



*Hello my dear student of Zero To Mastery 2018.
Before you start reading the guide, please remember the following statement:*

One GitHub Repository = One cloned repo!

*Do not work on the same cloned repo
from **two** different paths on your local PC*



Remember this, or It might cause unexpected results...

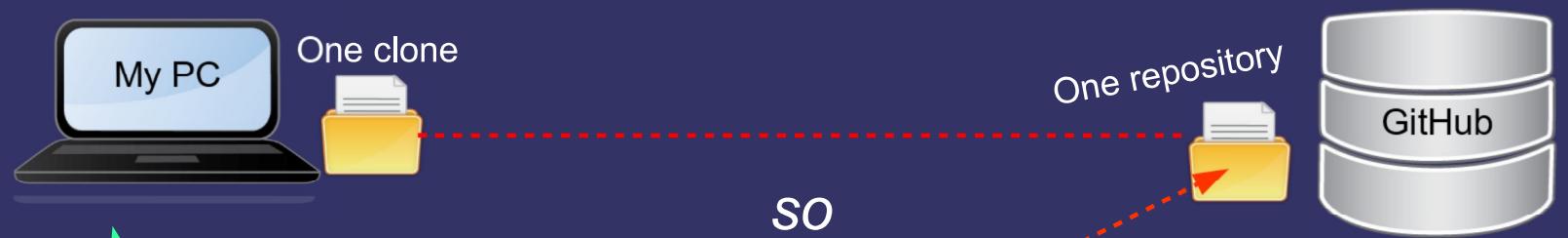


The very same cloned repo in different places on your PC will have different package references id's after you **manipulate the original repo!**

Second path =Desktop cloned repo

First path = Original cloned repo

The image shows two side-by-side Windows File Explorer windows. The left window's address bar is highlighted with a red box and contains the path `C:\Users\Dimo Mironov\Desktop\test\background-generator\.git`. The right window's address bar is also highlighted with a red box and contains the path `C:\Users\Dimo Mironov\Desktop\Dimo Studies\Andrei Negoie\The Complete Web D...`. Both windows show a list of files and folders within their respective .git directories. A red arrow points from the left window's list to the right window's list, specifically highlighting the 'packed-refs' file. Below the windows is a Notepad application showing two lines of text. A magnifying glass icon with a blue handle is positioned over the first line of text in the left Notepad window, which reads: `# pack-refs --peeled --fully-peeled --sorted`. The second line of text in the left Notepad window is partially visible as `fd8f4fe93b1c078114b0f846b7e5345c9367f9e refs/remote...`. A magnifying glass icon with a blue handle is also positioned over the first line of text in the right Notepad window, which reads: `# pack-refs --peeled --fully-peeled --sorted`. The second line of text in the right Notepad window is partially visible as `59dc737c740354f2ed7033ab04f5debca1947 refs/remote...`.



SO

One GitHub Repository will contain only one cloned version of that repo
on your local PC!

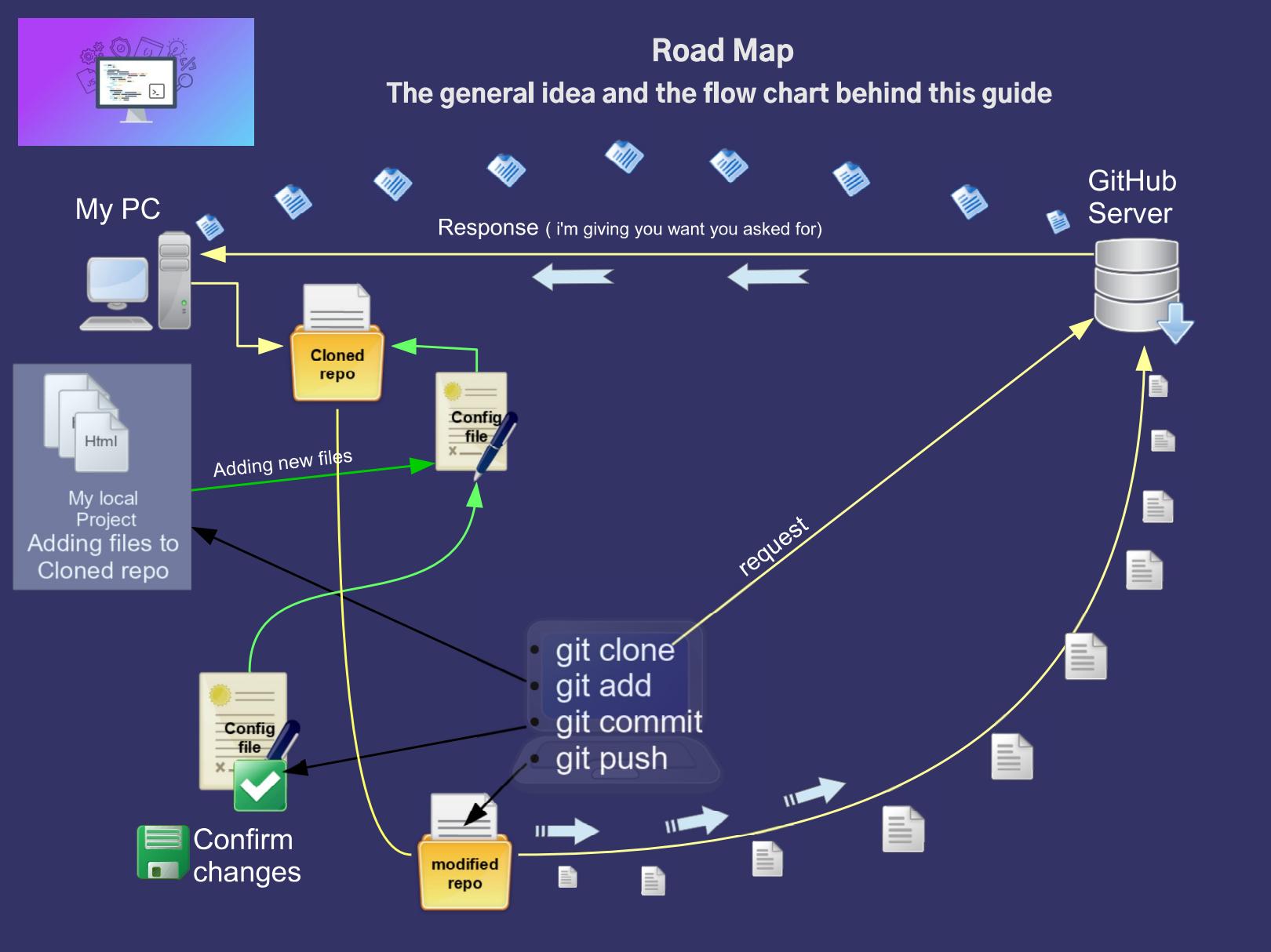
Don't make multiple clones of this repo on local machine.

Okay?



Road Map

The general idea and the flow chart behind this guide



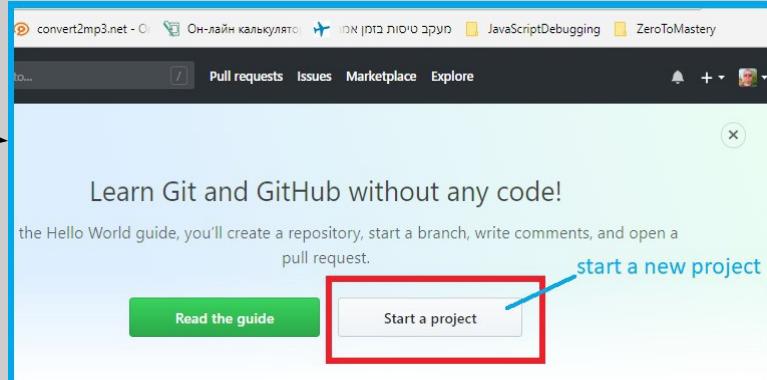
Git and GitHub Source Control

Part1 (Working on a new repository)

The Complete Web Developer in 2018: Zero to Mastery

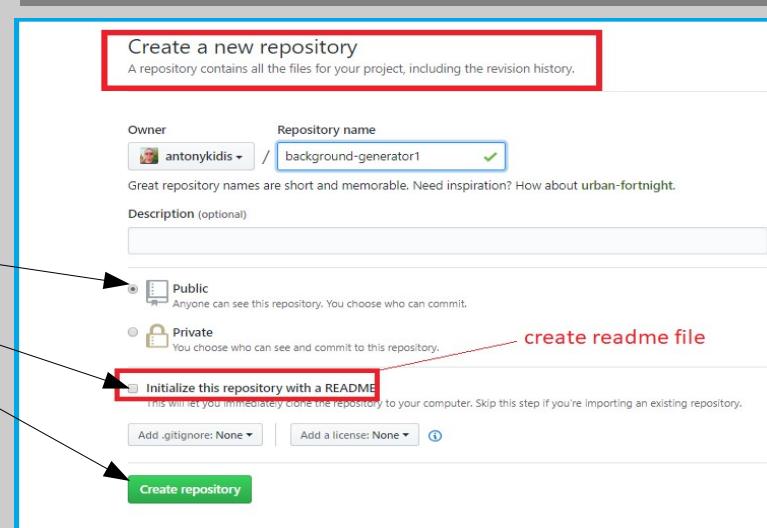
Step 1

- Navigate to <https://github.com>
- Start a new project



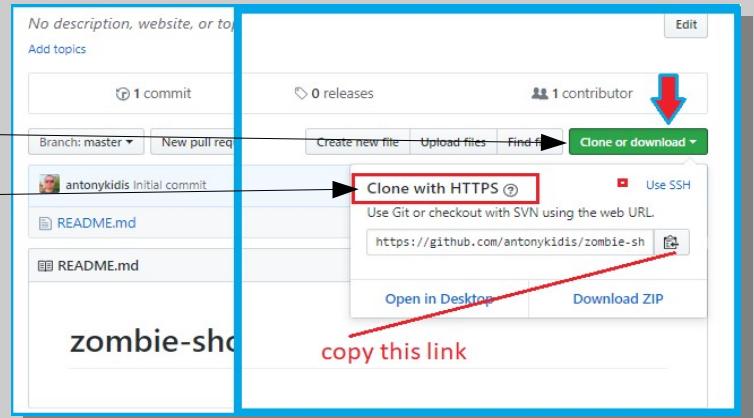
Step 2

- Name your project **background-generator**
- Select Public
- Check to create a **README file**
- Finally click **create repository** button



Step 3

- Click Clone or download button
- Select use HTTPS
- Copy the URL to a clipboard by clicking the Little icon



Step 4

- Open terminal(bash,or powershell) or other preferred terminal.
- Navigate to a project you currently work on.
- Enter the following command:

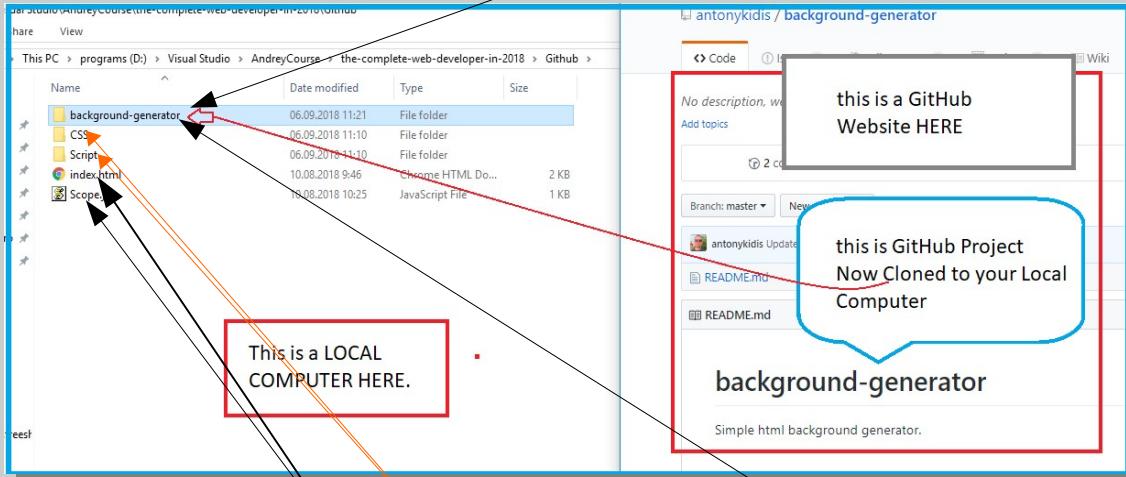
```
git clone https://github.com/antonykidis/background-generator.git
```

This will clone GitHub's repository to your computer.

