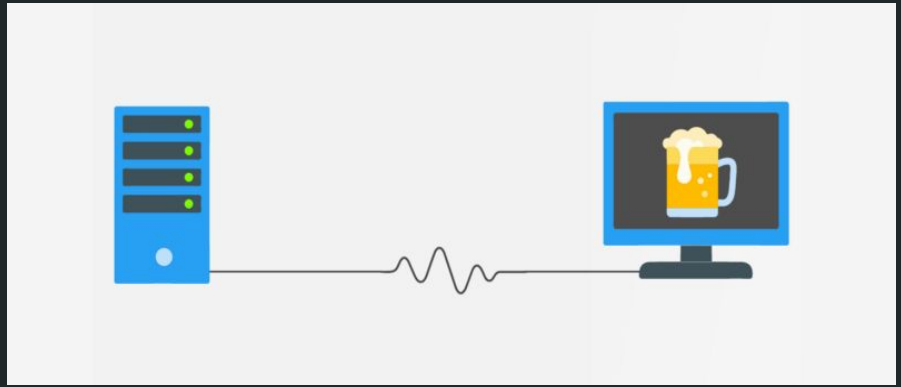
Advanced Javascript

Day 8 - API, XHR

API



Application Programming Interfaces (APIs) are constructs made available in programming languages to allow developers to create complex functionality more easily.

They abstract more complex code away from you, providing some easier syntax to use in its place.

Types of API

- Browser APIs — constructs built into the browser that sits on top of the JavaScript language and allows you to implement functionality more easily.
- Third-party APIs — constructs built into third-party platforms (e.g. Twitter, Facebook) that allow you to use some of those platform's functionality in your own web pages (for example, display your latest Tweets on your web page).
- JavaScript libraries — Usually one or more JavaScript files containing custom functions that you can attach to your web page to speed up or enable writing common functionality. Examples include jQuery, Mootools and React.
- JavaScript frameworks — The next step up from libraries, JavaScript frameworks (e.g. Angular and Ember) tend to be packages of HTML, CSS, JavaScript, and other technologies that you install and then use to write an entire web application from scratch. The key difference between a library and a framework is “Inversion of Control”. When calling a method from a library, the developer is in control. With a framework, the control is inverted: the framework calls the developer's code.

Let’s quickly recap HTTP and Client Server Model...

JavaScript can send network requests to the server and load new information whenever it’s needed.

We have already talked about HTTP requests like GET, POST, PUT, DELETE.

Right now our websites are simple and doesn’t have dynamic data.

For example, we can use a network request to:

- Submit an order,
- Load user information,
- Receive latest updates from the server,
- ...etc.

There's an umbrella term "AJAX" (abbreviated Asynchronous JavaScript And XML) for network requests from JavaScript.

There are multiple ways to send a network request and get information from the server.

XHR

What is XHR?

`XMLHttpRequest` is a built-in browser object that allows to make HTTP requests in JavaScript.

The XMLHttpRequest object is a developers dream, because you can:

- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

We have already talked about HTTP requests like GET, POST, PUT, DELETE.

4 Steps of XHR

Create XMLHttpRequest:

var xhr = new XMLHttpRequest();

Step 2

Initialize it, usually right after new XMLHttpRequest:

xhr.open(method, URL, [async, user, password])

Step 3

Send it out.

xhr.send(body)

This method opens the connection and sends the request to server. The optional body parameter contains the request body.

Some request methods like GET do not have a body. And some of them like POST use body to send the data to the server. We'll see examples of that later.

4 Steps of XHR

Listen to xhr events for response.

These three events are the most widely used:

load – when the request is complete (even if HTTP status is like 400 or 500), and the response is fully downloaded.

error – when the request couldn't be made, e.g. network down or invalid URL.

progress – triggers periodically while the response is being downloaded, reports how much has been downloaded.

Let’s take a look at a simple example where we send a simple GET request and retrieve some JSON data.

But we don’t just want to GET data, we sometimes want to POST to!

Let’s look at a POST request using XHR.