

## CPSC1520 – JavaScript 6 Exercise: Fetch API

### Introduction

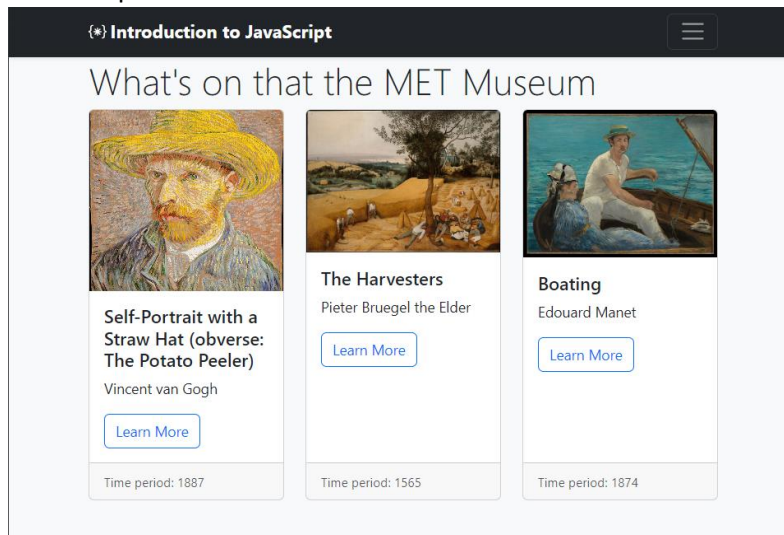
The Fetch API is essential to most client side JavaScript applications. In this in class exercise what you'll be doing is you'll be using fetch to get data and populate an application.

We'll be using the data from the Metropolitan Museum of Art <https://metmuseum.github.io/> to practice getting data and reading some documentation.

### Exercise Step 1: Fetch and Render the objects from the Met Museum REST API

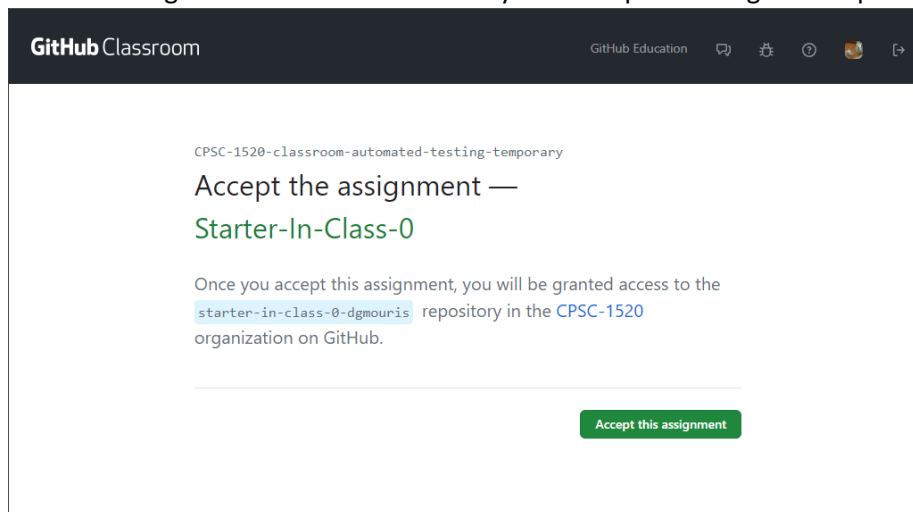
1. Create a function named "getMuseumObjectsData", this is going to be the function that will be doing our fetch requests.
  - a. You'll be needing to loop through the museum objects ids so that in this function so that you can access each id.
2. In the loop that you've created in the previous step you'll need to make a fetch request for each id to the "Object" endpoint of the MET Museum API.
  - a. You'll be using the documentation from the MET Museum REST API. Look at the documentation under "Object" (not "Objects") at this link <https://metmuseum.github.io/>. Note that you can use a REST API client to ensure that you are making the correct request.
  - b. To ensure that you've got the fetch request working, print out the resolved data to the console.
3. Create a function named "renderGalleryCard".
  - a. This function will take five parameters
    - i. Title
    - ii. Image
    - iii. artistName
    - iv. date
    - v. wikiDataUrl
  - b. Select the element with the class "museum-gallery"
  - c. Loop through each contact and use the "HTML for the project" in the main.js to continuously add a templateString to the element above.
  - d. When you call the function with the correct data the page:
    - i. Must have the correct title matching the screenshot below
    - ii. Must have the correct image matching the screenshot below
    - iii. Must have the correct artist name matching the screenshot below
    - iv. Must have the correct date matching the screenshot below
    - v. Must have the correct wiki data url matching the screenshot below
4. Call the function "renderGalleryCard" when you resolve your fetch request in the "getMuseumObjectsData" function.

