MIDDLESEX Community College

Tools and Technologies for Tech Writers 2024

Homework Helpers

Notices

This document was prepared as a handout for the Middlesex Community College Tools and Technologies for Technical Writers class, Winter semester 2024.

Prepared by Zoë Lawson, course instructor.

Contents

Homework Helpers

This course is very fast paced. A lot of the instructions are vague. This document is an attempt to flush out some details if you need it.

I reiterate this many times over the course: I am covering too much, too fast.

Every single topic I cover could be a full course unto itself.

The intention is that you learn a bit about how you need to learn whatever new tools come your way in the workplace. Do you need a few tutorials? Do you need to buy a book? Do you need someone to train you, hands on?

The other intention is that you figure out if there are tools you hate, so you know to avoid those when job hunting.

Note: This document is a work in progress. I am going to add to it chapter by chapter.

Homework requirements

For each assignment, I am looking to see if you tried to use the tool.

It is impossible to master any of these tools in a week.

- I do not expect you to get everything correct.
- I do not expect you to become an expert.
- I do not expect this to be an excellent example of your writing abilities.
- I do not expect the assignment to be coherent or logical. It could all be Lorem Ipsum with some formatting applied.

My intention is that it only takes an hour or two to make a simple file (or more, depending upon the tool) that includes the following:

- At least one heading
- A paragraph or two
- An unordered list
- A numbered list
- A table
- Some inline formatting
- A link to an external file
- An image

This text can be completely nonsensical. I just want to see if you tried (not even succeeded) at using the tool.

Building Blocks on page 5 explains why I selected this set of items.

Suggestion on page 8 shows an example set of content.

These are the minimum requirements.

If you choose, you can attempt to use each homework assignment to make a portfolio piece. Pick something to document and write it up in whatever the tool of the week is. One of the easiest things to document might be something about whatever tool you are using. If you ask, I will review your content for writing style and offer suggestions.

Building Blocks

There are a handful of tasks you need to learn how to do in any program you work in.

Every tool, and every implementation of a tool, has a unique way of doing things. For example, many content sets require some way to link to the software application they are documenting. This is so if you click the help button in your application, a specific page in your output appears. Most tools offer a way to do this, however how you do it is unique to the tool. Also, what is specifically required is unique to your work place. You may need to set up a mapping file that links application IDs to help IDs, and then do something to add the help IDs to your source files so the "magic" works in the output. You may just need to ensure that output files are named a certain way or put in a certain folder.

That said, there are a relatively small list of things that you need to be able to do in every tool. If you can do these things, you can use the tool.

As a Writer

If all you are concerned with is writing, these are the basic tasks you need to know how to do.

Required

This is the bare minimum you have to be able to do in any tool to be a technical writer.

Add a new paragraph	The English language is divided into chunks of text. These chunks are often paragraphs. You need to know how to make a new paragraph as needed. This is usually as simple as pressing **Enter**, but could require a bit more work, such as remembering to have a blank line between paragraphs in lightweight markup, or having to wrap the text in a <p> element in HTML.
Format the content	As you author content, you need to add formatting to it to help present the information better. You need to know both what formatting to apply, as well as how to do it. Some of this is subjective, such as determining if the content work better as a series of paragraphs, a bulleted list, or a table. Some of this should follow your company's style guide, such as when to apply inline formatting, and knowing the name of the style for a third level heading. Some of this is knowing how your tool works. When I'm describing adding formatting here, in general, I'm referring to applying the formats, styles, or template in your tool. Designing what your format actually looks like is generally something different.
Make text a title	The way we structure content in English, your content needs a title. This could be a chapter title, a book title, a section title, or a topic title. This

could be done by applying a style to a paragraph, filling in the correct field in a form, or making sure the line of text has a line of equal signs underneath it.

Make a list

A lot of technical content requires lists of information: lists of prerequisites, lists of options, lists of things to do next. In many ways, this is just a specialization of *Format the content*. However, list items are very common.

Make a procedure

Many would state that this is the core of technical writing. Our main job is to tell people how to do things. This is often done via a procedure. This could be as simple as making a numbered list, or following a set of styles prescribed by your style guide, or the complex structure of a DITA task topic. Similar to *Make a list*, this is a specialization of *Format the content*, but there are often special tools or techniques for working with procedures.

Make a table

Tables are a great way to present certain types of information. They also tend to be very complicated. You need to learn how to insert, format, etc. Again, it's another specialization of *Format the content*, but they are tricky. If you're working with lightweight markup, it can be the most complicated formatting. Most tools have wizards and various tools that take some time to learn.

Insert an image

A picture is worth a thousand words. You will need to include images in your content. Different tools do this in different ways.

In general, there are two ways you need to consider inserting images: inline or separately. An inline image is used often in procedures to assist with instructions such as "Click the **Save** (💾) icon".

For larger images, you need to insert them so that they stand alone. That could be inside of a figure with a caption, or just on a separate line. In rare cases, you may configure text wrapping to go around the image.

Insert a link

Content these days usually needs to refer to other things. This can be links to external web sites or links to other parts of your content. (Internal references are often called *cross-references*.)

Linking can be something you have to control manually, or it might be something that can be autogenerated by your tooling. It can also be something you may have to maintain over time, or maybe the tool helps you keep the links working.

Add inline formatting

Besides formatting giant blocks of text, such as paragraphs, lists, and tables, you often need to apply formatting to specific words or phrases in your giant blocks of text. Common examples include making things you click in your software bold, making variables italic, and making command names use a monospace font.

Structure your content

Most likely you don't have a lot to say about the big structure, such as new help systems, new books, or new output formats. However, within the area where you do have control, you will need to know how to structure your content. By structure, this generally means what makes a new section or chapter, and how section headings are nested. This could be making sure you use the right heading level style or could be structuring a bunch of topics into something that makes a table of contents.

Occasional

You may or may not need to do these tasks. These depend upon your tooling, your implementation, and how structured your work place is.

Make a new thing	Most of the time, as a technical writer you are working with content that is already semi-established. The User's Guide already exists, you're just adding new sections to it. You very rarely make a new thing. Also, there's usually a senior writer or information architect who minimally has extremely strong opinions and potentially corporate guidelines to follow about how new large things (new books, new help systems, new top level sections) can be created. However, if you are working in a topic based system, you will probably often need to make new topics.
Update the table of contents	You will have to structure your content. Depending upon your tooling, your table of contents could be automatically generated, or something you have update. You may also need to be aware of your heading structures to make sure the table of contents works correctly. For example, some tools can't handle if your headings go from heading level 1 to heading level 3, skipping heading level 2.
Indexing or tagging for search	This may not make an actual index any more, but you probably need to do something to improve search results. This can be marking index terms, adding keywords, or adding tags.
Add context sensitive links	If you have a context sensitive help system, you need to add the markers or metadata or whatever to ensure your software can open the right page in your content. This could be something complicated to identify a help ID, or just making sure a file is named something specific.
Work with reuse	This is entirely tooling dependent, but if your tool allows for reuse, you should probably take advantage of it. Reuse can be many different things. You could have certain words or phrases that need to be inserted a specific way, such as version numbers or product names. You could have a way to share topics or chapters between different books. You could be able to reuse specific paragraphs, or any other defined chunk. You will need to learn both how to do the reuse in the tool you are using, and how your company maintains and organizes reused things.
Work with conditional text	Some tools allow you to mark content so that it only appears when certain conditions are met. This means you can mark content that only appears for specific outputs, such as between HTML or PDF output, or based on product or component.

