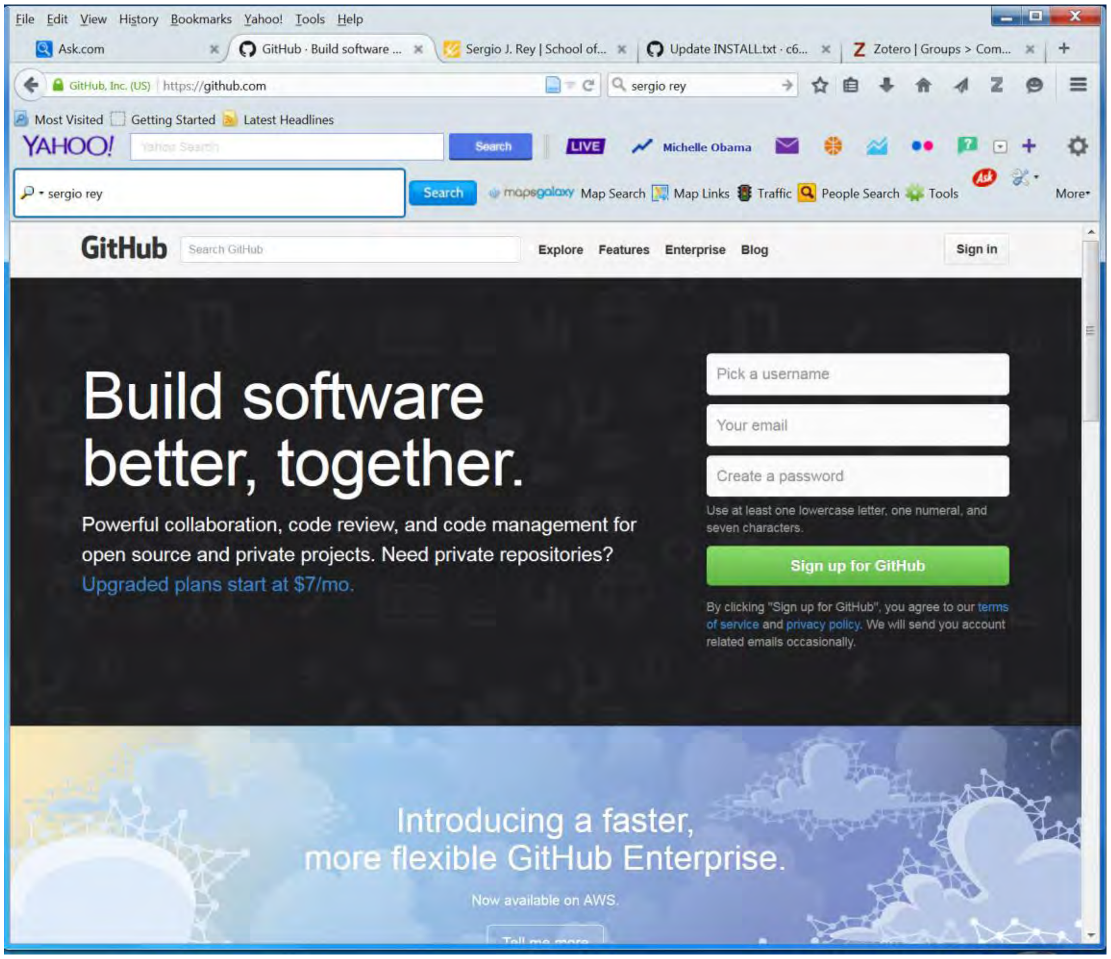
Tips on how to set up a GitHub account:

1. Go to the website https://github.com/, you will see the following page:


File Edit View History Bookmarks Yahoo! Tools Help
Ask.com
GitHub · Build software ...
Sergio J. Rey | School of...
Update INSTALL.txt · c6...
Zotero | Groups > Com...
GitHub, Inc. (US) https://github.com
sergio rey
Most Visited Getting Started Latest Headlines
YAHOO! Search LIVE Michelle Obama
sergio rey Search mapsgalaxy Map Search Map Links Traffic People Search Tools More
GitHub Search GitHub Explore Features Enterprise Blog Sign in
Pick a username
Your email
Create a password
Sign up for GitHub
Introducing a faster, more flexible GitHub Enterprise.
Now available on AWS.
Tell me more

Build software better, together.

Powerful collaboration, code review, and code management for open source and private projects. Need private repositories? Upgraded plans start at $7/mo.

Use at least one lowercase letter, one numeral, and seven characters.

By clicking "Sign up for GitHub", you agree to our terms of service and privacy policy. We will send you account related emails occasionally.

