Github Pages Tutorial

Prepared by Maksym Kornyev

Github Pages (GH Pages) is an awesome and free web hosting alternative if you'd like to host your own static[1] website. It is built on top of Github, which is a source-control platform that lets you host remote / cloud-based repositories[2] (ie. those on Max's github), and Git, which lets you keep a file revision history stashed locally on your computer and later publish it onto Github. This tutorial will cover the steps needed to deploy your static website in less than 20 min.

By the end of the tutorial you will…

- Have installed Git command line tools
- Have your own static Github domain
  - ie. https://kmornyev.github.io
- Have a really basic understanding of Git constructs

Pre-Requisites

Make Sure you have the Git command line interface installed. Check out the official Git installation instructions here (for Mac/Windows/Linux).

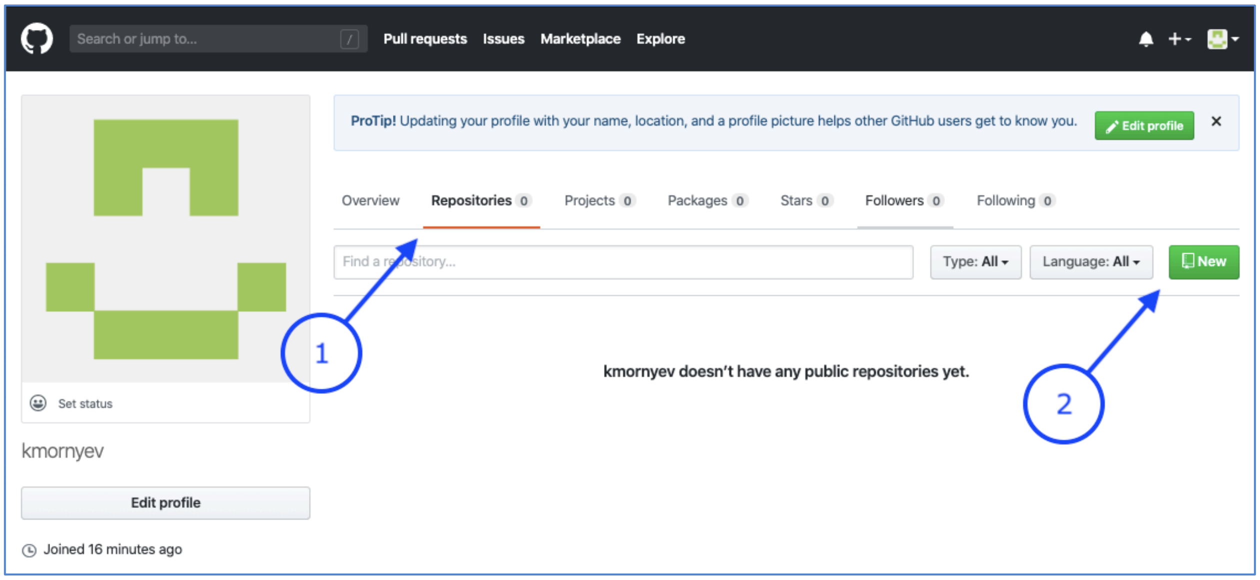
Steps

1. Come up with a clever username for your Github account. It *will* be part of the URL address of your GH Pages site: ie. *username.github.io*

2. Head over to Github.com and sign up for a Free account. (You can later get awesome perks with your andrew email account)