Suggestion

Make a simple set of content that you can keep reusing in different tools.

Every single one of these homework assignments can be portfolio pieces. You can write complicated instructions on how to do things to show off what a great writer you are.

However, to pass this course, all you need to do is try to make all the basic building blocks in different tools.

When trying to learn the different tools, it might be easier if you just have text you are copying and pasting instead of trying to write something new *and* learn how to format something in yet another tool.

I would very much prefer you didn't cut and paste the following, but this is the minimal amount I'm looking for:

Look a Title

The above is a title. It could be formatted as something specifically labeled a title, or maybe just a heading level one.

This is another new paragraph with something **bold** and *italic*. I now have two separate paragraphs, and have some inline formatting.

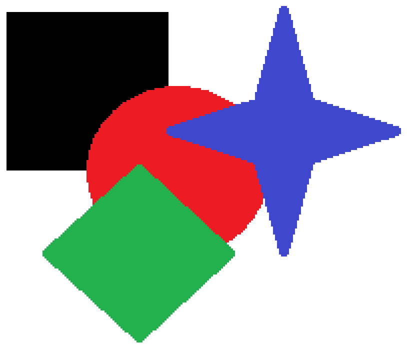
- I also need a list.
- This is a second list item.

  With a paragraph that lines up with the list item.
- And look, a third list item.

I have proven I can make an unordered list. And I can make another paragraph. Oh. I need a link to something, like http://www.google.com.

1. Now I need an ordered list.
2. I can use this for procedures.
3. Or to identify items in an image I don't want to translate.

There. Proof I can make a numbered list.



And there is an image. And I remembered to add alternative text.

Table Head 1	**Table Head Column 2**
Column 1	Column 2
More Column 1	More Column 2
And a third row for fun	With one more column 2

Assumptions

These instructions assume the following:

- You have already cloned the class repository and you are keeping it synced.
- You have Notepad++ installed.

  Instructions are in `Handouts/tips_and_tricks.pdf`.
- You have modified Windows File Explorer to always show known file type extensions.

  Instructions are in `Handouts/tips_and_tricks.pdf`.

Week 1: Getting Started with Git

The homework for week 1 is to set up your GitHub account.

The homework helper for this week is very minimal because I've tried to have a lot of information in various places.

- `mcc_tools_tech\Week01-IntroGitHub\Week1-IntroGitHub.html` presentation I use during class.

  Yes, you are going to have access to all the presentations. While there are versions in existence for the entire class, so you can read ahead and be completely bored during class, but I am updating the presentations before each class, so things will change.
- `mcc_tools_tech\Week01-IntroGitHub\using_git.pdf` (*Using Git*) document that provides more detail than the presentation.

  This document is also available in Blackboard.
- `mcc_tools_tech\Handouts\git_cheatsheet.pdf` (*Git Cheatsheet*) document that is more of a reference, but might give a detail in a different way that makes something click.
- https://drive.google.com/file/d/1ij3FnYD-0f0TFgDDzeVpo74udtSKuTGJ/view?usp=sharing – A video I threw together in 2022 to help you get started with GitHub.

  Everywhere it says "Winter2022", you should use "Winter2024".

  Many times in the work place, the training information is slightly out of date. Enjoy learning how to adapt to slightly inaccurate directions.
- Syncing Repos in GitHub – Another video about syncing repositories in GitHub I created in 2022. You probably don't need it for this week, but you'll definitely need it for next week.

  Everywhere it says "Winter2022", you should use "Winter2024".

Week 2: Progressive Information Disclosure

This homework is just being familiar with editing text files.

Be aware that when you're working with computers you often have to deal with *reserved characters*. To make code work, certain words and certain characters mean very specific things.

For example, in our properties file, the equals sign (=) has a specific meaning. Each line of the properties file has the format:

```
Name = Value
```

So, whenever whatever is processing runs into a =, the processor does something special.

For most properties files, the end of a line indicates "done now". So a line break is a reserved character.

Sometimes, you still need to use those special, reserved characters.

To do this, you need to *escape* the special or reserved character. This means you use a different reserved character to say "what follows should be emitted as what it is, don't treat it like the reserved character".

How you escape a reserved character is unique for every language. In general, I just search the internet for "How to escape equals sign in Java properties file" and find out that I generally need to use backslashes (\).

Be aware that the automation I am using is very simplistic. I cannot pass HTML in the properties file and have it work in the generate HTML file.

Week 3 Microsoft Word

Using styles to try and convince Microsoft Word to behave.

The majority of Microsoft Word users are not power users. Most people learned how to write a paper with maybe some running headers and footers and a few cross-references. While you may use it every day in a workplace, you are most likely starting from a template or by editing an existing document.

To produce quality content, you need to make sure you are applying your company's style guide. You need to produce content that is branded appropriately, and follow whatever guidance. You need to use the correct fonts, the right spacing, make sure figures have the correct labeling, etc.

You need to learn to both read the style guide and how to properly apply the styles.

Technically you can be aware of whatever the formatting is, and manually apply it to every paragraph as necessary, but then you're spending more than half your time applying formatting. With styles, you just have to apply the correct style.

Add Template to Custom Office Templates Folder

By default Microsoft Word has a specific folder where it expects to find templates. You need to add the template to this folder.

You should search the internet for the location of the `Custom Office Templates` folder for your version of Microsoft Word and your operating system.

Copy `mcc_tools_tech/Week03-WordOffice/Homework/MCC_WordTemplate.dotx` to the `Custom Office Templates` directory for your system.

If you cannot find this location, you can make your own custom one. Be aware that Microsoft Word can only configure one folder for custom templates. If you change it to a custom directory, you'll have to always put templates here.

Convince Word to find personal templates

These instructions will be different for every version of Word, but hopefully these get you headed in the right direction.

Make sure you have Word templates (.dotx) in your `Custom Office Templates`, which is usually `C:\Users\`*`User Name`*`\Documents\Custom Office Templates`.

These instructions were determined using Microsoft Office 2021.

1. In Microsoft Word, select **File** > **Options**.
2. Select **Save** from the navigation pane.
3. In the **Default personal templates location** field, enter the path to where your Word templates are located.
4. Click **OK**.

If all goes well, when you select **File** > **New**, there should be a **Personal** option.

If All Else Fails

If you are unable to find your Custom Office Templates folder or convince Word to use it, try this method.

1. Double-click the `MCC_WordTemplate.dotx` file.
   This should open a new (Document 1) Word document using the template.
2. Use **File** > **Save As** to save your file.

   Technically, you probably just need to use **Save**, but I'd rather be safe than sorry.

Working with Files in GitHub.com

You should learn how to use GitHub Desktop or some other tool to work with files in git. However, github.com does have limited browser-based tools you can use.

For full functionality, you absolutely need to use a local git client. For several of the assignments, you need to have a whole bunch of files locally, and to be able to turn in the multiple files edited, you need to use a local git client. To be able to say you can use git on your resume, you need to know how to use a local client.

