# Ajax

Ajax with XHR

# AJAX?

- Asynchronous
- JavaScript
- And
- XML

AJAX Response

- Data
  - XML <entry></entry>
  - JSON { property: data }
  - HTML <div></div>

# API?

**A**pplication

**P**rogramming

**I**nterface

*“In computer programming, an application programming interface is a set of subroutine definitions, protocols, and tools for building application software.” - Wiki*

# XMLHttpRequest (XHR)

***Create?***

```
Const asyncRequestObject  = new XMLHttpRequest();
```

# Methods (Open)

```
asyncRequestObject.open();
```

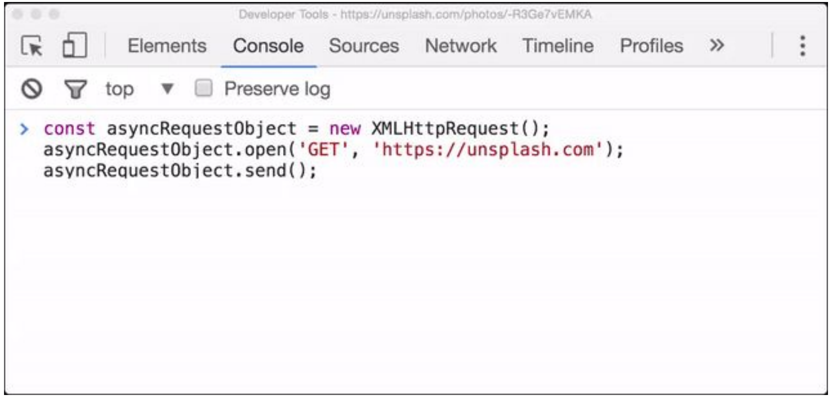
`.open()` takes a number of parameters, but the most important are its first two: the HTTP method URL to send the request.

If we want to asynchronously request the homepage from the popular high-res image site, Unsplash, we'd use a `GET` request and provide the URL:

```
    asyncRequestObject.open('GET', 'https://unsplash.com');
```

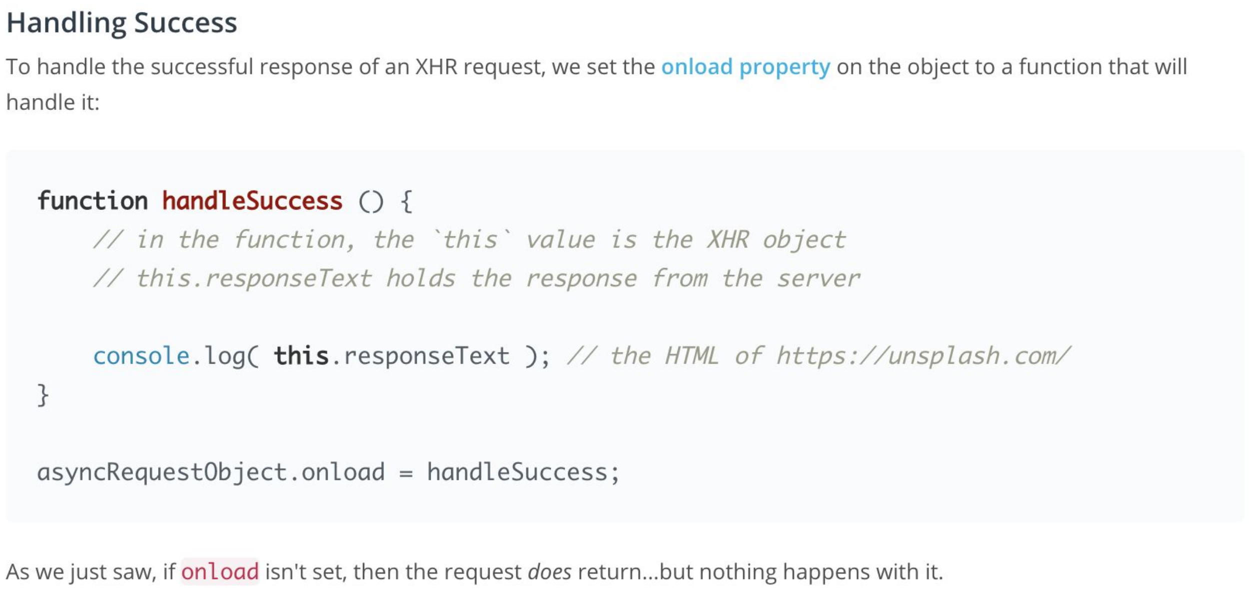
# Methods (Send)

```
asyncRequestObject.send();
```



```
> const asyncRequestObject = new XMLHttpRequest();
  asyncRequestObject.open('GET', 'https://unsplash.com');
  asyncRequestObject.send();
```