5. The final product should look like the screenshot below.



## Exercise Step 2 – Push up your code to GitHub


1. Open the link given and accept the assignment. Your link should look something like this. Note the image will be different because you'll accept the assignment specified.



You'll see a page like this.


**GitHub Classroom**

GitHub Education



You accepted the assignment, **Starter-In-Class-0**. We're configuring your repository now. This may take a few minutes to complete. Refresh this page to see updates.

Note: You may receive an email invitation to join [CPSC-1520](#) on your behalf. No further action is necessary.

**Join the GitHub Student Developer Pack**


Verified students receive free GitHub Pro plus thousands of dollars worth of the best real-world tools and training from GitHub Education partners — for free. [Learn more](#)

Apply

Once your repo is ready the page should look like the following.

**GitHub Classroom**


GitHub Education



## You're ready to go!


You accepted the assignment, **Starter-In-Class-0**.

Your assignment repository has been created:

 <https://github.com/CPSC-1520/starter-in-class-0-dgmouris>

We've configured the repository associated with this assignment (update).

Note: You may receive an email invitation to join [CPSC-1520](#) on your behalf. No further action is necessary.

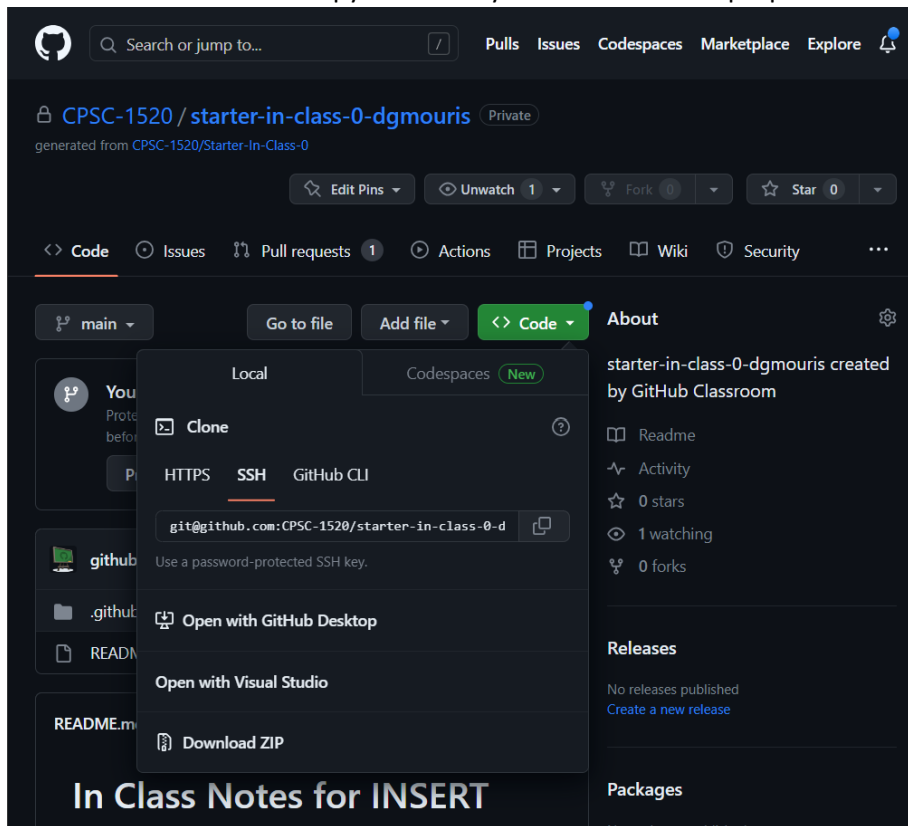
**Join the GitHub Student Developer Pack**

Verified students receive free GitHub Pro plus thousands of dollars worth of the best real-world tools and training from GitHub Education partners — for free. [Learn more](#)

Apply

- You should see the page below once you click on the link highlighted in blue. Click the button that says "Code." You'll need to select "HTTPS" unless you've set up "SSH" (you can also set up

GitHub CLI". Click on the copy icon once you've selected the proper icon.