That said, sometimes you need a work around. You might be travelling and need to fix something quickly. You can get your local git instance into a tangled mess and just want to get that file turned in (and ask me for help later). You may need to do something on the virual desktop, where you can't install a git client.

GitHub.com provides some basic tools in the web UI that you can use when you have to.

Create a New Text File

GitHub.com includes a text editor, so you can create new text files in the browser.

Remember, when I say text file, I mean any file that is "just text", even if it doesn't have a `.txt` extension.

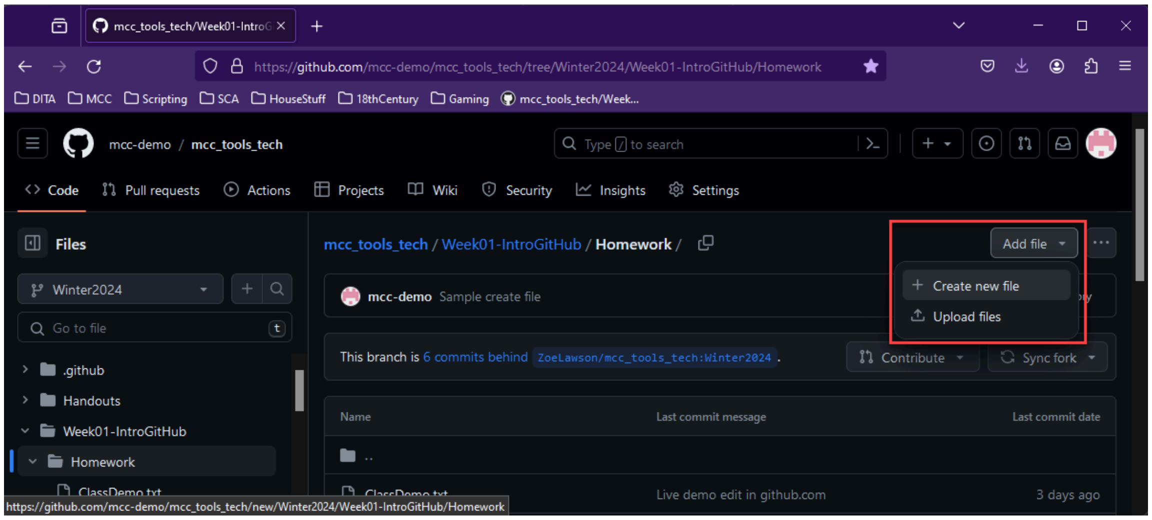
This includes, but is not limited to:

- `.css`
- `.dita`
- `.html`
- `.md`
- `.properties`
- `.rst`
- `.svg`

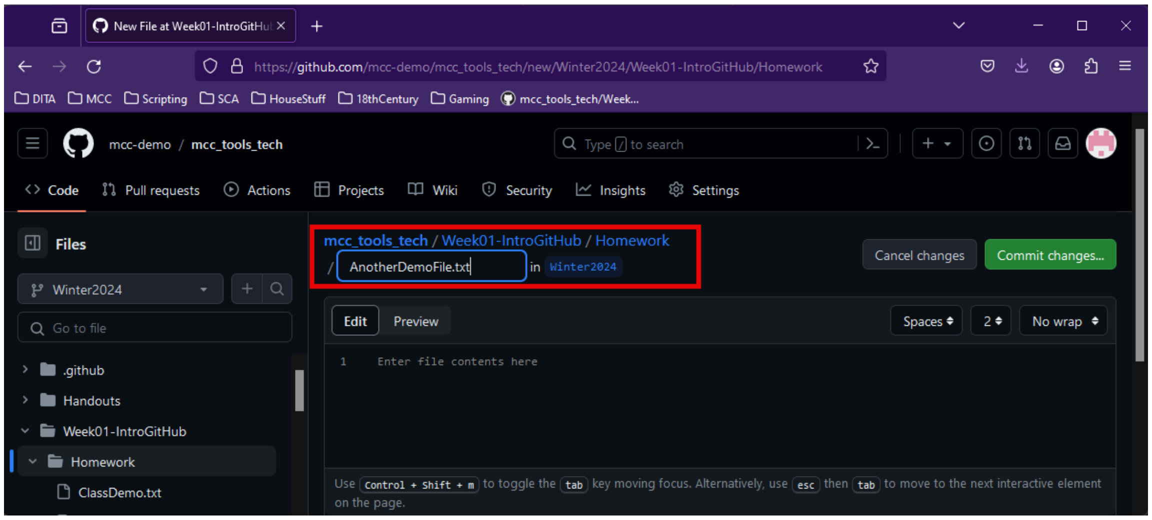
See https://docs.github.com/en/repositories/working-with-files/managing-files/creating-new-files, or you can try my simplified instructions.

1. Log in to GitHub.com, and navigate to your fork of the class repository.
   For example, using the demo user, mcc-demo, I would go to https://github.com/mcc-demo/mcc_tools_tech.
2. Navigate to where ever you need to create the file.
   For this example, I will use the Week 1 Homework folder, as that is the default place to play with working with files in git without confusing anything.

3. Select **Add file** > **Create new file**.


mcc_tools_tech/Week01-IntroG
https://github.com/mcc-demo/mcc_tools_tech/tree/Winter2024/Week01-IntroGitHub/Homework
mcc-demo / mcc_tools_tech
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
Winter2024
Go to file
.github
Handouts
Week01-IntroGitHub
Homework
ClassDemo.txt
mcc_tools_tech / Week01-IntroGitHub / Homework /
Add file
Create new file
Upload files
mcc-demo Sample create file
This branch is 6 commits behind ZoeLawson/mcc_tools_tech:Winter2024 .
Contribute
Sync fork
Name
Last commit message
Last commit date
ClassDemo.txt
Live demo edit in github.com
3 days ago
https://github.com/mcc-demo/mcc_tools_tech/new/Winter2024/Week01-IntroGitHub/Homework

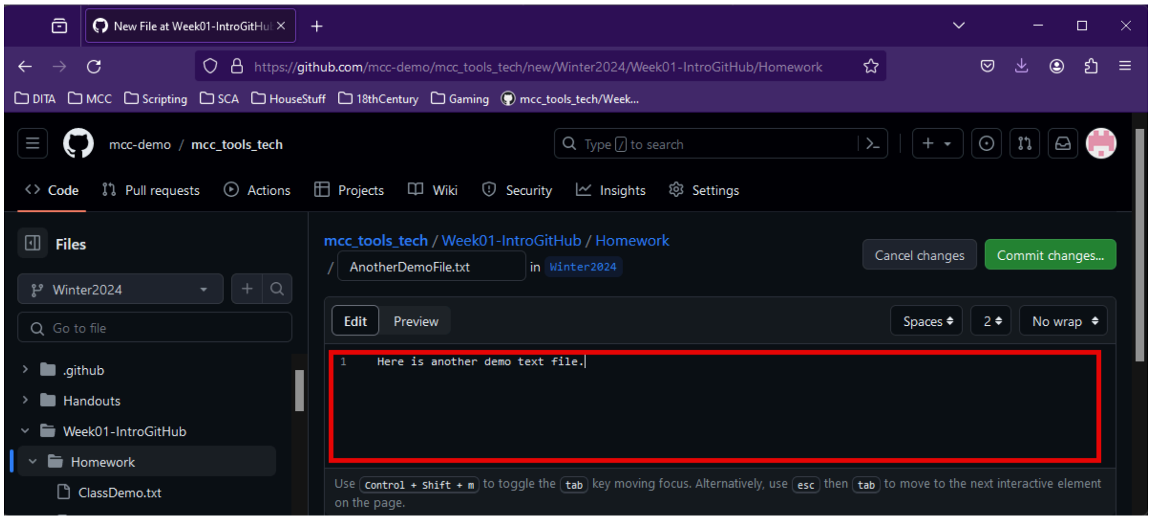
4. Provide a name for your file.


