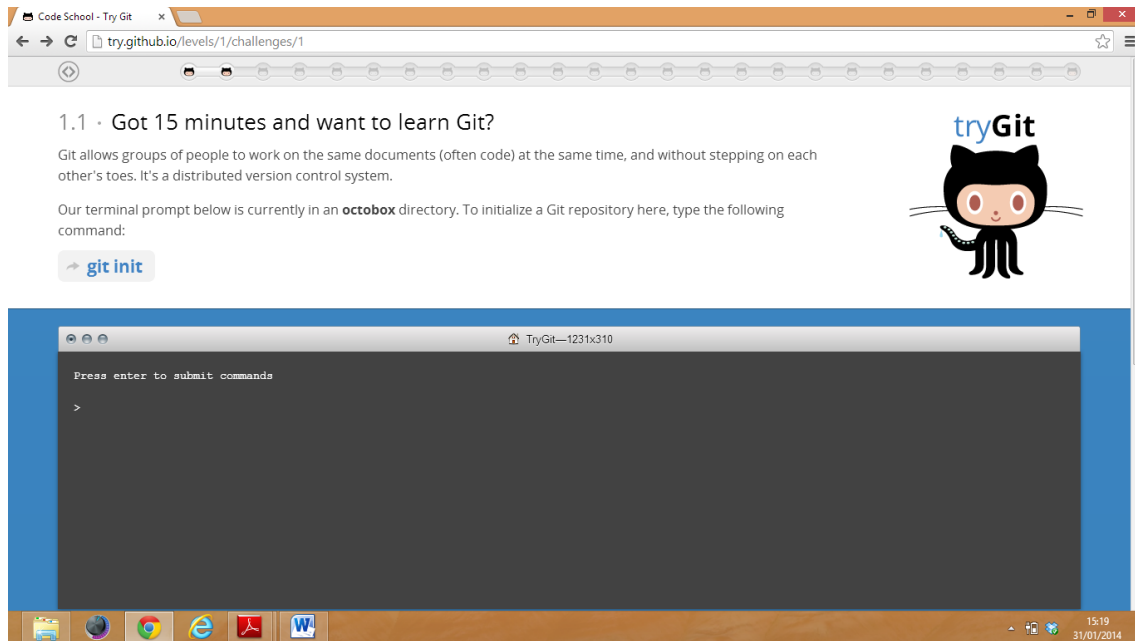


Nombre: Mary C. Amado Tije
Carné: 201313751
IPC1 Sección A.
Laboratorio de IPC.

25 Callenges de GitHub

01



1.1 · Got 15 minutes and want to learn Git?

Git allows groups of people to work on the same documents (often code) at the same time, and without stepping on each other's toes. It's a distributed version control system.

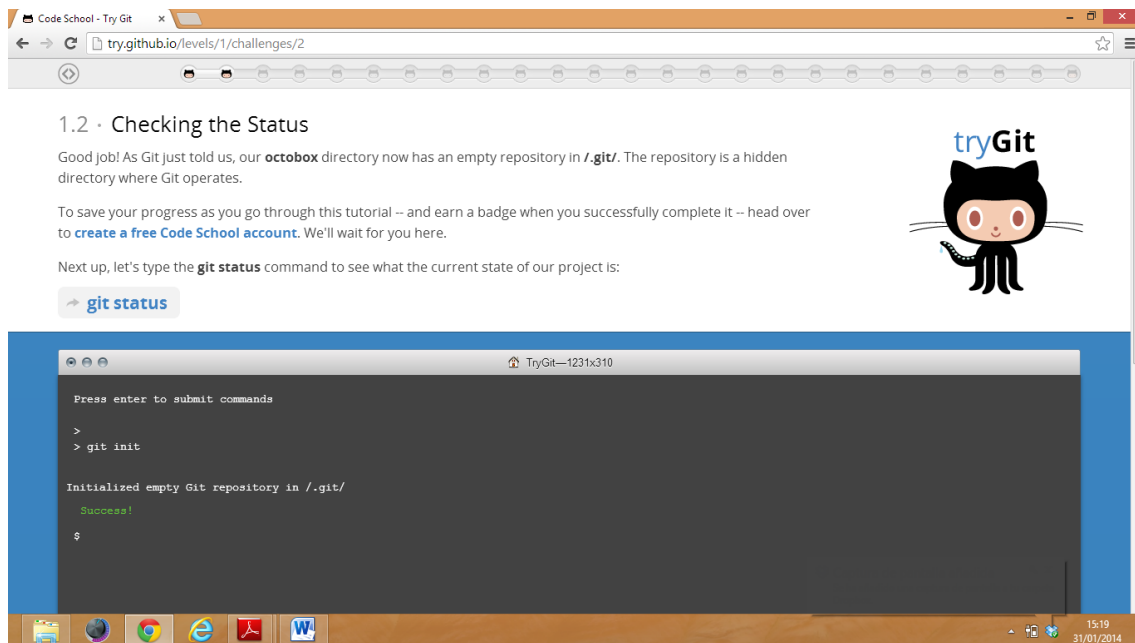
Our terminal prompt below is currently in an **octobox** directory. To initialize a Git repository here, type the following command:

