

# Playwright Interview Questions with Answers

## Section 1: Playwright Basics

### 1. What is Playwright?

**Answer:** Playwright is an open-source automation framework by Microsoft that is used for end-to-end testing of web applications. It supports multiple browsers (Chromium, Firefox, and WebKit) and allows testing in multiple languages (JavaScript, TypeScript, Python, C#, and Java).

### 2. How do you install Playwright?

**Answer:** You can install Playwright using npm:

```
npm init playwright@latest
```

Or install it separately:

```
npm install --save-dev playwright
```

### 3. How do you launch a browser in Playwright?

**Answer:**

```
const { chromium } = require('playwright');
(async () => {
  const browser = await chromium.launch();
  const page = await browser.newPage();
  await page.goto('https://example.com');
  await browser.close();
})();
```

### 4. What browsers does Playwright support?

**Answer:** Playwright supports:

- Chromium (Google Chrome, Microsoft Edge)
- Firefox
- WebKit (Safari)

## Section 2: Playwright Advanced Concepts

### 5. How do you handle authentication in Playwright?

Answer:

```
const { chromium } = require('playwright');
(async () => {
  const browser = await chromium.launch();
  const context = await browser.newContext({ storageState: 'auth.json' });
  const page = await context.newPage();
  await page.goto('https://example.com');
})();
```

### 6. How do you take screenshots in Playwright?

Answer:

```
await page.screenshot({ path: 'screenshot.png' });
```

### 7. How do you test API responses using Playwright?

Answer:

```
const { test, expect } = require('@playwright/test');

test('API testing with Playwright', async ({ request }) => {
  const response = await request.get('https://jsonplaceholder.typicode.com/posts/1');
  expect(response.status()).toBe(200);
  const body = await response.json();
  console.log(body);
});
```

### 8. How do you handle file uploads in Playwright?

Answer:

```
await page.setInputFiles('input[type="file"]', 'path/to/file.txt');
```

## 9. How do you handle multiple tabs in Playwright?

Answer:

```
const [newPage] = await Promise.all([
  context.waitForEvent('page'),
  page.click('a[target="_blank"]')
]);
await newPage.bringToFront();
```

## 10. How do you mock API responses in Playwright?

Answer:

```
await page.route('**/api/data', async route => {
  await route.fulfill({ json: { message: 'Mocked API response' } });
});
```

## 11. How do you handle alerts and pop-ups in Playwright?

Answer:

```
page.on('dialog', async dialog => {
  console.log(dialog.message());
  await dialog.accept();
});
```

## 12. How do you measure performance in Playwright?

Answer:

```
const metrics = await page.evaluate(() => JSON.stringify(window.performance));
console.log(metrics);
```

## 13. How do you handle dropdowns in Playwright?

Answer:

```
await page.selectOption('select#dropdown', 'optionValue');
```

#### 14. How do you scroll to an element in Playwright?

Answer:

```
await page.evaluate(() => document.querySelector('selector').scrollIntoView())
```

#### 15. How do you test PDF generation in Playwright?

Answer:

```
await page.pdf({ path: 'page.pdf', format: 'A4' });
```

#### 16. How do you handle iframes in Playwright?

Answer:

```
const frame = await page.frame({ name: 'iframe_name' });  
await frame.click('button');
```

#### 17. How do you handle timeouts in Playwright?

Answer:

```
await page.setDefaultTimeout(5000);
```

#### 18. How do you capture network requests in Playwright?

Answer:

```
page.on('request', request => console.log(request.url()));
```

#### 19. How do you retry failed tests in Playwright?

Answer:

```
npx playwright test --retries=2
```

## 20. How do you set cookies in Playwright?

Answer:

```
await context.addCookies([{ name: 'cookieName', value: 'cookieValue', url: 'https://example.com' }]);
```

## 21. How do you disable JavaScript in Playwright?

Answer:

```
const context = await browser.newContext({ javaScriptEnabled: false });
```

## 22. How do you validate an element's CSS properties?

Answer:

```
const color = await page.locator('selector').evaluate(el => getComputedStyle(el).color);
```

## 23. How do you test dark mode in Playwright?

Answer:

```
await page.emulateMedia({ colorScheme: 'dark' });
```

## 24. How do you emulate touch devices?

Answer:

```
await browser.newContext({ hasTouch: true });
```

## 25. How do you handle SSL errors?

Answer:

```
const browser = await chromium.launch({ ignoreHTTPSErrors: true });
```

## Section 3: More Advanced Playwright Concepts

26. How do you validate tooltips in Playwright?

Answer:

```
const tooltipText = await page.locator('selector').getAttribute('title');
console.log(tooltipText);
```