- Here is the following output
Unpacking objects 100% (6 / 6) done.
The background-generator cloned now to your computer

```
MINGW64:/d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github$ cd Github
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github$ ls
CSS/ index.html Scope.js Script/
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github$ git clone https://github.com/antonykidis/background-generator.git
Cloning into 'background-generator'...
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), done.
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github$ ls
background-generator/ CSS/ index.html Scope.js Script/
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github$
```

We've just copied our repo to a local computer.
You can see the [background-generator](#) appeared in the left window

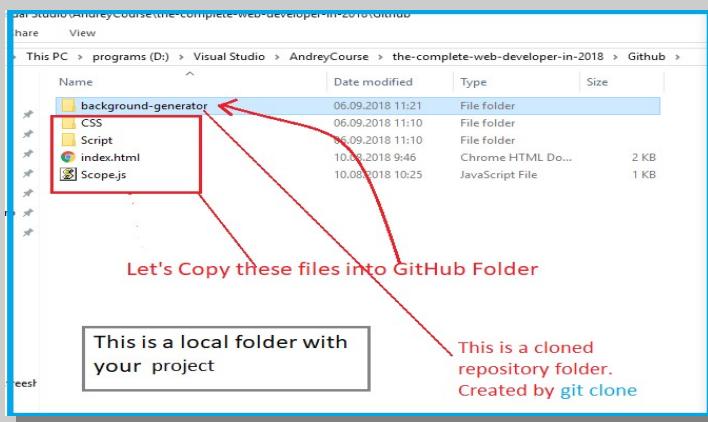


Step 5

Next Step is to take all of our local project's [files and folders](#), and copy them into the background-generator repository folder.
You can copy these files using the IDE, or a command line (terminal)

To copy files and folders via command line please refer to the terminal section of the course

Illustration



Step 6

1. Back to a terminal window and cd (navigate) to a newly created cloned directory named background-generator
Example:
`cd background-generator`
2. Then type `ls` (dir in windows) to list the contents of a background-generator folder.
3. You should see the similar output with all the copied files from the previous step.
(master) means that we now working in the repository folder

```
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github/background-generator (master)
$ ls
CSS/ index.html README.md Scope.js Script/
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github/background-generator (master)
$
```

This folder wil contain the project files plus the readme file

Step 7

- We now can check the status of our files since we've copied our files into the repository folder (communication folder)
- While in terminal type `git status`
- This will invoke the following output

You can see that we have untracked files and folders
We can now add these files to a GitHub **Configuration file** before upload it back to GitHub

```
MINGW64:/d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github/background-generator (master)
$ ls
CSS/ index.html README.md Scope.js Script/
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github/background-generator (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    CSS/
    Scope.js
    Script/
    index.html

nothing added to commit but untracked files present (use "git add" to track)
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github/background-generator (master)
$ |
```



Configuration file. We will explore it in the following sections.

Adding files and folders to configuration file

Step 8

- Let's Add untracked files to a special Configuration file

Type the following commands:

- git add CSS (folder)
- git add Script(folder)
- git add Scope.js(file)
- git add index.html (file)

To add all files and folders at once type **git add .**

```
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ ls
CSS/  index.html  README.md  Scope.js  Script/
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    • CSS/
      • Scope.js
      • Script/
        • index.html

nothing added to commit but untracked files present (use "git add" to track)

Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ |
```

Step 9

- After adding each file, and folder

1. Type **git status**

2. You should see the following output

3. Okay we've just set the configuration file. Great!

4. Next step is to commit these files into the configuration file.

```
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ ls
CSS/  index.html  README.md  Scope.js  Script/
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   CSS/main.css
    new file:   Scope.js
    new file:   Script/main.js
    new file:   index.html

Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ |
```

We added files
successfully
And our branch is up to date

Commit the changes we've made in previous steps

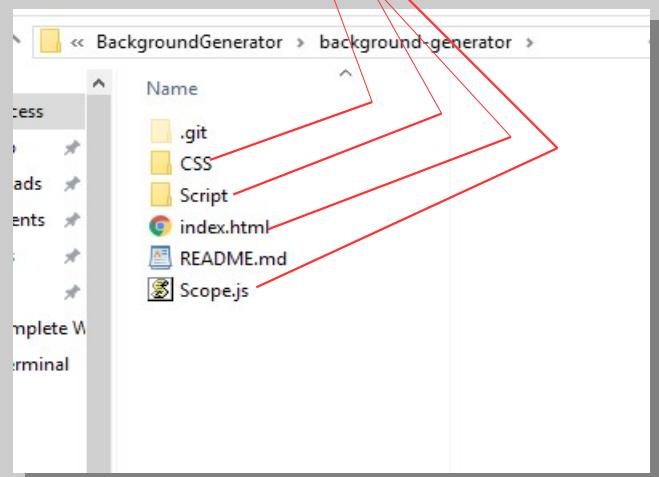
Step 10

- Let's commit previously added files and folders.
- Enter the following command
- git commit -m "adding starting project"

Commit means **Save all** the changes to config.file before pushing it back to the GitHub server

```
lma Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
loper-in-2018/Github/background-generator (master)
git commit -m "adding starting project. Say hi to marcy"
[master 31a4720] adding starting project. Say hi to marcy
 4 files changed, 82 insertions(+)
create mode 100644 CSS/main.css
create mode 100644 Scope.js
create mode 100644 Script/main.js
create mode 100644 index.html
```

```
lma Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
loper-in-2018/Github/background-generator (master)
```



We've just set up the configuration file

- The last step is to push these changes back to the GitHubServer. But let's explore the configuration file first.



Explore the configuration file



Name	Date modified	Type	Size
hooks	06.09.2018 11:19	File folder	
info	06.09.2018 11:19	File folder	
logs	06.09.2018 11:19	File folder	
objects	06.09.2018 13:51	File folder	
refs	06.09.2018 11:19	File folder	
COMMIT_EDITMSG	06.09.2018 13:51	File	1 KB
config	06.09.2018 11:19	File	1 KB
description	06.09.2018 11:19	File	1 KB
HEAD	06.09.2018 11:19	File	1 KB
index	06.09.2018 13:51	File	1 KB
packed-refs	06.09.2018 11:19	File	1 KB

```
MINGW64:/d/Visual Studio/AndreyCourse/the-complete-web-developer-in... - □ X
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ git add CSS
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ git add Script
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ ls
CSS/ index.html README.md Scope.js Script/
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:  CSS/main.css
    new file:  Scope.js
    new file:  Script/main.js
    new file:  index.html

Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ git commit -m "adding starting project. Say hi to marcy"
[master 31ad472] adding starting project. Say hi to marcy
 4 files changed, 82 insertions(+)
 create mode 100644 CSS/main.css
 create mode 100644 Scope.js
 create mode 100644 Script/main.js
 create mode 100644 index.html

Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-dev
eloper-in-2018/Github/background-generator (master)
$ :
```

DIRC

CSS/main.css

README.md

index.html

index.js

Think of it as a Configuration file
This file will be PUSHED to GitHub to make changes

index file = Configuration file

Push changes to GitHub

Step 11

- Finally Push the changes to the server
- Use the following command
- `git push`
- This will get you to the following output
- We've successfully wrote objects to [http://github.com...](http://github.com)

```
Dima Mironov@Dmitry-M MINGW64 /d/Visual Studio/AndreyCourse/the-complete-web-developer-in-2018/Github/background-generator (master)
$ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 1.77 KiB | 1.77 MiB/s, done.
Total 8 (delta 0), reused 0 (delta 0)
To https://github.com/antonykidis/background-generator.git
  b5d39e0..31a4720 master -> master
```

Great!
Now a background-generator is online

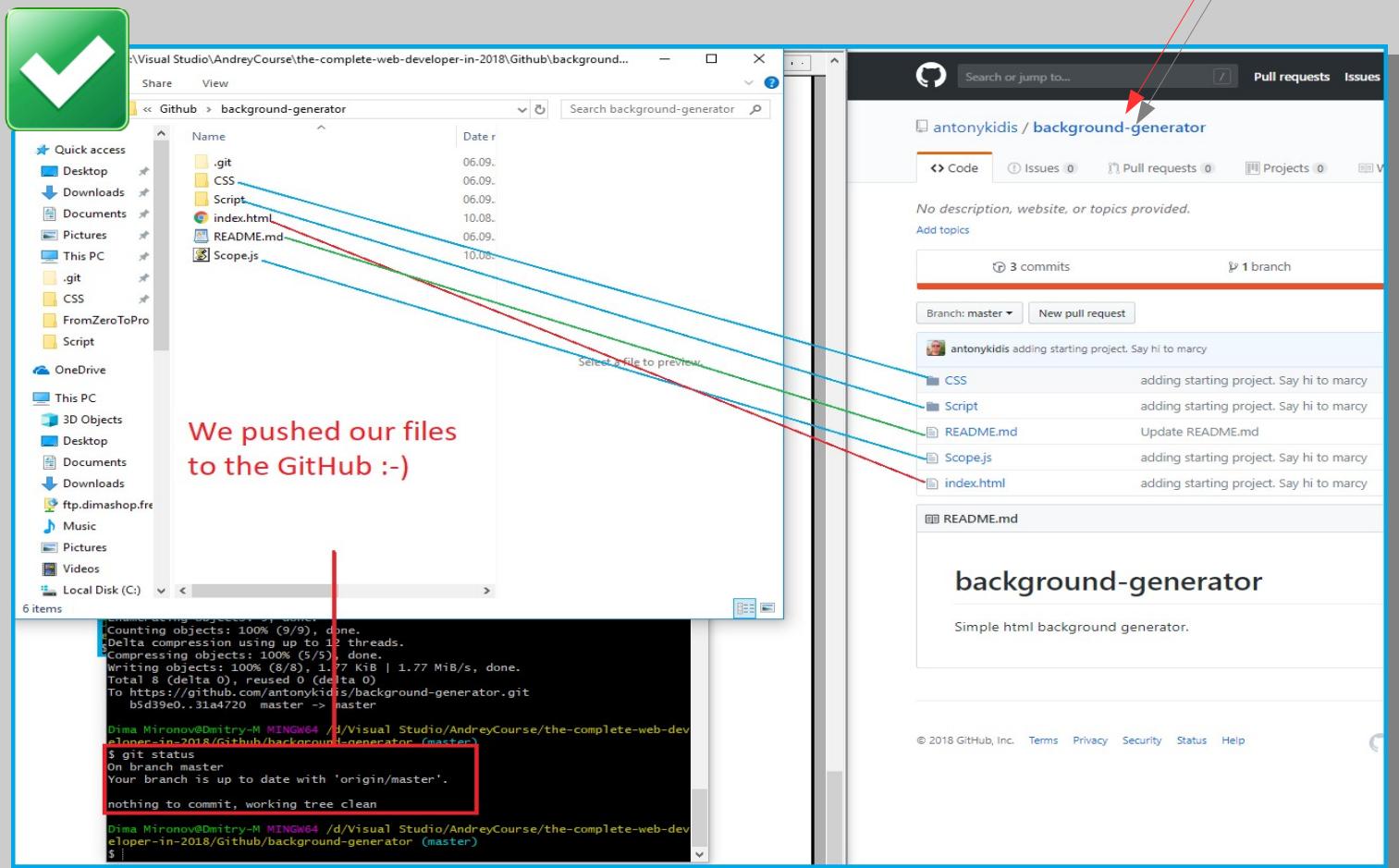
Step 12

- Go back to <https://github.com> and check what you've done so far
(Test pushed files on GitHub)