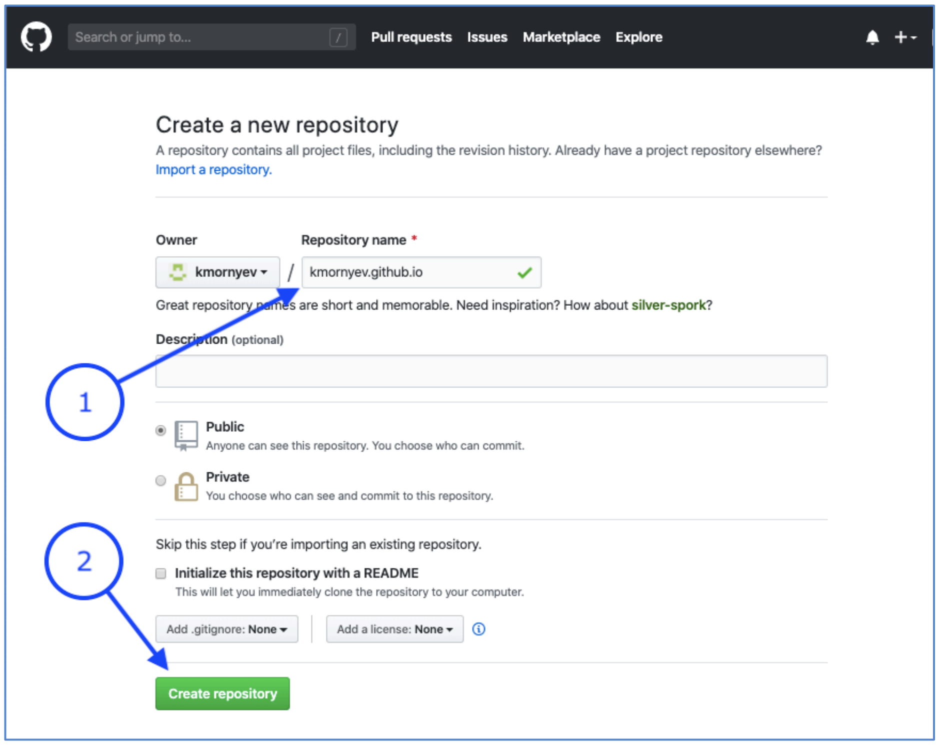
[1] A static website typically consists of just .html, .css, and .js files, among other website assets. While dynamic web applications require a dedicated server to perform additional processing: ie. request handling and server-side resource loading (think of web frameworks like AngularJS, React, or Ruby on Rails).
[2] For developers, a repository is just a collection of project assets. They are basically folders!

3. After you verify your email, you will be taken to the *Create a New Repository* page. If you aren't redirected, log into your Github and go to the *Repositories* tab. Then hit the *New* button as shown below.



kmornyev doesn't have any public repositories yet.

4. In the name field, enter your username followed by ".github.io" and hit *Create Repository*. The name format ***must*** match the following: **yourusername.github.io**



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: kmornyev / Repository name *: kmornyev.github.io

Great repository names are short and memorable. Need inspiration? How about silver-spork?

Description (optional)

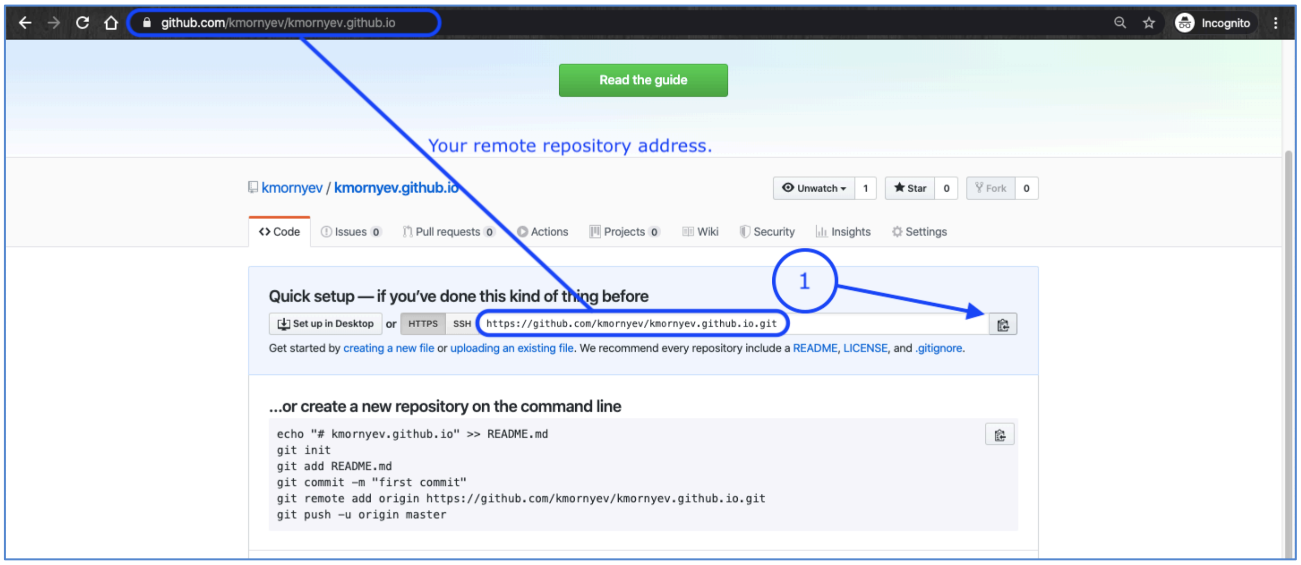
Public
Anyone can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer.