3. Clone the repository in your console (or if you're using GitHub Desktop) using the "git clone REPO\_URL" command.

```
DMOURIS@W309-DMORR2 C:\Users\dmouris\temp
$ git clone git@github.com:CPSC-1520/starter-in-class-0-dgmouris.git
Cloning into 'starter-in-class-0-dgmouris'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 8 (delta 0), reused 2 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), done.

DMOURIS@W309-DMORR2 C:\Users\dmouris\temp
$
```

And go into this folder.

4. Make your changes, then add them to staging (using "git add .") and commit them (using "git commit -m "CHANGE THIS MESSAGE"). Once committed, push them up to GitHub (using "git

push”) it should look like below.

```
DMOURIS@W309-DMORR2 C:\Users\dmouris\temp\starter-in-class-0-dgmouris
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

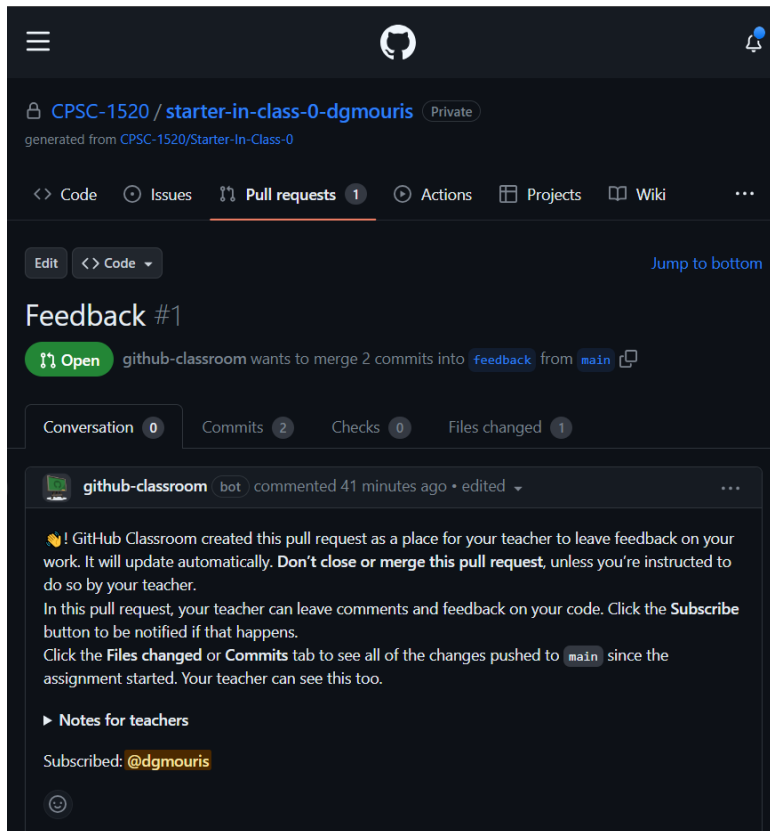
DMOURIS@W309-DMORR2 C:\Users\dmouris\temp\starter-in-class-0-dgmouris
$ git add README.md

DMOURIS@W309-DMORR2 C:\Users\dmouris\temp\starter-in-class-0-dgmouris
$ git commit -m "Made changes"
[main 9532c1b] Made changes
 1 file changed, 1 insertion(+), 3 deletions(-)

DMOURIS@W309-DMORR2 C:\Users\dmouris\temp\starter-in-class-0-dgmouris
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 20 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 373 bytes | 373.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:CPSC-1520/starter-in-class-0-dgmouris.git
   b6ef88e..9532c1b  main -> main

DMOURIS@W309-DMORR2 C:\Users\dmouris\temp\starter-in-class-0-dgmouris
$
```

5. If you click “Pull Requests” and then the first item called “Feedback” you should see your commit (seen at the bottom).



6. Upload the link of your repository to Moodle.

## Grading

I'll give full marks if:

- The code is correctly implemented as described above.
- The end result looks like the following picture

