
Domain 1: Essential JavaScript Principles and Practices

1.1: Identify characteristics of JavaScript and common programming practices.

- 1.1.1: List key JavaScript characteristics, including object-based nature, events, platform-independence, and differences between scripting languages and programming languages.
- 1.1.2: Identify common programming concepts, including objects, properties and methods.
- 1.1.3: Describe various JavaScript versions and flavors, including ECMA standards, JScript and similarities with proprietary scripting languages.
- 1.1.4: Distinguish between server-side and client-side JavaScript applications, including JavaScript interpreters and rendering engines.
- 1.1.5: Describe acceptable coding practices, including appropriate use of comment tags and the `<noscript>` tag.

1.2: Work with variables and data in JavaScript.

- 1.2.1: Use attributes and methods to communicate with users, including the *type* attribute, and the `alert()`, `prompt()` and `confirm()` methods.
- 1.2.2: Define variables.
- 1.2.3: Use data types, including *null* and *undefined*.
- 1.2.4: Obtain user input and store it in variables.
- 1.2.5: Report variable text to the client window.
- 1.2.6: Distinguish between concatenation and addition.
- 1.2.7: Use expressions.
- 1.2.8: Use operators, including string concatenation (`+=`), strict comparison (`===` , `!==`) and mathematical precedence.
- 1.2.9: Implement inline scripting.
- 1.2.10: Implement simple event handlers, including `onLoad()` and `onUnload()`.
- 1.2.11: Define keywords and reserved words.

1.3: Use JavaScript functions, methods, and events.

- 1.3.1: Use methods as functions.
- 1.3.2: Define functions.
- 1.3.3: Use data type conversion methods.
- 1.3.4: Call functions.

1.3.5: Pass arguments to functions, including argument creation, return values and the *calculateAvg()* function.

1.3.6: Return values from functions.

1.3.7: Distinguish between global and local variables.

1.3.8: Use the conditional operator.

1.3.9: Identify user events and event handlers.

1.3.10: Use built-in functions and cast variables.

Domain 2: Intermediate JavaScript Programming Techniques

2.1: Use JavaScript statements to control program flow.

2.1.1: Use the *if...* statement.

2.1.2: Use the *while...* statement.

2.1.3: Use the *do...while* statement.

2.1.4: Use the *for...* statement.

2.1.5: Use the *break* and *continue* statements.

2.1.6: Use the *switch...* statement.

2.2: Use the JavaScript Document Object Model (DOM).

2.2.1: Use JavaScript to manipulate the Document Object Model (DOM).

2.2.2: Use the *window* object of the DOM.

2.2.3: Manipulate properties and methods of the *document* object within the DOM.

2.2.4: Use the *with* statement.

2.2.5: Use the *image* object of the DOM, including image rollover creation.

2.2.6: Use the *history* object of the DOM.

2.2.7: Evaluate and change URL information with the *location* object of the DOM.

2.2.8: Use the *navigator* object of the DOM.

2.3: Use JavaScript language objects and create expressions.

2.3.1: Use the *String* object to test user input.

2.3.2: Evaluate strings, including use of the *length* property, and use of the *indexOf()*, *lastIndexOf()*, *substring()* and *charAt()* methods.

2.3.3: Identify basic regular expressions and the *RegExp* object.

2.3.4: Use the *Array* object to create more efficient code.

2.3.5: Identify uses for the *Date* and *Math* objects.

2.4: Create and use custom JavaScript objects.

- 2.4.1: Create a custom JavaScript object.
- 2.4.2: Define properties and methods of custom objects.
- 2.4.3: Create new object instances.
- 2.4.4: Create client-side arrays using custom objects.
- 2.4.5: Create functions and methods for manipulating client-side arrays.
- 2.4.6: Use the prototype property.

2.5: Debug and troubleshoot JavaScript code.

- 2.5.1: List common steps for debugging JavaScript code, including reviewing code and testing code in different browsers.
- 2.5.2: Describe and use various native and supplemental debugging tools, including enabling/disabling display.
- 2.5.3: Test code in multiple display platforms, including mobile devices.

Domain 3: Applied JavaScript

3.1: Use JavaScript to develop interactive forms.

- 3.1.1: Identify and use form controls, including X/HTML form elements.
- 3.1.2: Refer to form objects, including *form*, *radio*, *select*, *button*, *text*, *textarea* and *checkbox*.
- 3.1.3: Define the *form* object.
- 3.1.4: Use the *button* object.
- 3.1.5: Use the *checkbox* object.
- 3.1.6: Evaluate text with the *text* and *textarea* objects.
- 3.1.7: Process *radio* object options.
- 3.1.8: Capture choices from a select list with the *select* object.
- 3.1.9: Conduct form validation, including valid X/HTML code.

3.2: Modify X/HTML with JavaScript.

- 3.2.1: Identify steps and methods for changing X/HTML "on the fly," including the *getElementById*, *getElementsByName* and *getElementsByTagName* methods of the DOM.
- 3.2.2: Modify attributes in X/HTML using DOM elements.
- 3.2.3: Modify values in X/HTML using DOM elements.
- 3.2.4: Use the *innerHTML* element.

3.3: Address JavaScript security issues involving browsers and cookies.

3.3.1: Distinguish between the browser and the operating system in relation to the elements responsible for security.

3.3.2: Discuss browser security issues relevant to JavaScript, including script blocking, prohibition of frame-to-frame URL changing, and *document.write* behavior differences among browsers.

3.3.3: Define signed scripts.

3.3.4: Perform client-side browser detection and determine browser compatibility.

3.3.5: Identify common issues and procedures for creating secure JavaScript code.

3.3.6: Define cross-site scripting and the associated security risks.

3.3.7: Define the functions of cookies and manipulate them effectively, including testing for presence of cookies, clearing cookies, enabling/disabling cookies in the browser, and deleting cookies from your hard drive.

3.3.8: Assign a cookie using JavaScript.

3.3.9: Use cookies and passwords to restrict entry to a page.

Domain 4: JavaScript Technology Extensions

4.1: Implement JavaScript libraries.

4.1.1: Identify and evaluate the benefits and drawbacks of using predefined libraries and plug-ins, such as jQuery, Spry, Dojo, MooTools and Prototype.

4.1.2: Identify steps for using libraries (such as jQuery) and available plug-ins, including jQuery-friendly X/HTML and X/HTML optimization for faster JavaScript manipulation.

4.1.3: Identify steps for loading and referencing external scripts and pre-made external scripts.

4.2: Use JavaScript and AJAX to create interactive Web applications.

4.2.1: Define fundamental AJAX elements and procedures.

4.2.2: Diagram common interactions among JavaScript, XML and XHTML.

4.2.3: Identify key XML structures and restrictions in relation to JavaScript.

4.2.4: Explain how the *XMLHttpRequest* object interacts with XML.

4.2.5: Use the *XMLHttpRequest* object to retrieve data.

4.2.6: Describe typical AJAX-based requests.

4.2.7: Identify key server response issues related to AJAX-based requests.

4.2.8: Use JavaScript to communicate with databases.

4.2.9: Identify alternatives to XML-based AJAX.