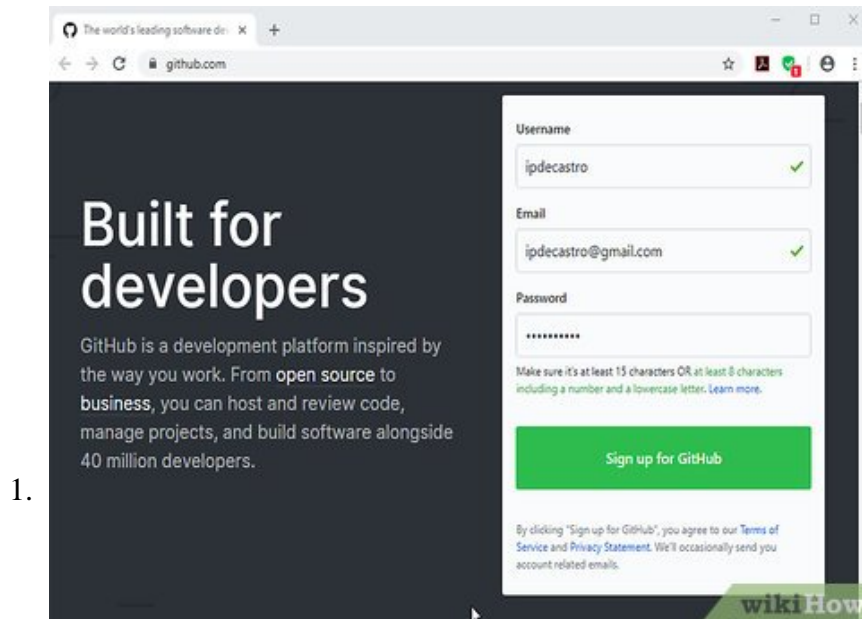


# How to Set Up and Use Git

Git is one of the most widely used version control systems for software development. Built by Linus Torvalds in 2005, Git focuses on speed, data integrity, and support for distributed, non-linear workflows. With its widespread use even for...

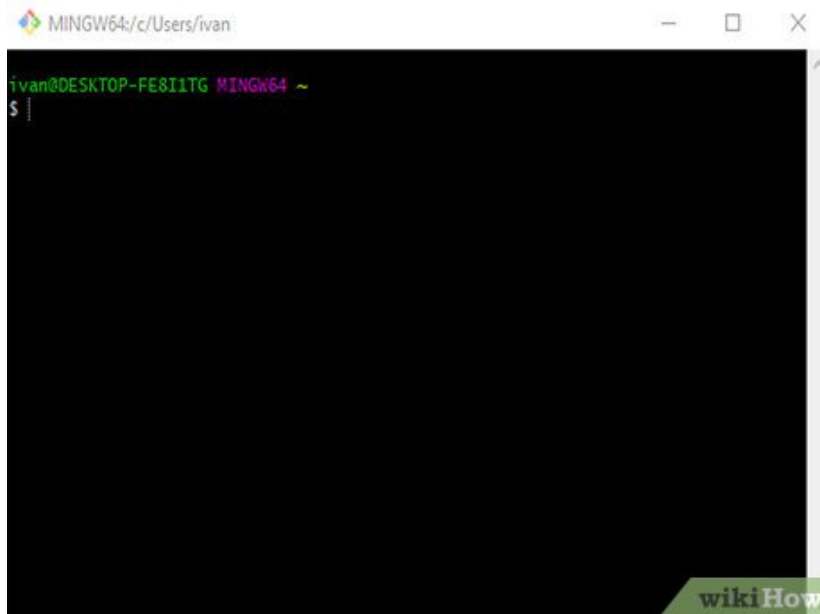
Part 1 of 3:

## Setting Up Your Account



**Set up a Github Account.** Visit GitHub and create an account. For the purposes of this tutorial a free account will work just fine.

2.



