# Basics

Synchronous programming, we need to wit until pervious tasks are over.

ret = getRemoteData();

// wait till remote data is fetched

// process returns from remote data

// next operation has to wait until

// all previous steps are done

nextOperation();

In async programming we don’t wait for blocking operation to finish, instead we register a function (callback) to be called when blocking operation is done and we continue executing another code.

getRemoteData(function(){

// wait till remote data is fetched

// process return from remote data

});

// next operation don’t have to wait

// for old operation to finish

nextOperaiton();

I JavaScript most of Async implementations are supported by different libraries.

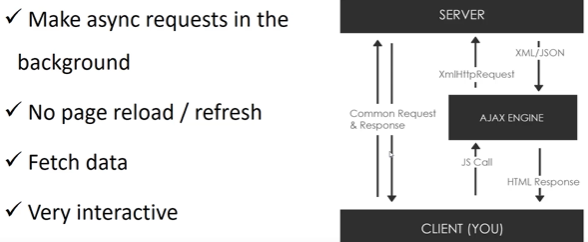
* XMLHTTPRequest & Fetch
* JQuery, Ajax, Axios, other HTTP libraries
* Node.js fs (filesystem) module

Following are ways we can handle Asynchronous code

* Callbacks
* Promises
* Async/Await

# AJAX

Asynchronous JavaScript and XML. AJAX  **is a buzzard for ‘making a HTTP request from Javascript without leaving or refreshing page’** . This is not any single library of framework, but set of web technologies to send and receive data between client and server. In Ajax mostly JSON is used rather than XML (even if it’s in name). Most of the API’s now return JSON data and not XML.



XMLHTTPRequest (XHR) is one of the (OLD technology now) API used for Ajax. This is API is supported/ is part of all the browser’s. This API is mainly used for transfer of data between client and server and can work with JSON, XML, or plain text.

There are **other newer/external** libraries as well which can help making Ajax call.

* Fetch API (popular and built-in to Javascript)
* Axios (external library)
* Superagent (external library)
* jQuery (not recommended)
* Node HTTP (for nodejs)

## XHR API

Basic syntax of getting data from API endpoint which doesn’t need any authentication

// create XHR object from built-in API

const xhr = new XMLHttpRequest();

// open the connection

xhr.open('GET', <<API URL>>, true);

// this runs when data is returned and loaded

xhr.onload = function(){

  // check if return is success

  if(this.status === 200) {

    console.log(this.responseText);

  }

}

//required to complete the request

xhr.send();

## CallBack function

It is a function passed to another function and it’s called by another function whenever needed (mostly called when another function does its execution)

// if we want 'SayName' function to wait for greet function to finish then

// we can pass 'SayName' function as callback to greet which then internally

// execute call back function

function greet(name, myFunction) {

    console.log('Hello world');

    // callback function

    // executed only after the greet() is executed

    myFunction(name);

  }

  // callback function

  function sayName(name) {

    console.log('Hello' + ' ' + name);

  }

  // calling the function after 2 seconds

  setTimeout(greet, 2000, 'John', sayName);

  //-> 'Hello world' followed by 'Hello John'