What we've done so far

We successfully added a new files into GitHub repository





Next Step - Work on Existing Repository

- Open Terminal(or cmd/bash/powrshell in windows)
- Navigate to the folder where you want to download a GitHub existing repository
- Copy repository's url from GitHub
- Type the following command (Example)
`git clone https://github.com/antonykidis/background-generator.git`
- Finally check if a cloned repository appears in the desired folder

we've just
cloned our repo to a local
computer

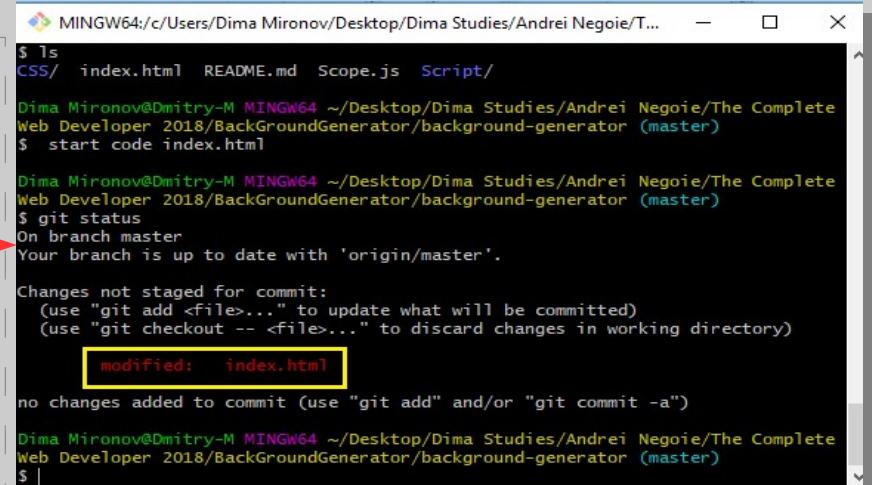
```
MINGW64:/c/Users/Dima Mironov/Desktop/test
$ cd test
Dima Mironov@Dmitry-M MINGW64 ~/Desktop/test
$ ls
index.html
Dima Mironov@Dmitry-M MINGW64 ~/Desktop/test
$ ls
background-generator/
Dima Mironov@Dmitry-M MINGW64 ~/Desktop/test
$ |
```

Next Step - Pushing a modified files back to a GitHub

1. For example:
open [index.html](#) and change the title<h1>here</h1>
to a cool generator
2. Save it
3. Comeback to a terminal window, and type
`git status`

You should see a similar output

4. Type `git add` (adding the index.html)
5. Type `git commit -m "title has been changed"`
(save changes)
6. Type `git push` (pushing the changes back to server)



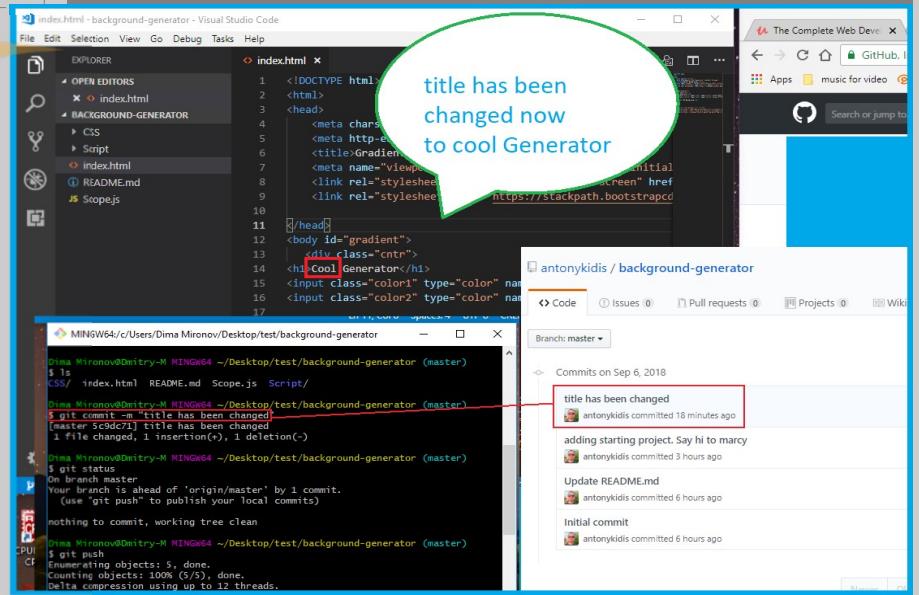
```
$ ls
$ CSS/  index.html  README.md  Scope.js  Script/
$ Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackGroundGenerator/background-generator (master)
$ start code index.html
$ Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackGroundGenerator/background-generator (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")

$ Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackGroundGenerator/background-generator (master)
$ |
```



title has been changed now to cool Generator

```
index.html - background-generator - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER index.html x
OPEN EDITORS x index.html
BACKGROUND-GENERATOR
CSS/ index.html
Script
index.html
README.md
Scope.js

MINGW64:/c/Users/Dima Mironov/Desktop/test/background-generator
$ ls
$ CSS/  index.html  README.md  Scope.js  Script/
$ Dima Mironov@Dmitry-M MINGW64 ~/Desktop/test/background-generator (master)
$ git commit -m "title has been changed"
[master 59d6c71] title has been changed
 1 file changed, 1 insertion(+), 1 deletion(-)
$ Dima Mironov@Dmitry-M MINGW64 ~/Desktop/test/background-generator (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads.
```

The GitHub repository page shows a commit history:

- Initial commit by antonykidis 6 hours ago
- Update README.md by antonykidis 3 hours ago
- adding starting project. Say hi to marcy by antonykidis 18 minutes ago
- title has been changed by antonykidis 18 minutes ago

7. Go to GitHub

4 commits

Navigate to commits link

And see the actual changes we've just made
As you can see the title has been changed to a Cool Generator

Showing 1 changed file with 1 addition and 1 deletion.

2 index.html

```
@@ -11,7 +11,7 @@
11   11   </head>
12   12   <body id="gradient">
13   13     <div class="cntr">
14 - 14   <h1>Background Generator</h1>
+ 14   <h1>Cool Generator</h1>
15   15     <input class="color1" type="color" name="color1" value="#00ff00">
16   16     <input class="color2" type="color" name="color2" value="#ff0000">
17   17
```

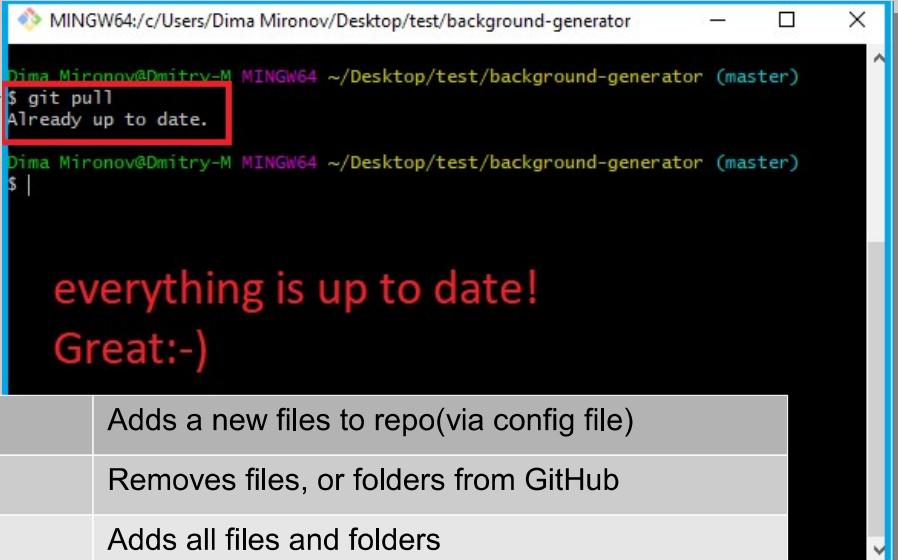
title has changed

0 comments on commit 5c9dc71

We can pull the latest version from GitHub

- Type `git pull`

Useful
commands



Dima Mironov@Dmitry-M MINGW64 ~/Desktop/test/background-generator (master)
\$ git pull
Already up to date.

Dima Mironov@Dmitry-M MINGW64 ~/Desktop/test/background-generator (master)
\$ |

**everything is up to date!
Great:-)**

git add <files, or folders>	Adds a new files to repo(via config file)
git rm <files or folders>	Removes files, or folders from GitHub
git add .	Adds all files and folders
git commit	Save changes to config file, and prepare pushing it to the server
git push	Pushes the new changes back to server, and updates the remote repository
git pull	Pulls the latest version of the repository from a GitHub
git status	Checks the overall status of repository
git branch -d branch_name or git branch -D branch_name	Removes the named branch
git revert dd61ab32	Removes the last commit mistakenly pushed to the GitHub By commit ID. So the commit id in this case is dd61ab32

git push origin --delete <branch_name>	Delete a remote GIT branch
git push <remote_name> :<branch_name> Remote name means this: git push origin :<branch_name>	Delete a remote GIT branch

The end of part 1
Please read the GitHub guide [part 2](#)

The complete Web Developer in 2018

Git and GitHub Guide

Part-2

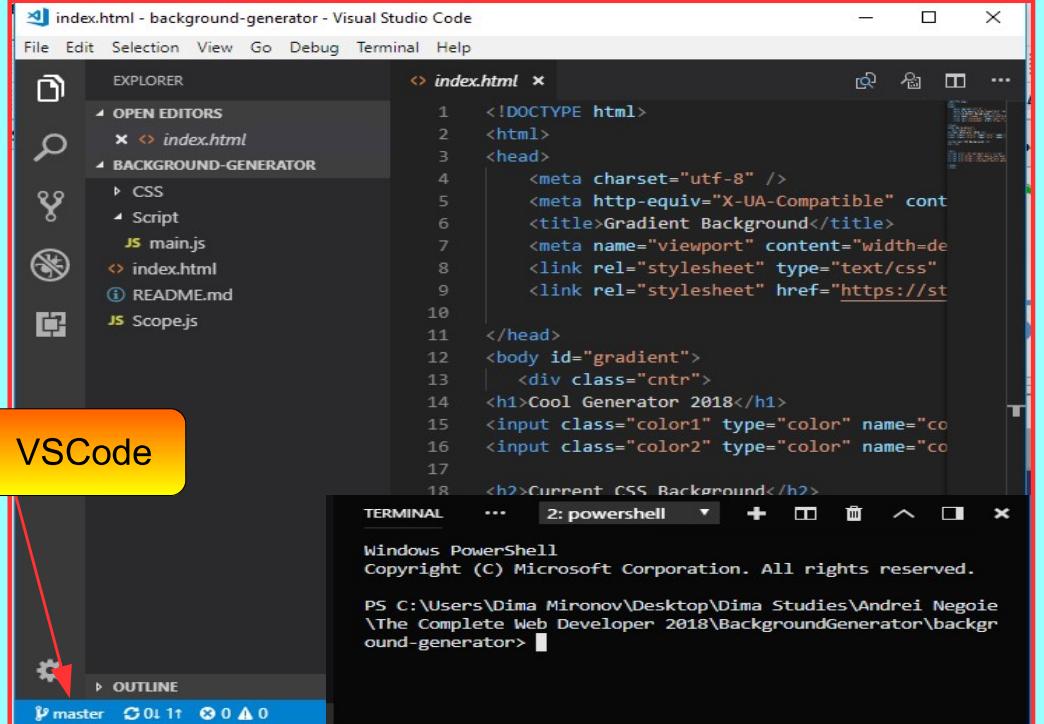
Branching

Info Page

For all the windows users.

You can use Gitbash terminal, or a Visual Studio Code terminal.
Download it here <https://git-scm.com/downloads>

GitBash for windows



VS Code displays a branch name.
However, It doesn't display a branch
name In the PowerShell terminal.

My suggestion is to work with the
Git Bash terminal.