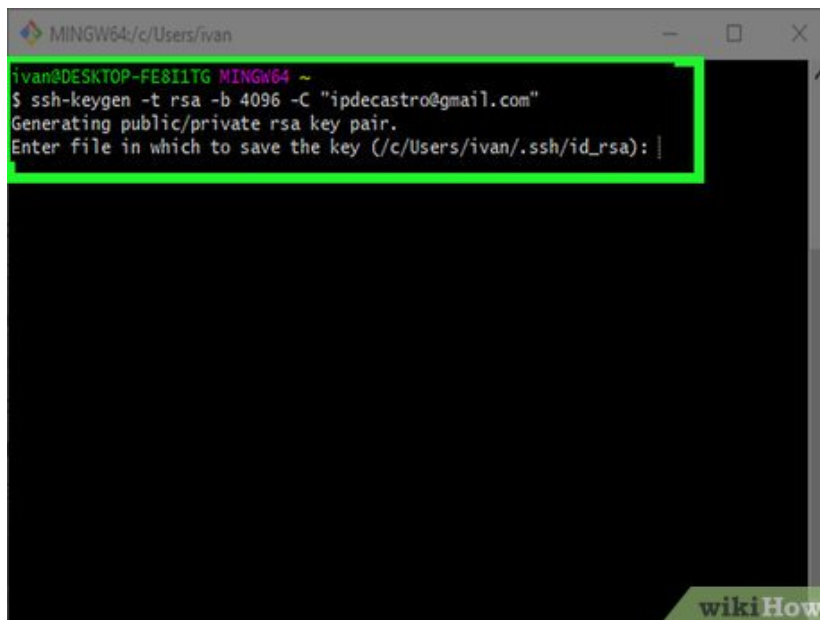
```
MINGW64/c/Users/ivan
ivan@DESKTOP-FE8I1TG MINGW64 ~
$ |
```

wikiHow

**Install Git Bash.** In order to get started you must first download and install Git Bash for windows. Go ahead and do that now by following this link: [Git Bash](#).

1. Once it's installed, run Git Bash. You should be looking at a black command prompt screen. Git Bash uses Unix commands to operate so some knowledge of Unix is important to have.

3.



```
MINGW64/c/Users/ivan
ivan@DESKTOP-FE8I1TG MINGW64 ~
$ ssh-keygen -t rsa -b 4096 -C "ipdecastro@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/ivan/.ssh/id_rsa): |
```

wikiHow

**Create an SSH Key.** In order to establish a secure encrypted connection between your GitHub account and Git Bash on your computer, you must generate and link an SSH key. In Git Bash, paste this code but substitute in the email you used with your GitHub account: `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`