[git init](#)

TryGit

```
Press enter to submit commands
>
```

#02



1.2 · Checking the Status

Good job! As Git just told us, our **octobox** directory now has an empty repository in **./git/**. The repository is a hidden directory where Git operates.

To save your progress as you go through this tutorial -- and earn a badge when you successfully complete it -- head over to [create a free Code School account](#). We'll wait for you here.

Next up, let's type the **git status** command to see what the current state of our project is:

[git status](#)

TryGit

```
Press enter to submit commands
>
> git init
Initialized empty Git repository in ./git/
Success!
$
```

#03

Code School - Try Git


trygithub.io/levels/1/challenges/3

1.3 · Adding & Committing

I created a file called **octocat.txt** in the octobox repository for you (as you can see in the browser below).

You should run the **git status** command again to see how the repository status has changed:

[git status](#)



```
TryGit-1231x310

Initialized empty Git repository in /.git/
Success!

$ git status

# On branch master
#
# Initial commit
nothing to commit (create/copy files and use "git add" to track)
Success!

$
```

Captura de pantalla añadida

Se ha añadido una captura de pantalla a tu carpeta Dropbox.

15:19

31/01/2014

#04

Code School - Try Git


trygithub.io/levels/1/challenges/4

1.4 · Adding Changes

Good, it looks like our Git repository is working properly. Notice how Git says **octocat.txt** is "untracked"? That means Git sees that **octocat.txt** is a new file.

To tell Git to start tracking changes made to **octocat.txt**, we first need to add it to the staging area by using **git add**.

[git add octocat.txt](#)



```
TryGit-1231x310

Success!

$ git status

# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#   octocat.txt
nothing added to commit but untracked files present (use "git add" to track)
Success!

$
```

15:20

31/01/2014

#05


Code School - Try Git

trygithub.io/levels/1/challenges/5

1.5 · Checking for Changes

Good job! Git is now tracking our **octocat.txt** file. Let's run **git status** again to see where we stand:

[git status](#)



```
TryGit-1231x310
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#   octocat.txt
nothing added to commit but untracked files present (use "git add" to track)
Success!
$ git add octocat.txt

Nice job, you've added octocat.txt to the Staging Area
$ git status
```

My Octocat Repository

Advice

15:20

31/01/2014

#06

Code School - Try Git


trygithub.io/levels/1/challenges/6

1.6 · Committing

Notice how Git says **changes to be committed**? The files listed here are in the **Staging Area**, and they are not in our repository yet. We could add or remove files from the stage before we store them in the repository.

To store our staged changes we run the **commit** command with a message describing what we've changed. Let's do that now by typing:

[git commit -m "Add cute octocat story"](#)



```
TryGit-1231x310

Nice job, you've added octocat.txt to the Staging Area
$ git status

# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   octocat.txt
#
Success!
$ git commit -m "Add cute octocat story"
```

My Octocat Repository

Advice


15:20

31/01/2014

#07

Code School - Try Git

try.github.io/levels/1/challenges/7



I put some in an **octofamily** directory and some others ended up in the root of our octobox. Luckily, we can add all the new files using a wildcard with **git add**. Don't forget the quotes!

[git add '*.txt'](#)

TryGit—1231x310

```
$ Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   octocat.txt
#
$ git add octocat.txt
$ git commit -m "Add cute octocat story"
[master (root-commit) 20b5cod] Add cute octocat story
1 file changed, 1 insertion(+)
create mode 100644 octocat.txt
$ git add '*.txt'
```

My Octobox Repository

.git

octofamily


blue_octocat.txt

octocat.txt

red_octocat.txt

Advice

Wildcards:
We need quotes so that Git will receive the wildcard before our shell can interfere with it. Without quotes our shell will only execute the wildcard search within the current directory. Git will receive the list of files the shell found




15:21

31/01/2014

#08

Code School - Try Git

try.github.io/levels/1/challenges/8



It looks good, go ahead and run.

