

# GETTING STARTED WITH GIT & UNITY

## STEP 1: CREATE A UNITY PROJECT

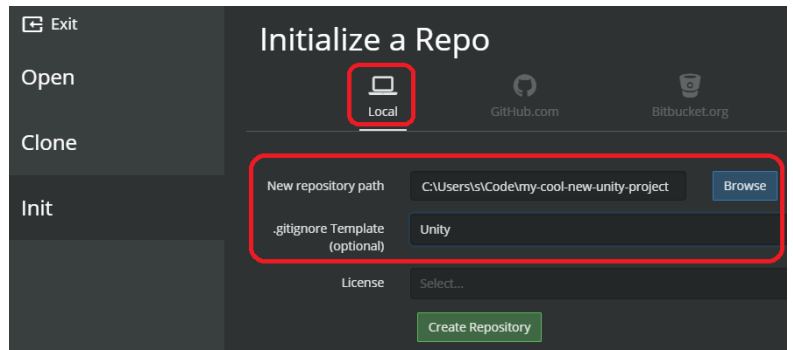
It doesn't need anything in it yet, but you should start by (at minimum) saving an empty scene. (Git doesn't keep track of folders unless they have at least one file in them, so this'll make sure your "Assets" folder gets added to Git and Unity won't complain if you share your project with someone else at this point.)

## STEP 2: GET GITKRAKEN

Go to [www.gitkraken.com](http://www.gitkraken.com) and download and install it! You need to register to use it but it's free (we won't be needing any premium features.) Let it walk you through the steps.

## STEP 3: SET UP YOUR LOCAL GIT REPO IN GITKRAKEN

Once you're set up in GitKraken, hit File > Init Repo (or click the folder icon in the top-left corner.) Set up a **Local** repo by browsing to the root of your Unity project (**not** the "Assets" folder, but the folder that contains "Assets", "ProjectSettings", etc.) Under ".gitignore template", pick "Unity".



(If you forget to pick a .gitignore template, that's okay! GitKraken itself grabs them from a freely-available online list of common ignore files. You can go to [www.github.com/github/gitignore](https://www.github.com/github/gitignore) to find the Unity one — just save it to the **root folder** of your repo.)

Now hit "Create Repository."

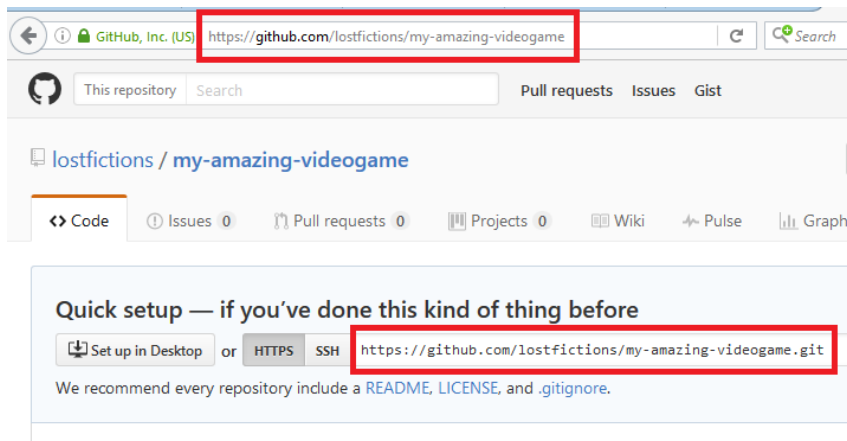
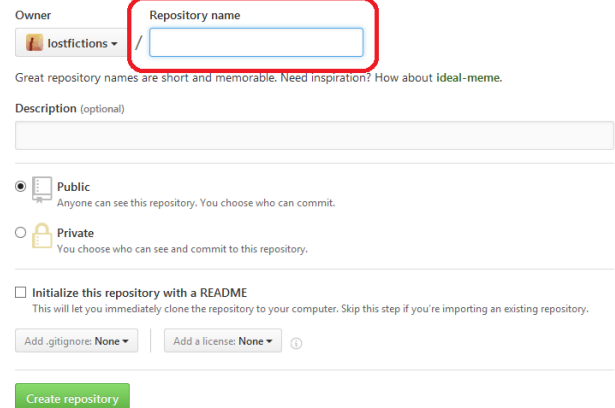
## STEP 4: MAKE A NEW REPOSITORY ON GITHUB.COM

Go to GitHub.com in your web browser, make sure you're logged in, hit the "+" in the top-right corner and pick "New Repository."