Figure 1: The GitHub main webpage (before you create an account and sign in)

Then choose your username, enter your email address, and create a password. Then click the green button Sign up for GitHub.

2. Choose your GitHub plan.

   Once you complete Step 1, you will be prompted to choose a plan. Below is an extract from the GitHub website that may help you make the decision:

   "GitHub provides two types of plans: free plans and paid plans.

   Both plans have the exact same features. They can have any number of public repositories, with unlimited collaborators.

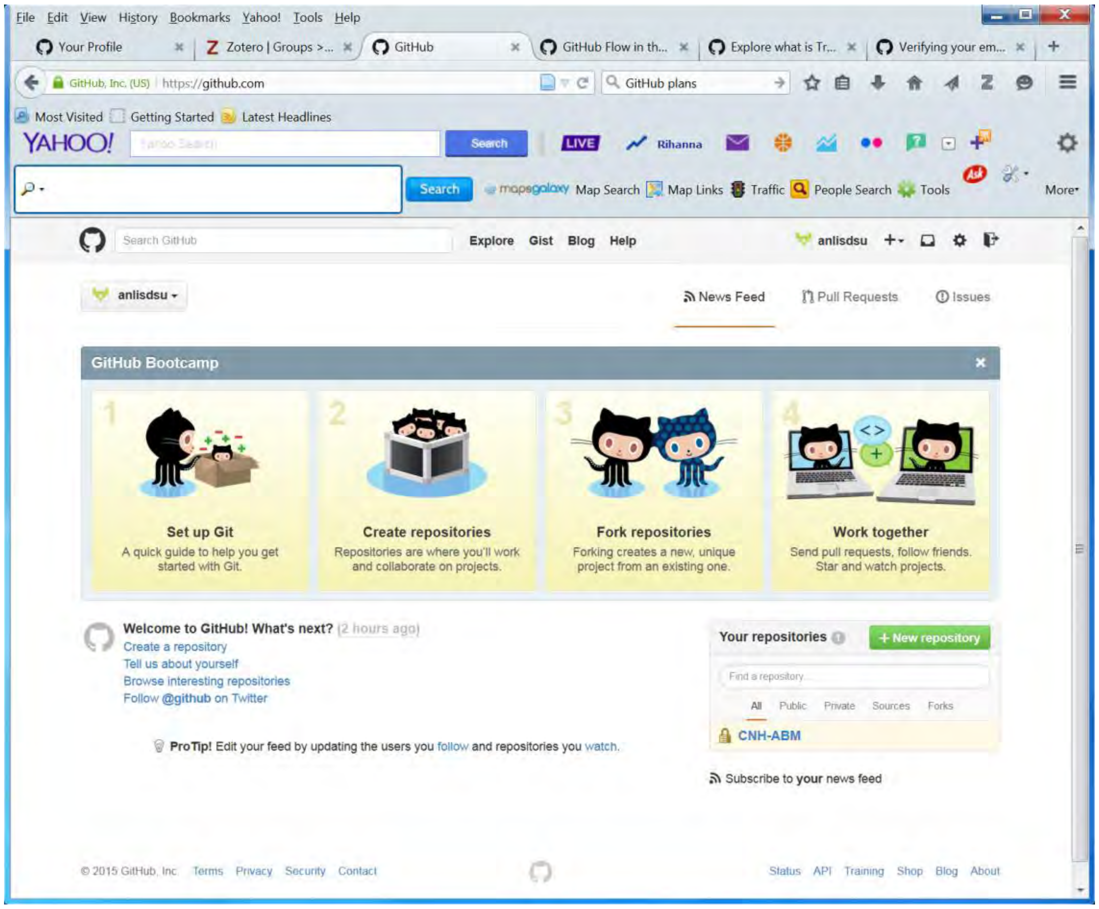
However, paid accounts can create private repositories. Public repositories are viewable by anyone, while private repositories have their visibility limited to just you and your collaborators.

The number of private repositories available is determined by specific paid plans. You can see all of our plans and pricing at github.com/plans. You can upgrade or downgrade your plan at any time."

Therefore for beginners I recommend you choose a Free plan. Later you can upgrade to any plan you deem appropriate.

3. Start use of GitHub.

   You can go back to the GitHub website (note to use GitHub, all you need is a web browser) at https://github.com/. Click the Sign in Button on the upper-right corner of the page (when prompted, enter your email and password). Then you will be able to access the GitHub main page (Figure 2).



A quick guide to help you get started with Git.

Repositories are where you'll work and collaborate on projects.

Forking creates a new, unique project from an existing one.

Send pull requests, follow friends. Star and watch projects.