[git commit -m 'Add all the octocat txt files'](#)

TryGit—1231x310

```
$ git commit -m "Add cute octocat story"
[master (root-commit) 20b5cod] Add cute octocat story
1 file changed, 1 insertion(+)
create mode 100644 octocat.txt
$ git add '*.txt'
$ git commit -m 'Add all the octocat txt files'
```

My Octobox Repository

.git

octofamily


blue_octocat.txt

octocat.txt

red_octocat.txt

Advice

Check all the things!
When using wildcards you want to be extra careful when doing commits. Make sure to check what files and folders are staged by using `git status` before you do the actual commit. This way you can be sure you're committing exactly the things you want



15:22

31/01/2014

#09

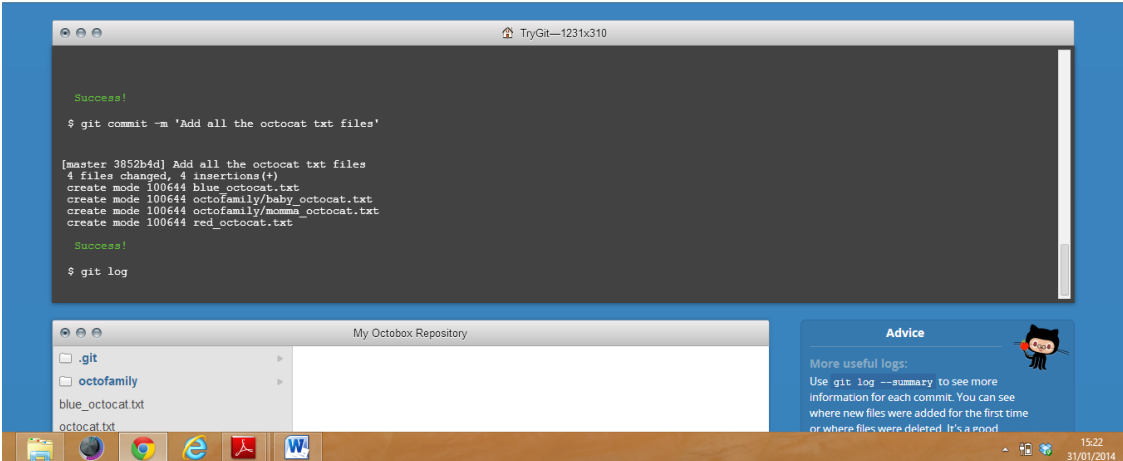
Code School - Try Git x

try.github.io/levels/1/challenges/9

So we've made a few commits. Now let's browse them to see what we changed.

Fortunately for us, there's **git log**. Think of Git's log as a journal that remembers all the changes we've committed so far, in the order we committed them. Try running it now:

[git log](#)



The terminal window shows the following output:

```
Success!
$ git commit -m 'Add all the octocat txt files'

[master 3852b4d] Add all the octocat txt files
4 files changed, 4 insertions(+)
create mode 100644 blue_octocat.txt
create mode 100644 octofamily/baby_octocat.txt
create mode 100644 octofamily/locomotive_octocat.txt
create mode 100644 red_octocat.txt

Success!
$ git log
```

The file explorer shows the following structure:

```
My Octobox Repository
├── .git
├── octofamily
├── blue_octocat.txt
└── octocat.txt
```

Advice

More useful logs:

Use `git log --summary` to see more information for each commit. You can see where new files were added for the first time or where files were deleted. It's a good

15:22 31/01/2014

#10

Code School - Try Git x

try.github.io/levels/1/challenges/10

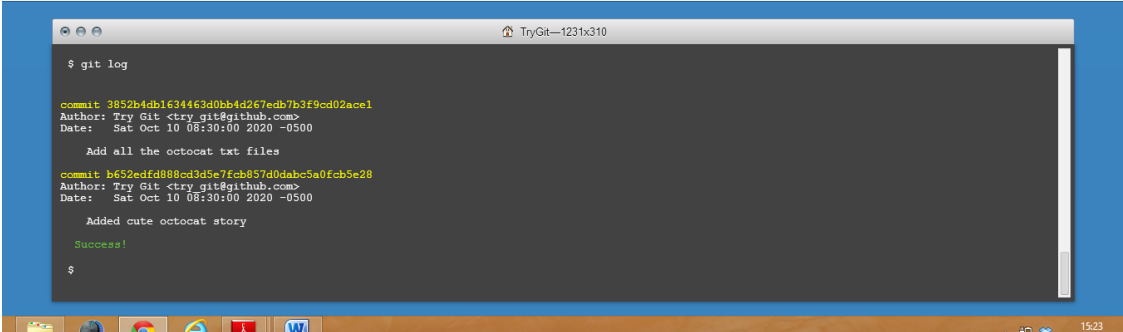
1.10 - Remote Repositories

Great job! We've gone ahead and created a new empty GitHub repository for you to use with Try Git at https://github.com/try-git/try_git.git. To push our local *repo* to the GitHub server we'll need to add a remote repository.

