Cypress Interview Questions and Answers

General Cypress Concepts

1. What is Cypress, and how is it different from Selenium?

- Cypress is a modern JavaScript-based end-to-end testing framework built specifically for web applications. Unlike Selenium, it operates directly in the browser, offering faster test execution and easier debugging. Selenium works via WebDriver APIs, supporting multiple languages and browsers, while Cypress primarily supports JavaScript/TypeScript and Chromium-based browsers.

2. What are the key advantages of using Cypress for end-to-end testing?

- Fast execution: Direct browser interaction.
- Automatic waits: No need for explicit waits.
- Developer-friendly: Built-in debugging tools.
- Simple setup: No complex drivers required.
- Readable syntax: Easy to write and maintain tests.

3. Explain the Cypress execution process. How does it work internally?

- Cypress runs within the same browser as your application, leveraging JavaScript's event loop. It injects its own proxy layer to interact with the DOM and network traffic, making it faster and more reliable.

4. What are Cypress commands, and how are they different from JavaScript functions?

- Cypress commands are asynchronous and run in a chained, promise-like manner. They manage their execution order internally, whereas JavaScript functions execute immediately.

Test Automation Fundamentals

5. How does Cypress handle asynchronous operations?

   - Cypress uses a built-in queue to manage command execution. It automatically waits for commands and assertions to complete before moving to the next command.

6. Explain the importance of the `cy.wrap()` command.

   - `cy.wrap()` allows you to work with non-Cypress promises or objects in Cypress chains, enabling compatibility with external libraries or asynchronous operations.

7. What is the role of the `cy.request()` command in API testing?

   - `cy.request()` is used to perform HTTP requests directly without relying on the UI. It helps test backend endpoints or precondition data setup.

8. How do you handle uncaught exceptions or application errors during Cypress test execution?

   - Use `Cypress.on('uncaught:exception', (err, runnable) => false)` to suppress errors that don't affect your test outcomes.

9. Can Cypress be used to test mobile applications? Why or why not?

   - Cypress does not support testing native mobile applications directly. However, it can test mobile web applications on browsers with responsive viewports.

[Content continues with the remaining questions and answers...]