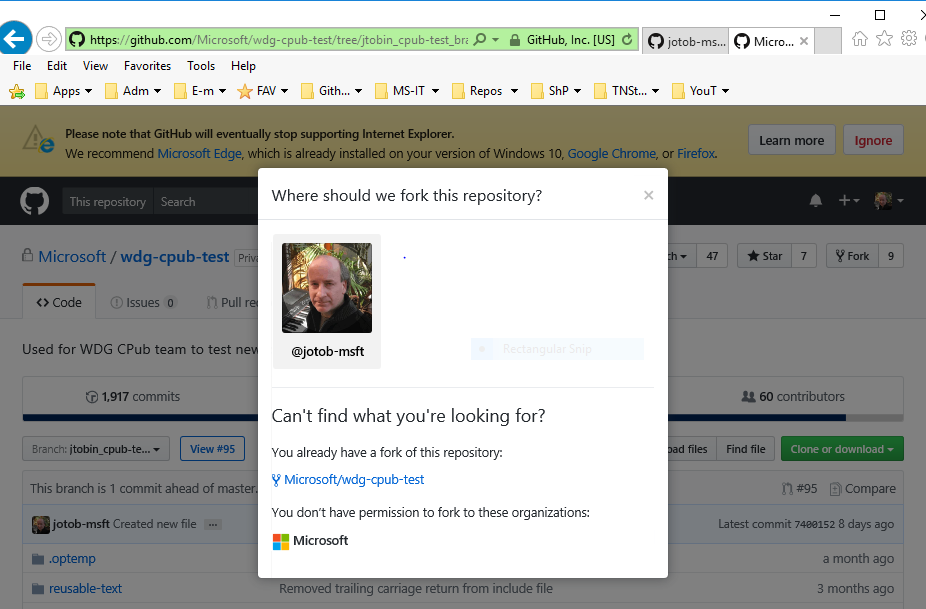
1. From the Github website, you need to fork the repository **win-cpub-itpro-docs from the**  Microsoft master repo:

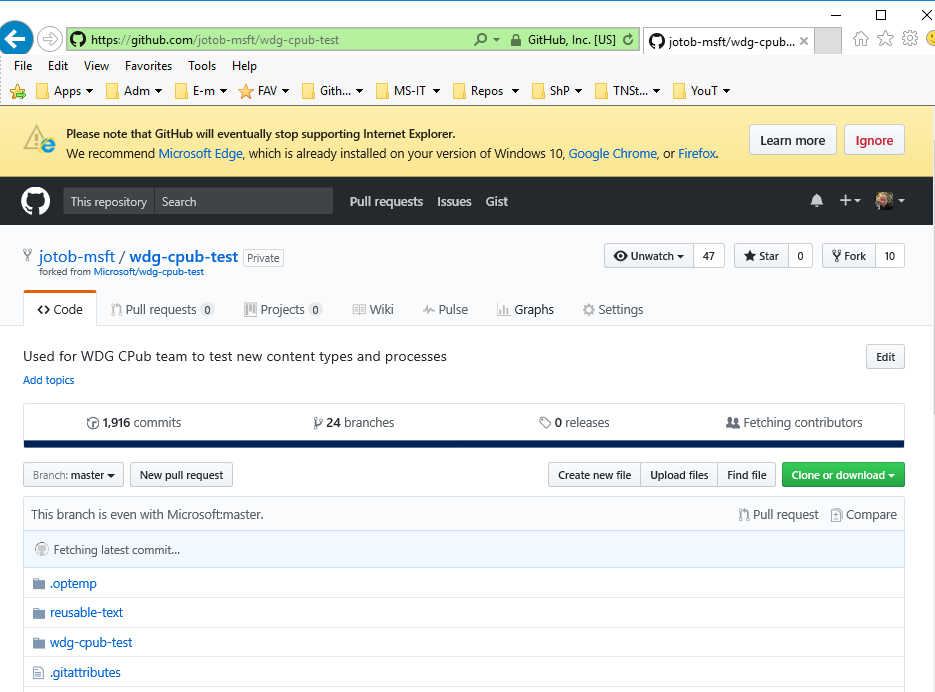
Microsoft/**win-cpub-itpro-docs**

**That is, you are forking from the Microsoft repository, but your fork will be named after your profile.** jotob-msft/win-cpub-itpro-docs



An onscreen message will say **“Forking Microsoft/wdg-cpub-test”.** It only takes a few seconds to fork the repo.