New File at Week01-IntroGitHu
https://github.com/mcc-demo/mcc_tools_tech/new/Winter2024/Week01-IntroGitHub/Homework
mcc-demo / mcc_tools_tech
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
Winter2024
Go to file
.github
Handouts
Week01-IntroGitHub
Homework
ClassDemo.txt
mcc_tools_tech / Week01-IntroGitHub / Homework
/ AnotherDemoFile.txt in Winter2024
Cancel changes
Commit changes...
Edit
Preview
Spaces
2
No wrap
Enter file contents here
Use control + shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.

You have to enter the extension.

You can also add a folder to the path to make a new sub-folder, such as typing `NewFolder/`. The instant you type the folder separator (`/`), GitHub recognizes it as a folder. The new folder name is added to the path and the file name field is emptied.

5. Enter your text in the text field, where it says "Enter file contents here".

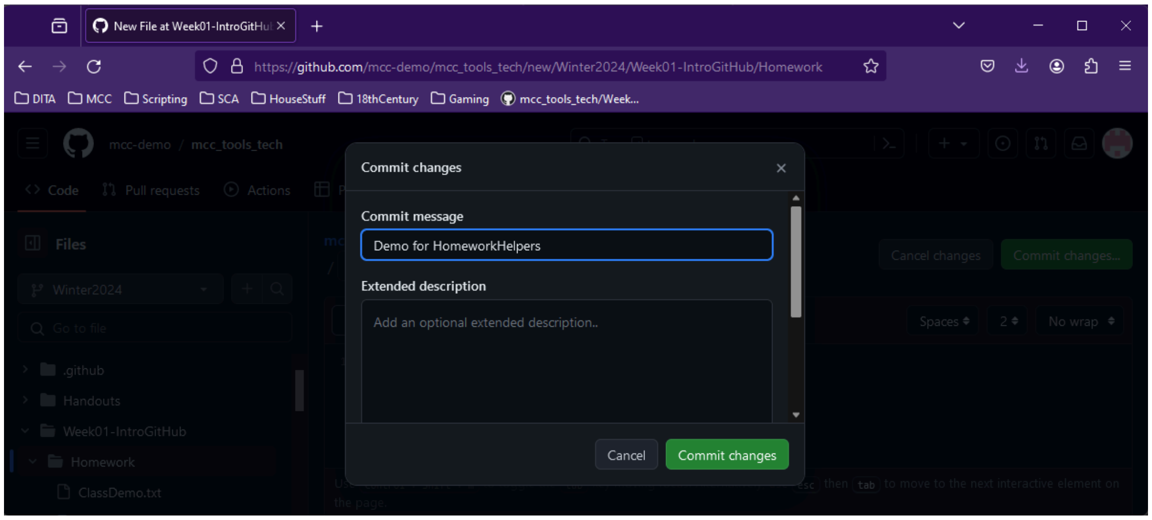

New File at Week01-IntroGitHub
https://github.com/mcc-demo/mcc_tools_tech/new/Winter2024/Week01-IntroGitHub/Homework
mcc-demo / mcc_tools_tech
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
Winter2024
Go to file
.github
Handouts
Week01-IntroGitHub
Homework
ClassDemo.txt
mcc_tools_tech / Week01-IntroGitHub / Homework
AnotherDemoFile.txt in Winter2024
Cancel changes
Commit changes...
Edit
Preview
Spaces
2
No wrap

```
Here is another demo text file.
```

Use Control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.

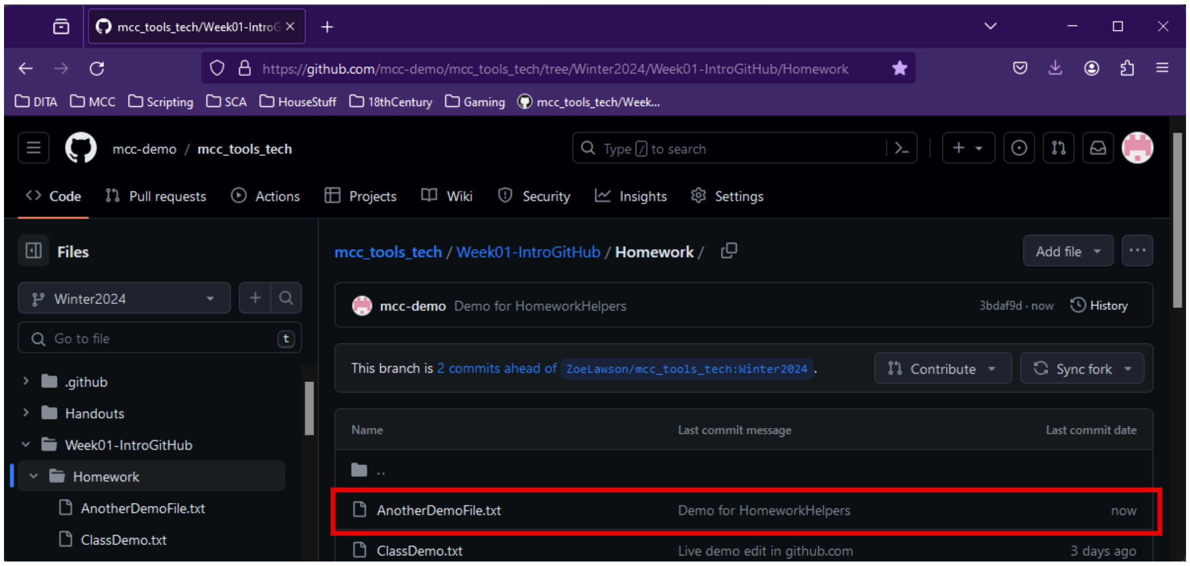
If you're using a file format GitHub knows how to display, such as Markdown, it can show what it will look like.

6. When you're done making changes, click **Commit changes**.
7. Provide a good commit message and click **Commit changes**.