5. Save your repository address to your clipboard by either copying the URL, or clicking the *copy* button to the right of the *Quick setup* dialog.



Quick setup — if you've done this kind of thing before

https://github.com/kmornyev/kmornyev.github.io.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# kmornyev.github.io" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/kmornyev/kmornyev.github.io.git
git push -u origin master
```

At this point, you just created your own *remote* GH repo. The next few sections will focus on initializing a local git repo using the Git CLI (command line interface) which you installed as a pre-requisite, and "pushing" the files to your remote repo. Contact the TA's if you have any trouble with the command line.

6. Fire up your Terminal (MacOS) or command line equivalent and **cd** into an empty project directory. I just use my desktop: **cd ~/Desktop/**. (See the terminal output at the end of Step 7 c.)

7. Read the headings and type the bolded commands into your console before pressing enter. If you get confused, check the sample terminal output at the end of step *c.* and step *i.*:

    a. Using the repo address from Step 5, clone your remote repo to your local directory. It will appear as a new folder in your current directory.

git clone https://github.com/kmornyev/kmornyev.github.io.git

Side Note:

Typically, you would create a *new* git repository in an empty directory (using **git init**) and add your repo address as a *remote* (**git remote add origin https://github.com/usr/repo**) before adding your project files (**git add .**), committing (**git commit -m "my msg"**), and pushing (**git push**) to the remote.

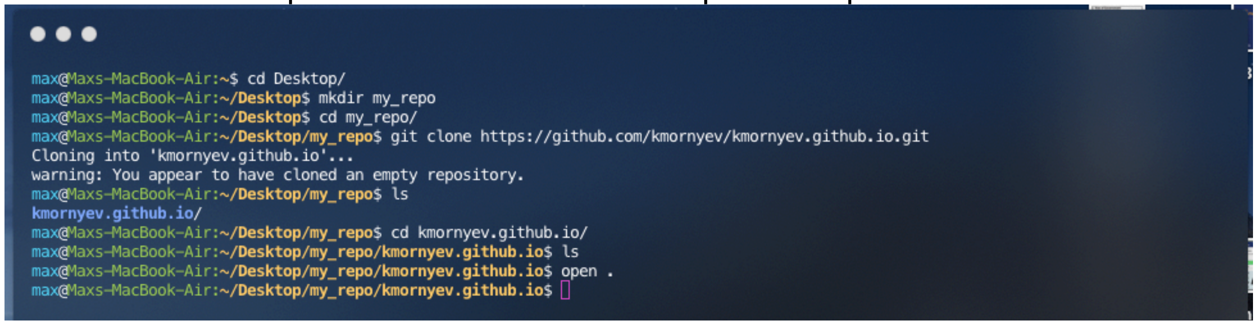
b. **cd** into the newly cloned repository (it should have the same name as your remote repo)

cd kmornyev.github.io/

c. Drag all of your project files into this new directory. You can open a new Finder/File Explorer window here by typing **open .** (MacOS) or **start .** (Windows), then add all of your project files.

** Your project *must* have an ***index.html*** file. This will be the default homepage of your GH pages domain **

Your terminal output should look like this up to this point:



```
max@Maxs-MacBook-Air:~$ cd Desktop/
max@Maxs-MacBook-Air:~/Desktop$ mkdir my_repo
max@Maxs-MacBook-Air:~/Desktop$ cd my_repo/
max@Maxs-MacBook-Air:~/Desktop/my_repo$ git clone https://github.com/kmornyev/kmornyev.github.io.git
Cloning into 'kmornyev.github.io'...
warning: You appear to have cloned an empty repository.
max@Maxs-MacBook-Air:~/Desktop/my_repo$ ls
kmornyev.github.io/
max@Maxs-MacBook-Air:~/Desktop/my_repo$ cd kmornyev.github.io/
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ ls
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ open .
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ 
```

d. Use **ls** (MacOS) or **dir** (Windows) to print all of the files in your current directory, and verify that all your site assets are in place.

e. Add your files to git. With this command, you add all your files to git's staging area. Staged files are files that will make up commits.

→ You can always check the status of git before and after any of the following commands with **git status**.

git add . → the trailing period is important here

f. Create a commit with all of the files you just added. A commit is just a series of relevant changes that a developer has made.

git commit -m "[add a creative commit message here]"

→ If git gives you the error "fatal: The current branch has no upstream," don't be afraid. Just execute the prompted command that git gives you (ie. **git branch --unset-upstream**).

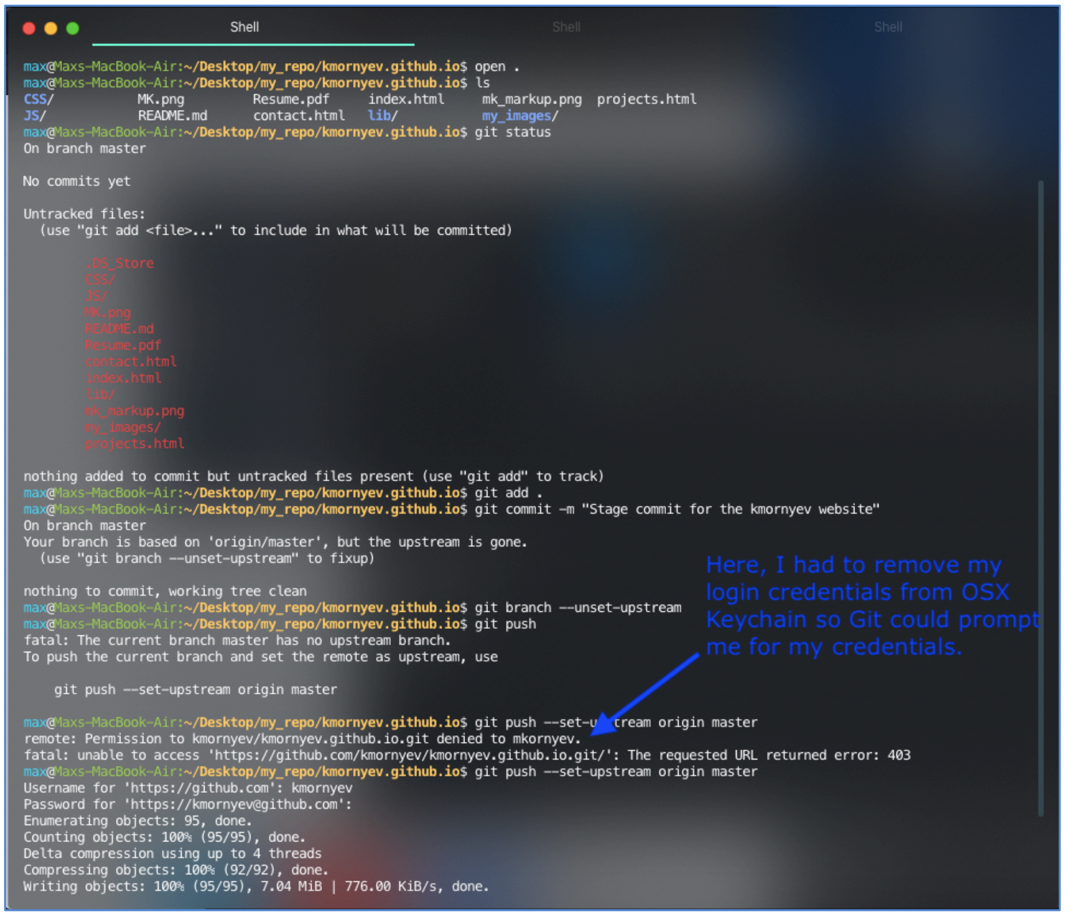
g. Now, push the changes to your remote: **git push**

   → If git gives you the error "fatal: The current branch has no upstream," do the same. (ie. execute **git push --set-upstream origin master**)

h. Now, git may ask you to log in. Use the same login credentials that you created your Github with (from Step 2).

   → If you are already logged in with another Github account locally, you may have to [change the global user credentials]() to be able to push to your new repo.

i. Congrats! Your remote repository should now be set up. Head to your website's URL, which will look like *https://**your-username**.github.io* (this is *not* your remote repo URL), and check the status of your website. Typically, Github indexes your website instantaneously, but changes to your repository may take several hours to propagate across Github's servers.
   → Read on to find out how to update your website.