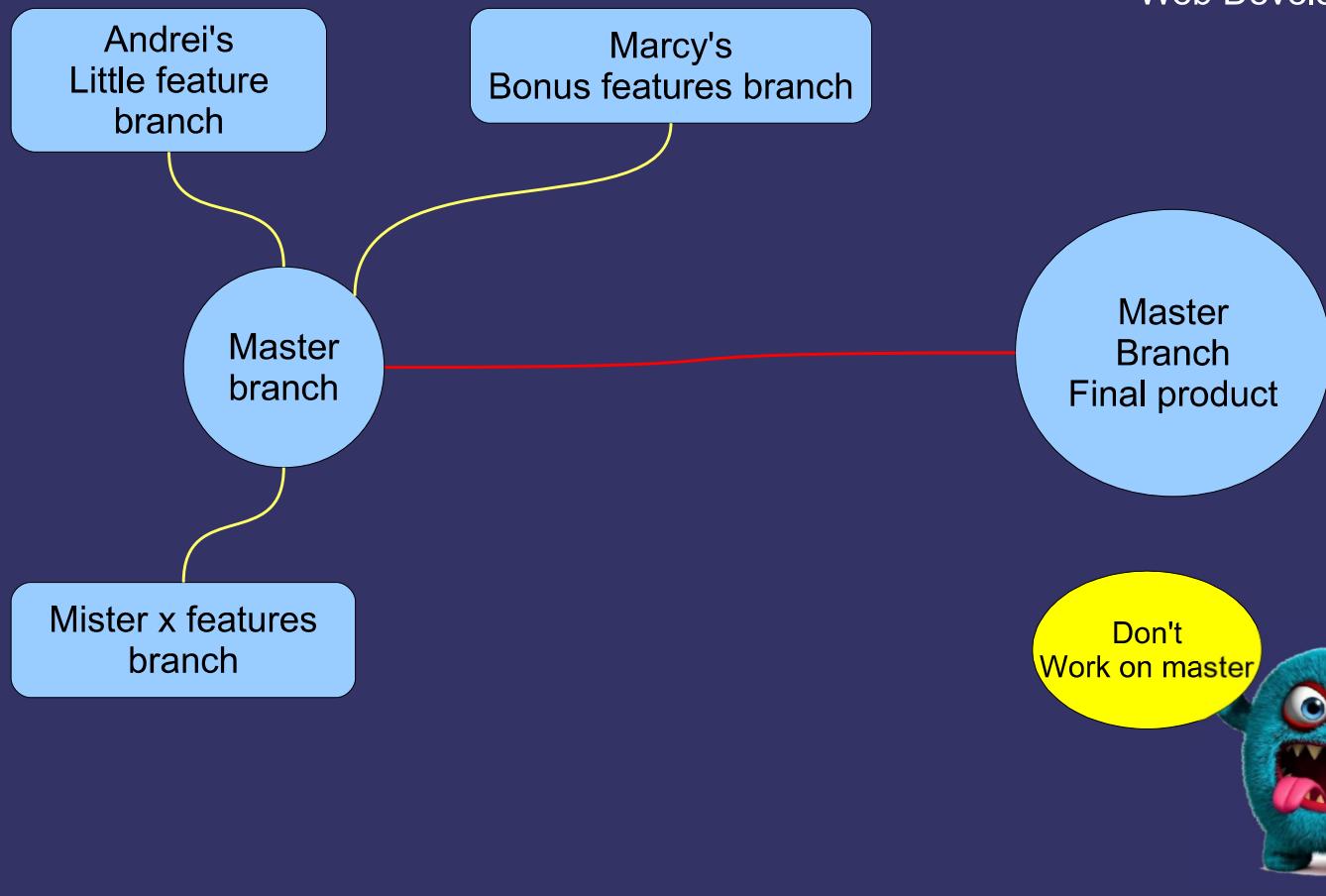
VSCode

GitBash for windows

Refers to Section 15, video #145
Git and GitHub Guide Part-2 Branching



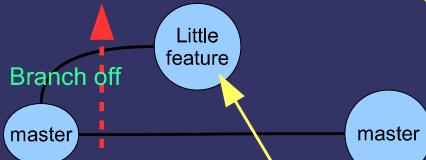
The complete
Web Developer in 2018



Create a new branch

Let's create a new branch (terminal)

1. Type `git branch littlefeature` (this will create a new branch)
2. Type `git branch` (will display all branches available)
3. Type `git checkout littlefeature` (Switch to a new branch)



* `master` means we currently on master branch.

Note:

If something went wrong and you need to delete the branch, type `git branch -d littlefeature`
The above command will delete the mistaken branch

```
Dima Mironov@Dmitry-M MINGW64 ~/DEsktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (master)
$ git branch littlefeature

Dima Mironov@Dmitry-M MINGW64 ~/DEsktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (master)
$ git branch
* master
  littlefeature

Dima Mironov@Dmitry-M MINGW64 ~/DEsktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (master)
$
```

4. We successfully switched to a new branch

```
Dima Mironov@Dmitry-M MINGW64 ~/DEsktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (master)
$ git checkout littlefeature
Switched to branch 'littlefeature'

Dima Mironov@Dmitry-M MINGW64 ~/DEsktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (littlefeature)
$
```

Let's modify index.html now

1. Open index.html (use IDE or terminal) Type: start code
index.html

2. Locate the <h2> tag and change the title to
<h2>This is the background</h2>

3. Save all, then do the following procedure...

4. git add index.html
5. git commit -m "changing title"
6. git push

If it doesn't work and you get an error

fatal: The current branch littlefeature has no upstream branch.

To push the current branch and set the remote as upstream,
use git push --set-upstream origin littlefeature

Fix.

Type git push --set-upstream origin littlefeature

After using the above command once,
you can then use a regular
git push command (without notifications)

Finally, you can cheat and type this
git push origin littlefeature

```
12  <body id="gradient">
13  |   <div class="cntr">
14  |     <h1>Cool Generator 2018</h1>
15  |     <input class="color1" type="color" name="color1" value="#00f
16  |     <input class="color2" type="color" name="color2" value="#ff8
17
18  |     <h2>This is the background</h2>
19  |     <h3></h3>
20
21 |
```

```
Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete
Web Developer 2018/BackgroundGenerator/background-generator (littlefeature)
$ git push
fatal: The current branch littlefeature has no upstream branch.
To push the current branch and set the remote as upstream, use

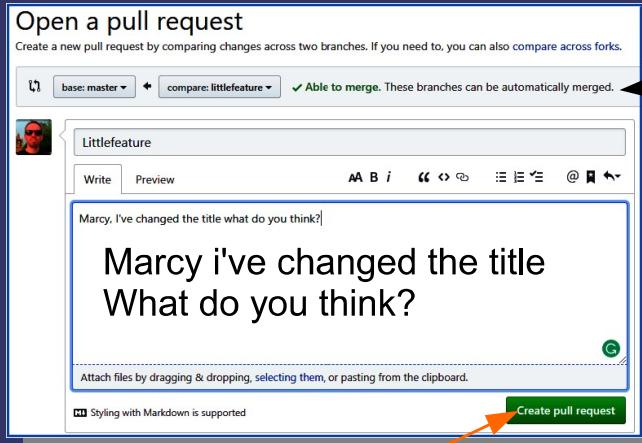
    git push --set-upstream origin littlefeature

Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete
Web Developer 2018/BackgroundGenerator/background-generator (littlefeature)
$ git push origin littlefeature
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 302 bytes | 302.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
```

Go Back to <https://github.com>

1. We now see a yellow notification
You recently pushed branches

2. If we click the green button
Compare & pull request
It will open the a pull request window



3. Enter a message, check the changes
And finally click the create pull request button

Note:

We've made a change to index.html
Marcy will decide approve it or not.



The GitHub page displays the following details:

- Repository: antonykidis / background-generator
- Code tab selected
- Issues: 0
- Pull requests: 0
- Projects: 0
- Wiki: 0
- Insights: 0
- Settings: 0
- No description, website, or topics provided.
- Manage topics
- 6 commits, 2 branches, 0 releases
- Your recently pushed branches: **littlefeature (less than a minute ago)**
- Branch: littlefeature
- New pull request
- Upload files, Find file, Clone or download
- This branch is 2 commits ahead of master.
- 16 hours ago, 9 days ago, 9 days ago, 9 days ago, 6 hours ago
- index.html diff showing 1 changed file with 2 additions and 2 deletions:
 - Line 14: - <h1>Cool Generator</h1> + <h1>Cool Generator 2018</h1>
 - Line 18: - <h2>Current CSS Background</h2> + <h2>This is the background</h2>

Okay now STOP!

Do you really understand what is going on here!?

Hey marcy I made a pull request

Do something about it

No...

Yes or Yes!?



Compare & pull request

Hey Marcy I made a pull request. Please look at my html file,
compare new changes to the previous one.

And if they good enough please **pull them, and merge them** into the master branch.

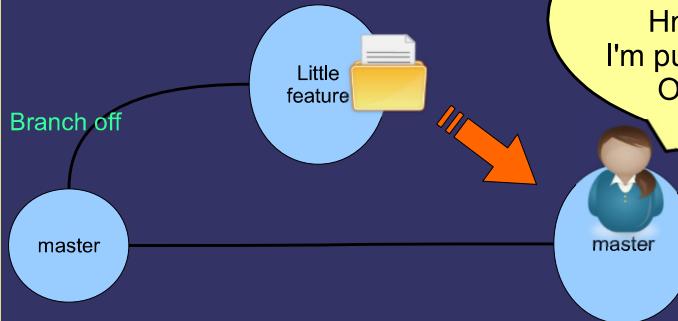
Marcy is an administrator and she will open your pushed files on a littlefeature branch.
She will compare them to the previous one, and decide whether to prove these changes or not

Important

At this point you as a developer
Just need to wait for Marcy's approval.
That's it.
Don't be confused by the rest of the
diagrams.

Let's see How Marcy handle my
request on her side as Administrator.

With this information in mind let's
Proceed to the next step.

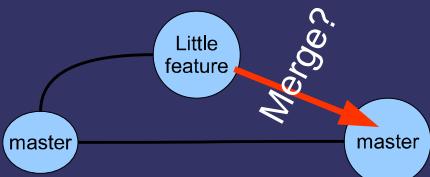


Hi Andrei!
Let's compare your changes
Hmm..okay looks good!
I'm pulling this into the master
Okay it's merged now!

We successfully created a pull request (PR)

As you can see antonykidis(Andrei) wants to merge 3 commits into **master** from **littlefeature**

Hey Marcy I did a little Change on little feature, is it okay for me to Merge it to master?



Let's see what Marcy will probably do..

The screenshot shows a GitHub pull request page for a repository named "background-generator". The title of the PR is "Littlefeature #1". A green "Open" button is visible. The description states: "antonykidis wants to merge 3 commits into master from littlefeature". Below this, the commit history is shown:

- antonykidis commented 4 minutes ago: "Marcy, I've changed the title what do you think?"
- antonykidis added some commits 19 hours ago:
 - update the title
 - changing title
 - changing title to background

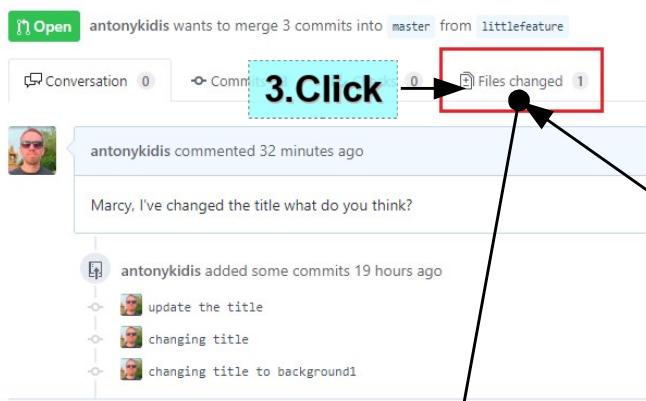
On the right side of the commit list, a callout box highlights "1 commit", "2 commit", and "3 commit". At the bottom of the PR page, there is a summary of CI status and mergeability:

- Continuous integration has not been set up (with a note: "Several apps are available to automatically catch bugs and enforce style").
- This branch has no conflicts with the base branch (with a note: "Merging can be performed automatically").