Welcome to GitHub! What's next? (2 hours ago)
Create a repository
Tell us about yourself
Browse interesting repositories
Follow @github on Twitter

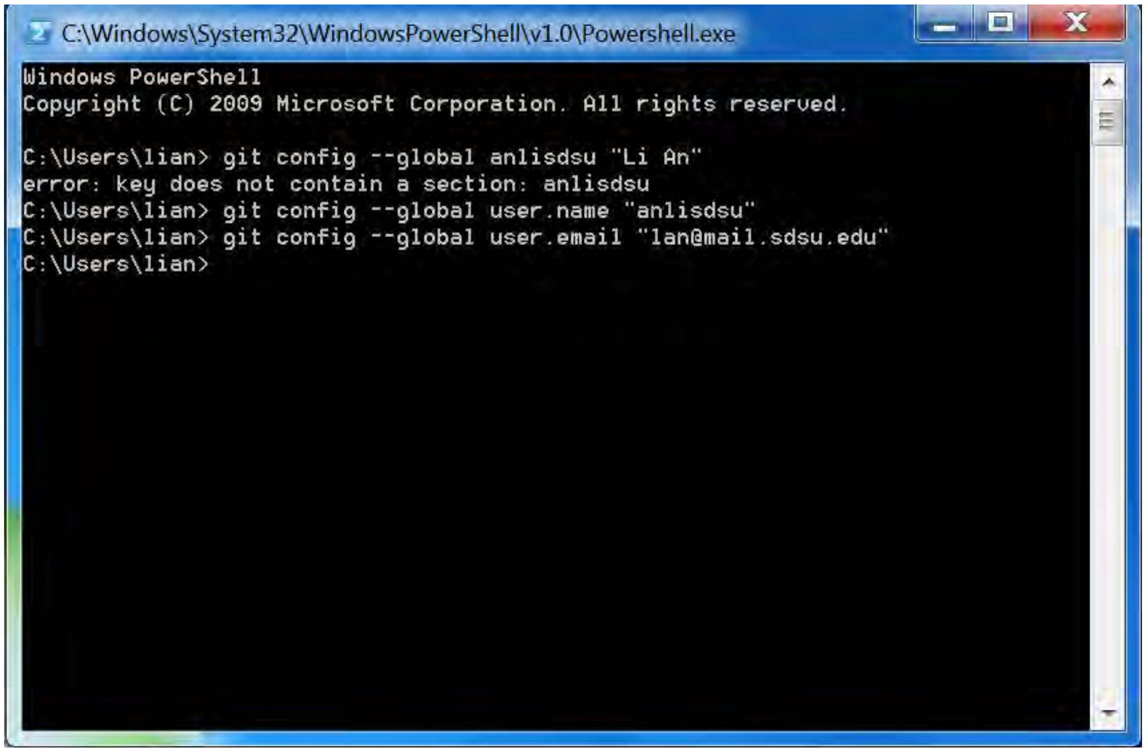
ProTip! Edit your feed by updating the users you follow and repositories you watch.

Figure 2: The GitHub webpage (after sign-in)

Before we start using GitHub, I recommend you click on the four steps on the above page, i.e., 1. Set up Git, 2. Create repositories, 3. Fork repositories, and 4. Work together. Then read these instructions and understand basic concepts such as repository (a place to store, save, edit, and retrieve your code and work with other people) and fork repositories (a fork is a copy of a repository for proposing changes to someone else's project).

3.1. Sign up Git.

Click the Set up Git page (see Figure 2), you will get to a page with four steps. Simply follow the instructions for installing Git, open Git Shell, and telling the system your information.



