Git and Github Workflow:

1- Clone the repo to your computer

2- Make a new branch for your story with the following naming convention [Name]_[IssueNumber]_[DiscriptiveWords].
Example yahia_12_add_comments

```
$ git branch branch_name
$ git checkout branch_name
```

<u>Note on branches:</u>
Branches are what naturally happens when you want to work on multiple features at the same time. You wouldn't want to end up with a master branch which has Feature A half done and Feature B half done.

3- Build an awesome feature
4- Check what have changed

```
$ git status
```

`$ git diff` (For what has changed before `add`ing)
`$ git diff --cached` (For what has changed after `add`ing and before `commit`ing)
You can give the `git diff` program filenames as optional arguments (e.g
`$ git diff file_name1 file_name2`)
Note on diff: If you prefer using a specific GUI-based diff tool, you can use (more info about difftool here):

```
$ git difftool
```

Everything is good?

5- Add/Remove affected files.

```
$ git add file_name1 file_name2
$ git rm --cached file_name1 file_name2
```

6- Commit your changes with **<u>a descriptive and short commit message</u>**

```
$ git commit -m 'Issue #<number> very descriptive and short commit
message'
```

The “`Issue #issue_number`” is **<u>very important</u>** it hooks the commit to the issue.
Go to the issue in github repo and update it’s labels if needed.
Examples of short descriptive messages:

```
$ git commit -m 'Issue #5, Fix duplicate comments bug'
```

Tip: think of your commit message as the following, “Applying this commit will <what is does>” e.g

Applying this commit will `Fix duplicate comments bug`.

Now you need to pull the latest code from master in order to have your code mergeable.

7- Checkout to master
```
$ git checkout master
```

8- Pull the latest code
```
$ git pull origin master
```

9- Back to your feature branch
```
$ git checkout C1_yahia_12_add_comments
```

10- Merge master with your current branch
```
$ git merge master
```

Conflicts may/will happen, don't panic. If there is a conflict, your options are :

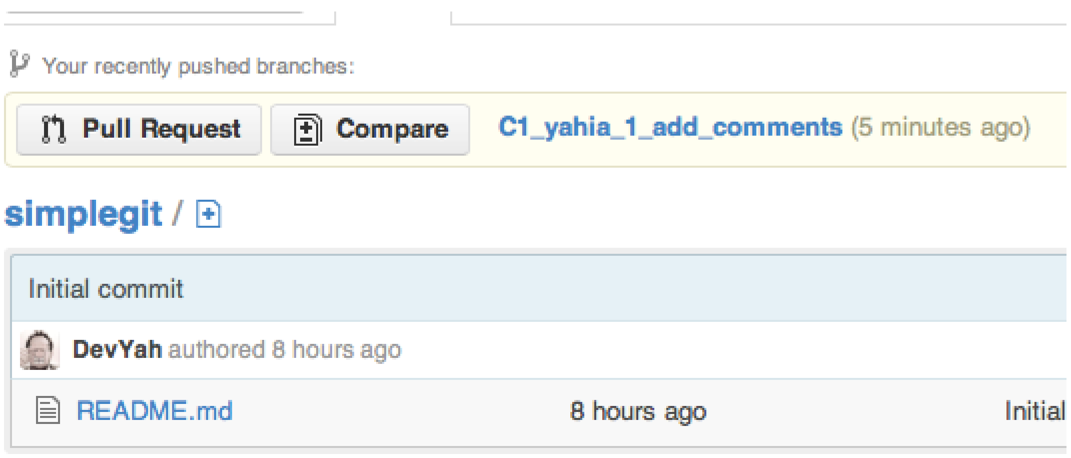
a- Decide not to merge: git reset HEAD
b- Decide to merge, edit the files (check here) then use "git add" to add them and then git commit
OR
b'- use git mergetool.
More info here

11- As mentioned in the try-git course, use `$ git log` and `$ git log --summary` to get an overview of your history. (Tip: you might be interested in tig)

So far everything you've done is local. It's time to show it to the world and ask for a pull request (to have your code in the master)
12- Make sure you are in the right branch
```
$ git branch
```

13- Push your branch to the remote repo on github
```
$ git push origin branch_name
```

14- Go to github and make a pull request.
There are many ways to make a pull request, the easiest is
 a. go to repo on github and click on pull request
 b. click on pull_request.