At the bottom right, there is a "Merge pull request" button and a note: "You can also open this in GitHub Desktop or view command line instructions".

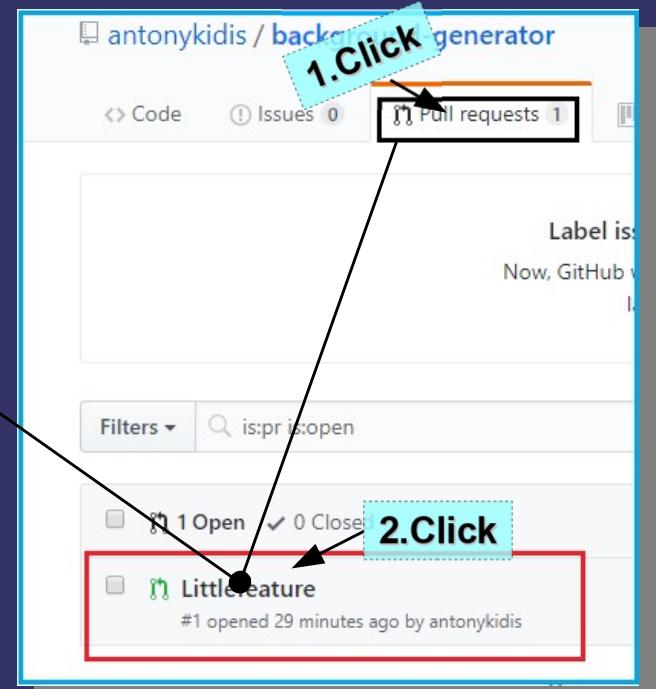
Marcy will Open-up a pull request link

Littlefeature #1



A screenshot of a GitHub diff for the file "index.html". It shows 2 additions and 2 deletions. The diff highlights a change in line 18 where the text "Current CSS Background" is deleted and "This is the background!" is added. A callout "Hover on plus sign" points to the '+' sign in the diff interface.

```
@@ -11,11 +11,11 @@
</head>
<body id="gradient">
  <div class="cntn">
    <h1>Cool Generator</h1>
    <h1>Cool Generator 2018</h1>
    <input class="color1" type="color" name="color1" value="#00ff00">
    <input class="color2" type="color" name="color2" value="#ff0000">
    <h2>Current CSS Background</h2>
-   <h2>This is the background!</h2>
+   <h3></h3>
```



The screenshot shows a code editor with a diff view and a review interface. The diff view highlights changes in the code:

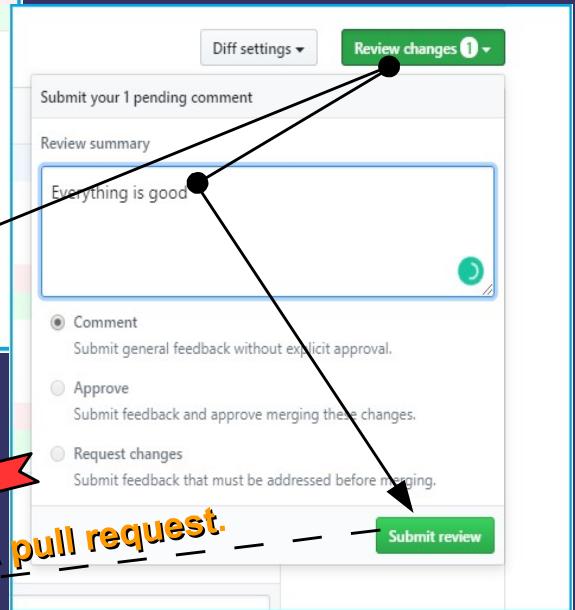
```
@@ -11,11 +11,11 @@
11   11   </head>
12   12   <body id="gradient">
13   13     <div class="cntn">
14 - 14     - <h1>Cool Generator</h1>
15 + 14     + <h1>Cool Generator 2018</h1>
16   15     <input class="color1" type="color" name="color1" value="#00ff00">
17   16     <input class="color2" type="color" name="color2" value="#ff0000">
18 - 17     - <h2>Current CSS Background</h2>
19 + 18     + <h2>This is the background</h2>
```

A red arrow points to the line with the new h2 tag. A blue button with a plus sign is highlighted, indicating it's being used to add a comment. The comment box contains the text "Looks good".

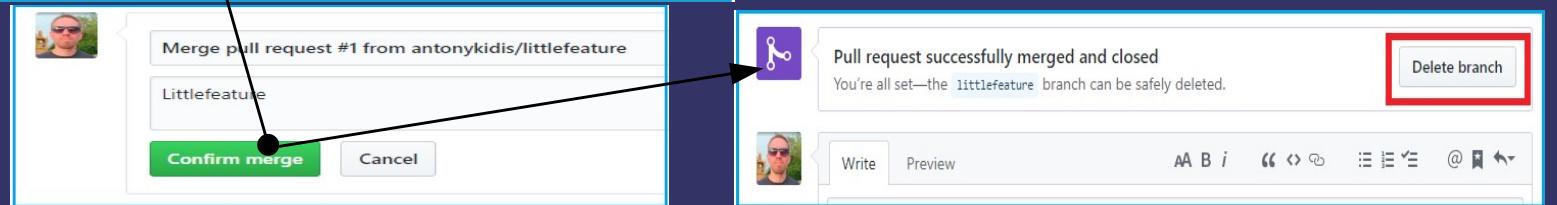
Below the code editor, there are tabs for "Write" and "Preview", and a toolbar with various icons. The bottom of the screen has a message bar and a footer.

Hey everything Looks good!
(Click a plus sign)

- (Click a plus sign)
 - 1. Strat a review, add some comments, submit the Review.
 - 2. Click Review changes, add a comment, and Finally click Submit review
 - 3. Click Merge pull request, then confirm merge.



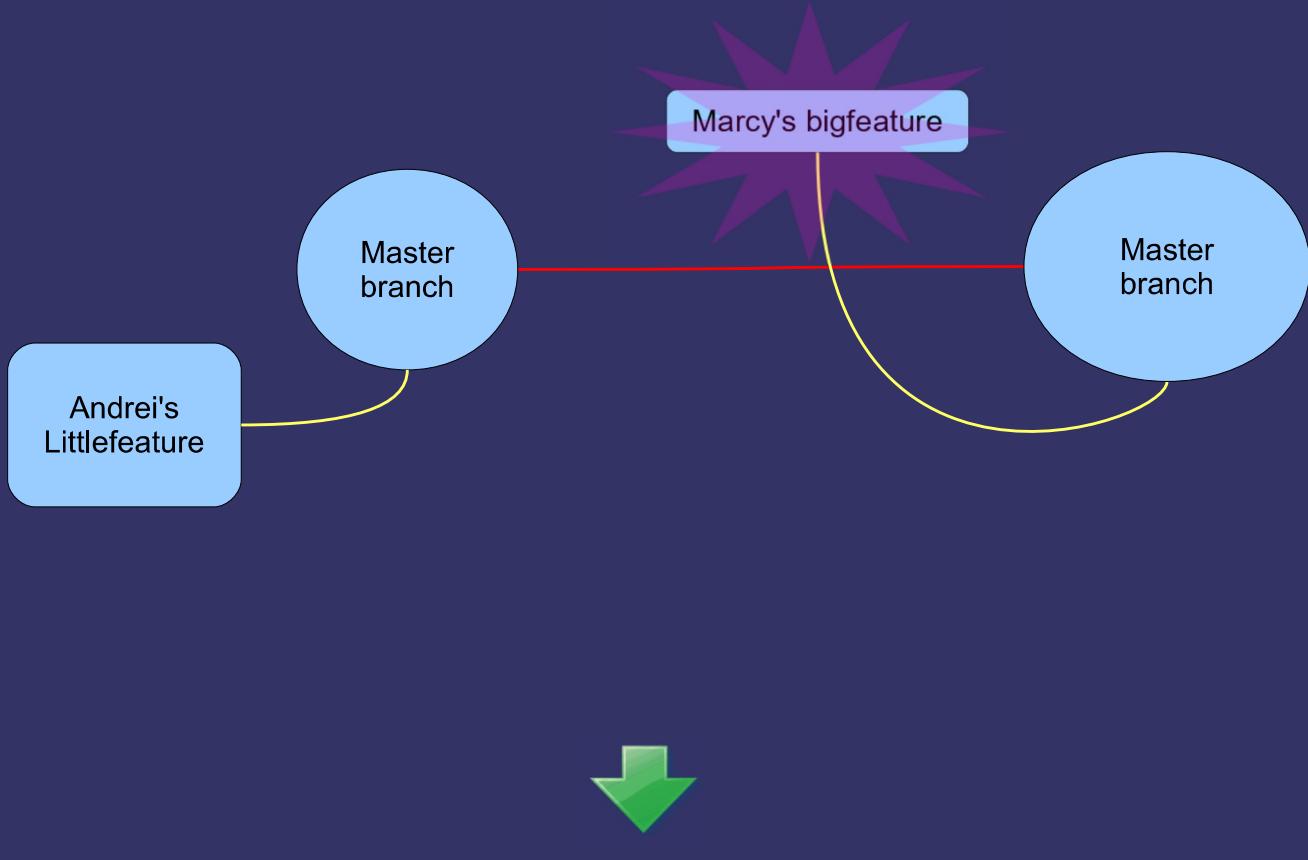
Pull request is now merged into the master.
Marcy can now delete littlefeature branch if she want.



Merge Conflicts 07:45 (time)



What if Marcy wants to create her own feature!?

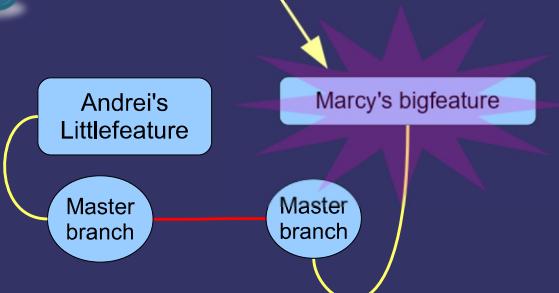




I want to create my own „bigfeature“ branch.

Marcy will create a new branch like this

Step 1



- 1 git branch bigfeature
- 2 git checkout bigfeature

```
Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (littlefeature)
$ git branch bigfeature

Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (littlefeature)
$ git checkout bigfeature
Switched to branch 'bigfeature'

Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/BackgroundGenerator/background-generator (bigfeature)
$ |
```

Step 2

Marcy changed the title back to the Background generator

```
<div class="cntr">
<h1>Background Generator</h1>
<input class="color1" type="color"
<input class="color2" type="color"
```

Step 3 She will then do a regular procedure, to save the changes.

- 1 git add index.html
- 2 git commit -m "reverting back to old title"
- 3 git push

If something went wrong, and you want to delete the branch, you can use The following commands. And start all over again.

git branch -d branch_name
git branch -D branch_name



Now go back to a GitHub. We again see The pull request. **Don't compare and pull request This time!**

Before comparing and pulling a request Marcy decided to add an exclamation mark! to the title

Marcy adds exclamation mark

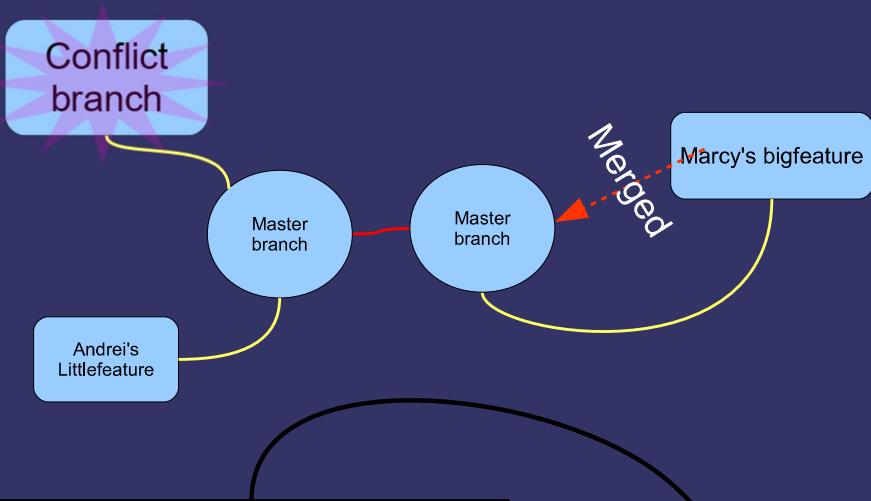
She then Save all...and follow the below steps

1	git add index.html
2	git commit -m "adding exclamation mark"
3	git push

Let's go back to GitHub.
So now we have 2 commits
1. is the **background generator**
2. and the **background generator!** with the exclamation mark

1. Click **Compare and pull request** button,
2. Add a comment.
3. Click **Create Pull Request** button.
4. **Merge pull request** button.
5. Finally click **Confirm merge** button.