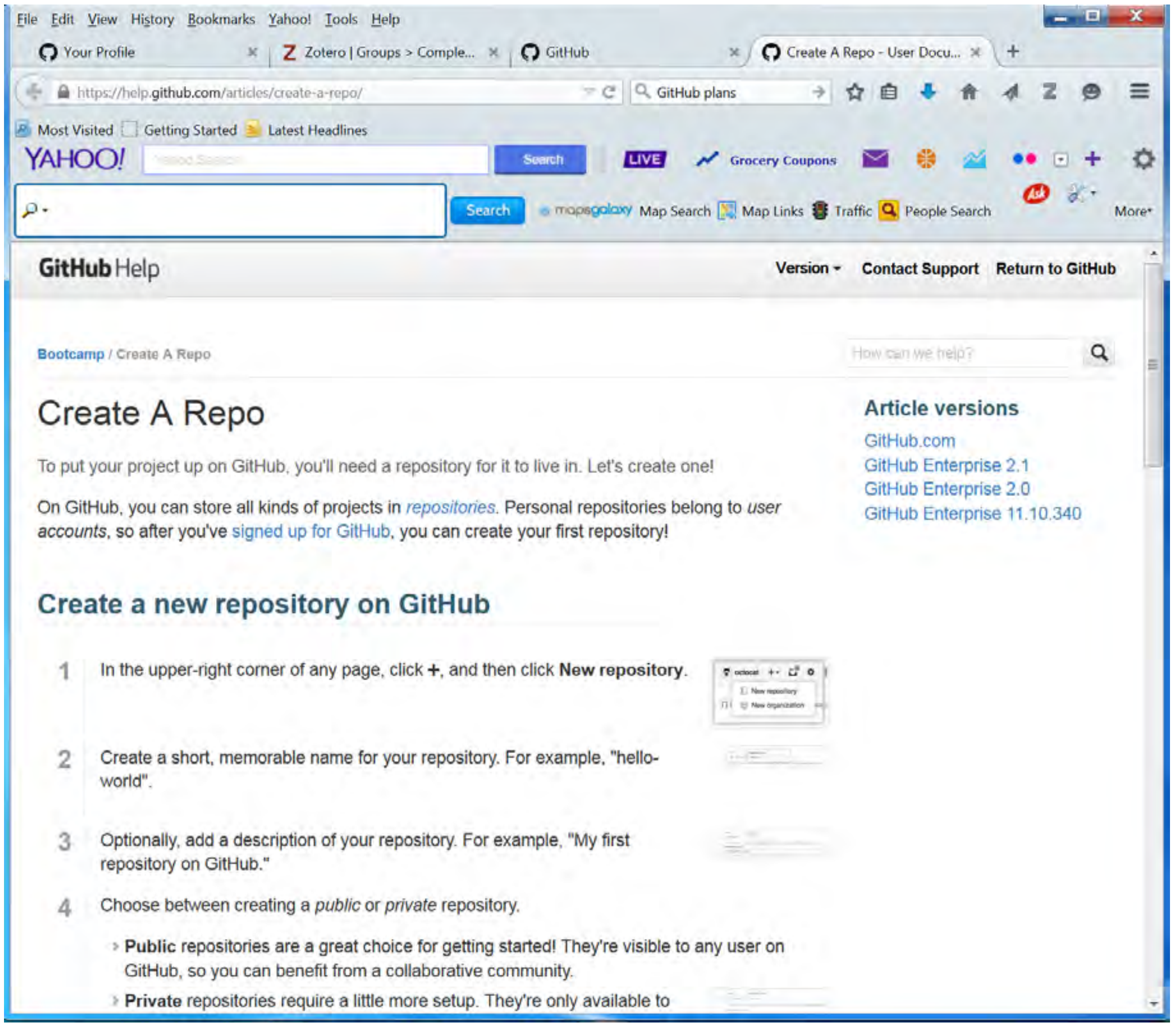
```
C:\Windows\System32\WindowsPowerShell\v1.0\Powershell.exe
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

C:\Users\lian> git config --global anlisdsu "Li An"
error: key does not contain a section: anlisdsu
C:\Users\lian> git config --global user.name "anlisdsu"
C:\Users\lian> git config --global user.email "lan@mail.sdsu.edu"
C:\Users\lian>
```

Figure 3: The Git Shell window after installation of GitHub for Windows.

Once the installation is done, two icons (one for Git Shell and one for GutHub) will be placed on your desktop (also a log file icon is placed, but I moved it to a folder inside). The above graph (Figure 3) shows Steps 3 and 4 after installation of GitHub for Windows. Note the GitHub username and email address should be in quotation marks.

3.2. Create repositories.

Click the button for 2. Create repositories (Figure 2), and you will see a page for instructions like the page below (Figure 4):



Bootcamp / Create A Repo

Create A Repo

To put your project up on GitHub, you'll need a repository for it to live in. Let's create one!

On GitHub, you can store all kinds of projects in *repositories*. Personal repositories belong to *user accounts*, so after you've signed up for GitHub, you can create your first repository!

Article versions
GitHub.com
GitHub Enterprise 2.1
GitHub Enterprise 2.0
GitHub Enterprise 11.10.340

Create a new repository on GitHub

1 In the upper-right corner of any page, click **+**, and then click **New repository**.

2 Create a short, memorable name for your repository. For example, "hello-world".

3 Optionally, add a description of your repository. For example, "My first repository on GitHub."

4 Choose between creating a *public* or *private* repository.

- **Public** repositories are a great choice for getting started! They're visible to any user on GitHub, so you can benefit from a collaborative community.
- **Private** repositories require a little more setup. They're only available to

Figure 4: Webpage for instructions on how to create a repository.