Give your project a useful name! Ignore the other options — the only other thing you need to do on this page is hit "Create repository."

### Create a new repository

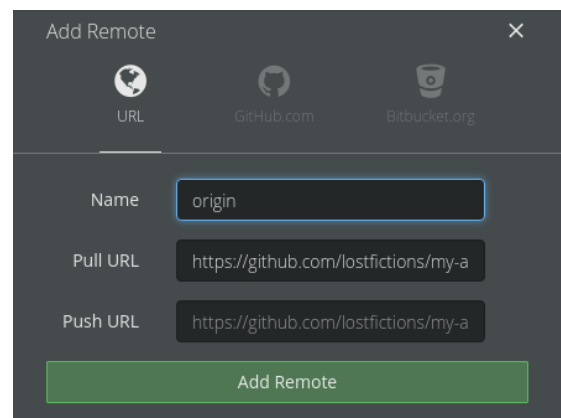
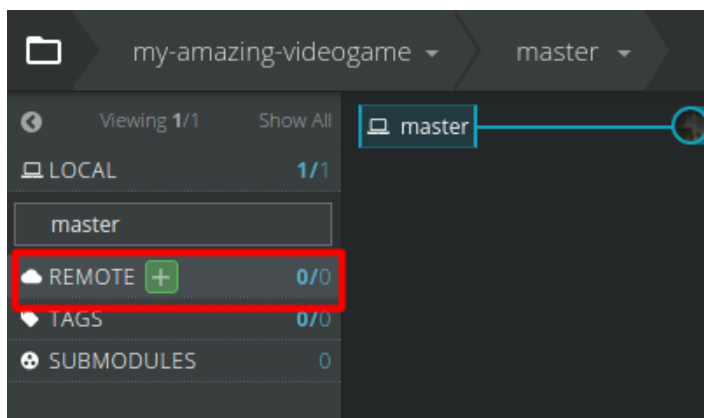
A repository contains all the files for your project, including the revision history.



On the next page, grab the URL of the page (you can also get the URL it gives you under "quick setup" — same thing.) Just make sure it starts with "https"!

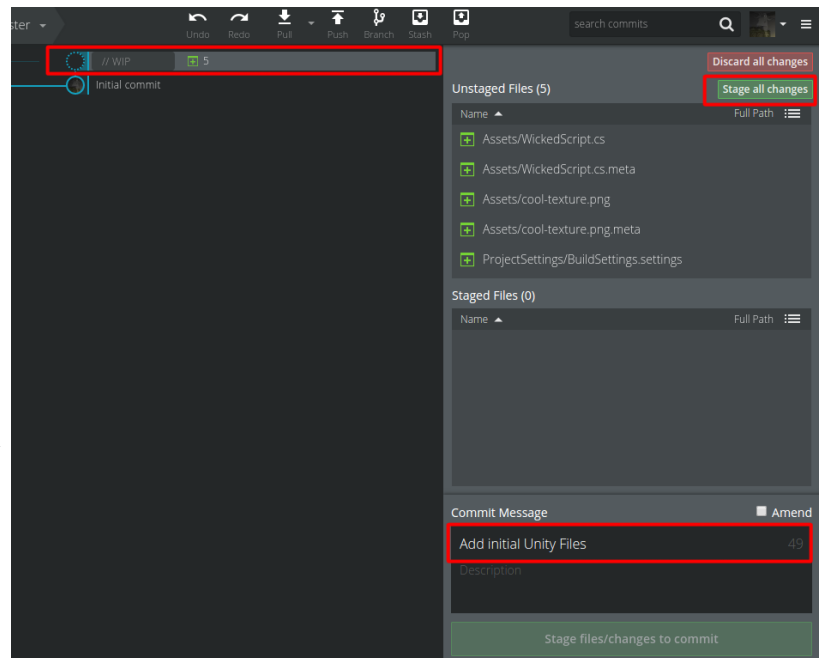
## STEP 5: ADD GITHUB REPO AS A REMOTE TO YOUR LOCAL REPO

Back in GitKraken, in the left-hand sidebar hover over "Remote" and hit the "+" icon to add a new remote. Paste your GitHub URL in the "Pull URL" field and the "Push URL" field should populate automatically. In the "Name" field you can call the remote whatever you like (eg. "github", "internet") but by convention default Git remotes are just called "origin", so let's go with that.