Marcy just merged a pull request from a bigfeature branch. Meanwhile, Andrei decided to create a new branch too



He will then go back to a terminal
And enter the following commands:

5. `git add index.html`
6. `git commit -m“deleting title”`
7. `git checkout master`
8. `git pull`
9. `git checkout conflict`
10. `.merge master into the conflict branch.`
11. type `git merge master`



In the terminal window, Andrei wil enter the following commands:
1. `git branch conflict`
2. `git checkout conflict`

- 3 Andrei decided to completely remove the background generator title
4. save all

The terminal window shows the following output:

```
<body id="gradient">
  <div class="crr">
    <h1>Background Generator!</h1>
    <input class="color1" type="color" name="color1">
    <input class="color2" type="color" name="color2">
  </div>
</body>
</html>
```

Index.html

```
Web Developer 2018/BackgroundGenerator/background-generator (conflict)
$ git merge master
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

Conflict
branch

merging

littlefeature

6. By now you should receive the following error
7. If you Open a (Visual Studio Code/Sublime)
You could see the the following output:

```
Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/backgroundGenerator/background-generator (conflict)
$ git merge master
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

```
Dima Mironov@Dmitry-M MINGW64 ~/Desktop/Dima Studies/Andrei Negoie/The Complete Web Developer 2018/backgroundGenerator/background-generator (conflict|MERGING)
$
```

Important to read!

Let's understand the problem.



1. We Made a conflict branch.
 2. Then we removed the title from HTML.
 3. Then Andrei did `git add .`. Then `git commit -m „deleting title“`
- But before pushing these changes back to a GitHub, he decided to Merge whatever in the master, into his conflict branch. Yes, he could push the title-less index.html back to a GitHub. But he came up with another idea: He said let's see what is in master these days....
- Let's see what is inside the master now. Let's pull it into the conflict just to be updated. Yes sure, he didn't know that master is now different from a conflict, because he did not pulled nothing for a while (just an example)
4. So we got back into the master branch.
 5. We then pulled master's **current version** (to a local computer)
 6. Then we merged the master into the conflict. Just to be sure the conflict is synchronized with the master. But we have invoked an error. Because master contains a title, but a conflict branch doesn't have a title.
- So we get an intuitive user friendly notification from VS code or sublime. Saying that there is a changes to be made, and your files different from master.
- So!...Where is the catch? Why we all need this???** you say...
- Okay, You worked on some branch, so once in a while you want to update Your branch with a latest changes. So it is easier to merge a master into Your **OWN** branch, rather than download files manually, copy all of its contents, then Paste all of these into your own branch. **You see the difference???** :-)
- What if a master branch contains thousands of files? You will get nuts until you copy them one by one into your branch. So We simply merging the master, and it does all the dirty work for us. without worrying we missed something.

```
1 </head>
2 <body id="gradient">
3   <div class="cntr">
4     Accept Current Change | Accept Incoming Change | Accept Both Changes | Co
4     <<<<< HEAD (Current Change)
5     =====
6     <h1>Background Generator!</h1>
7     >>>>> master (Incoming Change)
8     <input class="color1" type="color" name="color1" value="#000000">
9     <input class="color2" type="color" name="color2" value="#ff0000">
0     <input class="color2" type="color" name="color2" value="#ffff00">
1
```





```
1 </head>
2 <body id="gradient">
3   <div class="cntr">
4     Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
5     <<<<< HEAD (Current Change)
6     =====
7     >>>>> master (Incoming Change)
8   <h1>Background Generator!</h1>
9   <input class="color1" type="color" name="color1" value="#00ff00">
10  <input class="color2" type="color" name="color2" value="#ffff00">
11  <input class="color2" type="color" name="color2" value="#ff0000">
12
```

<<<Head This is conflict branch

>>>master This is master branch

I don't know you

I don't know you

conflict

master

Andrei's Littlefeature

Marcy's bigfeature

Master branch

Master branch

Conflict branch

Merging master
into conflict

Master has a title
<h1>background Generator!</h1>

Conflict has NO title

Ask Marcy first if she want to leave the title.
If yes, we can accept the Incoming changes by
clicking the Accept incoming link,Or delete
manually the following:

<<<<<< Head Current change

=====

>>>>>> Master Incoming change

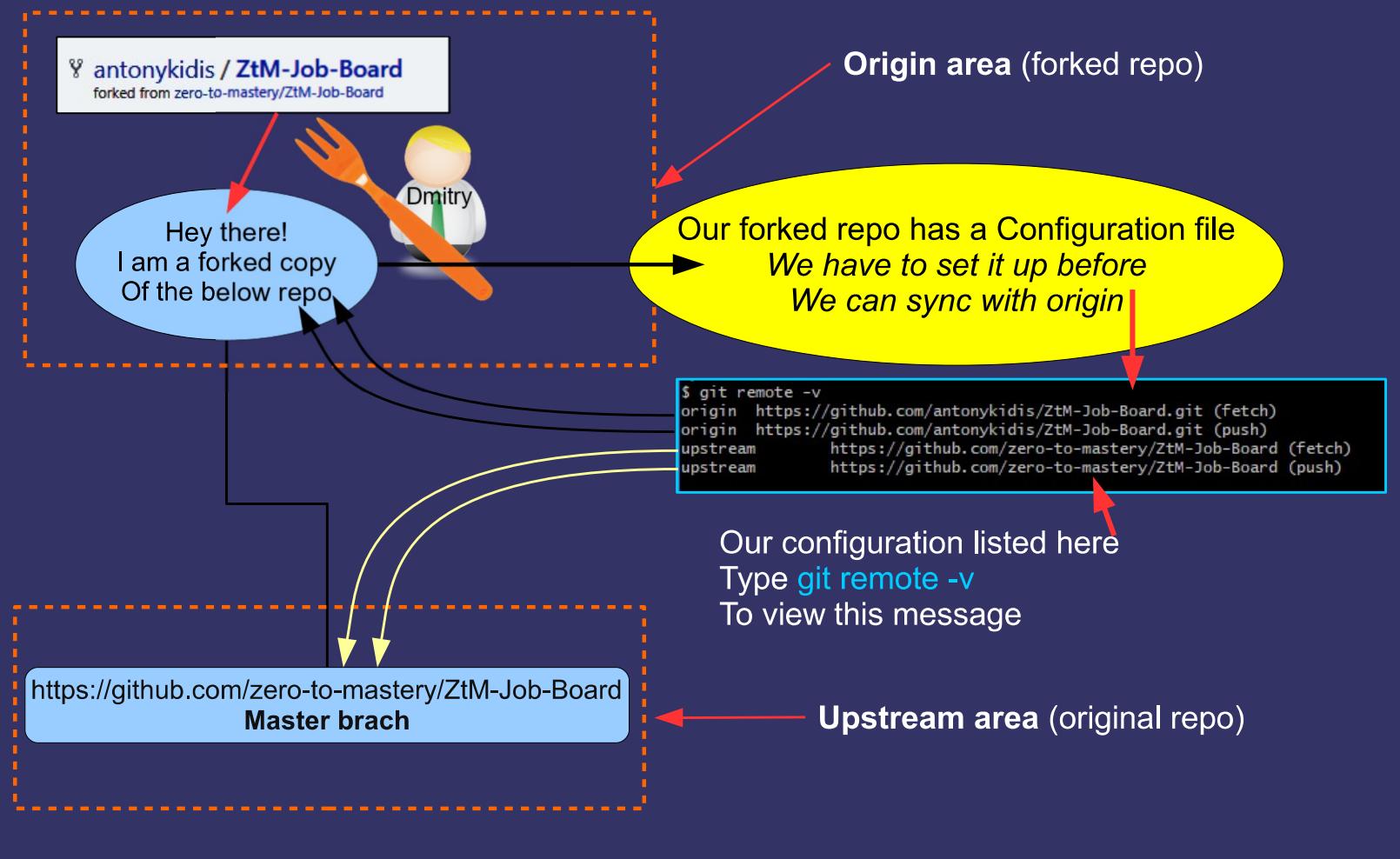
Leaving the
<h1>background Generator</h1>
In place
If you edit the file manually

Finally use the following commands

1	git add index.html
2	git commit -m "Okay we leaving the title as is"
3	git push

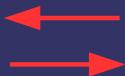
In the next section We'll see
how to keep your fork up to date

keep your fork up to date



#Step1 Configuring a remote for a fork

Fetch == pull
Push == push



MAC | WINDOWS | LINUX

You must configure a remote that points to the upstream repository in Git to sync changes you make in a fork with the original repository. This also allows you to sync changes made in the original repository with the fork.



1 Open Git Bash.

2 List the current configured remote repository for your fork.

```
$ git remote -v
origin  https://github.com/YOUR_USERNAME/YOUR_FORK.git (fetch)
origin  https://github.com/YOUR_USERNAME/YOUR_FORK.git (push)
```

3 Specify a new remote *upstream* repository that will be synced with the fork.

```
$ git remote add upstream https://github.com/ORIGINAL_OWNER/ORIGINAL_REPOSITORY
```

4 Verify the new upstream repository you've specified for your fork.

```
$ git remote -v
origin  https://github.com/YOUR_USERNAME/YOUR_FORK.git (fetch)
origin  https://github.com/YOUR_USERNAME/YOUR_FORK.git (push)
upstream  https://github.com/ORIGINAL_OWNER/ORIGINAL_REPOSITORY.git (fetch)
upstream  https://github.com/ORIGINAL_OWNER/ORIGINAL_REPOSITORY.git (push)
```

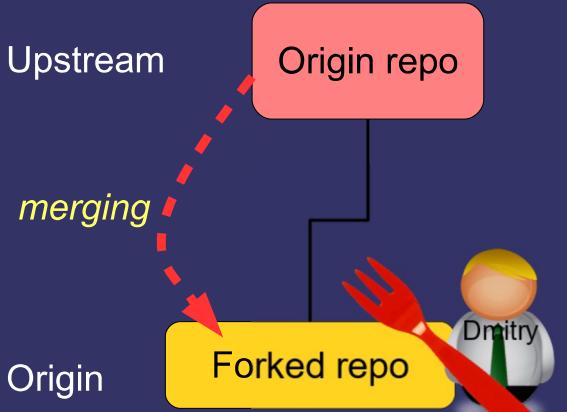
We all set up now

But we still have to pull
From upstream, and
Merge with origin.



step 2

Let's Pull, and merge



Done!
We now synced with the original
repository

- 1 Open Git Bash.
- 2 Change the current working directory to your local project.
- 3 Fetch the branches and their respective commits from the upstream repository. Commits to `master` will be stored in a local branch, `upstream/master`.

```
$ git fetch upstream
remote: Counting objects: 75, done.
remote: Compressing objects: 100% (53/53), done.
remote: Total 62 (delta 27), reused 44 (delta 9)
Unpacking objects: 100% (62/62), done.
From https://github.com/ORIGINAL_OWNER/ORIGINAL_REPOSITORY
 * [new branch]      master    -> upstream/master
```
- 4 Check out your fork's local `master` branch.

```
$ git checkout master
Switched to branch 'master'
```
- 5 Merge the changes from `upstream/master` into your local `master` branch. This brings your fork's `master` branch into sync with the upstream repository, without losing your local changes.

```
$ git merge upstream/master
Updating a422352..5fdff0f
Fast-forward
 README           |   9 -----
 README.md        |   7 ++++++
 2 files changed, 7 insertions(+), 9 deletions(-)
 delete mode 100644 README
 create mode 100644 README.md
```