1. You will then be prompted as to where you want to save the key. The default location will suffice so just hit `?Enter`. Next, Git Bash will ask you to enter and confirm a passphrase. While you don't have to include one, it's highly recommended that you do.

```

MINGW64/c/Users/ivan
SHA256:65JL1UYIEDwsgcsZwgF9KwVps50H205Vkj76D7bEXy4 ipdecastro@gmail.com
The key's randomart image is:
+---[RSA 4096]-----+
|.o^"=oo|
|o+.BB o |
|o."+++o .|
|.o .o^ o|
| . + o S o|
| +. . o|
| o=...|
| oo^Eo|
| .o=o.|
+---[SHA256]-----+

ivan@DESKTOP-FE8I1TG MINGW64 ~
$ eval "$(ssh-agent -s)"
Agent pid 382

ivan@DESKTOP-FE8I1TG MINGW64 ~
$ ssh-add ~/.ssh/id_rsa
Identity added: /c/Users/ivan/.ssh/id_rsa (ipdecastro@gmail.com)

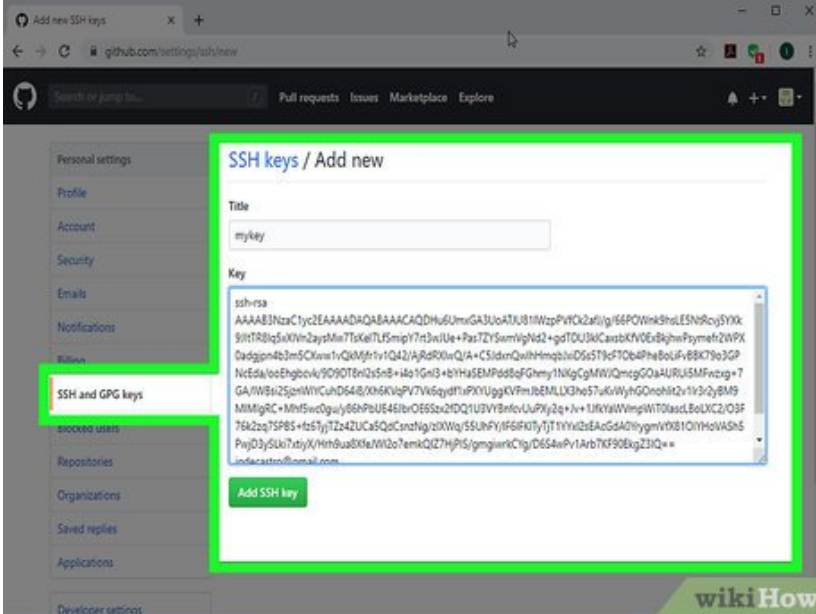
ivan@DESKTOP-FE8I1TG MINGW64 ~
$

```

4.

**Add your SSH key to the ssh-agent.** This will authorize your computer to use that SSH key. Enter the following command to start the SSH Agent: `eval "$(ssh-agent -s)"` Then enter in `ssh-add ~/.ssh/id_rsa` to add your created key.

1. If your key has a different name besides `id_rsa` or you saved it in a different location, make sure you use that instead.



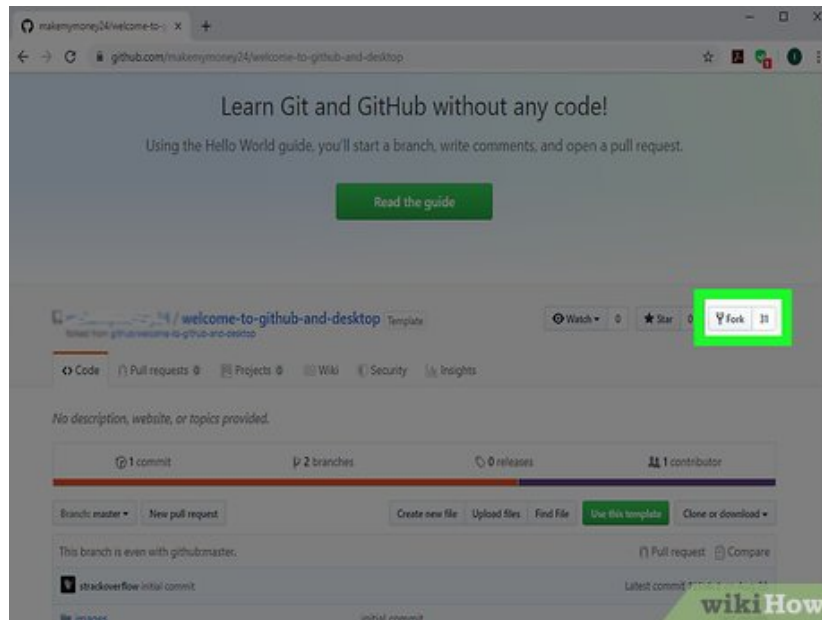
The screenshot shows the GitHub 'SSH keys / Add new' page. The 'Title' field contains 'mykey'. The 'Key' field contains a long alphanumeric string representing the public key. The 'SSH and GPG keys' option in the left sidebar is highlighted with a green box. A green 'Add SSH key' button is visible at the bottom of the form.

