
  - Event Loop: The event loop continuously checks the callback queue and moves callback functions to the call stack when it's empty. This process ensures that asynchronous tasks are executed in the correct order, maintaining the program's responsiveness.

References

- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Event_loop

Questions and Answers

Thank You!