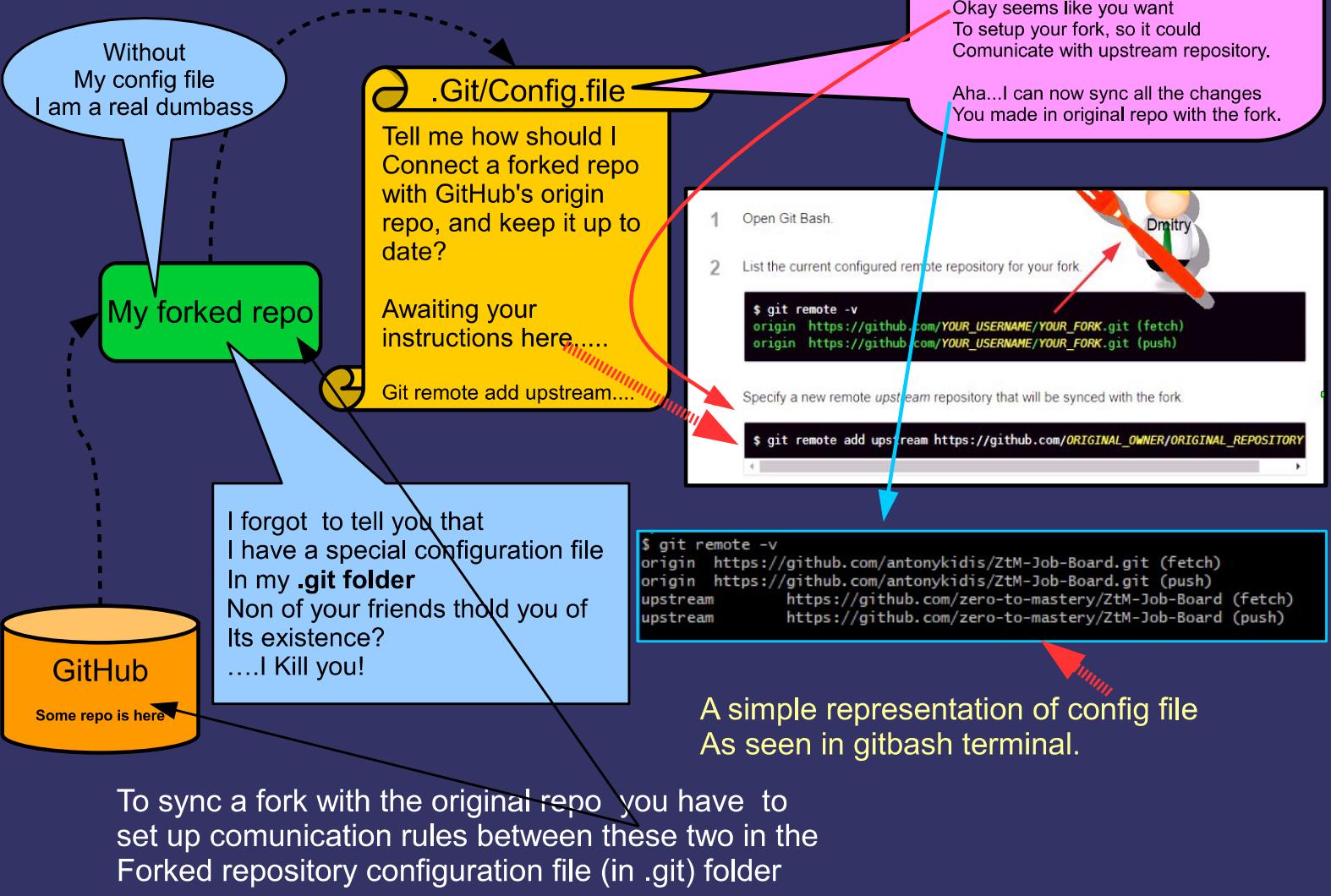
If your local branch didn't have any unique commits, Git will instead perform a "fast-forward":

```
$ git merge upstream/master
Updating 34e91da..16c56ad
Fast-forward
 README.md          |   5 +++--
 1 file changed, 3 insertions(+), 2 deletions(-)
```

What the hell Was that?

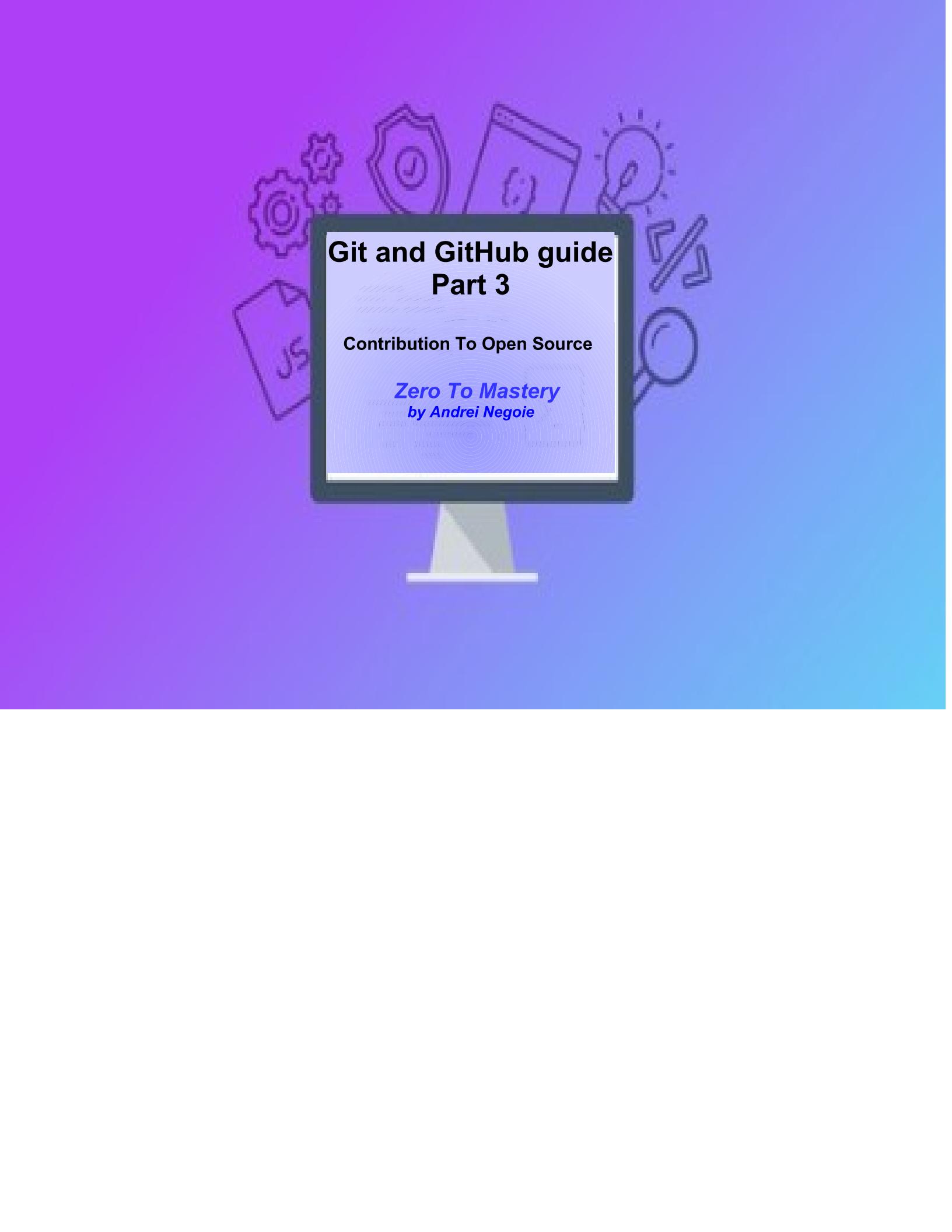
Okay let's try „explanation for dummies“

As I had a real hard time to understand this the easy way





***TO BE CONTINUED
IN PART 3 (Contribute to the open source)***



Git and GitHub guide

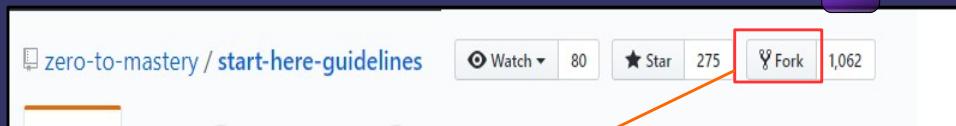
Part 3

Contribution To Open Source

Zero To Mastery
by Andrei Negoie

Lecture 148. We start from 05:41 minutes (follow these steps)

1



3. git clone <https://github.com/antonykidis/start-here-guidelines.git>

4. Cd into start-here-guidelines.

5. ls

CODE_OF_CONDUCT.md CONTRIBUTORs.md 'Get Started.md' LICENSE README.md

6. Type **start code**. (for VS Code should open the entire folder)

7. Branch out git checkout -b test

8. Add your name to the list of contributors

9. Save all CTRL+S

10. git add .

11. git commit -m "test"

12. git push origin test

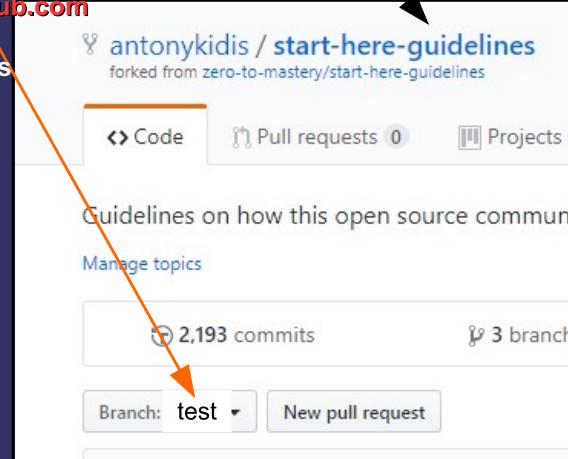
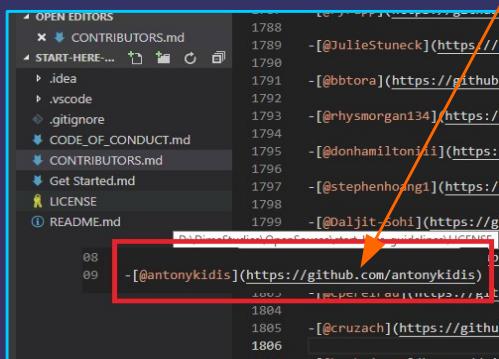
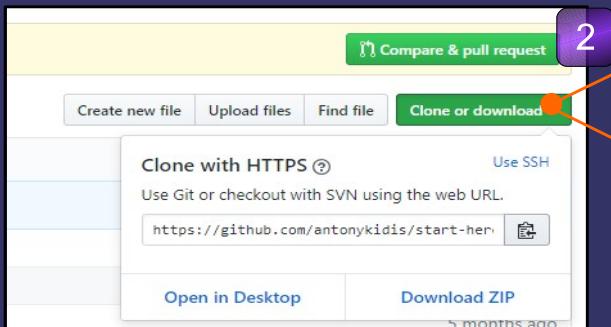
13. git push

14. go back to <https://github.com>

15. See a notification

16. Finish a pull request as appears in the video.

2



A different way adding yourself to CONTRIBUTORS.MD

We could skip the previous page and do the following

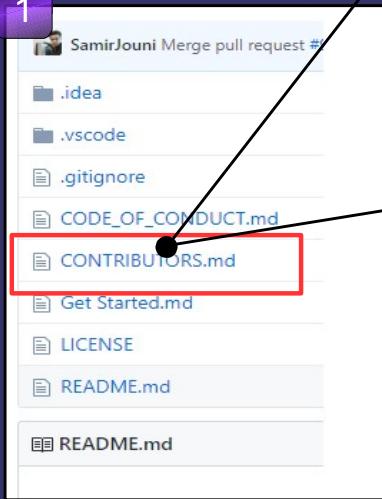
I've decided
to add myself this way



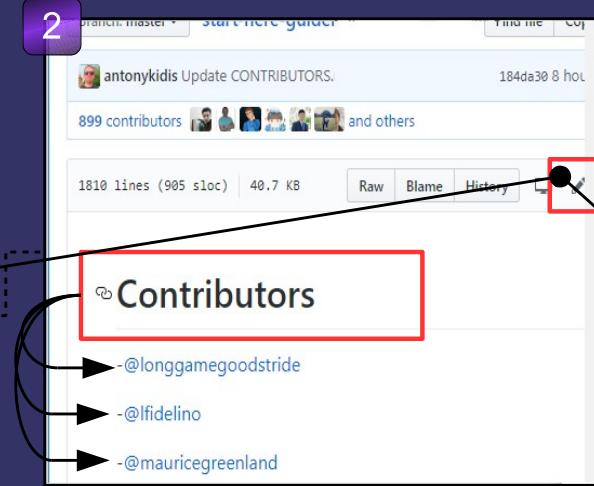
* Another way adding yourself to a CONTRIBUTORS.md list would be editing it straight on a start-here-guidelines GitHub repository.