# Handling Response



### Handling Success

To handle the successful response of an XHR request, we set the onload property on the object to a function that will handle it:

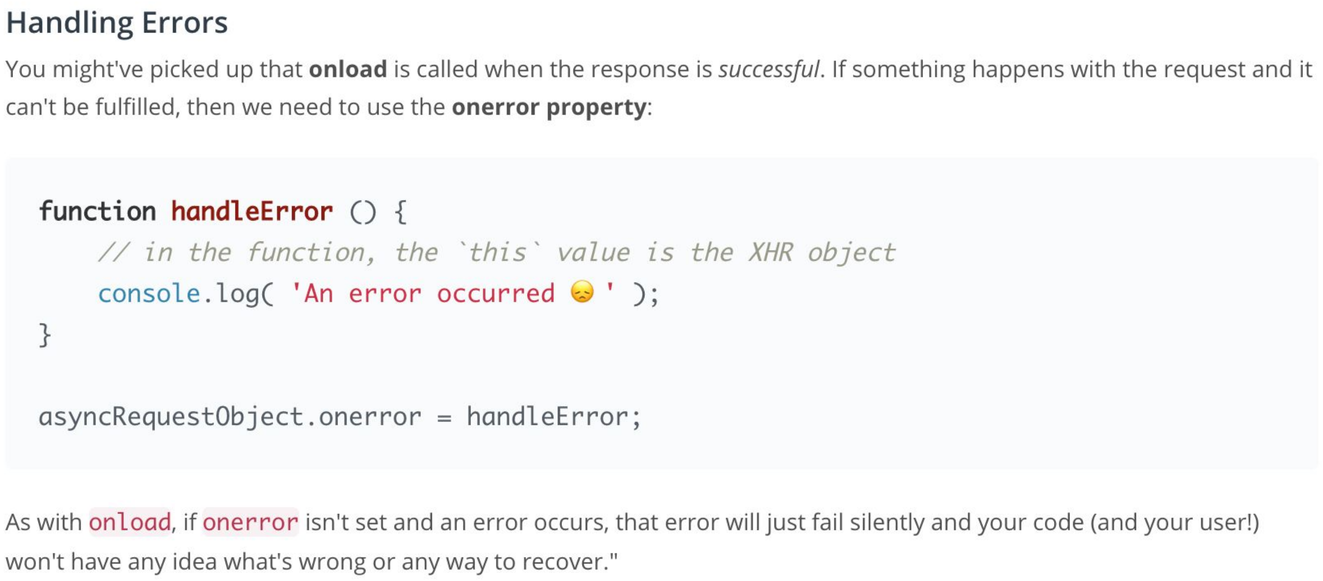
```
function handleSuccess () {
    // in the function, the `this` value is the XHR object
    // this.responseText holds the response from the server

    console.log( this.responseText ); // the HTML of https://unsplash.com/
}

asyncRequestObject.onload = handleSuccess;
```

As we just saw, if `onload` isn't set, then the request *does* return...but nothing happens with it.

