Solutions to Quick Checks

Table of Contents

[Quick Check Answers 2](#_Toc66133779)

[Quick Check 1 2](#_Toc66133780)

[Quick Check 2 2](#_Toc66133781)

[Quick Check 3 3](#_Toc66133782)

[Quick Check 4 4](#_Toc66133783)

# Quick Check Answers

Quick Check 1

1. What is caching?

Caching refers to the temporary storage of data for faster access. Most web browsers reduce the amount of data that needs to be retrieved from a server by caching retrieved data on a local computer.

**Feedback**: If caching is enabled in a web browser, the browser will attempt to locate any necessary data in its cache before making a request from a web server.

1. What are the three foundations of AJAX?

Asynchronous, JavaScript, and XML

**Feedback**: Asynchronous: The client is free to use the other contents of the website without waiting for a response from the server. All responses are managed in separate data streams; JavaScript: Programming can be managed on the client side using only the JavaScript language. No other client-side programming language or app is necessary and XML: Data can be stored in XML, a markup language like HTML for creating structure documents using element tags.

1. Provide a statement to create a new request object named MyReq.

let MyReq = new XMLHttpRequest();

**Feedback**: A request object is created using the object constructor: new XMLHttpRequest()

1. Provide a statement to open the MyReq object to the url *http://www.example.com* using the GET method.

MyReq.open("get", "http://www.example.com");

**Feedback**: A request object begins with the open() method defining the resource to be retrieved from the server and the method by which that resource should be retrieved.

1. What property value indicates that a complete response has been received from the server?

xhr.readyState === 4

**Feedback**: If the readyState value is 4, a completed response has been received. However, a complete response does not necessarily mean a successful response, so you also must use the status property to confirm that a successful connection with the server was made.

Quick Check 2

1. Write the following function in arrow function syntax:

function display(msg) {  
 alert(mg);  
}

let display = msg => alert(msg);

**Feedback**: Arrow function syntax removes extraneous characters from a function expression, resulting in smaller code files that can be quickly handled by the JavaScript and the web server.

1. What are the three states of a promise object?

pending, resolved, rejected

**Feedback:** pending (the promise has been given but not yet fulfilled or rejected), resolved (the promise has been fulfilled), and rejected (the promise has been rejected and will not be fulfilled.

1. Provide code to create a promise object named myPromise.

let myPromise = new Promise((*resolved*, *rejected*) => {  
 *promise statements*  
});

**Feedback**: A promise object is defined using the new Promise() object constructor with *resolved* and *rejected* callback functions that are run once the promise is resolved or rejected.

1. Provide code to send a promise to the https://jsonshow/photos resource using fetch.

fetch("https://jsonshow/photos")

**Feedback:** The Fetch API uses promises to manage request and responses from server resources and, thus, avoids some of the problems associated with callback hell. Requests are made to a server resource using fetch(*url*, *options*) where *url* is the location of the server resource and *options* is an optional object defining values for the HTTP message**.**

1. What method should be applied to a response object to return a promise that can be parsed as text?

*response.*text()

**Feedback**: The text of the response object is parsed using the *response*.text() method, creating another promise.

Quick Check 3

1. Provide code to create an XML parser.

let *parser* = new DOMParser()

**Feedback:** The *parser* object stores properties and methods that can be used to convert an XML text string into a XML DOM.

1. What method converts an XML text string into a DOM?

*parser*.parseFromString(*text*, *mimeType*)

**Feedback:** In the parseFromString() method, *text* is the text string to be parsed and *mimeType* identifies the type of structured data stored in the text string. Mime type values include "text/html", "text/xml", "image/svg+xml" for working with HTML, XML, and SVG documents.

1. What method converts an XML DOM back into a text string?

*serializer*.serializeToString(*dom*)

**Feedback:** In the serializeToString() method, *serializer* is an XML object with properties and methods for converting a DOM and *dom* is the node tree containing elements to convert to text.

1. Provide a method to parse JSON data received from a response object.

*response*.json()

**Feedback:** In the json() method, *response* is the Promise object containing the server response. Once parsed, the JSON data can be handled as an object literal.

1. When would you use the PUT method in a Fetch request?

Use the PUT method to change existing data on the server, such as changing a user profile in a server database.

**Feedback:** Five of the most important methods for interacting with a server resource are GET, POST, PUT, PATCH, and DELETE for getting data, adding new data, revising existing data in whole or in part, and for deleting existing data.

Quick Check 4

1. What is a freemium service?

A service that is free or essentially free for limited use but increasing in cost with increased access and use.

**Feedback:** A freemium service helps developers evaluate an API become committing it to public use and allows users with more limited applications to test and use the service without incurring significant costs.

1. What is an API endpoint?

An API endpoint is the point of contact between the client device and a server resource; essentially an endpoint is the url to which a server request is made.

**Feedback:** A server might provide different endpoints for its resources with each endpoint dedicated to a particular server operation.

1. What are three ways that an app can deal with the same-origin policy?

Cross Origin Sharing, JSONP, and XHR proxies

**Feedback:** Cross Origin Sharing (CORS) uses HTTP headers to ensure that a website from one origin can exchange data with a resource of a different origin. JSONP folds a server request within a script element that is attached to a callback function. XHR proxies funnel their requests through a secondary server that does have access to the original server’s resources.