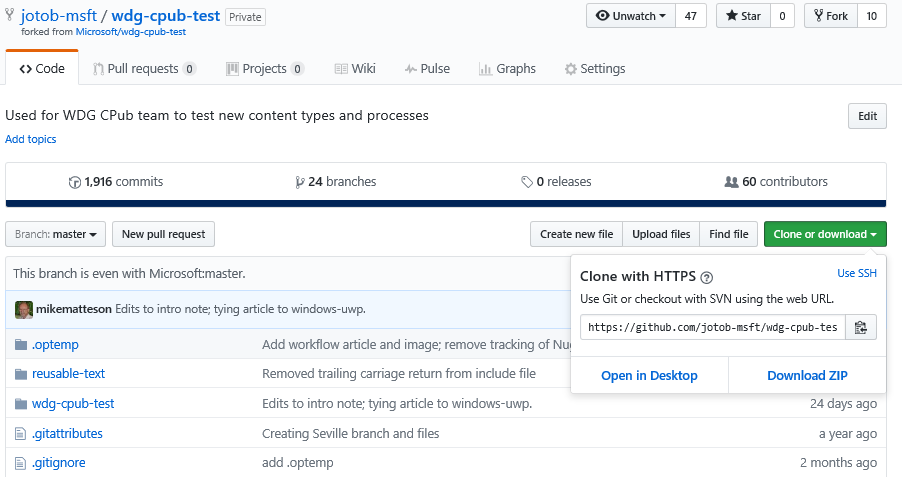
1. Click the fork button at the top right of the github.com site.
2. Click you profile picture to move the fork into your account profile. Next, you clone that remote fork, jotob-msft/win-cpub-itpro-docs using its URL, and copy it down to your local machine (using either the Github website or using the command line).



1. Now, you need to clone the fork. Before cloning, create a local directory repo in **File Explorer** to hold the clone.

Copy the URL, then go to the command line and type:

Git clone [URL]



1. At the git command line, cd to the folder to which you have cloned the repo and then do a **dir** to confirm that the repo has been successfully copied down.

v-jotob@SEA-STT-JTobin MINGW64 /c/repo

$ **git clone** <https://github.com/jotob-msft/win-cpub-itpro-docs.git> --- git clone [URL]

v-jotob@SEA-STT-JTobin MINGW64 /c

$ cd test-repo

v-jotob@SEA-STT-JTobin MINGW64 /c/test-repo

$ git clone https://github.com/jotob-msft/wdg-cpub-test.git

Cloning into 'wdg-cpub-test'...

remote: Counting objects: 10610, done.

remote: Total 10610 (delta 0), reused 0 (delta 0), pack-reused 10610

Receiving objects: 100% (10610/10610), 33.06 MiB | 2.75 MiB/s, done.

Resolving deltas: 100% (6955/6955), done.

Checking out files: 100% (527/527), done.

v-jotob@SEA-STT-JTobin MINGW64 /c/test-repo

$ cd wdg-cpub-test

v-jotob@SEA-STT-JTobin MINGW64 /c/test-repo/wdg-cpub-test (master)

$ dir

HelloWorld.md reusable-text wdg-cpub-test

README.md test.txt yaml\ block\ template.md

v-jotob@SEA-STT-JTobin MINGW64 /c/test-repo/wdg-cpub-test (master)

$

Before you push or pull any content, you must first confirm that your remote fork is listed as what is known as ‘origin’. That is, you must confirm the presence and identity of origin, which is your remote fork which bears your profile name. ‘Origin’ is the default remote repository where you will make your changes, and from which you will be pulling content and pushing content to:

jotob-msft/win-cpub-itpro-docs

To confirm the origin, use the command:

**git remote -v**

which give similar output as follows:

**origin** https://github.com/**jotob-msft**/win-cpub-itpro-docs.git (fetch)

**origin**  https://github.com/**jotob-msft**/win-cpub-itpro-docs.git (push)

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (**master**)

The first line of output shows the repo you are pulling data from (origin) (a fetch operation). A fetch is a get.

The second line: ‘origin’ shows the repo you are pushing data to (origin) (in a push operation).

A pull in Git is Fetch + Merge. So Pull is like a shortcut, and performs both operations at once.

So you are going to make your changes and push them to your remote fork (origin). But you will first make your changes and add them your local fork or branch, with

**git add .**

and

**commit -v** [ “comment”]

However, before you push up that content to your remote fork, to ultimately merge it with the Microsoft master repo, you will want to make sure that you don’t over-write any other changes made to that repo by anyone else, or by multiple people.