5.

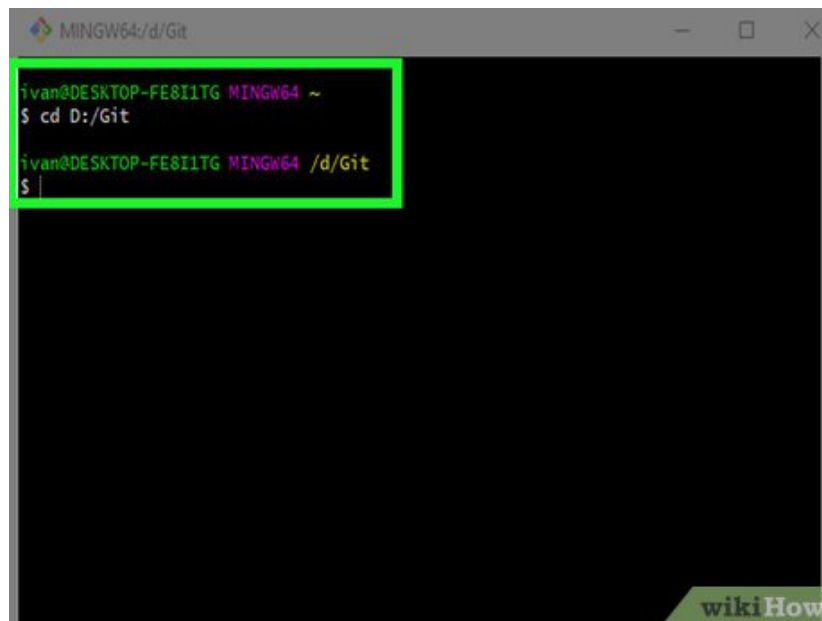
**Add your SSH key to your account.** You will now need to configure your account to use your newly created key. Copy the ssh key to your clipboard: `clip ~/.ssh/id_rsa.pub`. Then, in the top right corner of any GitHub page, click on your profile photo, and then click *Settings*. In the user settings sidebar, click *SSH and GPG keys*. Then click *New SSH Key*. Now you can enter in a descriptive name for your key then paste your key into the key field, and press "Add SSH Key". Confirm it, and you're all set!

Part 2 of 3:

## Setting Up a Project

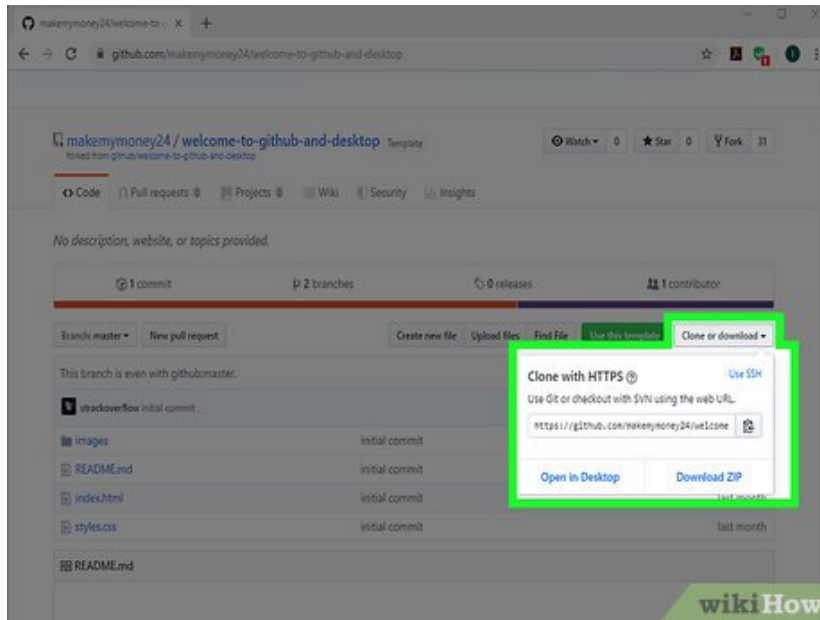


**Fork a repository.** In order to make changes to a project in GitHub, it must be forked. Go to the repository you want to work on, and fork the repository by pressing *fork* in the top right part of the page. This will make a copy of that repository on your account.



