Live D3 Visuals using GitHub Pages

The simplest way to see your D3 visuals on the web is by using GitHub Pages feature available for FREE on GitHub.com. Just follow these simple steps to create your website with your D3 Visuals. *GitHub* is a website where you can publish your code for public download and possible collaboration.

Method 1: *without the use of Git repository*
This is the simplest method and does not require using or creating a local Git directory.
In this case, you create all the code locally, say using brackets, and store it in a local directory *RegionalPopulation*.

1. Create a GitHub account on https://github.com and confirm your login information.
2. On github, create a **new repository by clicking the plus icon** that is next your username on the top right hand corner. If a drop down menu appears, then click on "**New Repository**".
3. Give your repository a name (i.e. *RegionalPopulation*) and **make it a public repository**.
4. Add a description.
5. Initialize this repository with (or you can do this later):
   README.md (to explain your repository to viewers in github.com)
   If you push some files to github that you want github to ignore, list them inside .gitignore (for now, we assume that you will not need to ignore any file).

6. Go to Upload files to add your files. Drag and drop the files you want to be published. One of these files must be named index.html. This is the file that will be invoked with your visualization link**.**
7. Once you have uploaded the all the required files, **click on "settings",** which is located right above the HTTPS clone URL. Under the section GitHub Pages, change the source to master branch. Click Save.
8. Now you can go to https://<username>.github.io/<reponame>/ to check your visualization.

Method 2: http://www.instructables.com.id/introduction-to-Github
Git is a source code version control system, which a series of "commits" or snapshots of your code that resides on your laptop/desktop. You make the commits manually.

The website describes 10 simple steps which are summarized below.
On Github

1. Create a GitHub account on https://github.com and confirm your login information.
2. On github, create a **new repository by clicking the plus icon** that is next your username on the top right hand corner. If a drop down menu appears, then click on "**New Repository**".
3. Give your repository a name (i.e. *RegionalPopulation*) and **make it a public repository**.
4. Add a description.
5. Initialize this repository with (or you can do this later):
   README.md (to explain your repository to viewers in github.com)
   If you push some files to github that you want github to ignore, list them inside .gitignore (for now, we assume that you will not need to ignore any file).

On your laptop,

6. Install Git on your laptop (or desktop) following the commands at https://git-scm.com/book/en/v2/Getting-Started-Installing-Git.
7. Start the laptop terminal. Change Directory to go to the directory where you have your work. You can use "cd" command such as cd *RegionalPopulation* (or specify the full pathname) to enter this directory.
8. Type *Git init*
   This method starts by creating a local git directory. This command creates hidden files that Git uses to manage source code control
9. Although these files are inside the directory, they have not yet been added to git. Add files to git by git add filename or git add . (dot) to add all the files in the directory)

   To communicate with the outside world, Git uses *remotes*. These are repositories other than the one on your local disk which you can push your changes into (so that other people can see them) or pull from (so that you can get changes from others). This is

essentially how this system works for Git and GitHub. Github directory is the *origin* and the local Git directory is the *master*.

10. Create a local commit by git commit -m "whatever_message" (typically “whatever_message” is “First Commit”).
11. Push the commit remotely to Github by git push origin master. (Think of this command as git push to origin from master. Github will ask you to authenticate yourself by asking your username and password before allowing you to clone the directory.
12. Go to settings page of your git repository in github.com. Go to https://github.com/<username>/<reponame>/settings. Under the section GitHub Pages, change the source to master branch. Click Save.
13. Now you can go to https://<username>.github.io/<reponame>/ to check your visualization.

Method 3: This method is the reverse of the above method where you clone a github directory on your local Git repository. This method is ideally suited when you have a github repository that you want to download locally. In our example, we will start with an essentially empty github directory. Cloning will not be permitted in a directory where you have already created the code. So, you will have to create a Git repository. Let us call this repository *FinishedRegionalPopulation*.

For the purpose of this method, you will have three working areas. The first working area is Github (on the web). Second working area is your local directory *RegionalPopulation* where you have created the necessary files for transfer. You will create a third working area on your laptop that will be referred to as *Git area FinishedRegionalPopulation*. There are three steps needed to push and publish your results using Github.

As a first step, we assume that you have created a directory on your laptop where you have the necessary files that you want to push to github for publication. We will refer to this directory as *RegionalPopulation*. We assume that you have created files such as index.html, index.css, index.js and other files as you see fit to complete your visualization task inside this directory.

As a second step you will move the files from your *RegionalPopulation* to be published to your local Git directory *FinishedRegionalPopulation*. As a final step, you will push these files to Github.

On Github

1. Create a GitHub account on https://github.com and confirm your login information.
2. On github, create a **new repository by clicking the plus icon** that is next your username on the top right hand corner. If a drop down menu appears, then click on “**New Repository**”.
3. Give your repository a name (i.e. *RegionalPopulation*) and **make it a public repository**.
4. Add a description.
5. Initialize this repository with (or you can do this later):
   README.md (to explain your repository to viewers in github.com)
   If you push some files to github that you want github to ignore, list them inside .gitignore (for now, we assume that you will not need to ignore any file).

On your laptop,

5. Install Git on your laptop (or desktop) following the commands at https://git-scm.com/book/en/v2/Getting-Started-Installing-Git.
6. On your laptop terminal type git clone https://github.com/<username>/*FinishedRegionalPopulation*.git to clone the github repository to your laptop (or specify the full path wherever you want this Git directory to be created). This command creates hidden files that Git uses to manage source code control. Github will ask you to authenticate yourself by asking your username and password before allowing you to clone the directory. Alternatively, you can authenticate yourself once and for all using
   git config -- global user.name “username”
   git config – global user.email your_email@ucsc.edu
7. Change Directory to go to the directory where you have your work. You can use “cd” command such as cd *FinishedRegionalPopulation* to enter this directory.
8. Move the files from *RegionalPopulation* to *FinishedRegionalPopulation* Directory.

9. Although these files are inside the directory, they have not yet been added to git. Add files to git by git add filename or git add . (dot) to add all the files in the directory).
10.Create a local commit by git commit -m "whatever_message" (typically “whatever_message” is “First Commit”).
11. Push the commit remotely to Github by git push origin master. (Think of this command as git push to origin from master. Github will again ask you to authenticate yourself by asking your username and password before allowing you to clone the directory.
12. Go to settings page of your git repository in github.com. Go to https://github.com/<username>/<reponame>/settings. Under the section GitHub Pages, change the source to master branch. Click Save[1].
13. Now you can go to https://<username>.github.io/<reponame>/ to check your visualization.

[1] Note that the default branch is master branch. Generally the repository is non-web repository. In those cases the web application of the repo is separate than the code. Therefore in such scenarios, you may either add a new branch (gh-pages) and keep committing the web application relevant code in that branch and change the source to that branch, or you may add a docs directory in the root repo directory in master branch whereby you may keep website relevant files and change the source to that docs folder in master branch. However in our case this would not be needed since our code itself is web application.