27. How do you interact with shadow DOM elements in Playwright?

Answer:

```
const shadowElement = await page.locator('css=shadow-root-element').shadowRoot
await shadowElement.click();
```

28. How do you handle geolocation testing in Playwright?

Answer:

```
await context.grantPermissions(['geolocation']);
await context.setGeolocation({ latitude: 37.7749, longitude: -122.4194 });
```

29. How do you test responsiveness in Playwright?

Answer:

```
await page.setViewportSize({ width: 375, height: 812 });
```

30. How do you block requests in Playwright?

Answer:

```
await page.route('**/*.jpg', route => route.abort());
```

**31. How do you handle keyboard events in Playwright?**

Answer:

```
await page.keyboard.press('Enter');
```

**32. How do you handle mouse hover in Playwright?**

Answer:

```
await page.hover('selector');
```

**33. How do you slow down execution in Playwright for debugging?**

Answer:

```
npx playwright test --debug
```

**34. How do you interact with modals in Playwright?**

Answer:

```
await page.locator('selector').click();  
await page.waitForSelector('modal-selector');
```

**35. How do you validate console logs in Playwright?**

Answer:

```
page.on('console', msg => console.log(msg.text()));
```

### 36. How do you handle downloads in Playwright?

Answer:

```
const [download] = await Promise.all([
  page.waitForEvent('download'),
  page.click('selector')
]);
await download.saveAs('path/to/save/file');
```

### 37. How do you test drag-and-drop functionality in Playwright?

Answer:

```
await page.dragAndDrop('#source', '#destination');
```

### 38. How do you validate text content in Playwright?

Answer:

```
const text = await page.locator('selector').textContent();
console.log(text);
```

### 39. How do you test auto-complete dropdowns in Playwright?

Answer:

```
await page.fill('input-selector', 'search-text');
await page.waitForSelector('dropdown-selector');
await page.click('option-selector');
```

### 40. How do you check network request responses in Playwright?

Answer:

```
const response = await page.waitForResponse(response =>
response.url().includes('api-endpoint'));
console.log(await response.json());
```



**41. How do you handle authentication popups in Playwright?**

Answer:

```
await page.authenticate({ username: 'user', password: 'pass' });
```

**42. How do you capture full-page screenshots in Playwright?**

Answer:

```
await page.screenshot({ path: 'screenshot.png', fullPage: true });
```

**43. How do you disable JavaScript while testing in Playwright?**

Answer:

```
await context.overridePermissions('https://example.com', []);
```

**44. How do you validate CSS properties of an element in Playwright?**

Answer:

```
const color = await page.locator('selector').evaluate(el =>
getComputedStyle(el).color);
console.log(color);
```

**45. How do you test dark mode in Playwright?**

Answer:

```
await page.emulateMedia({ colorScheme: 'dark' });
```

**46. How do you emulate a touch device in Playwright?**

Answer:

```
await page.emulate({ viewport: { width: 375, height: 812 }, hasTouch: true });
```

#### 47. How do you handle SSL errors in Playwright?

Answer:

```
const browser = await chromium.launch({ ignoreHTTPSErrors: true });
```

#### 48. How do you simulate a slow network in Playwright?

Answer:

```
await page.route('**/*', route => route.continue({ delay: 3000 }));
```

#### 49. How do you check if an element is visible in Playwright?

Answer:

```
const isVisible = await page.isVisible('selector');  
console.log(isVisible);
```

#### 50. How do you get the count of elements matching a selector?

Answer:

```
const count = await page.locator('selector').count();  
console.log(count);
```

#### 51. How do you wait for an API response in Playwright?

Answer:

```
const response = await page.waitForResponse(response =>  
response.url().includes('api-endpoint'));  
console.log(await response.json());
```

#### 52. How do you scroll to an element in Playwright?

Answer:

```
await page.locator('selector').scrollIntoViewIfNeeded();
```

### 53. How do you interact with an iframe in Playwright?

Answer:

```
const frame = await page.frame({ name: 'iframe-name' });  
await frame.click('selector');
```

### 54. How do you simulate mobile gestures in Playwright?

Answer:

```
await page.touchscreen.tap('selector');
```

### 55. How do you check local storage values in Playwright?

Answer:

```
const localStorageValue = await page.evaluate(() => localStorage.getItem('key'))  
console.log(localStorageValue);
```

### 56. How do you handle right-click context menus in Playwright?

Answer:

```
await page.click('selector', { button: 'right' });
```

### 57. How do you test a webpage in offline mode using Playwright?

Answer:

```
await page.context().setOffline(true);
```

### 58. How do you use environment variables in Playwright tests?

Answer:

```
const baseURL = process.env.BASE_URL;  
await page.goto(baseURL);
```