This command takes a *remote name* and a *repository URL*, which in your case is https://github.com/try-git/try_git.git.

Go ahead and run **git remote add** with the options below:

[git remote add origin https://github.com/try-git/try_git.git](#)



The terminal window shows the following output:

```
$ git log

commit 3852b4cb1634463d0bb4d267edb7b3f9cd02ace1
Author: Try Git <try_git@github.com>
Date: Sat Oct 10 08:30:00 2020 -0500
    Add all the octocat txt files

commit b652edfd880cd3d5e7fcb857d0dabc5a0fcb5e28
Author: Try Git <try_git@github.com>
Date: Sat Oct 10 08:30:00 2020 -0500
    Added cute octocat story

Success!
$
```

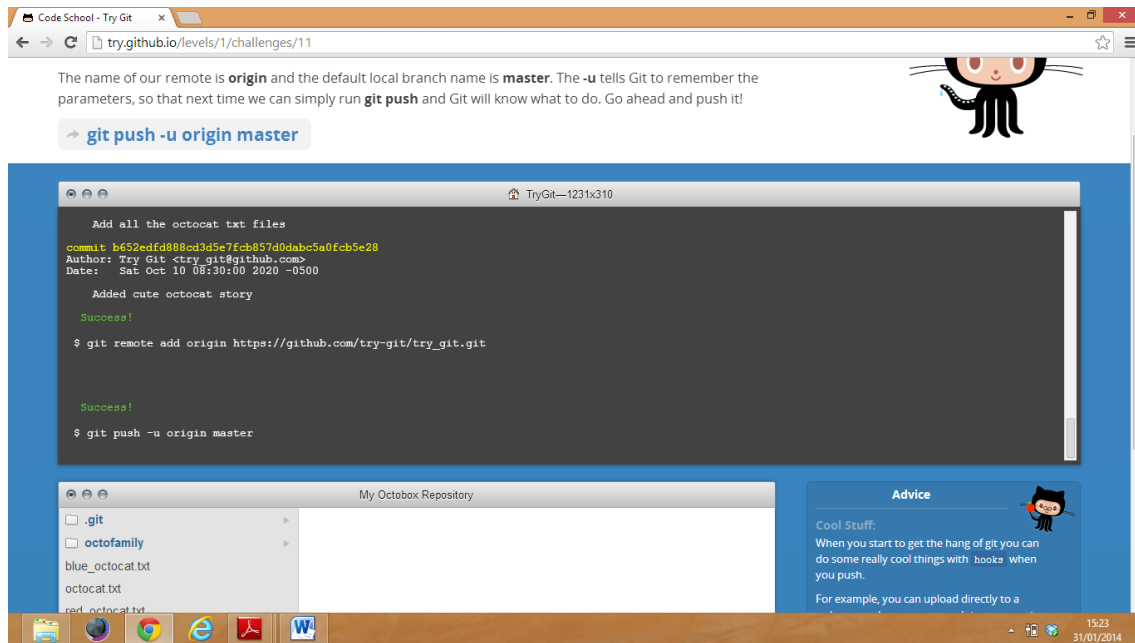
The file explorer shows the following structure:

```
My Octobox Repository
├── .git
├── octofamily
├── blue_octocat.txt
└── octocat.txt
```

tryGit

15:23 31/01/2014

#11



The name of our remote is **origin** and the default local branch name is **master**. The **-u** tells Git to remember the parameters, so that next time we can simply run **git push** and Git will know what to do. Go ahead and push it!