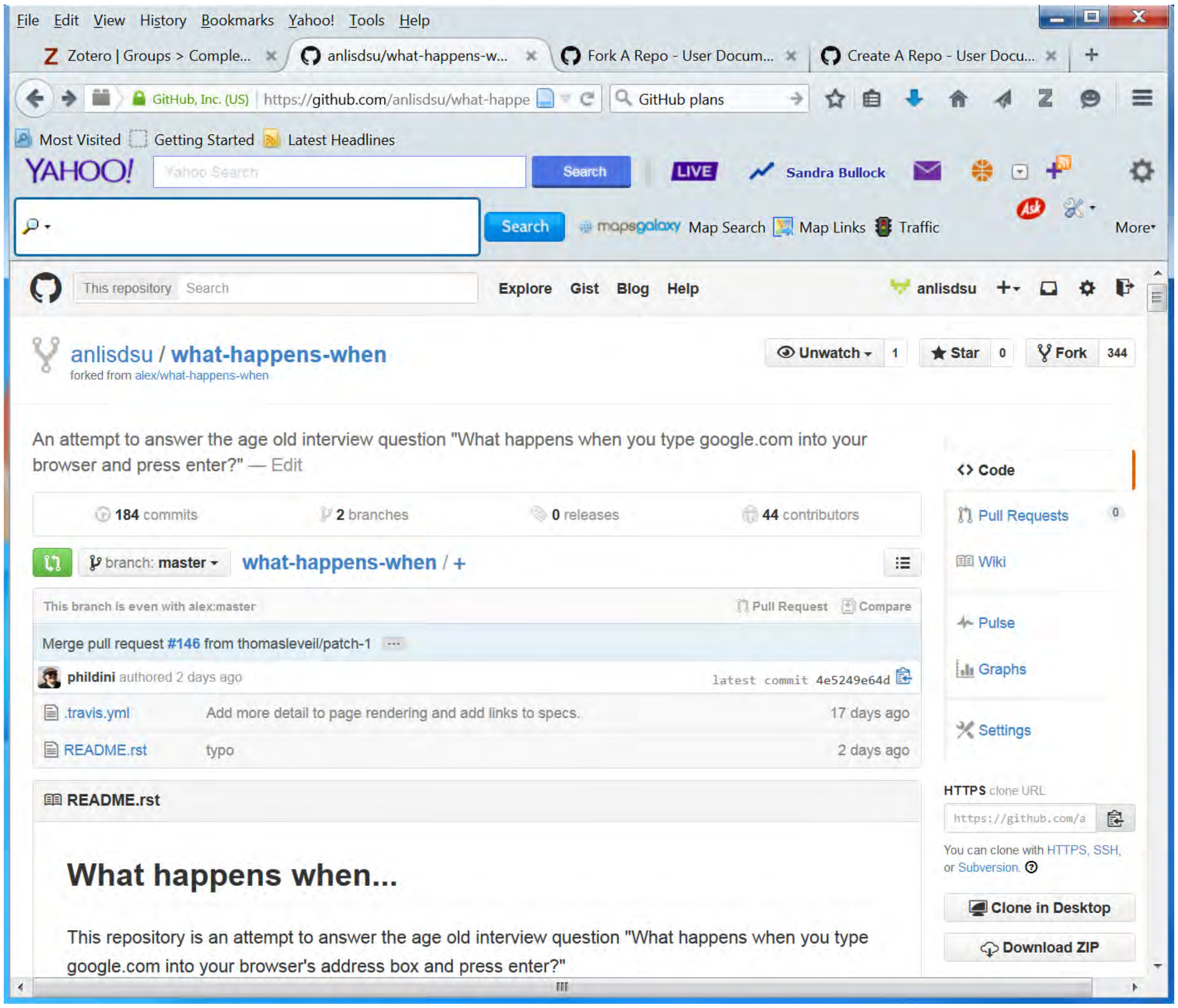
But be aware that the instructions are given on the basis on the main webpage https://github.com/ (the same as Figure 2). Follow the instructions on this page. Here I created a repository named **CNH-ABM**, which you can see from near the lower-right bottom of Figure 2. This was created to be private (only shared with people within the group); let us create one that is public, which is named **ABM-Modules**.

The next thing to do during this step is to commit a change. Automatically, GitHub creates a README.md file. Click on this file, and write something on the box. Then scroll down to the end of the page, click the green button “Commit changes”. Then all we have written will be saved to the file README.md. There are other types of files you can change and commit.

3.3. Fork repositories.

The idea of fork repositories is to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea. Therefore for practice purpose, let us go to the main page and click on Browse interesting repositories (near the lower-left corner). Among the many repositories, let us hit alex/what-happens-when as an example.

