**Cypress Interview Questions and Answers**

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

**Answer:**  
Cypress is a modern JavaScript-based end-to-end testing framework designed for web applications. It differs from Selenium in the following ways:

* Runs directly in the browser, whereas Selenium communicates with browsers through WebDriver.
* Has built-in automatic waiting, removing the need for explicit waits like in Selenium.
* Uses JavaScript only, while Selenium supports multiple languages.
* Provides real-time test execution with detailed debugging features.

**2. What are the key features of Cypress?**

**Answer:**

* Automatic waiting (no need for explicit waits)
* Real-time reloading
* Easy debugging with browser dev tools
* API testing capabilities
* Network request stubbing and mocking
* Snapshot testing and time travel

**3. How do you handle dynamic elements in Cypress?**

**Answer:**  
Dynamic elements can be handled using:

* **Using .should() for retries:**

cy.get('.dynamic-element').should('be.visible')

* **Using { timeout: value } for extended waits:**

cy.get('.dynamic-element', { timeout: 10000 }).should('exist')

* **Using .contains() for elements with changing attributes:**

cy.contains('button', 'Click Me').click()

**4. How do you perform API testing in Cypress?**

**Answer:**  
Cypress has a built-in cy.request() command for API testing:

cy.request('GET', 'https://api.example.com/users').then((response) => {

expect(response.status).to.eq(200);

expect(response.body).to.have.property('users');

});

**5. How do you handle authentication in Cypress?**

**Answer:**

* **Using UI-based login:** Perform login using Cypress commands.
* **Using API-based authentication:** Store authentication tokens.

cy.request('POST', '/api/auth', { username: 'user', password: 'pass' })

.then((response) => {

window.localStorage.setItem('authToken', response.body.token);

});

**6. How do you work with iframes in Cypress?**

**Answer:**  
Cypress does not support iframes directly, but you can use cy.iframe() with the cypress-iframe plugin:

cy.frameLoaded('#iframeID');

cy.iframe().find('.button-class').click();

**7. How do you handle file uploads in Cypress?**

**Answer:**  
Using the cypress-file-upload plugin:

cy.get('input[type="file"]').attachFile('test-file.pdf');

**8. What are fixtures in Cypress, and how do you use them?**

**Answer:**  
Fixtures store external test data in JSON files.

cy.fixture('user.json').then((user) => {

cy.get('#username').type(user.username);

});

**9. How do you integrate Cypress with CI/CD pipelines?**

**Answer:**  
Cypress can be integrated with GitHub Actions, Jenkins, or GitLab CI.

Example for GitHub Actions:

- name: Run Cypress Tests

run: npx cypress run

**10. How do you handle environment variables in Cypress?**

**Answer:**  
Use cypress.env.json or set environment variables in cypress.config.js:

const apiUrl = Cypress.env('API\_URL');

**11. What is the Page Object Model (POM) in Cypress?**

**Answer:**  
POM is a design pattern that organizes test code:

class LoginPage {

login(username, password) {

cy.get('#user').type(username);

cy.get('#pass').type(password);

cy.get('#login').click();

}

}

export default new LoginPage();

**12. How do you mock network requests in Cypress?**

**Answer:**  
Using cy.intercept():

cy.intercept('GET', '/api/users', { fixture: 'users.json' }).as('getUsers');

**13. What are Cypress custom commands, and how do you create them?**

**Answer:**  
Custom commands enhance reusability.

Cypress.Commands.add('login', (username, password) => {

cy.get('#username').type(username);

cy.get('#password').type(password);

cy.get('#login-button').click();

});

**14. How do you perform data-driven testing in Cypress?**

**Answer:**  
Using .each() or reading from CSV/JSON files:

cy.fixture('data.json').then((data) => {

data.forEach((user) => {

cy.log(user.username);

});

});

**15. How do you generate Allure reports in Cypress?**

**Answer:**  
Install allure-cypress and update cypress.config.js:

const allureWriter = require('@shelex/cypress-allure-plugin/writer');

module.exports = defineConfig({

e2e: {

setupNodeEvents(on, config) {

allureWriter(on, config);

return config;

},

},

});