Commit message: Demo for HomeworkHelpers

Ta-da! Your text file has been added to your fork.


mcc_tools_tech/Week01-IntroG
https://github.com/mcc-demo/mcc_tools_tech/tree/Winter2024/Week01-IntroGitHub/Homework
DITA MCC Scripting SCA HouseStuff 18thCentury Gaming mcc_tools_tech/Week...
mcc-demo / mcc_tools_tech
Type / to search
Code Pull requests Actions Projects Wiki Security Insights Settings
Files
Winter2024
Go to file
.github
Handouts
Week01-IntroGitHub
Homework
AnotherDemoFile.txt
ClassDemo.txt
mcc_tools_tech / Week01-IntroGitHub / Homework /
Add file
mcc-demo Demo for HomeworkHelpers 3bdaf9d · now History
This branch is 2 commits ahead of ZoeLawson/mcc_tools_tech:Winter2024 .
Contribute
Sync fork

Name	Last commit message	Last commit date
..		
AnotherDemoFile.txt	Demo for HomeworkHelpers	now
ClassDemo.txt	Live demo edit in github.com	3 days ago

Remember that this just adds the file to your fork. You still need to make a pull request.

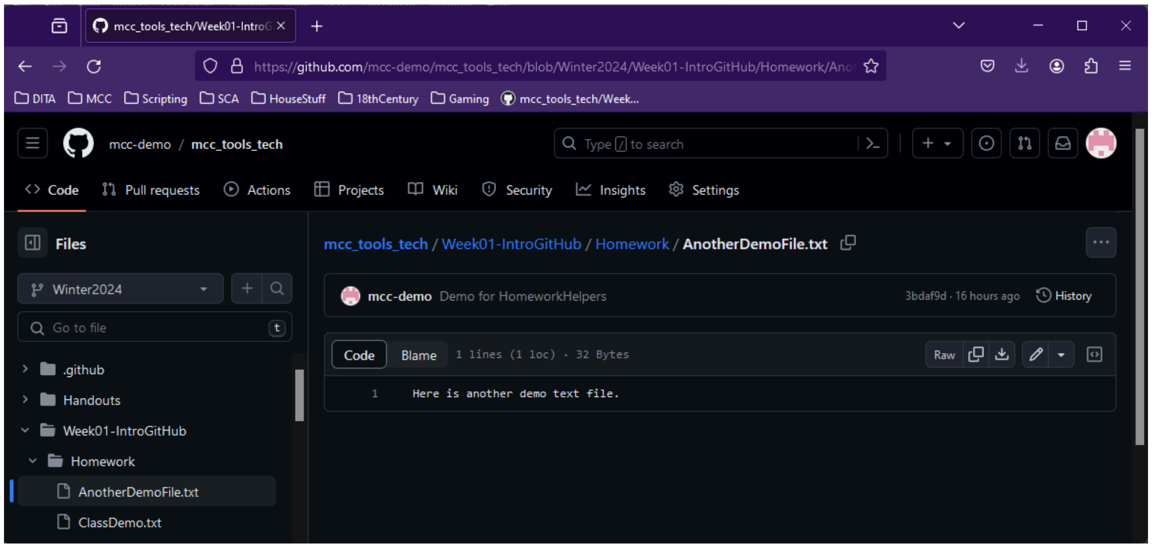
If you need a refresher on how to do a pull request, see `mcc_tools_tech\Week01-IntroGitHub\using_git.pdf` (Using Git) or `mcc_tools_tech\Handouts\git_cheatsheet.pdf` (Git Cheatsheet).

Edit an Existing Text File

With the GitHub text editor, you can edit text-based files in your browser.

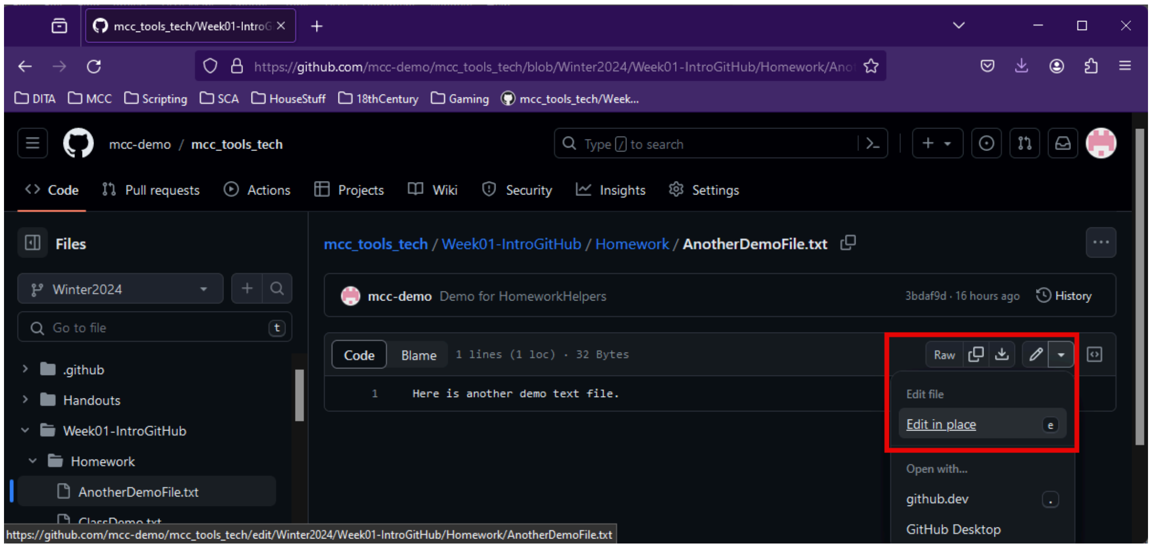
You can read the GitHub.com docs here: https://docs.github.com/en/repositories/working-with-files/managing-files/editing-files or you can look at my simplified procedure.

1. Log in to GitHub.com and navigate to the file you want to edit in your fork.
   In this example, I am going to edit the file I just made, `mcc_tools_tech/Week01-IntroGitHub/Homework /AnotherDemoFile.txt`.


mcc_tools_tech/Week01-IntroG
https://github.com/mcc-demo/mcc_tools_tech/blob/Winter2024/Week01-IntroGitHub/Homework/Ano
DITA MCC Scripting SCA HouseStuff 18thCentury Gaming mcc_tools_tech/Week...
mcc-demo / mcc_tools_tech
Type / to search
Code Pull requests Actions Projects Wiki Security Insights Settings
Files
Winter2024
Go to file
.github
Handouts
Week01-IntroGitHub
Homework
AnotherDemoFile.txt
ClassDemo.txt
mcc_tools_tech / Week01-IntroGitHub / Homework / AnotherDemoFile.txt
mcc-demo Demo for HomeworkHelpers
3bdaf9d · 16 hours ago
History
Code
Blame
1 lines (1 loc) · 32 Bytes
Raw

