Contribute to someone's repository

Say you want to contribute changes to someone else's repository (eg, this one (https://github.com/kbroman/github_tutorial)).

- Go to the repository on github. (Say it's by `myfriend`, and is called `the_repo`, then you'll find it at `https://github.com/myfriend/the_repo`.)
- Click the "Fork" button at the top right.
- You'll now have your own copy of that repository in your github account.
- Open a terminal/shell.
- Type

  ```
  $ git clone git@github.com:username/the_repo
  ```

  where `username` is *your* username.
- You'll now have a local copy of *your version* of that repository.
- Change into that project directory (`the_repo`):

  ```
  $ cd the_repo
  ```

- Add a connection to the original owner's repository.

  ```
  $ git remote add myfriend git://github.com/myfriend/the_repo
  ```

- Note the distinction between `git@github.com:` in the first case and `git://github.com/` in the second case. I'm not sure why these need to be the way they are, but that's what works for me.
- Also note the first myfriend does not need to be the same as the username of `myfriend`. You could very well choose:

  ```
  $ git remote add repo_nickname git://github.com/myfriend/the_repo
  ```

- To check this `remote add` set up:

  ```
  $ git remote -v
  ```

- Make changes to files.
- `git add` and `git commit` those changes

- `git push` them back to github (https://github.com). These will go to *your version* of the repository.
- Note: if you get an error like:

```
error: src refspec master does not match any.
error: failed to push some refs to 'git@github.com:username/the_repo'
```

Then try `git push origin HEAD:gh-pages` (see stackoverflow. (https://stackoverflow.com/questions/4181861/src-refspec-master-does-not-match-any-when-pushing-commits-in-git)). Typing `git show-ref` can show what reference to put after HEAD.

- Go to *your version* of the repository on github.
- Click the “Pull Request” button at the top.
- Note that your friend’s repository will be on the left and *your repository* will be on the right.
- Click the green button “Create pull request”. Give a succinct and informative title, in the comment field give a short explanation of the changes and click the green button “Create pull request” again.

Pulling others’ changes

Before you make further changes to the repository, you should check that your version is up to date relative to your friend’s version.

Go into the directory for the project and type:

```
$ git pull myfriend master
```

This will pull down and merge all of the changes that your friend has made.

Now push them back to your github repository.

```
$ git push
```

Handling pull requests

Say your friend has suggested some changes to your code.

Ask them to get a github account (first_use.html) and follow the instructions above: fork your repository, make the changes, and submit a pull request.

Once they do that, you’ll get an email about it. How to handle it?

Using the github website:

- Go to your version of the repository.
- Click on “Pull Requests” at the top.
- Click on the particular request.
- You’ll see their comments on the pull request, and can click to see the exact changes.

- If you want them to make further changes before you merge the changes into your repository, add a comment.
- If you hate the whole idea, just click the “Close” button.
- If you want to merge the changes into your repository, click the “Merge pull request” button.
- Your github repository will now be fixed, but you’ll want to get them into your local repository, too.
- Open a terminal/shell, and type

```
$ git pull
```

Using the command line

You don’t have to use the github website for this.

- Open a terminal/shell.
- Go into the directory for your project.
- Add a connection to your friend’s version of the github repository, if you haven’t already.

```
$ git remote add myfriend git://github.com/myfriend/the_repo
```

- Pull his/her changes.

```
$ git pull myfriend master
```

- Push them back to your github repository.

```
$ git push
```

- The pull request on github will be automatically closed.

Next: Handling merge conflicts (merge_conflicts.html)