Therefore, before making any changes to your fork, you first need to make a copy of the Microsoft repo that Microsoft publishes from (upstream) **Microsoft**/win-cpub-itpro-docs, so that after you make your changes to your fork (origin), you will add those changes back to the Microsoft master repository.

To do that, you carry out the command **git remote add upstream** [Microsoft Remote upstream repository URL]

$ **cd** https://github.com/**Microsoft**/win-cpub-itpro-docs.git

So the **add** command actually copies down the **Microsoft** repository.

One might think that this upstream remote would get created at the time you create your fork, but it doesn't. Therefore, you have to add the upstream remote manually after you clone.

could confirm your new upstream remote is the MS repo you forked. What? Does **git remote add** **upstream** actually fork the Microsoft master repo as well?

$ **git remote -v**

**origin**  https://github.com/**jotob-msft**/win-cpub-itpro-docs.git (fetch) -- your remote fork

**origin** https://github.com/**jotob-msft**/win-cpub-itpro-docs.git (push) -- your remote fork

**upstream**  https://github.com/**Microsoft**/win-cpub-itpro-docs.git (fetch) -- your newly-added upstream remote

**upstream**  https://github.com/**Microsoft**/win-cpub-itpro-docs.git (push) -- your newly-added upstream remote

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (**master**)

So if you run a git remote -v directly after adding the upstream microsoft repo (master), you now see that remote repo in the output.

Next, you need to pull down the newly-added upstream **Microsoft** master repo.

**Microsoft/**win-cpub-itpro-docs

$ **git pull upstream master:working**

Unpacking objects: 100% (15/15), done.

From https://github.com/**Microsoft**/win-cpub-itpro-docs

\* [new branch] master -> working

\* [new branch] master -> upstream/master

This command copies/pulls down any changes made by others from the Microsoft upstream master

**Microsoft/**win-cpub-itpro-docs

into your local fork (master) so that you are seeing all the current most updated changes before you push your own changes up.

v-jotob@SEA-STT-JTobin MINGW64 /c/test-repo/wdg-cpub-test/wdg-cpub-test/jotob (master)

$ git pull upstream master:wdg-cpub-test

From https://github.com/Microsoft/wdg-cpub-test

\* [new branch] master -> wdg-cpub-test

\* [new branch] master -> upstream/master

**Already up-to-date.**

The output will tell you whether the Master repo is up to date.

This is the equivalent of syncing with the upstream master repo. This copies down an exact copy of the Microsoft master repo. It is a sync. This is pulling (fetch + merge) from the master branch of the upstream repo into the local branch you will have created, named **working**. If a branch called ‘working’ does not exist, it will be now be created.

Next, we switch to the local branch we had created (working) with the **git checkout** command.

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (master)

$ **git checkout working**

Switched to branch 'working'

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (working)

**Maybe we don’t need a git push origin working right here at this stage because we haven’t yet added any content.**

~~Next, you need to tell your remote fork (in Github, not locally) to create a corresponding new branch named working (which has heretofore only existed~~ *~~locally)~~* ~~and to push or copy my new local content up to the remote ‘working’ folder.~~

~~Yes, if you made changes AND ran git add . and git commit -m "comment" then the changes would push right.~~

$ **git push origin working Maybe not valid command placed right here.**

~~remote: Resolving deltas: 100% (12/12), completed with 6 local objects.~~

~~To https://github.com/~~**~~jotob-msft~~**~~/win-cpub-itpro-docs.git~~

~~\* [new branch] working -> working~~

Now we add some content or make our changes with a **git add .**

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (working)

$ **git add .**

Next, running the ‘**commit**’ command commits your changes to your local ***branch* (working)**

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (working)

$ **git commit -m "comment added new topics to March change history"**

[working 2bd8eef1] comment added new topics to March change history

Next we push our commit to origin (to our remote fork) or more specifically, to ‘working’, the remote branch of that fork, which iw a copy of the local branch we had initially created, also named ‘working’

jtob-msft/**working**

The remote branch keeps the same branch name in your remote fork as locally. You pushed that branch name to your fork when you first ran **git push origin branch**.

Now we create a pull request from the Github website.

**Microsoft**\cpub repo\master < **jtob-msft**\cpub repo\working

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (working)

$ **git push origin working**

Counting objects: 5, done.

Total 5 (delta 4), reused 0 (delta 0)

remote: Resolving deltas: 100% (4/4), completed with 4 local objects.

To https://github.com/jotob-msft/win-cpub-itpro-docs.git

f90e3c0f..2bd8eef1 working -> working

v-jotob@SEA-STT-JTobin MINGW64 /c/repo/win-cpub-itpro-docs (working)