In the top-right corner of the page under alex/what-happens-when, click Fork. Then a pop-up window shows the progress of downloading the repo and everything inside it. When this process is done, you will see a page like below (Figure 5):



An attempt to answer the age old interview question "What happens when you type google.com into your browser and press enter?" — Edit

This branch is even with alex:master

Merge pull request #146 from thomasleveil/patch-1 ...

phildini authored 2 days ago — latest commit 4e5249e64d

.travis.yml	Add more detail to page rendering and add links to specs.	17 days ago
README.rst	typo	2 days ago

You can clone with HTTPS, SSH, or Subversion.

README.rst

What happens when...

This repository is an attempt to answer the age old interview question "What happens when you type google.com into your browser's address box and press enter?"

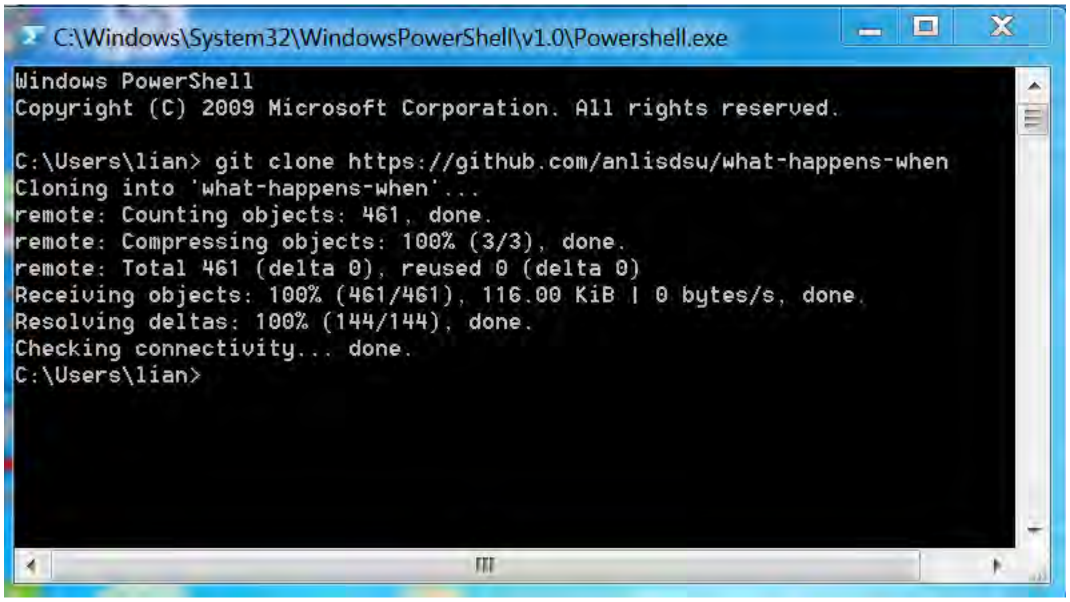
Figure 5: Outcome of a fork opertation: Another person's repository downloaded to my repository

From here we can open a file (such as README.md file), click the pen icon (for editing purpose), add a few words, and then hit the green button "Commit changes".
So far we have a fork of the alex/what-happens-when repository, but we do not have the files in that repository on our local computer. Below we clone a version to our local computer following the steps below:

a. On GitHub, navigate to your fork of the alex/what-happens-when repository.
b. In the right sidebar of your fork's repository page, click to copy the clone URL for your fork.
c. Open the command line (i.e., double click the icon Git Shell that is often on your Desktop for Windows users).
d. Type the following command to clone:

   git clone https://github.com/anlisdsu/what-happens-when

   Then you will see the following progress (Figure 6):