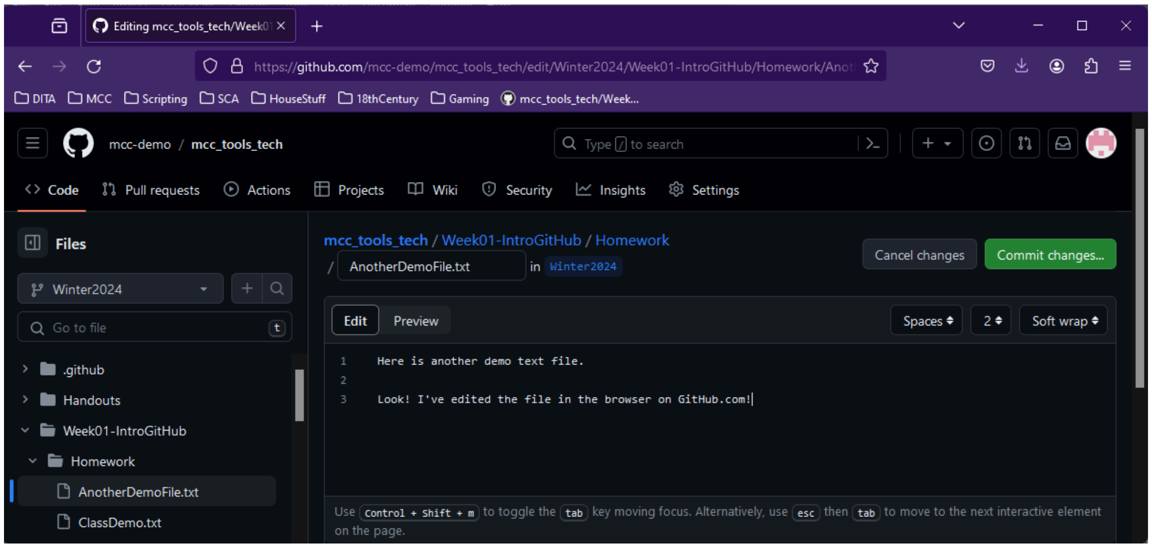
```
Here is another demo text file.
```

2. Click the pencil icon, or the arrow next to it and select **Edit in place**.


mcc_tools_tech/Week01-IntroG
https://github.com/mcc-demo/mcc_tools_tech/blob/Winter2024/Week01-IntroGitHub/Homework/Ano
DITA MCC Scripting SCA HouseStuff 18thCentury Gaming mcc_tools_tech/Week...
mcc-demo / mcc_tools_tech
Type / to search
Code Pull requests Actions Projects Wiki Security Insights Settings
Files
Winter2024
Go to file
.github
Handouts
Week01-IntroGitHub
Homework
AnotherDemoFile.txt
ClassDemo.txt
mcc_tools_tech / Week01-IntroGitHub / Homework / AnotherDemoFile.txt
mcc-demo Demo for HomeworkHelpers
3bdaf9d · 16 hours ago
History
Code
Blame
1 lines (1 loc) · 32 Bytes
Raw
Edit file
Edit in place
e
Open with...
github.dev
.
GitHub Desktop
https://github.com/mcc-demo/mcc_tools_tech/edit/Winter2024/Week01-IntroGitHub/Homework/AnotherDemoFile.txt

```
Here is another demo text file.
```

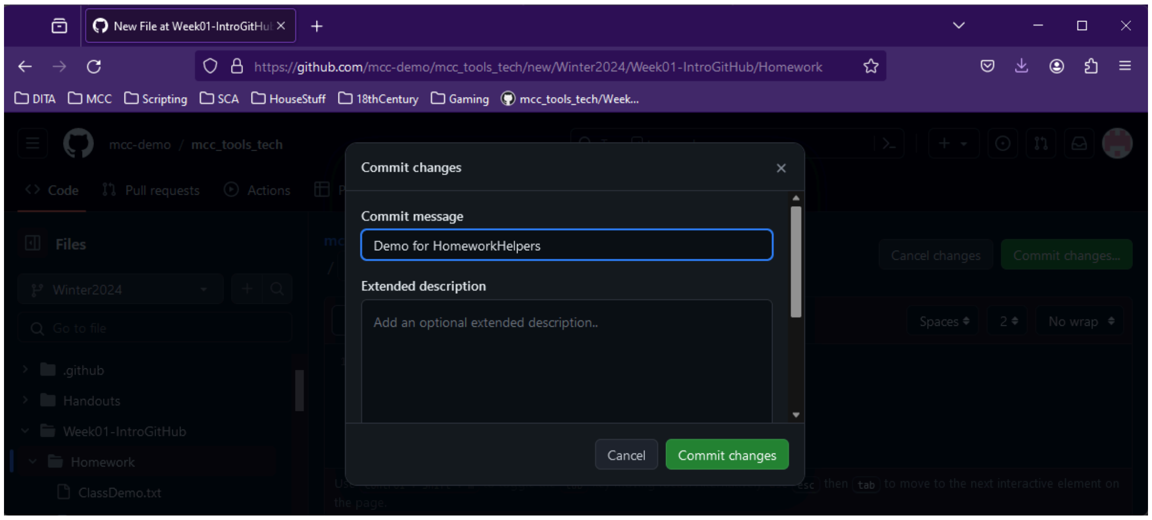
3. Make your changes to the file.


Editing mcc_tools_tech/Week0
https://github.com/mcc-demo/mcc_tools_tech/edit/Winter2024/Week01-IntroGitHub/Homework/Anot
DITA
MCC
Scripting
SCA
HouseStuff
18thCentury
Gaming
mcc_tools_tech/Week...
mcc-demo / mcc_tools_tech
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
Winter2024
Go to file
.github
Handouts
Week01-IntroGitHub
Homework
AnotherDemoFile.txt
ClassDemo.txt
mcc_tools_tech / Week01-IntroGitHub / Homework
/ AnotherDemoFile.txt in Winter2024
Cancel changes
Commit changes...
Edit
Preview
Spaces
2
Soft wrap
Use Control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.

```
Here is another demo text file.

Look! I've edited the file in the browser on GitHub.com!
```

4. Provide a good commit message and click **Commit changes**.


New File at Week01-IntroGitHu
https://github.com/mcc-demo/mcc_tools_tech/new/Winter2024/Week01-IntroGitHub/Homework
Commit changes
Commit message
Extended description
Add an optional extended description..
Cancel
Commit changes

Commit message: Demo for HomeworkHelpers

You have now edited a text file on GitHub.com.

Remember that this just edits the file to your fork. You still need to make a pull request.

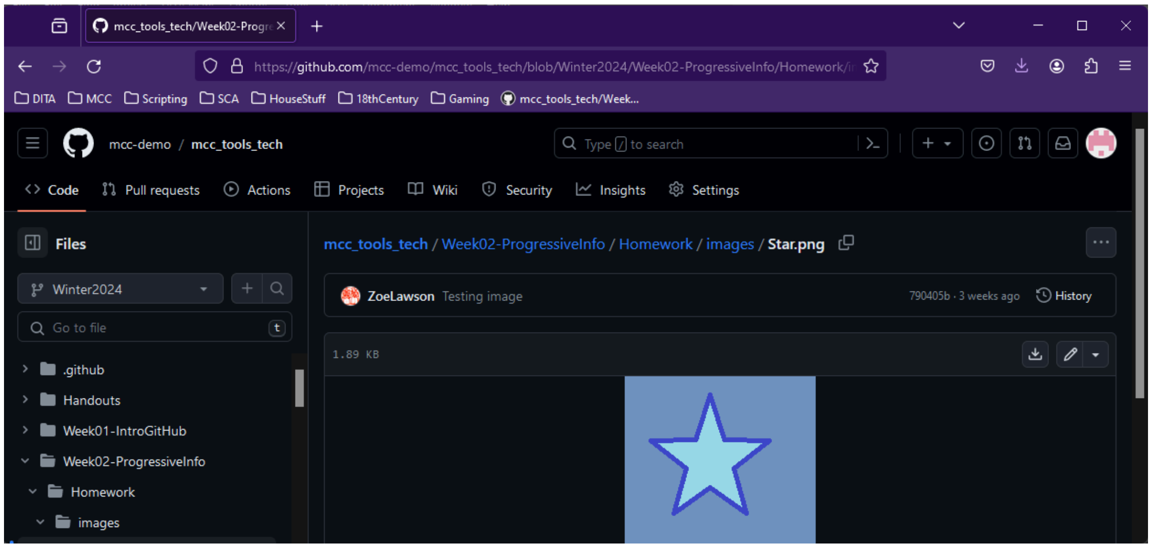
If you need a refresher on how to do a pull request, see `mcc_tools_tech\Week01-IntroGitHub\using_git.pdf` (Using Git) or `mcc_tools_tech\Handouts\git_cheatsheet.pdf` (Git Cheatsheet).

