Downloading from Git:

The most basic example is to download the entirety of the most recent revision of a single public repo. For this example, we'll be downloading the git repo:
https://github.com/synewmark-resources/resources
To do so, we must identify the username, and repo portion of the URL.

The username is the GitHub user the repo is under, in this case it will be "synewmark-resources", and the repo name is just "resources".
Finally, we'll also need the branch name. By default this is main. But you'll need to check.

Now let's enter all this into the command line:
First, we'll add –g to indicate we're performing a Git request, then -u for username and -b for branch. Leaving us with:

-g ' ' -u synewmark-resources -b main

Note that because -g expects an OAuth token it's necessary to follow it with a blank space. Most Terminal Emulators strip spaces so you must use the quotes to indicate it. On Windows the same can be accomplished with " "

Now for repo name, because this is a batch downloader, it instead accepts a path to a text file containing a series of new-line separated usernames. We only want the one, so let's create a file named Students.txt that contains just the line "resources" (see /TestResources in the git repo for examples)

-g ' ' -u synewmark-resources -b main -r "Students.txt"

And you'll get this:

```
root@LAPTOP-TUFIVNJG:~/JAR# java -jar AutoSubmit.jar -g ' ' -u synewmark-resources -b main -r "Students.txt"
Executing Git request...

Downloading files from: https://github.com/synewmark-resources/resources to directory: /tmp/MOSSAutoSubmit_GitFiles1943122713679892963

Git download completed you can find your files at this directory:
/tmp/MOSSAutoSubmit_GitFiles1943122713679892963
```

By default the Git download will go to a newly created temporary directory. The details of that directory are OS specific, but most will not automatically delete it when done, that must be done manually.

If you don't want to download the entire repo you can specify the subdirectory with the --subdirectory flag. In that case, only files nested beneath that directory will be downloaded. You can also pass in a text file with a list of files to download with the --files flag.
If both an --subdirectory and --files flag are set the files will be nested on top of the subdirectory specified. I.e. if --subdirectory is set to: "dir/dir2" and --files has a listing: "dir2/file1.java"

AutoSubmit will look for your file in the directory dir/dir2/dir2/file1.java. Therefore, be careful not to duplicate directories.

You can also specify where to download the files with the flag -d or --directory

If your repo is private you'll need to pass an OAuth token with valid read permissions to the -g flag:

-g gho_x -u synewmark-resources -b main -r "Students.txt"

Where **gho_x** your OAuth token

If you want to download a specific revision you can pass in an ISO 8061 timestamp with the flag --timestamp and the latest commit prior to that timestamp will be downloaded.

Uploading to MOSS:

MOSS uploads are done by uploading entire directories. This is convenient because it means we can upload the entire directory from a Git Repo. In fact, when grouping Moss and Git calls, this is exactly what happens. However, if we want to run MOSS as a standalone program, we can. Moss requires just 3 parameters, the language of the student code, the Student File Directory and your MOSS ID with the flags: -l, -sfd, and -id respectively.

For example:
-m 884640278 -l Java -sfd StudentCode

If we're linking a Moss upload with a Git download we can omit the -sfd flag leaving us with:

-g -u synewmark-resources -b main -r "Students.txt" -m 884640278 -l Java

```
shmuel@DESKTOP-45A9977:~/JAR$ java -jar AutoSubmit.jar --g -u synewmark-resources -b main -r "Students.txt" --m -l Java -id 884640278
Executing Git request...

Downloading files from: https://github.com/synewmark-resources/resources to directory: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869

Git download completed you can find your files at this directory:
/tmp/MOSSAutoSubmit_GitFiles4439403260426961869

Executing MOSS request...

Student File Directory not supplied. Using working directory: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869

uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student0/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student1/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student2/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student3/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student4/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student5/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student6/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student7/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student8/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/synewmark-resources/resources/studentCode/student9/HelloWorld.java

Local results saved to: /tmp/MOSSAutoSubmit_GitFiles4439403260426961869/MossRequestResults.htm

MOSS request completed
You can view your results here: http://moss.stanford.edu/results/4/1062293849858
```

The completion of the Moss request will leave you with a URL to your results that will only be saved for 7 days. Therefore AutoSubmit will also save the request locally to the Student File Directory. Take care submitting multiple requests with the same root directory as each will overwrite the locally saved file.

