 Practice making the API call you'll make in code in your application.  You may need to get an API key for the API.  Document a sample URL for the API and any parameters that your application may need to modify.

 Analyze the JSON results from the API call.  Document the structure of the JSON object and identify any of the specific data that your application will need.  Document the structure of the JSON response when an invalid API call is made too.

Text to Speech

I made an account at tech.iaith. On their portal, you can apply for a key to each of the web services they run. The form asked for a description of my project and the link to the website it will be used on. I published [a stub html page to citstudent](http://citstudent.lanecc.net/~williamsm312/cs233js/project/), so that I would know the URL for this field.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter name | English translation of parameter name | Description | Notes |
| Testun | Text | the text you want to hear as a wav file; this needs to be encoded so that whitespace characters are replaced by ‘%20’ | I think the way users will interact with this is entering simple vocabulary words, not whole sentences |
| Siaradwr | Speaker | This sets a particular speaker’s voice. The API uses a default voice if you don’t enter a parameter value here. | The default is a woman speaking in a southern welsh accent. The documentation for other text to speech api’s from this organization mention plans to expand the number of speakers available. I will use the default |
|  |  |  |  |

Sample call:

Text to speak = “Dw I eisiau glywed rhywbeth”

English translation: I want to hear something

<https://tts.techiaith.cymru/coqui-tts/api/v1?testun=Dw%20i%20eisau%20glywed%20rhywbeth&siaradwr=&api_key=6ec83e1b-9879-4bc8-b28a-134fa218a5ed>

This returns a .wav file.

Testing:

I can make a call by building the string and entering into the browser. When I make the request, my browser downloads a .wav

There are a lot of options for getting the audio on the page. I haven’t successfully used fetch yet, though. The simplest way is to build the api call in a string and then use a template literal in the audio tag’s innerHTML like this:

function addSource(){

let audioElement=document.querySelector("audio");

const call='https://tts.techiaith.cymru/coqui-tts/api/v1?testun=Dw%20i%20canu&siaradwr=&api\_key=6ec83e1b-9879-4bc8-b28a-134fa218a5ed';

const string=`<source src=${call} type="audio/wav"> Your browser does not support the audio element.`;

audioElement.innerHTML = string;

}

This was me testing in a script element on the hmtl page, so it isn’t encapsulated in an ES6 class yet.

However, there might be a more elegant way to do this using fetch and promises.

Vocab API

I had planned on using a vocab API from techiaith. It’s a widget, not a web service and it doesn’t meet my needs.

Pexels API

<https://www.pexels.com/api/>

<https://www.pexels.com/api/documentation/>

API key: 563492ad6f91700001000001d4be53950c3d4ace99ac5c52efdf1558

Endpoint address start: <https://api.pexels.com/v1/>

The documentation suggests adding a js library that I can install as a package

The example call doesn’t look like what I expect.

curl -H "Authorization: 563492ad6f91700001000001d4be53950c3d4ace99ac5c52efdf1558" \

"https://api.pexels.com/v1/search?query=nature&per\_page=1"

Search by keyword

|  |  |  |
| --- | --- | --- |
| Query | String/required | The search query. Ocean, Tigers, Pears, etc. |
| Orientation | String/optional | Desired photo orientation. The current supported orientations are: landscape, portrait or square. |
| Size | String/optional | Minimum photo size. The current supported sizes are: large(24MP), medium(12MP) or small(4MP). |
| Color | String/optional | Desired photo color. Supported colors: red, orange, yellow, green, turquoise, blue, violet, pink, brown, black, gray, white or any hexidecimal color code (eg. #ffffff). |
| Locale | String/optional | The locale of the search you are performing. The current supported locales are: 'en-US' 'pt-BR' 'es-ES' 'ca-ES' 'de-DE' 'it-IT' 'fr-FR' 'sv-SE' 'id-ID' 'pl-PL' 'ja-JP' 'zh-TW' 'zh-CN' 'ko-KR' 'th-TH' 'nl-NL' 'hu-HU' 'vi-VN' 'cs-CZ' 'da-DK' 'fi-FI' 'uk-UA' 'el-GR' 'ro-RO' 'nb-NO' 'sk-SK' 'tr-TR' 'ru-RU'. |
| Page | Int/optional | The page number you are requesting. Default: 1 |
| Per page | Int/optional | The number of results you are requesting per page. Default: 15 Max: 80 |

**A JSON object gets returned.**

**Sample response:**

{

"total\_results": 10000,

"page": 1,

"per\_page": 1,

"photos": [

{

"id": 3573351,

"width": 3066,

"height": 3968,

"url": "https://www.pexels.com/photo/trees-during-day-3573351/",

"photographer": "Lukas Rodriguez",

"photographer\_url": "https://www.pexels.com/@lukas-rodriguez-1845331",

"photographer\_id": 1845331,

"avg\_color": "#374824",

"src": {

"original": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png",

"large2x": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png?auto=compress&cs=tinysrgb&dpr=2&h=650&w=940",

"large": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png?auto=compress&cs=tinysrgb&h=650&w=940",

"medium": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png?auto=compress&cs=tinysrgb&h=350",

"small": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png?auto=compress&cs=tinysrgb&h=130",

"portrait": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png?auto=compress&cs=tinysrgb&fit=crop&h=1200&w=800",

"landscape": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png?auto=compress&cs=tinysrgb&fit=crop&h=627&w=1200",

"tiny": "https://images.pexels.com/photos/3573351/pexels-photo-3573351.png?auto=compress&cs=tinysrgb&dpr=1&fit=crop&h=200&w=280"

},

"liked": false,

"alt": "Brown Rocks During Golden Hour"

}

],

"next\_page": "https://api.pexels.com/v1/search/?page=2&per\_page=1&query=nature"

}