JSON Learning Objectives

The objective of this lesson is to familiarize you with the JSON format and how to serialize to and deserialize from that format.

The learning objectives for this lesson are that you can:

1. Identify and generate valid JSON-formatted strings
2. Use `JSON.parse` to deserialize JSON-formatted strings
3. Use `JSON.stringify` to serialize JavaScript objects
4. Correctly identify the definition of "deserialize"
5. Correctly identify the definition of "serialize"

This lesson is relevant because JSON is the *lingua franca* of data interchange.

Storage Lesson Learning Objectives

Below is a complete list of the terminal learning objectives for this lesson. When you complete this lesson, you should be able to perform each of the following objectives. These objectives capture how you may be evaluated on the assessment for this lesson.

1. Write JavaScript to store the value "I <3 falafel" with the key "eatz" in the browser's local storage.
2. Write JavaScript to read the value stored in local storage for the key "paper-trail".

Browser Basics Lesson Learning Objectives

Below is a complete list of the terminal learning objectives for this lesson. When you complete this lesson, you should be able to perform each of the following objectives. These objectives capture how you may be evaluated on the assessment for this lesson.

1. Explain the difference between the BOM (browser object model) and the DOM(document object model).
2. Given a diagram of all the different parts of the Browser identify each part. Use the Window API to change the innerHeight of a user's window.
3. Identify the context of an anonymous functions running in the Browser (the window).
4. Given a JS file and an HTML file, use a script tag to import the JS file and execute the code therein when all the elements on the page load (using`DOMContentLoaded`)
5. Given a JS file and an HTML file, use a script tag to import the JS file and execute the code therein when the page loads
6. Identify three ways to prevent JS code from executing until an entire HTML page is loaded
7. Label a diagram on the Request/Response cycle.
8. Explain the Browser's main role in the request/response cycle. (1.Parsing HTML,CSS, JS 2. Rendering that information to the user by constructing a DOM tree and rendering it)
9. Given several detractors - identify which real-world situations could be implemented with the Web Storage API (shopping cart, forms saving inputs etc.)
10. Given a website to visit that depends on cookies (like Amazon), students should be able to go to that site add something to their cart and then delete that cookie using the Chrome Developer tools in order to empty their cart.

Element Selection Lesson Learning Objectives

Below is a complete list of the terminal learning objectives for this lesson. When you complete this lesson, you should be able to perform each of the following objectives. These objectives capture how you may be evaluated on the assessment for this lesson.

1. Given HTML that includes `<div id=”catch-me-if-you-can”>HI!</div>`, write a JavaScript statement that stores a reference to the HTMLDivElement with the id “catch-me-if-you-can” in a variable named “divOfInterest”.
2. Given HTML that includes seven SPAN elements each with the class “cloudy”, write a JavaScript statement that stores a reference to a NodeList filled with references to the seven HTMLSpanElements in a variable named “cloudySpans”.
3. Given an HTML file with HTML, HEAD, TITLE, and BODY elements, create and reference a JS file that in which the JavaScript will create and attach to the BODY element an H1 element with the id "sleeping-giant" with the content "Jell-O, Burled!".
4. Given an HTML file with HTML, HEAD, TITLE, SCRIPT, and BODY elements with the SCRIPT's SRC attribute referencing an empty JS file, write a script in the JS file to create a DIV element with the id "lickable-frog" and add it as the last child to the BODY element.
5. Given an HTML file with HTML, HEAD, TITLE, SCRIPT, and BODY elements with no SRC attribute on the SCRIPT element, write a script in the SCRIPT block to create a UL element with no id, create an LI element with the id "dreamy-eyes", add the LI as a child to the UL element, and add the UL element as the first child of the BODY element.
6. Write JavaScript to add the CSS class "i-got-loaded" to the BODY element when the window fires the DOMContentLoaded event.
7. Given an HTML file with a UL element with the id "your-best-friend" that has six non-empty LIs as its children, write JavaScript to write the content of each LI to the console.
8. Given an HTML file with a UL element with the id "your-worst-enemy" that has no children, write JavaScript to construct a string that contains six LI tags each containing a random number and set the inner HTML property of ul#your-worst-enemy to that string.
9. Write JavaScript to update the title of the document to the current time at a reasonable interval such that it looks like a real clock.

Event Handling Lesson Learning Objectives

Below is a complete list of the terminal learning objectives for this lesson. When you complete this lesson, you should be able to perform each of the following objectives. These objectives capture how you may be evaluated on the assessment for this lesson.

1. Given an HTML page that includes `<button id="increment-count">I have been clicked <span id="clicked-count">0</span> times</button>`, write JavaScript that increases the value of the content of span#clicked-count by 1 every time `button#increment-count` is clicked.
2. Given an HTML page that includes `<input type="checkbox" id="on-off"><div id="now-you-see-me">Now you see me</div>`, write JavaScript that sets the display of div#now-you-see-me to "none" when input#on-off is checked and to "block" when input#on-off is not checked.
3. Given an HTML file that includes `<input id="stopper" type="text" placeholder="Quick! Type STOP">`, write JavaScript that will change the background color of the page to cyan five seconds after a page loads unless the field input#stopper contains only the text "STOP".
4. Given an HTML page with that includes `<input type=”text” id=”fancypants”>`, write JavaScript that changes the background color of the textbox to #E8F5E9 when the caret is in the textbox and turns it back to its normal color when focus is elsewhere.
5. Given an HTML page that includes a form with two password fields, write JavaScript that subscribes to the forms submission event and cancels it if the values in the two password fields differ.
6. Given an HTML page that includes a div styled as a square with a red background, write JavaScript that allows a user to drag the square around the screen.
7. Given an HTML page that has 300 DIVs, create one click event subscription that will print the id of the element clicked on to the console.
8. Identify the definition of the bubbling principle.