**Create a local directory.** Create a folder somewhere on your computer where you want to house the repository. Then use Git Bash to navigate to that folder. Remember Git Bash accepts UNIX commands, so in order to get into your directory, use the CD command like so: `$ cd /path/to/directory`

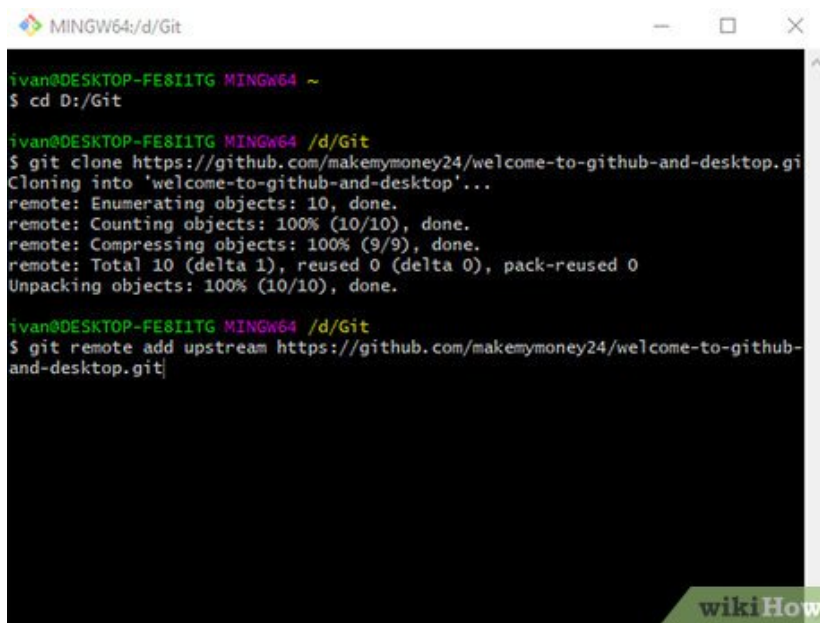
3.



**Clone the fork.** In GitHub, navigate to your fork and under the repository name, click *Clone or download*, and copy the link it gives you.

1. Next, in Git Bash, enter in the following command using your copied URL: `$ git clone https://github.com/YOUR-USERNAME/REPOSITORY_NAME`. Press `Enter`, and your local clone will be created.

4.