```
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ open .
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ ls
CSS/          MK.png        Resume.pdf     index.html     mk_markup.png  projects.html
JS/           README.md     contact.html   lib/           my_images/
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .DS_Store
        CSS/
        JS/
        MK.png
        README.md
        Resume.pdf
        contact.html
        index.html
        lib/
        mk_markup.png
        my_images/
        projects.html

nothing added to commit but untracked files present (use "git add" to track)
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ git add .
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ git commit -m "Stage commit for the kmornyev website"
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ git branch --unset-upstream
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ git push
fatal: The current branch master has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin master

max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ git push --set-upstream origin master
remote: Permission to kmornyev/kmornyev.github.io.git denied to mkornyev.
fatal: unable to access 'https://github.com/kmornyev/kmornyev.github.io.git/': The requested URL returned error: 403
max@Maxs-MacBook-Air:~/Desktop/my_repo/kmornyev.github.io$ git push --set-upstream origin master
Username for 'https://github.com': kmornyev
Password for 'https://kmornyev@github.com':
Enumerating objects: 95, done.
Counting objects: 100% (95/95), done.
Delta compression using up to 4 threads
Compressing objects: 100% (92/92), done.
Writing objects: 100% (95/95), 7.04 MiB | 776.00 KiB/s, done.
```

Here, I had to remove my login credentials from OSX Keychain so Git could prompt me for my credentials.

Site Maintenance

1. You may continue to develop your website from that same project directory, and use a combination of substeps from Step 7 to make changes. In totality, you will…

    a. Make changes locally
    b. Stage your changed files to be committed: **git add .**
    c. Commit the changes: **git commit -m "…"**
    d. And push them online: **git push**

2. You can also delete your local repository (but only after pushing all changes to your remote), and later **git pull** the remote to your local machine.

    Git is a really powerful tool that makes open-source and parallel development a breeze, regardless of whether it is in the context of websites or other software. Read up on the git documentation to find out more:

    Open Source Git website: https://git-scm.com/
    Official Github website: https://github.com/
    Official Github git guides: https://guides.github.com/activities/hello-world/