**59. How do you execute JavaScript code inside Playwright?**

Answer:

```
const result = await page.evaluate(() => document.title);  
console.log(result);
```

**60. How do you automate form submission in Playwright?**

Answer:

```
await page.fill('#username', 'testuser');  
await page.fill('#password', 'password123');  
await page.click('button[type=submit]');
```

## Section 4: Even More Advanced Playwright Concepts

**61. How do you clear cookies in Playwright?**

Answer:

```
await context.clearCookies();
```

**62. How do you assert a page title in Playwright?**

Answer:

```
const title = await page.title();  
console.log(title);
```

**63. How do you test a redirect in Playwright?**

Answer:

```
await page.waitForNavigation();  
console.log(page.url());
```

#### 64. How do you interact with Shadow DOM elements?

Answer:

```
const shadowElement = await  
page.locator('shadow-host').shadowRoot().locator('inner-element');  
await shadowElement.click();
```

#### 65. How do you test lazy-loaded elements in Playwright?

Answer:

```
await page.waitForSelector('selector', { state: 'visible' });
```

#### 66. How do you capture console errors in Playwright?

Answer:

```
page.on('console', msg => console.log(msg.text()));
```

#### 67. How do you perform hover actions in Playwright?

Answer:

```
await page.hover('selector');
```

#### 68. How do you assert page load performance in Playwright?

Answer:

```
const perfTiming = await page.evaluate(() =>  
JSON.stringify(window.performance.timing));  
console.log(perfTiming);
```

#### 69. How do you handle browser console logs in Playwright?

Answer:

```
page.on('console', msg => console.log(msg.text()));
```

**70. How do you block specific requests in Playwright?**

**Answer:**

```
await page.route('**/*.png', route => route.abort());
```

**71. How do you test download functionality in Playwright?**

**Answer:**

```
const [download] = await Promise.all([
  page.waitForEvent('download'),
  page.click('a#download')
]);
await download.saveAs('downloaded-file.ext');
```

**72. How do you test page reloads in Playwright?**

**Answer:**

```
await page.reload();
```

**73. How do you wait for multiple requests in Playwright?**

**Answer:**

```
await Promise.all([
  page.waitForResponse('https://example.com/api1'),
  page.waitForResponse('https://example.com/api2')
]);
```

**74. How do you run Playwright tests in parallel?**

**Answer:**

```
npx playwright test --workers=4
```

**75. How do you close a specific tab in Playwright?**

Answer:

```
await page.close();
```

**76. How do you retrieve network request headers in Playwright?**

Answer:

```
page.on('request', request => console.log(request.headers()));
```

**77. How do you validate a text inside an element in Playwright?**

Answer:

```
const text = await page.textContent('selector');  
console.log(text);
```

**78. How do you generate HTML reports from Playwright tests?**

Answer:

```
npx playwright test --reporter=html
```

**79. How do you use Playwright with Cucumber?**

Answer:

```
npm install --save-dev @cucumber/cucumber
```

**80. How do you wait for an element to disappear?**

Answer:

```
await page.waitForSelector('selector', { state: 'hidden' });
```

**81. How do you check if an element is disabled?**

Answer:

```
const isDisabled = await page.isDisabled('selector');  
console.log(isDisabled);
```

**82. How do you handle keyboard shortcuts in Playwright?**

Answer:

```
await page.keyboard.press('Control+A');
```

**83. How do you test web components using Playwright?**

Answer:

```
await page.locator('custom-component').click();
```

**84. How do you disable animations in Playwright?**

Answer:

```
await page.evaluate(() => document.body.style.animation = 'none');
```

**85. How do you automate Google authentication in Playwright?**

Answer:

```
await page.fill('input[type="email"]', 'your-email@gmail.com');  
await page.click('button[type="submit"]');
```

**86. How do you handle reCAPTCHA in Playwright?**

Answer:

```
await page.waitForTimeout(5000);
```



**87. How do you test scrolling behavior in Playwright?**

Answer:

```
await page.evaluate(() => window.scrollTo(0, document.body.scrollHeight));
```

**88. How do you use Playwright to check accessibility compliance?**

Answer:

```
npx playwright test --accessibility
```

**89. How do you validate text inside an input field?**

Answer:

```
const value = await page.inputValue('selector');  
console.log(value);
```

**90. How do you handle dynamically loaded elements?**

Answer:

```
await page.waitForSelector('selector', { state: 'attached' });
```

---