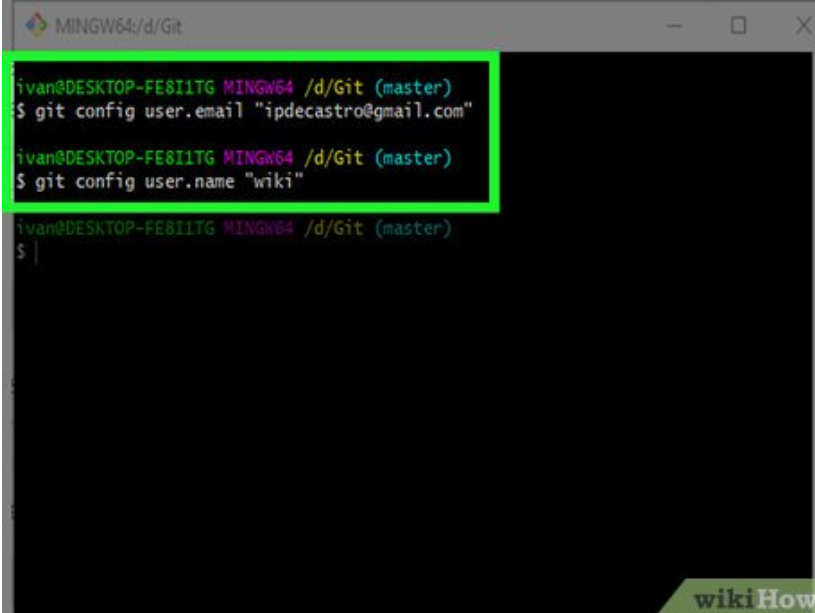


**Sync your fork with the original.** You need to be able to propose changes to the original repository. Navigate to the original repository that you forked in GitHub, then hit *Clone or download* and copy the URL.

1. Now navigate into the actual repository folder in GitHub. You'll know you're in the right spot when you see a (master) to the right of your command prompt.
2. Now simply run `$ git remote add upstream https://github.com/user/repositoryName` using the original URL of the

repository.

5.

A terminal window titled 'MINGW64:/d/Git' showing the execution of two git config commands. The first command is '\$ git config user.email "ipdecastro@gmail.com"' and the second is '\$ git config user.name "wiki"'. Both commands and their outputs are highlighted with a green box. The terminal prompt is 'ivan@DESKTOP-FE8I1TG MINGW64 /d/Git (master)'. A 'wikiHow' logo is visible in the bottom right corner of the terminal window.

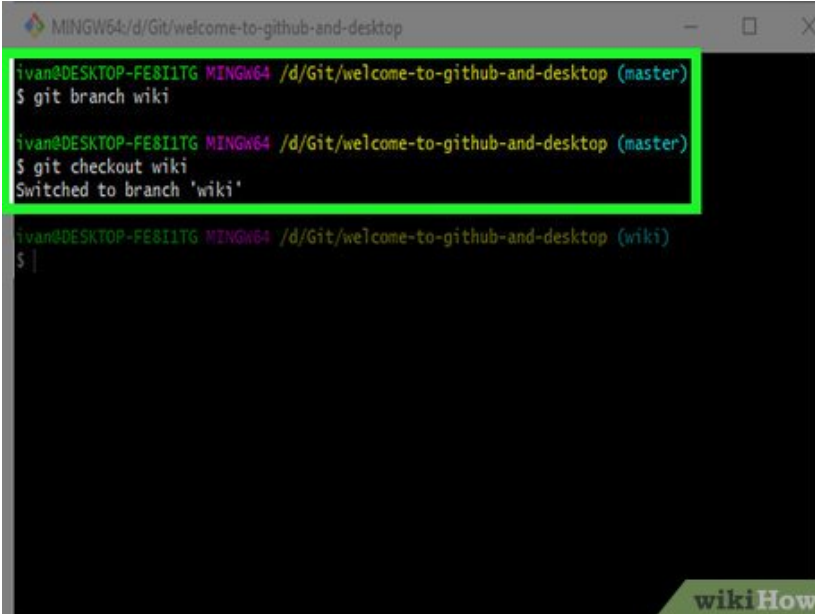
```
ivan@DESKTOP-FE8I1TG MINGW64 /d/Git (master)
$ git config user.email "ipdecastro@gmail.com"

ivan@DESKTOP-FE8I1TG MINGW64 /d/Git (master)
$ git config user.name "wiki"

ivan@DESKTOP-FE8I1TG MINGW64 /d/Git (master)
$ |
```

**Create a user.** Next you should create a user to track who made the changes to the repository. Run the following two commands. `$ git config user.email 'you@example.com'` and `$ git config user.name 'Your Name'`. Make sure the email you use is the same one that's on your git hub account.

6.