## STEP 6: COMMIT ALL YOUR CHANGES

Whenever you make changes to your project and want to take a snapshot, you'll **commit** them. We have some changes to commit already even in our empty Unity project, so let's make a first commit. Click the "WIP" entry at the top in GitKraken, and then choose the files you want to include in the commit by **staging** them. (For this first commit

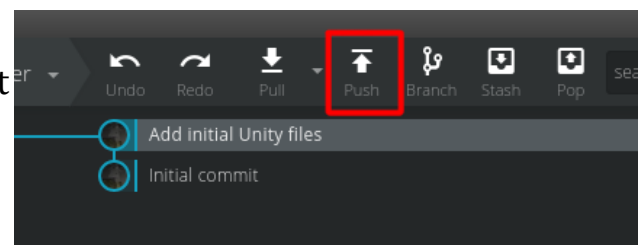


— and in most circumstances — you'll want to stage all your changes, which you can do with the green "Stage all changes" button at the top.)

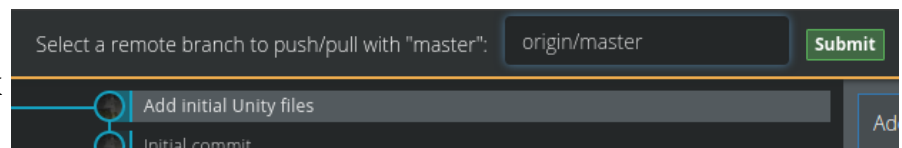
Now type a description of what you changed. For this first commit, something like "Add all Unity files" is perfect. You can leave the description blank. Now hit the big green "Commit" button at the bottom. Hooray!

## STEP 7: PUSH YOUR CHANGES TO GITHUB

Almost there! Hit "Push" in the toolbar at the top. The first time you do this it might ask you to login — use your GitHub user name and password.



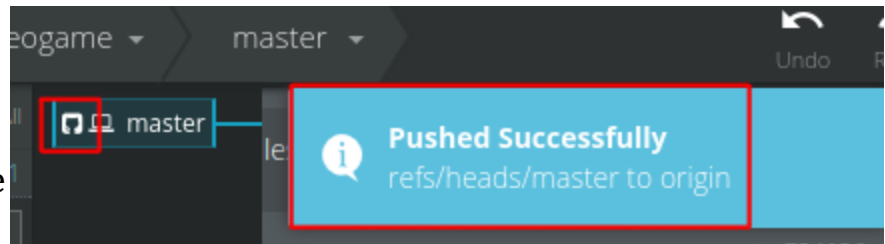
On your first push, GitKraken might also ask you a weird, needless,



confusing question: "Select a remote branch to push/pull with 'master!'" The correct answer to this is "origin/master" (or whatever name you gave to your

remote, eg. “github/master”.) GitKraken should never ask you this question again for this repo.

If all goes well, after a short time you should see a “Pushed Successfully”



message (and a little GitHub icon next to the “master” tag indicating that your remote repo is up-to-date with your local one.) Browse to your repo on github.com and check that your changes are visible there. Savour the sweet taste of Git success! Repeat steps 6 and 7 whenever you make new changes to your project to take new snapshots and push them to GitHub.

## GLOSSARY

**Version control system (VCS):** A program that tracks changes made to a set of files, making it easy to undo errors, collaborate with others, share code and make backups, find out who made which changes, etc.

**Git:** The most ubiquitous VCS, used by millions of people. Distributed as a command-line application with no graphical interface.

**GitKraken:** One of many pieces of software designed to provide a friendlier interface to Git.

**GitHub:** A company that runs a website that provides free hosting for open-source/public Git projects (with an option for paid hosting for private projects.) One of the most popular websites on the Internet, a big chunk of the world’s software is hosted there.

**Repo:** Short for “repository”. A folder that Git is keeping track of. It includes all your files, as well as the history of all the previous versions of the project. Can be “local” (on your computer) or “remote” (somewhere else, like on GitHub.com.)