`git push -u origin master`

```
TryGit-1231x310
Add all the octocat txt files
commit b659e1fd080cd3d5e7fcb657404abc5a0fcb5e28
Author: Try Git <try_git@github.com>
Date: Sat Oct 10 08:30:00 2020 -0500
    Added cute octocat story
Success!
$ git remote add origin https://github.com/try-git/try_git.git
Success!
$ git push -u origin master
```

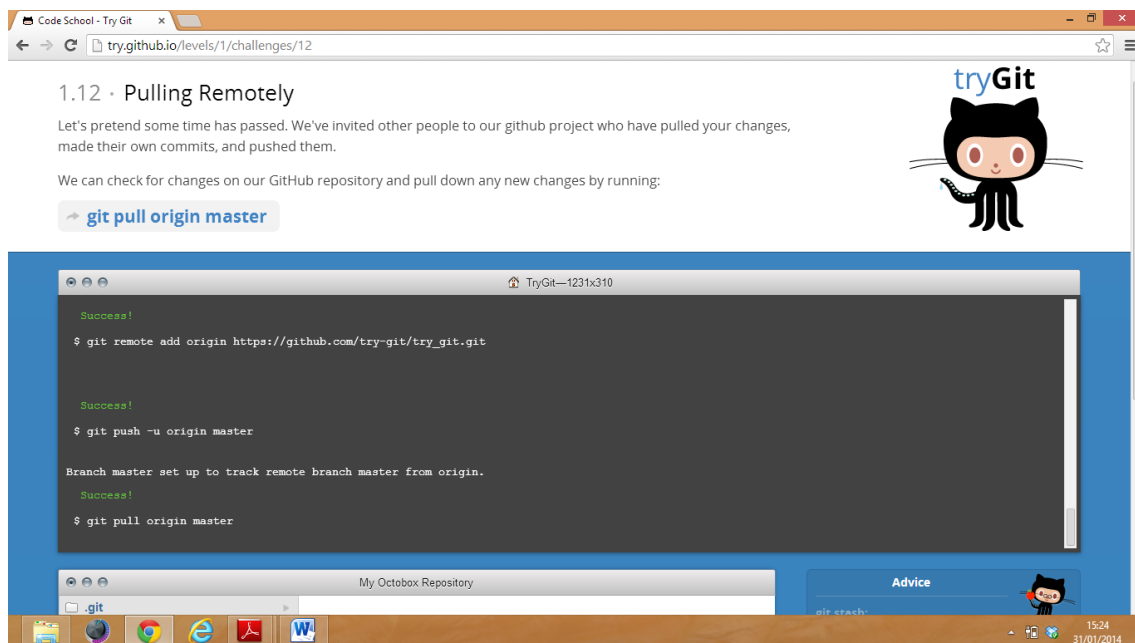
My Octobox Repository

- .git
- octofamily
- blue_octocat.txt
- octocat.txt
- red_octocat.txt

Advice

Cool Stuff:
When you start to get the hang of git you can do some really cool things with hooks when you push.
For example, you can upload directly to a

#12



1.12 · Pulling Remotely

Let's pretend some time has passed. We've invited other people to our github project who have pulled your changes, made their own commits, and pushed them.

We can check for changes on our GitHub repository and pull down any new changes by running:

`git pull origin master`

```
TryGit-1231x310
Success!
$ git remote add origin https://github.com/try-git/try_git.git
Success!
$ git push -u origin master
Branch master set up to track remote branch master from origin.
Success!
$ git pull origin master
```

My Octobox Repository

- .git

Advice

git crash

#13

Code School - Try Git


try.github.io/levels/1/challenges/13

1.13 • Differences

Uh oh, looks like there has been some additions and changes to the octocat family. Let's take a look at what is **different** from our last commit by using the **git diff** command.

In this case we want the diff of our most recent commit, which we can refer to using the **HEAD** pointer.

➔ **git diff HEAD**

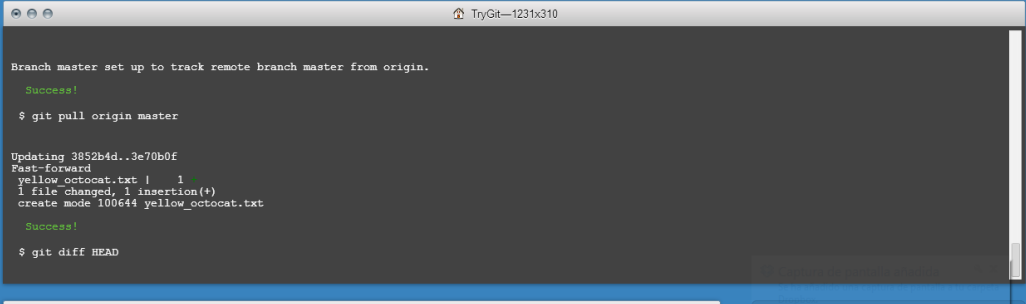


```
Branch master set up to track remote branch master from origin.
Success!

$ git pull origin master

Updating 3852b4d..3e70b0f
Fast-forward
 yellow_octocat.txt | 1
 1 file changed, 1 insertion(+)
 create mode 100644 yellow_octocat.txt
Success!

$ git diff HEAD
```



15:24 31/01/2014

#14

Code School - Try Git


try.github.io/levels/1/challenges/14

1.14 • Staged Differences

Another great use for **diff** is looking at changes within files that have already been staged. Remember, staged files are files we have told git that are ready to be committed.