1. Go to the forked repo <https://github.com/antonykidis/start-here-guidelines>
2. At your left hand side click the CONTRIBUTORS.md file.
3. Click Edit (pencil icon)
4. Add your name to the very end of the contributors list (using the mark down syntax)
5. Commit changes.

1



2



3

-[@rhysmorgan134](https://github.com/rhysmorgan134)
-[@donhamiltonii](https://github.com/donhamiltonii)
-[@stephenhoang1](https://github.com/stephenhoang1)
-[@Daljit-Sohi](https://github.com/Daljit-Sohi)
-[@ryub3n](https://github.com/ryub3n)
-[@cpereirau](https://github.com/cpereirau)
-[@cruzach](https://github.com/cruzach)
-[@hmahajan99](https://github.com/hmahajan99)
-[@antonykidis](https://github.com/antonykidis)

Commit changes

Update CONTRIBUTORS.md

Add an optional extended description...

Commit directly to the `master` branch.

Create a new branch for this commit and start a pull request

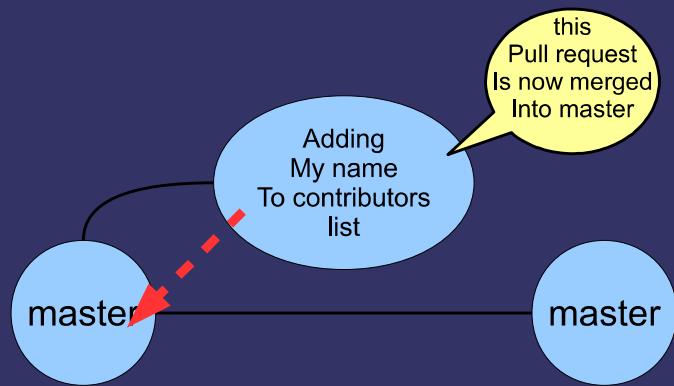
Commit changes **Cancel**

6. You will receive a new Pull Request notification.
7. Finish this procedure by clicking Compare and Pull request button.
8. Administrators will accept this pull request and merge it into master.

Congratulations on adding yourself as a new contributor!

*...Meanwhile somewhere in the cosmos
One of the administrators have merged
our pull request into the master.*

We've been added to a CONTRIBUTORS.md list.





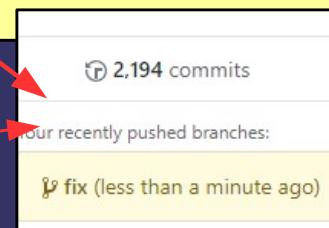
After adding myself to a contributors list via GitHub website, I finally decided to repeat these steps, but this time by cloning a repository, adding myself to the Contributors list, and pushing it back to a GitHub (as it appears in the original video)

I can say that I've edited the contributor's list two times so far.

1. By editing it on a GitHub website.
2. By cloning a repo, modifying a contributors list, and pushing it back to a GitHub
3. This caused the notification to appear once again.

Okay I have a new pull request notification.
But I don't need it because my name is already in the contributors' list
Hmmm...

Is there any way to get rid of it?
It's a good question...

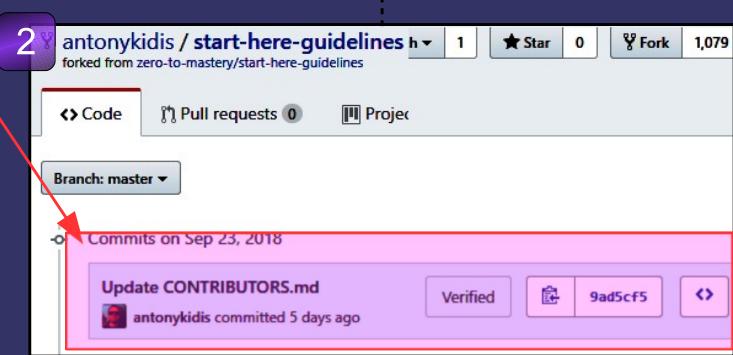
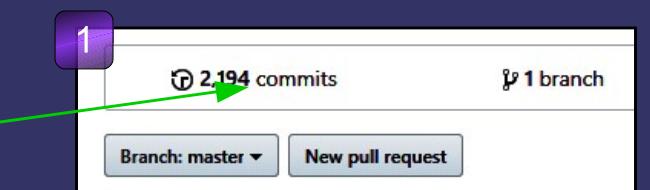
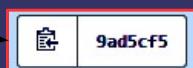


We could try to delete the commit like this

1. Go to a forked repo <https://github.com/zero-to-mastery/start-here-guidelines>
2. Click commits link.
2. Locate your commit.
3. Copy the commit ID
4. Go to terminal.
5. type `git push origin +9ad5cf5^:master`

+ Specify what destination ref to update with what source object.
The format of a <refspec> parameter is an optional plus +, followed by the source object <src>, followed by a colon :, followed by the destination ref <dst>

^A HEAD^A means the first parent of the tip of the current branch.



6. You will get the following output
7. Go back to commits (GitHub) and check if it's been deleted.
8. Repeat the previous steps again to delete another desired commits you want.

9. Once every commits is been removed, the pull request notification will no longer appear on the GitHub website.

10. You can then delete a remote branch like this:

11. Type `git push origin --delete fix`

In my case `fix` was the name of the branch

As you can see I've successfully deleted commits and a remote branches.
We now left with only one commit, and one master branch on our forked repository.

The screenshot shows the GitHub repository 'antonykidis / start-here-guidelines'. It has 2,194 commits and 1 branch. A red circle highlights the 'Branch: master' dropdown menu, and a red box highlights the '1 branch' count. A red arrow points from the 'Branch: master' dropdown to the GitHub interface below.

```
Dima Mironov@Dmitry-M MINGW64 /d/DimaStudies/OpenSource/start-here-guidelines (fix)
$ git push origin +19fe22b^:fix
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/antonykidis/start-here-guidelines.git
+ 19fe22b...e79238a 19fe22b^ -> fix (forced update)

Dima Mironov@Dmitry-M MINGW64 /d/DimaStudies/OpenSource/start-here-guidelines (fix)
```

The screenshot shows the GitHub repository 'antonykidis / start-here-guidelines' with the 'Branch: fix' selected. The commit history on Sep 23, 2018, shows several commits, but the commit on Sep 22, 2018, titled 'playing with branches' is missing. A red arrow points from the 'fix' branch selection to the commit history.

Commit Date	Commit Message	Author	SHA	Actions
Sep 23, 2018	Revert "removing my name"	antonykidis	ad76cbb	View Edit Copy
Sep 23, 2018	removing my name	antonykidis	0536746	View Edit Copy
Sep 23, 2018	adding and deleting the commit	antonykidis	ae62ef5	View Edit Copy
Sep 23, 2018	Revert "adding back my name"	antonykidis	964e559	View Edit Copy
Sep 23, 2018	adding back my name	antonykidis	de4dd71	View Edit Copy
Sep 23, 2018	Update CONTRIBUTORS.md	antonykidis	339b6e3	View Edit Copy
Sep 23, 2018	Revert "playing with branches"	antonykidis	b268c40	View Edit Copy
Sep 23, 2018	playing with branches	antonykidis	b9c51f6	View Edit Copy
Sep 22, 2018	(no commit message)			

Additional Information

You could encounter with some unexpected things while working on the Git and GitHub section of the course.

They are as follows:

1. What if we mistakenly entered a bad commit's message. How to fix that without pushing a whole new files into the GitHub again?
2. Your bash terminal showed a weird vim editor window (only for win. Users) How to use it?

Solution: How to Edit a bad commit message.

1. Go to a forked repository, open commits link, and find a commit you want to fix.
2. Copy a commit's ID to a clipboard
3. Open terminal window.
4. Use this command. `git revert fd8f4fe`
5. hit ENTER.
6. The vim editor window will appear.
7. The yellow text will represent a commit message.
8. Hit **insert** button on your keyboard. This will allow you to manipulate a text.(use arrows to navigate)
9. Delete the yellow text, and write `hello world` (for example)
10. Press ESC button to exit the text editor mode.
11. Hold down **SHIFT+I**: this will bring you to the vim script mode(check the left bottom corner, you will put commands there)
12. Type `wq` (w= write) (q=quit saving changes) and hit **ENTER**
13. You will receive the following output, with updated commit message
14. type `git push` to apply changes

The image consists of several windows illustrating the workflow:

- Top Right:** A GitHub commit history for "Commits on Sep 27, 2018". It shows a commit titled "reverting the title into master" by "antonykidis committed a day ago".
- Middle Left:** A terminal window showing the command `git revert fd8f4fe` being run. The output shows the commit message "reverting the title into master" is being reverted.
- Middle Center:** A vim editor window with the text "SHIFT+:". The status bar at the bottom shows the command `<cr>or background-generator/.git/COMMIT_EDITMSG :wq`.
- Middle Right:** A terminal window showing the result of the vim session. The commit message has been changed to "My new commit message is Hello World :-)".
- Bottom Right:** A GitHub commit history for "Commits on Sep 27, 2018". It shows a new commit titled "My new commit message is Hello World :-)" by "antonykidis committed 31 minutes ago".

A red dashed arrow points from the GitHub commit history at the top right to the vim editor window in the middle center. A blue arrow points from the vim editor window to the GitHub commit history at the bottom right. A red box highlights the commit message "My new commit message is Hello World :-)" in the terminal output of the vim session. A red box also highlights the commit message in the GitHub commit history at the bottom right. A red arrow points from the vim editor window to the GitHub commit history at the bottom right.

Step 14. Go back to GitHub and see the changes

antonykidis / **background-generator** Unwatch 1 Star

[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#) [Settings](#)

My new commit message is Hello World :-):
This reverts commit `fd8f4fe`.

master antonykidis committed 38 minutes ago 1 parent `fd8f4fe` commit `9f57c63d3bc36e11a98b1`

Showing 1 changed file with 6 additions and 0 deletions.

index.html

```
diff --git a/index.html b/index.html
@@ -11,7 +11,13 @@
 11   11     </head>
 12   12     <body id="gradient">
 13   13       <div class="cntr">
 14 +     <<<<< HEAD
 15 +     <<<<< HEAD
 16 +     + =====
 17       <h1>Background Generator!</h1>
 18 +     + >>>> master
 19 +     + =====
 20 +     + >>>>> 17720583859bd152a1731c8c054a77840d9ac9f8
 21       <input class="color1" type="color" name="color1" value="#00ff00">
 22       <input class="color2" type="color" name="color2" value="#ff0000">
 23       <input class="color2" type="color" name="color2" value="#ff0000">
```

We've just successfully changed the commit's message.
But the files are left the same.
Great. It works.