Your recently pushed branches:

Pull Request | Compare | C1_yahia_1_add_comments (5 minutes ago)

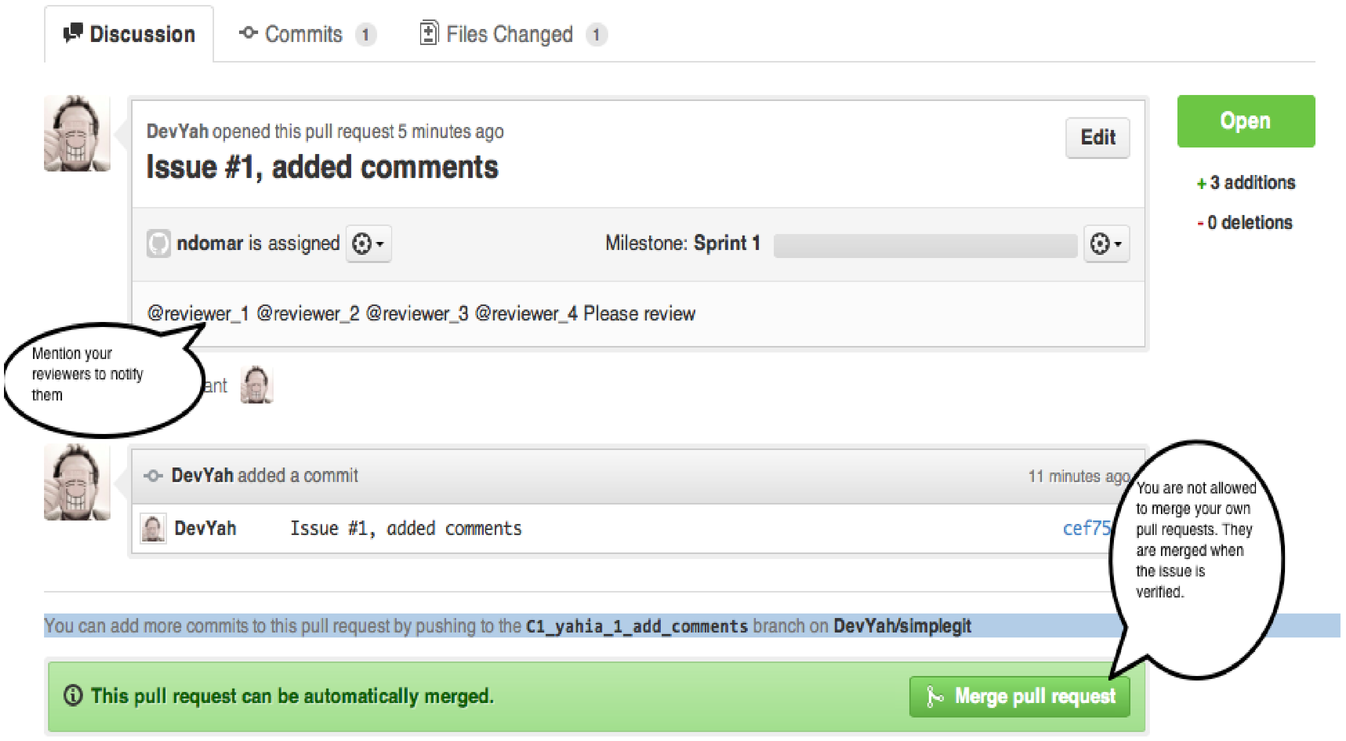
simplegit /

Initial commit

DevYah authored 8 hours ago

README.md 8 hours ago Initial

c. Add some info (issue number, mention reviewers, any notes)


Discussion | Commits 1 | Files Changed 1

DevYah opened this pull request 5 minutes ago

Issue #1, added comments

Edit

Open

+3 additions

-0 deletions

ndomar is assigned

Milestone: Sprint 1

@reviewer_1 @reviewer_2 @reviewer_3 @reviewer_4 Please review

DevYah added a commit 11 minutes ago

DevYah Issue #1, added comments cef75

You can add more commits to this pull request by pushing to the C1_yahia_1_add_comments branch on DevYah/simplegit

This pull request can be automatically merged.

Merge pull request

Mention your reviewers to notify them
You are not allowed to merge your own pull requests. They are merged when the issue is verified.

It is your responsibility to have a "This pull request can be automatically merged." message. If you don't have it, make sure that you merged master then commit and push the merge.

pull request
https://help.github.com/articles/using-pull-requests