Let's use **git add** to stage **octofamily/octodog.txt**, which I just added to the family for you.

➔ **git add octofamily/octodog.txt**

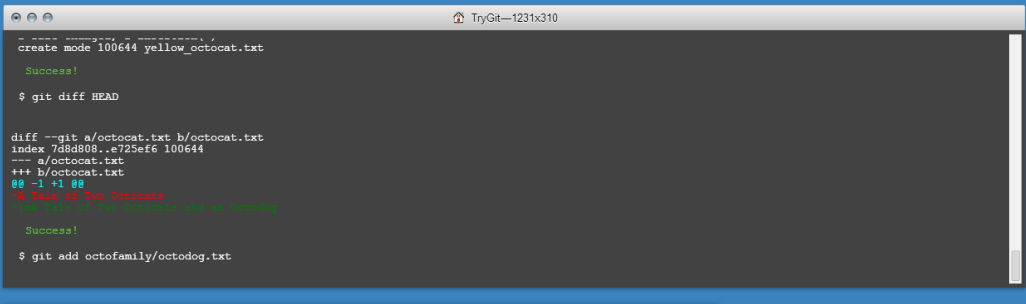


```
create mode 100644 yellow_octocat.txt
Success!

$ git diff HEAD

diff --git a/octocat.txt b/octocat.txt
index 7d8d800..e725ef6 100644
--- a/octocat.txt
+++ b/octocat.txt
@@ -1,1 @@
-Hello! I'm Octocat
+Hello! I'm Octocat and an Octodog
Success!

$ git add octofamily/octodog.txt
```



15:24 31/01/2014

#15


Code School - Try Git

try.github.io/levels/1/challenges/15

1.15 · Staged Differences (cont'd)

Good, now go ahead and run **git diff** with the **--staged** option to see the changes you just staged. You should see that **octodog.txt** was created.

[git diff --staged](#)



```
diff --git a/octocat.txt b/octocat.txt
index 7a8d808..e725ef6 100644
--- a/octocat.txt
+++ b/octocat.txt
@@ -1,1 @@
-Octocat
+Octocat and an Octodog

Success!

$ git add octofamily/octodog.txt

Success!

$ git diff --staged
```

#16

Code School - Try Git


try.github.io/levels/1/challenges/16

1.16 · Resetting the Stage

So now that octodog is part of the family, octocat is all depressed. Since we love octocat more than octodog, we'll turn his frown around by removing **octodog.txt**.

You can unstage files by using the **git reset** command. Go ahead and remove **octofamily/octodog.txt**.

[git reset octofamily/octodog.txt](#)



```
Success!

$ git diff --staged

diff --git a/octofamily/octodog.txt b/octofamily/octodog.txt
new file mode 100644
index 0000000..cfbc74a
--- /dev/null
+++ b/octofamily/octodog.txt
@@ -0,0 +1 @@
+Octocat

Success!

$ git reset octofamily/octodog.txt
```


#17

Code School - Try Git


try.github.io/levels/1/challenges/17

1.17 · Undo


git reset did a great job of unstaging octodog.txt, but you'll notice that he's still there. He's just not staged anymore. It would be great if we could go back to how things were before octodog came around and ruined the party.


Files can be changed back to how they were at the last commit by using the command: **git checkout -- <target>**. Go ahead and get rid of all the changes since the last commit for **octocat.txt**

→ **git checkout -- octocat.txt**



TryGit-1231x310

```
diff --git a/octofamily/octodog.txt b/octofamily/octodog.txt
new file mode 100644
index 0000000..cfbc74a
--- /dev/null
+++ b/octofamily/octodog.txt
@@ -0,0 +1 @@
+
+
+Success!
+
+$ git reset octofamily/octodog.txt
+
+Success!
+
+$ git checkout -- octocat.txt
```



15:26
31/01/2014

#18

Code School - Try Git


try.github.io/levels/1/challenges/17

1.17 · Undo


git reset did a great job of unstaging octodog.txt, but you'll notice that he's still there. He's just not staged anymore. It would be great if we could go back to how things were before octodog came around and ruined the party.