Download an Existing File

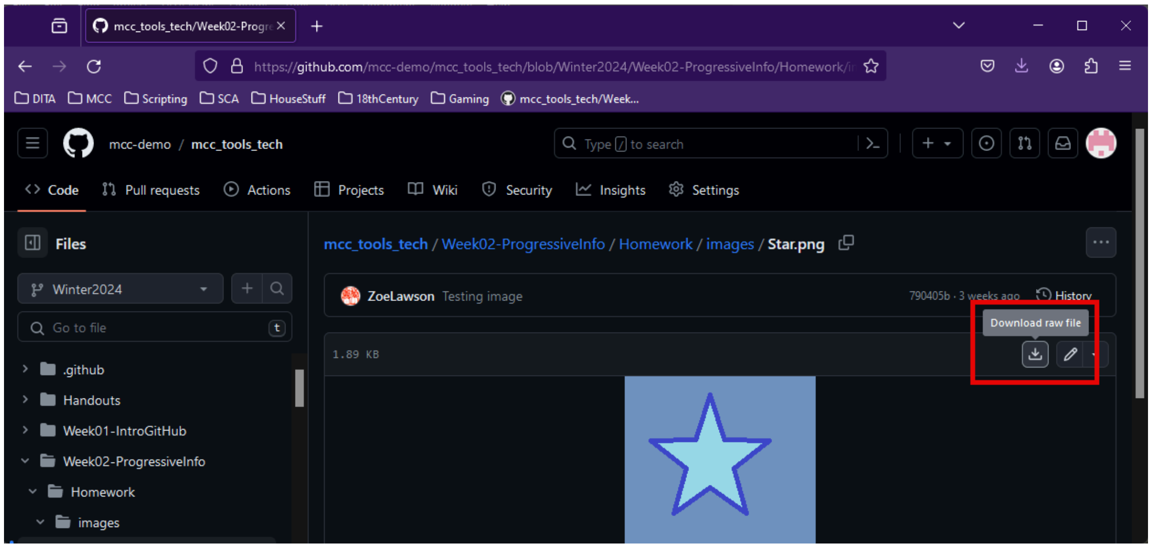
If you need to use a different computer than you usually do you can download a specific file from GitHub.com.

This is also useful if you need to restore something from the class repository. Instead of going through the mess of undoing commits, etc., you can go to the class repository and download the original file. You can then overwrite the file in your own fork to fix it.

1. Log in to GitHub.com and navigate to the file you want to download.



2. Click **Download raw file**.



The File Browser opens.

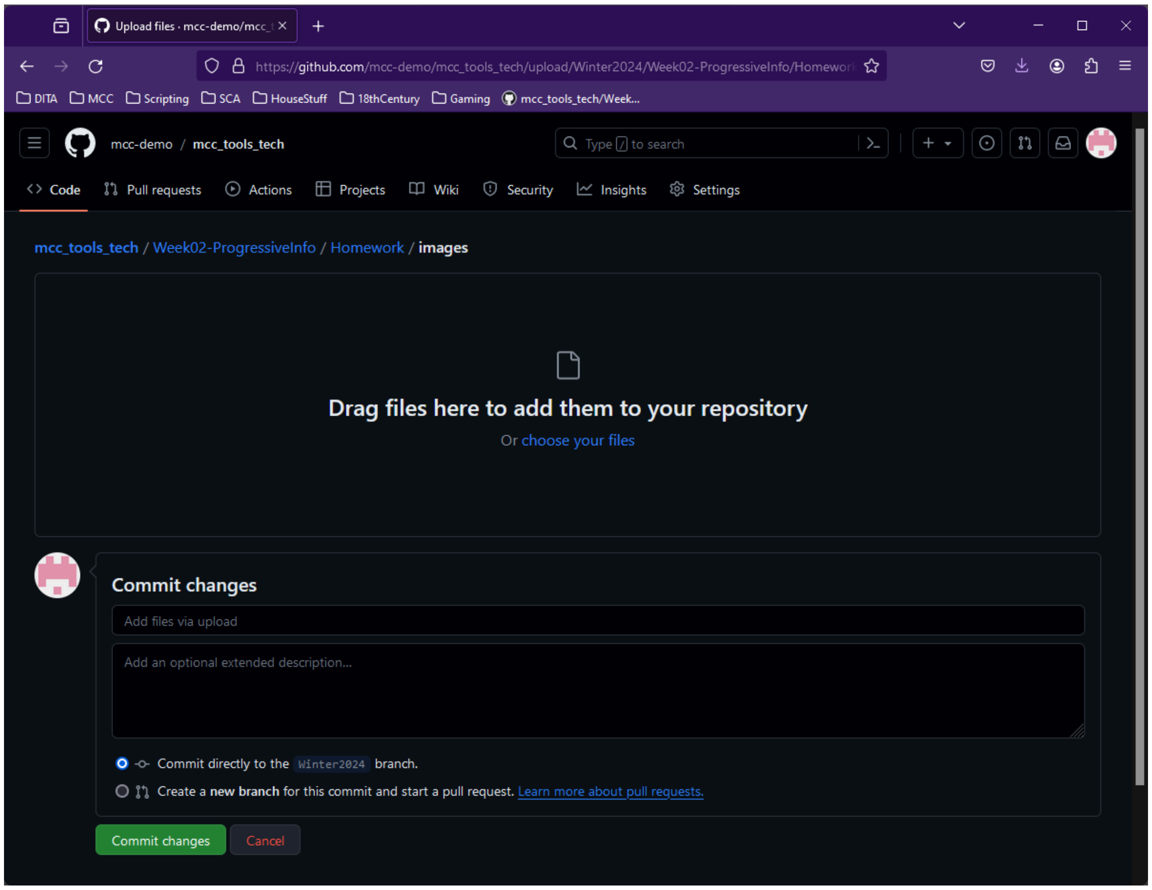
3. Navigate to where you want to save the file and click **Save**.

You now have the file locally, which you can work with as you need.

Upload a New File

Use this to add a new file to your repository, or to update a file of the same name.

1. Log in to GitHub and navigate to the folder where you want to add or update the file.
2. Select **Add file** > **Upload files**.