A terminal window titled 'MINGW64:/d/Git/welcome-to-github-and-desktop' showing the execution of two git commands. The first command is '\$ git branch wiki' and the second is '\$ git checkout wiki', which outputs 'Switched to branch 'wiki''. Both commands and their outputs are highlighted with a green box. The terminal prompt is 'ivan@DESKTOP-FE8I1TG MINGW64 /d/Git/welcome-to-github-and-desktop (master)'. A 'wikiHow' logo is visible in the bottom right corner of the terminal window.

```
ivan@DESKTOP-FE8I1TG MINGW64 /d/Git/welcome-to-github-and-desktop (master)
$ git branch wiki

ivan@DESKTOP-FE8I1TG MINGW64 /d/Git/welcome-to-github-and-desktop (master)
$ git checkout wiki
Switched to branch 'wiki'

ivan@DESKTOP-FE8I1TG MINGW64 /d/Git/welcome-to-github-and-desktop (wiki)
$ |
```

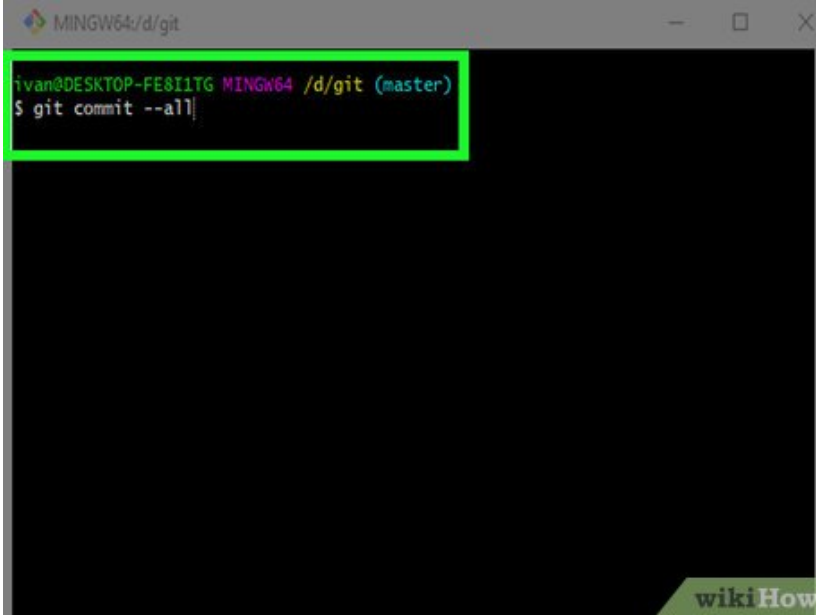
**Create a new branch.** Next you should create a new branch off of our master branch. As an actual branch of a tree. This branch will hold all of the specific changes you will make. You should create a new branch off of the master every time you work on a new problem. Whether it's a bug fix or the addition of a new feature, each task must get its own unique branch.

1. In order to make a branch, simply run: `$ git branch feature_x`. Replace *feature\_x* with a descriptive name of your feature.

2. Once you've made your branch use `$ git checkout feature_x`. This will switch you into the `feature_x` branch. You are now free to make changes to your code.

Part 3 of 3:

## Pushing Your Changes

```
1. A screenshot of a terminal window titled 'MINGW64/d/git'. The prompt is 'ivan@DESKTOP-FE8I1TG MINGW64 /d/git (master)'. The command '$ git commit --all' is entered and highlighted with a green box. A '1.' is positioned to the left of the terminal. A 'wikiHow' logo is in the bottom right corner of the terminal window.
```

**Commit your changes.** Once you've finished making changes, or you want to switch branches and work on something else, your changes must be committed. Run `$ git commit --all`. This will automatically commit all the changes you've made to the repository.