Files can be changed back to how they were at the last commit by using the command: **git checkout -- <target>**. Go ahead and get rid of all the changes since the last commit for **octocat.txt**

→ **git checkout -- octocat.txt**



TryGit-1231x310

```
diff --git a/octofamily/octodog.txt b/octofamily/octodog.txt
new file mode 100644
index 0000000..cfbc74a
--- /dev/null
+++ b/octofamily/octodog.txt
@@ -0,0 +1 @@
+
+
+Success!
+
+$ git reset octofamily/octodog.txt
+
+Success!
+
+$ git checkout -- octocat.txt
```



15:26
31/01/2014

#19

Code School - Try Git


trygithub.io/levels/1/challenges/19

1.19 · Switching Branches

Great! Now if you type **git branch** you'll see two local branches: a main branch named **master** and your new branch named **clean_up**.

You can switch branches using the **git checkout <branch>** command. Try it now to switch to the **clean_up** branch:

```
git checkout clean_up
```




```
TryGit-1231x310

Success!
$ git checkout -- octocat.txt

Success!
$ git branch clean_up

Success!
$ git checkout clean_up
```



#20

Code School - Try Git


trygithub.io/levels/1/challenges/20

1.20 · Removing All The Things

Ok, so you're in the **clean_up** branch. You can finally remove all those pesky octocats by using the **git rm** command which will not only remove the actual files from disk, but will also stage the removal of the files for us.

You're going to want to use a wildcard again to get all the octocats in one sweep, go ahead and run:

```
git rm '*.txt'
```




```
TryGit-1231x310

Success!
$ git branch clean_up

Success!
$ git checkout clean_up

Switched to branch 'clean_up'
Success!
$ git rm '*.txt'
```



#21

Code School - Try Git


trygithub.io/levels/1/challenges/21

1.21 · Committing Branch Changes

Now that you've removed all the cats you'll need to commit your changes.

Feel free to run **git status** to check the changes you're about to commit.

[git commit -m "Remove all the cats"](#)



TryGit—1231x310

```
Switched to branch 'clean_up'

Success!

$ git rm '*.txt'

rm 'blue_octocat.txt'
rm 'octocat.txt'
rm 'octofamily/baby_octocat.txt'
rm 'octofamily/momma_octocat.txt'
rm 'red_octocat.txt'

Success!

$ git commit -m "Remove all the cats"
```

My Octobox Repository

.git

octofamily

Advice

The '-a' option

If you happen to delete a file without using

15:27

31/01/2014

#22

Code School - Try Git


trygithub.io/levels/1/challenges/22

1.22 · Switching Back to master

Great, you're almost finished with the cat... er the bug fix, you just need to switch back to the **master** branch so you can copy (or **merge**) your changes from the **clean_up** branch back into the **master** branch.

Go ahead and checkout the **master** branch:

[git checkout master](#)



TryGit—1231x310

```
rm 'red_octocat.txt'

Success!

$ git commit -m "Remove all the cats"

[clean up 63540fe] Remove all the cats
5 files changed, 5 deletions(-)
delete mode 100644 blue_octocat.txt
delete mode 100644 octocat.txt
delete mode 100644 octofamily/baby_octocat.txt
delete mode 100644 octofamily/momma_octocat.txt
delete mode 100644 red_octocat.txt

Success!

$ git checkout master
```

My Octobox Repository

Advice

The '-a' option

If you happen to delete a file without using

15:27

31/01/2014

#23

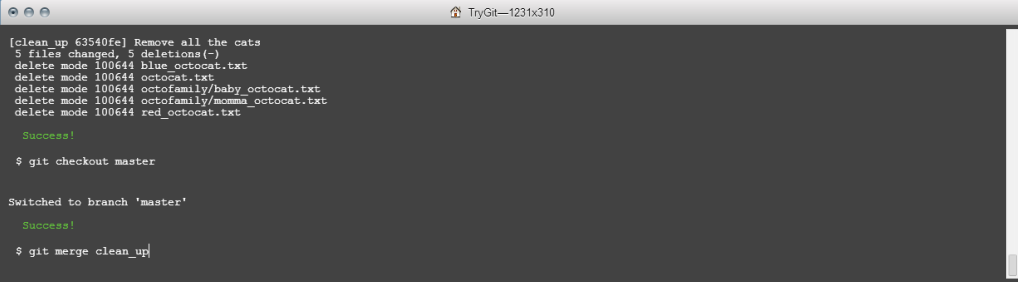
Code School - Try Git

try.github.io/levels/1/challenges/23

Alright, the moment has come when you have to merge your changes from the **clean_up** branch into the **master** branch. Take a deep breath, it's not that scary.