# Handling Response



### Handling Errors

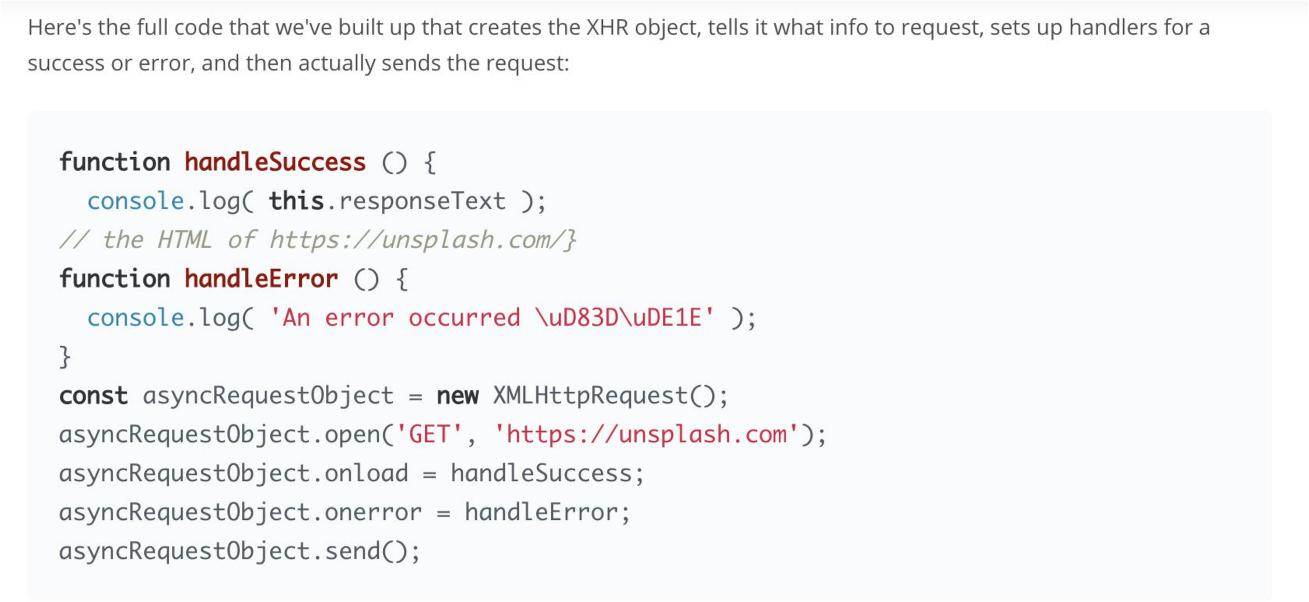
You might've picked up that **onload** is called when the response is *successful*. If something happens with the request and it can't be fulfilled, then we need to use the **onerror property**:

```
function handleError () {
    // in the function, the `this` value is the XHR object
    console.log( 'An error occurred 😞 ' );
}

asyncRequestObject.onerror = handleError;
```

As with `onload`, if `onerror` isn't set and an error occurs, that error will just fail silently and your code (and your user!) won't have any idea what's wrong or any way to recover."

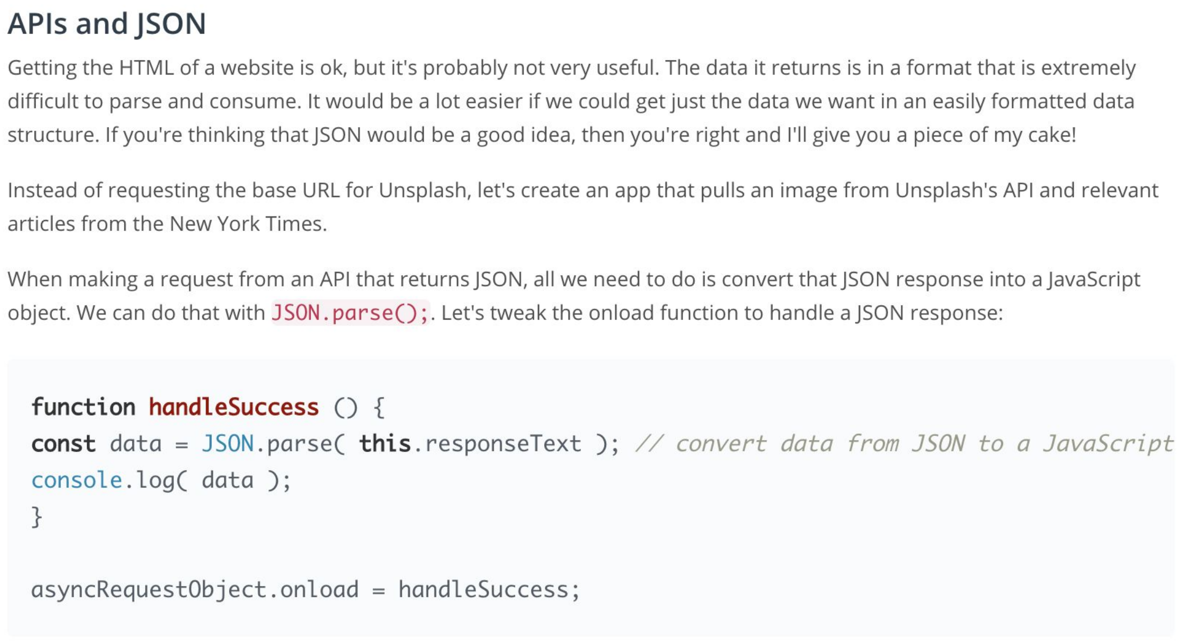
# Full code



Here's the full code that we've built up that creates the XHR object, tells it what info to request, sets up handlers for a success or error, and then actually sends the request:

```
function handleSuccess () {
  console.log( this.responseText );
// the HTML of https://unsplash.com/}
function handleError () {
  console.log( 'An error occurred \uD83D\uDE1E' );
}
const asyncRequestObject = new XMLHttpRequest();
asyncRequestObject.open('GET', 'https://unsplash.com');
asyncRequestObject.onload = handleSuccess;
asyncRequestObject.onerror = handleError;
asyncRequestObject.send();
```

# APIs and JSON



## APIs and JSON

Getting the HTML of a website is ok, but it's probably not very useful. The data it returns is in a format that is extremely difficult to parse and consume. It would be a lot easier if we could get just the data we want in an easily formatted data structure. If you're thinking that JSON would be a good idea, then you're right and I'll give you a piece of my cake!