C:\Windows\System32\WindowsPowerShell\v1.0\Powershell.exe

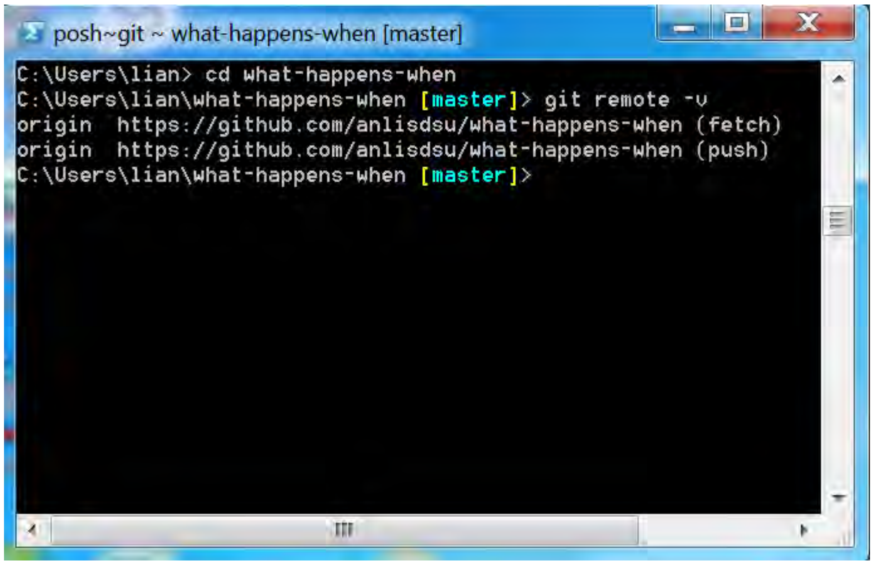
```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

C:\Users\lian> git clone https://github.com/anlisdsu/what-happens-when
Cloning into 'what-happens-when'...
remote: Counting objects: 461, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 461 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (461/461), 116.00 KiB | 0 bytes/s, done.
Resolving deltas: 100% (144/144), done.
Checking connectivity... done.
C:\Users\lian>
```

Figure 6: The clone process (downloading a repository to a local computer)

After doing this, let us verify it by going to C:\Users\lian at my computer: Yes, there is a folder what-happens-when there, which contains two files: one is .travis.yml, and the other is the readme file.

e. Next sync our fork with the original what-happens-when repository:

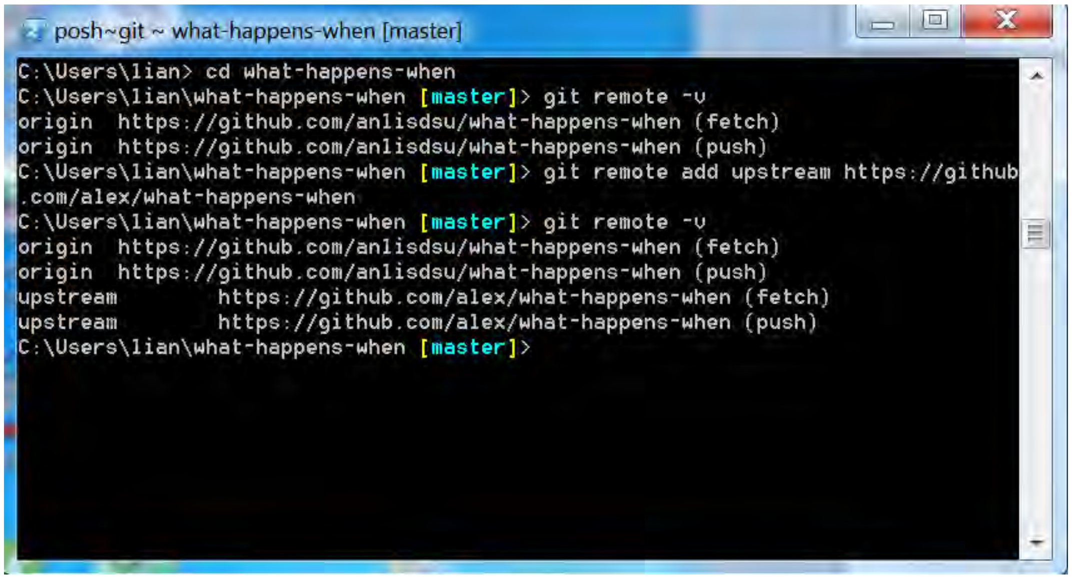
[Same as Steps a-c above: On GitHub, navigate to our fork of what-happens-when repository. Then on the right sidebar of this fork's repository page, click to copy the clone URL for your fork. Open the the command line]. Go to the directory for what-happens-when (now C:\Users\lian\what-happens-when), and type the command: git remove –v, you will see the following:



```
C:\Users\lian> cd what-happens-when
C:\Users\lian\what-happens-when [master]> git remote -v
origin  https://github.com/anlisdsu/what-happens-when (fetch)
origin  https://github.com/anlisdsu/what-happens-when (push)
C:\Users\lian\what-happens-when [master]>
```