We're already on the **master** branch, so we just need to tell Git to merge the **clean_up** branch into it:

```
git merge clean_up
```



```
[clean up 63540fe] Remove all the cats
5 files changed, 5 deletions(-)
delete mode 100644 blue_octocat.txt
delete mode 100644 octocat.txt
delete mode 100644 octofamily/baby_octocat.txt
delete mode 100644 octofamily/momma_octocat.txt
delete mode 100644 red_octocat.txt

Success!

$ git checkout master

Switched to branch 'master'

Success!

$ git merge clean_up
```

My Octobox Repository

- .git
- octofamily
- blue_octocat.txt
- octocat.txt

Advice

Merge Conflicts

Merge Conflicts can occur when changes are made to a file at the same time. A lot of people get really scared when a conflict happens, but

15:28 31/01/2014

#24

Code School - Try Git

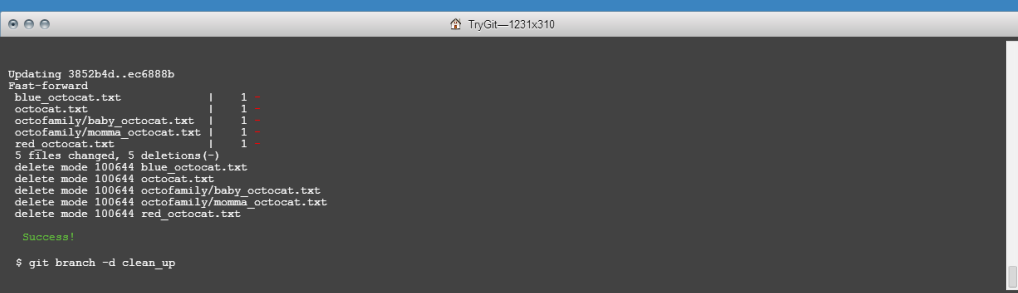
try.github.io/levels/1/challenges/24

1.24 - Keeping things clean

Congratulations! You just accomplished your first successful bugfix and merge. All that's left to do is clean up after yourself. Since you're done with the **clean_up** branch you don't need it anymore.

You can use **git branch -d <branch name>** to delete a branch. Go ahead and delete the **clean_up** branch now:

```
git branch -d clean_up
```



```
Updating 3852b4d..ec6888b
Fast-forward
 blue octocat.txt | 1 -
 octocat.txt      | 1 -
 octofamily/baby octocat.txt | 1 -
 octofamily/momma octocat.txt | 1 -
 red octocat.txt  | 1 -
 5 files changed, 5 deletions(-)
 delete mode 100644 blue_octocat.txt
 delete mode 100644 octocat.txt
 delete mode 100644 octofamily/baby_octocat.txt
 delete mode 100644 octofamily/momma_octocat.txt
 delete mode 100644 red_octocat.txt

Success!

$ git branch -d clean_up
```

My Octobox Repository

- .git
- octofamily

Advice

Force delete

What if you have been working on a feature

15:29 31/01/2014

#25



Code School - Try Git

trygithub.io/levels/1/challenges/25

1.25 · The Final Push

Here we are, at the last step. I'm proud that you've made it this far, and it's been great learning Git with you. All that's left for you to do now is to push everything you've been working on to your remote repository, and you're done!

[git push](#)



TryGit—1231x310

```
red_octocat.txt      1
5 files changed, 5 deletions(-)
delete mode 100644 blue_octocat.txt
delete mode 100644 octocat.txt
delete mode 100644 octofamily/baby_octocat.txt
delete mode 100644 octofamily/mama_octocat.txt
delete mode 100644 red_octocat.txt

Success!

$ git branch -d clean_up

Deleted branch clean_up (was ec6888b).

Success!

$ git push
```

15:29

31/01/2014

#26

The screenshot shows a web browser window with the address bar displaying `try.github.io/levels/1/challenges/25`. The page content includes a heading "Wrap it all Up" and a button labeled "Wrap it all Up". Below the browser window, a terminal window titled "TryGit—1231x310" shows the following commands and output:

```

$ git branch -d clean_up
Deleted branch clean_up (was ec6888b).
Success!
$ git push
To https://github.com/try-git/try_git.git
3e70b0f..c91bcd8 master -> master
Success!
>

```

Below the terminal window, a file explorer window titled "My Octobox Repository" shows a list of files and folders:

- .git
- octofamily
- yellow_octocat.txt

In the bottom right corner, there is a blue box titled "Advice" with the text: "Learning more about Git. We only scratched the surface of Git in this course. There is so much more you can do with it. Check out the Git documentation for a..."