Upload files · mcc-demo/mcc_
https://github.com/mcc-demo/mcc_tools_tech/upload/Winter2024/Week02-ProgressiveInfo/Homewor
DITA MCC Scripting SCA HouseStuff 18thCentury Gaming mcc_tools_tech/Week...
mcc-demo / mcc_tools_tech
Type / to search
Code Pull requests Actions Projects Wiki Security Insights Settings
mcc_tools_tech / Week02-ProgressiveInfo / Homework / images
Drag files here to add them to your repository
Or choose your files
Commit changes
Add files via upload
Add an optional extended description...
Commit directly to the Winter2024 branch.
Create a new branch for this commit and start a pull request. Learn more about pull requests.
Commit changes
Cancel

3. Drag and drop your file to the browser, or click **choose your files** to select files using a file browser.

   If you add a file that is the same name as an existing file, you will update it. Remember that Git cares about file name case. (`YourFile.txt` is different than `yourfile.txt`.)
4. After adding all the files you want to update, provide a commit message.
5. Click **Commit changes**.

Congrats, you have added or updated files in your GitHub repository.

Remember that this just adds the file to your fork. You still need to make a pull request.

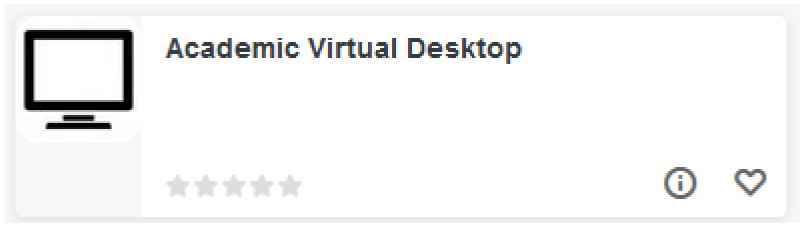
If you need a refresher on how to do a pull request, see `mcc_tools_tech\Week01-IntroGitHub\using_git.pdf` (Using Git) or `mcc_tools_tech\Handouts\git_cheatsheet.pdf` (Git Cheatsheet).

Using the Virtual Desktop

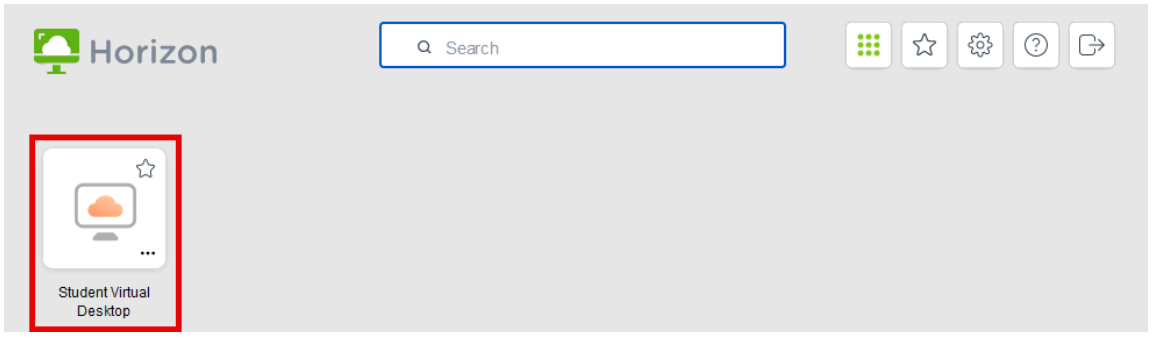
Middlesex Community College provides a virtual student desktop to access applications provided by the school.

The virutual desktop gives you access to all the Microsoft Office applications and Techsmith Camtasia.

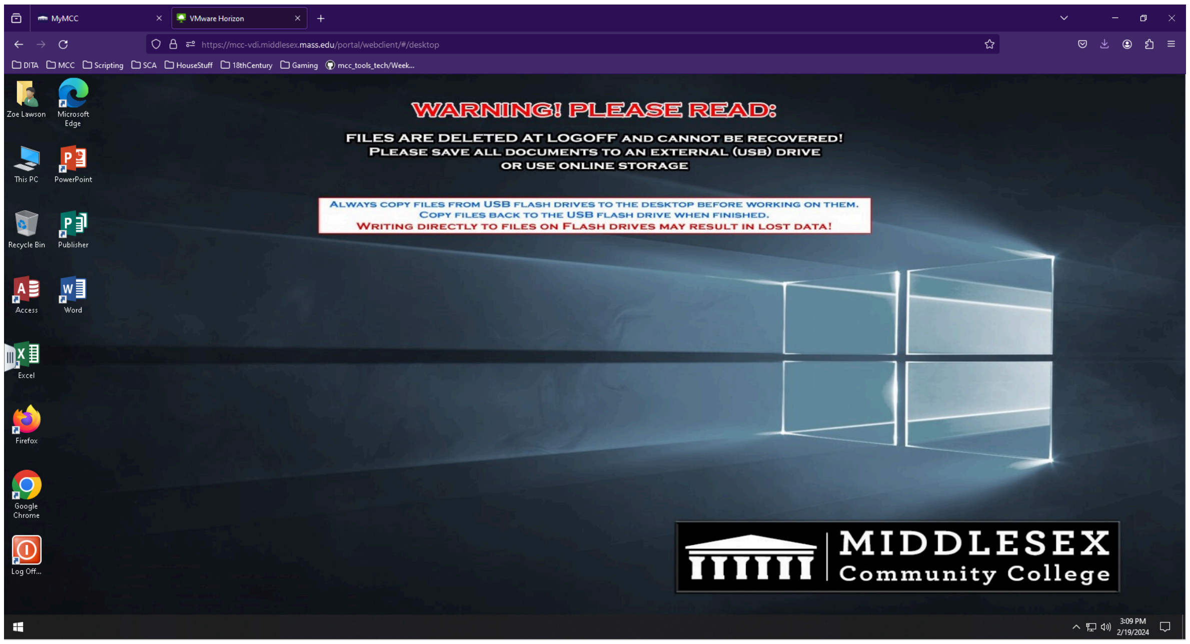
1. Log in to MyMCC (https://mymcc.middlesex.edu/).
2. Click **Academic Virtual Desktop**.



3. Sign in to MCC (again).
4. Click **Student Virtual Desktop**.



5. You may be prompted to choose between the desktop client and the browser client. This example uses the browser client.



WARNING! PLEASE READ:

FILES ARE DELETED AT LOGOFF AND CANNOT BE RECOVERED!
PLEASE SAVE ALL DOCUMENTS TO AN EXTERNAL (USB) DRIVE
OR USE ONLINE STORAGE

ALWAYS COPY FILES FROM USB FLASH DRIVES TO THE DESKTOP BEFORE WORKING ON THEM.
COPY FILES BACK TO THE USB FLASH DRIVE WHEN FINISHED.
WRITING DIRECTLY TO FILES ON FLASH DRIVES MAY RESULT IN LOST DATA!

You now have access to the virtual desktop in your browser window.

Remember that while you might be able to save files locally, there is no guarantee they will still be there the next time you log in. Save any work to your own cloud drive (OneDrive, Google Drive, etc.), or take advantage of the GitHub.com browser tools.