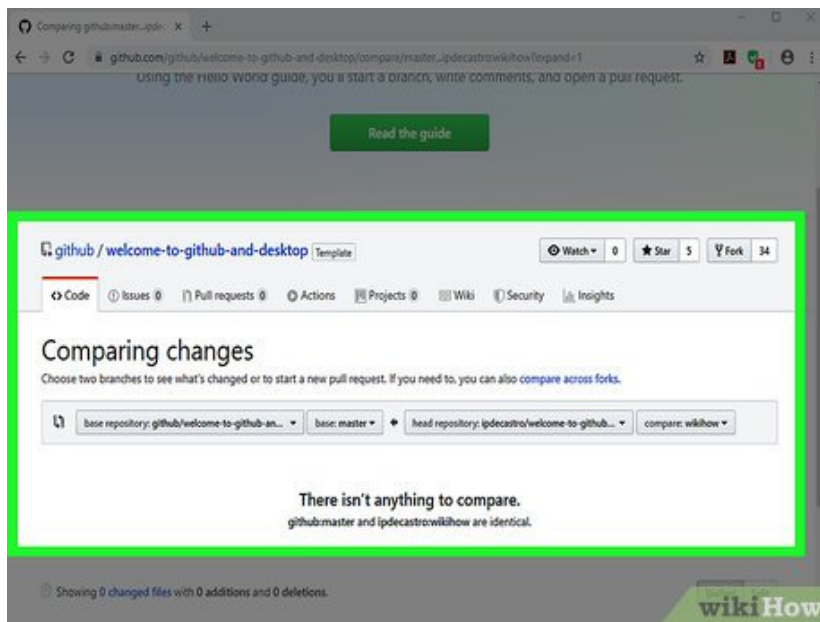
1. You will get a prompt to enter in a commit message using vim. This message should be short and descriptive. Use the arrow keys to navigate to the top line, and then hit `i` on your keyboard. You may now type your message. Once it's typed, hit `Esc` and then hit the colon key, `:`. Now type in the letters `wq` and hit `Enter`. This will save your commit message and quit the vim editor.

```
MINGW64/d/Git/welcome-to-github-and-desktop
ivan@DESKTOP-FE811TG MINGW64 /d/Git/welcome-to-github-and-desktop (wiki)
$ git push origin wikihow
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'wikihow' on GitHub by visiting:
remote:   https://github.com/ipdecastro/welcome-to-github-and-desktop/pull/new/wikihow
remote:
To https://github.com/ipdecastro/welcome-to-github-and-desktop.git
 * [new branch]   wikihow -> wikihow

ivan@DESKTOP-FE811TG MINGW64 /d/Git/welcome-to-github-and-desktop (wiki)
$
```

2.

**Make a push request.** Now that your changes have been committed, you should push them! Enter in `$ git push origin`.



3.

**Merge with the master branch.** Go back to GitHub and you should soon see a message pop up with your push. Hit "Compare & pull request". On this page you will have the opportunity to review your changes, as well as change your commit message and add comments. Once everything looks in order, and GitHub doesn't detect any conflicts, go ahead and make the request. And that's it!

1. Now it will be up to your other contributors and the owner of the repository to review your change and then merge it with the master repository.

```
MINGW64:/d/Git/welcome-to-github-and-desktop
ivan@DESKTOP-FE8I1TG MINGW64 /d/Git/welcome-to-github-and-desktop (wiki)
$ git push origin wikihow
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'wikihow' on GitHub by visiting:
remote:   https://github.com/ipdecastro/welcome-to-github-and-desktop/pull/new/wikihow
remote:
To https://github.com/ipdecastro/welcome-to-github-and-desktop.git
 * [new branch]   wikihow -> wikihow

ivan@DESKTOP-FE8I1TG MINGW64 /d/Git/welcome-to-github-and-desktop (wiki)
$ git fetch upstream && git rebase upstream/master
```

4.

**Always remember to fetch and rebase.** It's extremely important to always be working on the latest version of a file. Before you make any push requests, or you've just started a new branch or switched to a branch, always run the following command `git fetch upstream && git rebase upstream/master`.

You finished reading the article "**How to Set Up and Use Git**" edited by the [TipsMake](#) team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.