Instead of requesting the base URL for Unsplash, let's create an app that pulls an image from Unsplash's API and relevant articles from the New York Times.

When making a request from an API that returns JSON, all we need to do is convert that JSON response into a JavaScript object. We can do that with `JSON.parse();`. Let's tweak the onload function to handle a JSON response:

```
function handleSuccess () {
const data = JSON.parse( this.responseText ); // convert data from JSON to a JavaScript
console.log( data );
}

asyncRequestObject.onload = handleSuccess;
```

# Create Your Account - Unsplash

You'll need accounts with Unsplash and The New York Times to use their APIs

## Unsplash

- Create a developer account here - https://unsplash.com/developers
- Next, create an application here - https://unsplash.com/oauth/applications
    - this will give you an "Application ID" that you'll need to make requests

# Create Your Account - NY Times

## The New York Times

- Create a developer account here - https://developer.nytimes.com/
- They'll email you your api-key (you'll need this to make requests)

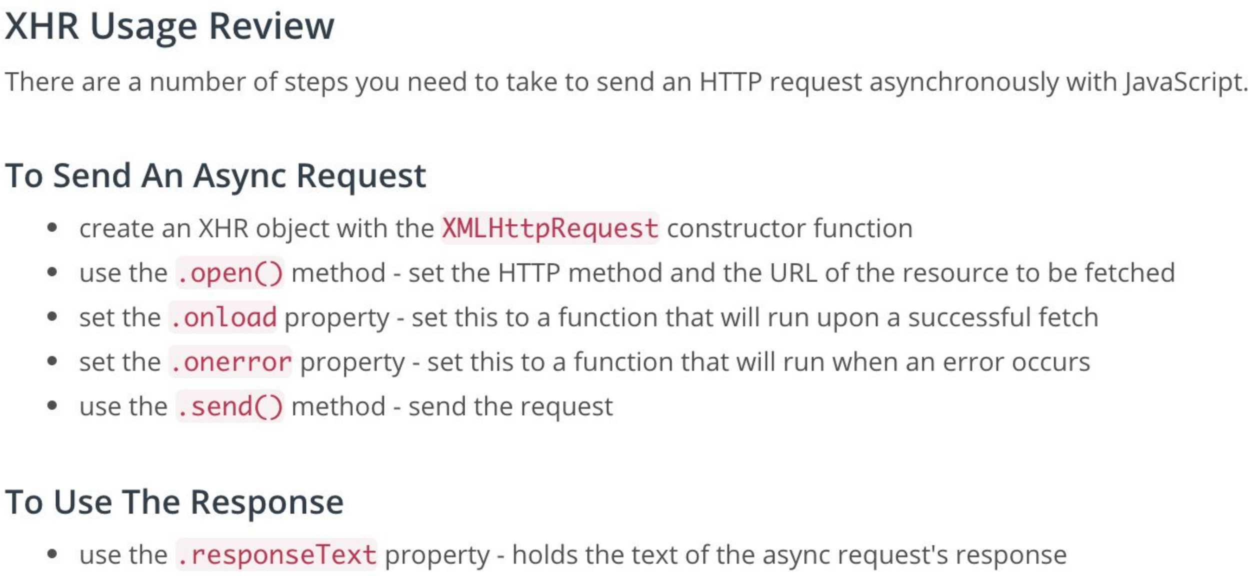
# Sending Request

The request needs an HTTP header to be sent along. What is the XHR method to add a header to the request?

***.setRequestHeader()***

XMLHttpRequest (documentation)

# XHR (Summary)



## XHR Usage Review

There are a number of steps you need to take to send an HTTP request asynchronously with JavaScript.

### To Send An Async Request

- create an XHR object with the `XMLHttpRequest` constructor function
- use the `.open()` method - set the HTTP method and the URL of the resource to be fetched
- set the `.onload` property - set this to a function that will run upon a successful fetch
- set the `.onerror` property - set this to a function that will run when an error occurs
- use the `.send()` method - send the request

### To Use The Response

- use the `.responseText` property - holds the text of the async request's response