We can also manually specify the student directory with the -d flag
-m 884640278 -l Java -d StudentFiles/Assignment1
Will upload all the student files in the folder Assignment

Note: MOSS servers are frequently overloaded being an extremely popular and helpful free service. If the code hangs or throws a socketexception/connection refused while executing a Moss request it's usually due to the servers being overloaded.

Uploading to Codequiry:

Codequiry uploads require each individual student directory to be zipped to their own zip file. Because of that AutoSubmit must be more aware of the directory structure of the Student Files when Submitting to Codequiry than MOSS. Make sure the folder passed in to the -d flag only contains student files. Like with MOSS, if Codequiry is run after a Git call, it can automatically collect the list of student directories.

Aside from the directories of student files Codequiry also requires your Codequiry API key: -c, and the Language of the student files -l.

Let's imagine we have 10 students folders nested in a directory structure number from 0-9 as follows: StudentFiles/studentCode/student0…student9

To submit all of those students we must run:
-c db7ef9e417b2e292465e1bd9ffd2f93eea71279324614dbbb5be3a8f1c8f0733 -d StudentFiles/studentCode -l Java

```
shmuel@DESKTOP-45A9977:~/JAR$ java -jar AutoSubmit.jar --c -sd StudentFiles/studentCode -l Java -key db7ef9e417b2e292465e1bd9ffd2f93eea71279324614dbbb5be3a8f1c8f0733
Executing Codequiry request...


Codequiry check creation+upload completed
You can finish execute your request at the following URL: https://dashboard.codequiry.com/course/30771/assignment/57741/submission
```

Note that we passed "StudentFiles/studentCode" into the -d flag and not just "StudentFiles". This is because AutoSubmit requires the immediate parent of the student files/directories passed.

Unlike with MOSS, Codequiry does not actually execute the request or save the results. Following the link will bring you to a page to manually run the checks.

Warning: at this time, Codequiries treatment -d flag is very naive. It will treat every file and folder in the directory as an individual student. In our previous example if there was also a

metadata.txt file in the studentCode folder next to the student folders it would also treat "metadata" as a student. Make sure the directory only contains student files.

Like how when using Moss and Codequiry together with Git obviates the need to pass in an explicit directory, using Codequiry after MOSS makes it unnecessary to pass in an explicit language.

For example:
-g ‘ ’ -u synewmark-resources -b main -r "Students.txt" -m 884640278 -l Java -c db7ef9e417b2e292465e1bd9ffd2f93eea71279324614dbbb5be3a8f1c8f0733

```
root@LAPTOP-TUFIVNJG:~/JAR# java -jar AutoSubmit.jar -g ‘ ’ -u synewmark-resources -b main -r "Students.txt" -m 884640278  -l Java -c db7ef9e417b2e2
92465e1bd9ffd2f93eea71279324614dbbb5be3a8f1c8f0733
Executing Git request...

Downloading files from: https://github.com/synewmark-resources/resources to directory: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315

Git download completed you can find your files at this directory:
/tmp/MOSSAutoSubmit_GitFiles14105687083270402315

Executing MOSS request...

uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student1/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student7/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student0/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student9/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student2/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student3/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student6/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student4/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student5/HelloWorld.java
uploading file: /tmp/MOSSAutoSubmit_GitFiles14105687083270402315/resources/studentCode/student8/HelloWorld.java

Local results saved to: /tmp/MossRequestResults_1643656125.htm

MOSS request completed
You can view your results here: http://moss.stanford.edu/results/8/3169165103203

Executing Codequiry request...


Codequiry check creation+upload completed
You can finish execute your request at the following URL: https://dashboard.codequiry.com/course/30771/assignment/58125/submission
```

Will download from all the repos listed in Students.txt and pass the files into Moss and Codequiry.

General Flag Rules:

The order of the flags does not matter. The runtime order will always be 1. Git, 2. MOSS, 3. Codequiry.

-d, --directory is shared by Github, MOSS, and Codequiry. -l, --language is shared by MOSS and Codequiry

Running --help will print all command options and brief descriptions of each

All ‘mandatory flags’ have short options ‘-’ and long options “--”. Optional flags just have long options.