Figure 7: The current configured remote repository for our fork

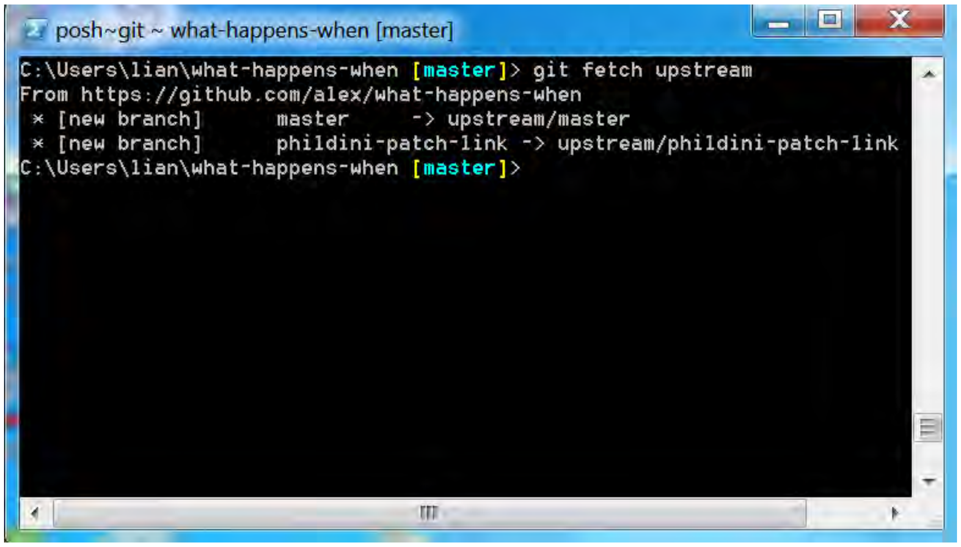
Then type git remote add upstream/ https://github.com/alex/ what-happens-when, then all should be done to set up the upstream source. To make sure this, type again git remote –v, you will see the changes made thus far (Figure 8):



```
C:\Users\lian> cd what-happens-when
C:\Users\lian\what-happens-when [master]> git remote -v
origin  https://github.com/anlisdsu/what-happens-when (fetch)
origin  https://github.com/anlisdsu/what-happens-when (push)
C:\Users\lian\what-happens-when [master]> git remote add upstream https://github
.com/alex/what-happens-when
C:\Users\lian\what-happens-when [master]> git remote -v
origin  https://github.com/anlisdsu/what-happens-when (fetch)
origin  https://github.com/anlisdsu/what-happens-when (push)
upstream        https://github.com/alex/what-happens-when (fetch)
upstream        https://github.com/alex/what-happens-when (push)
C:\Users\lian\what-happens-when [master]>
```

Figure 8: The outcome of setting up the new upstream repository

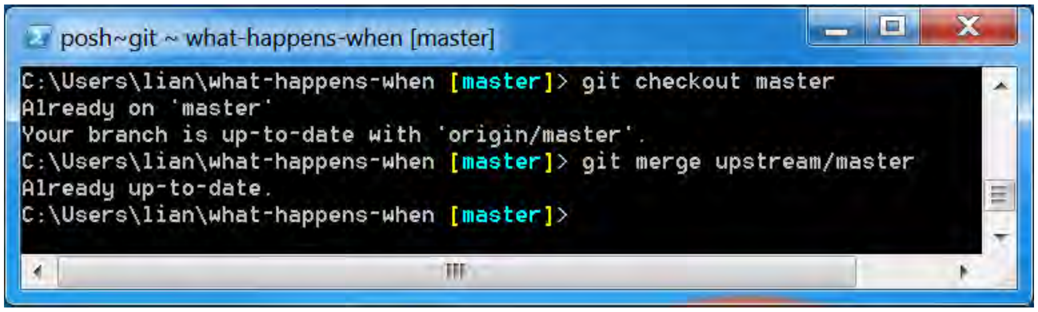
f. Then we can do the sync work
Still go to the terminal (for Windows users), stay at the same local directory (here C:\Users\lian\what-happens-when), and type git fetch upstream to fetch the branches and their respective commits from the upstream repository. We will see the following outcome after some time (Figure 9):



```
C:\Users\lian\what-happens-when [master]> git fetch upstream
From https://github.com/alex/what-happens-when
 * [new branch]      master     -> upstream/master
 * [new branch]      phildini-patch-link -> upstream/phildini-patch-link
C:\Users\lian\what-happens-when [master]>
```

Figure 9: The command and outcome for sync.

Then we should type git checkout master to check out our fork's local master branch and git merge upstream/master to merge the changes from upstream/master into your local master branch:



```
C:\Users\lian\what-happens-when [master]> git checkout master
Already on 'master'
Your branch is up-to-date with 'origin/master'.
C:\Users\lian\what-happens-when [master]> git merge upstream/master
Already up-to-date.
C:\Users\lian\what-happens-when [master]>
```

Figure 10: The outcome for merging upstream changes to local branch.

3.4. Work with other people

Click 4. Work together (see Figure 2), there are a bunch of things we can do, such as Follow People on GitHub (thus get notification about what s/he does), Watch a Project (say up-to-date with a project), Pull Requests (letting the original authors know the changes we add or make), and Issues (often telling the HitHub community some problems that need to be fixed).