



# **Practical Web Development**

## **Github**

Presented by ARMS (THAILAND) Co., Ltd.

# Index

## 1. Preparing a Github ID

- 1-1. What are Git and Github
- 1-2. Create a Github ID

## 2. Preparing a Git

- 2-1. Configure

## 3. Getting started with Git and GitHub

- 3-1. Create a new repository on Github
- 3-2. Get local repository from Github
- 3-3. Record Changes to the repository
- 3-4. Pushing to a remote
- 3-5. Team collaboration

## 4. Git commands

- 4-1. Git commands

# 1. Preparing a Github ID

## 1-1 What are Git and Github

- Git is a distributed revision control system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows
- As with most other distributed revision control systems, and unlike most client–server systems, every Git working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server
- See reference at "[https://en.wikipedia.org/wiki/Git\\_\(software\)](https://en.wikipedia.org/wiki/Git_(software))"

# 1. Preparing a Github ID

## 1-1 What are Git and Github

- GitHub is a web-based Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.
- Unlike Git, which is strictly a command-line tool, GitHub provides a web-based graphical interface and desktop as well as mobile integration. It also provides access control and several collaboration features such as wikis, task management, and bug tracking and feature requests for every project
- See reference at <https://en.wikipedia.org/wiki/GitHub>

In a nutshell, **Git** is a distributed version management system, **Github** is a web service name based on Git.

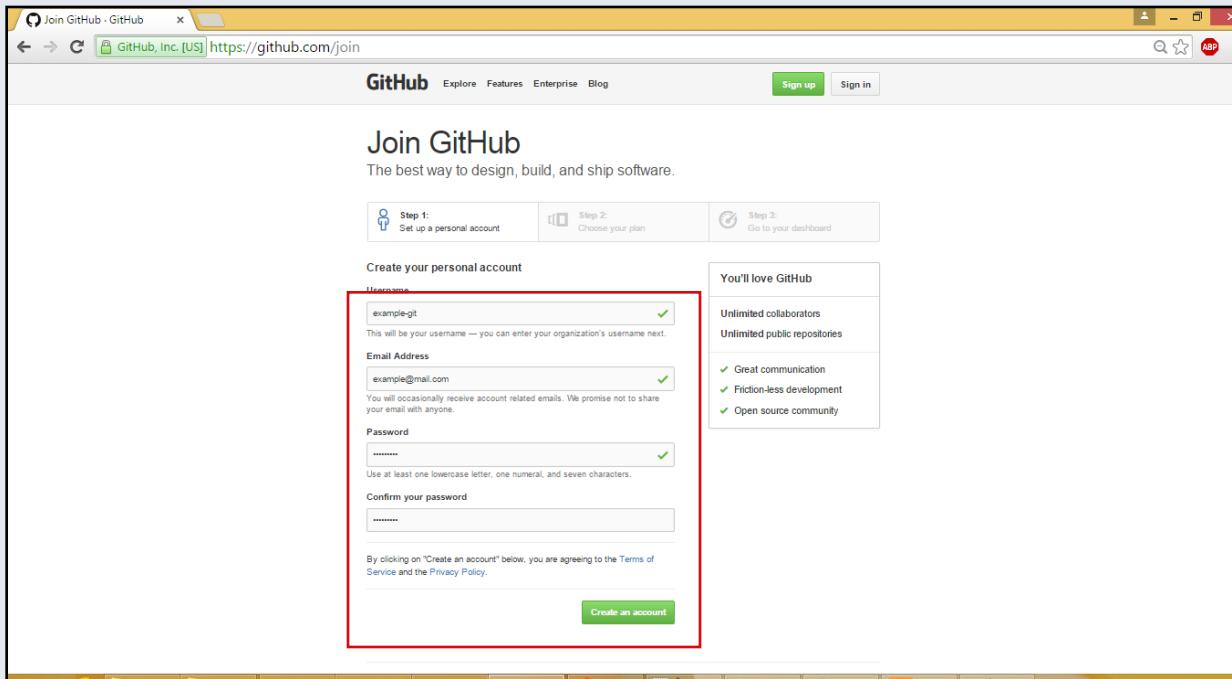
Git is developed by Linux development team.

Github is run by Github company

# 1. Preparing a Github ID

## 1-2 Create a Github ID

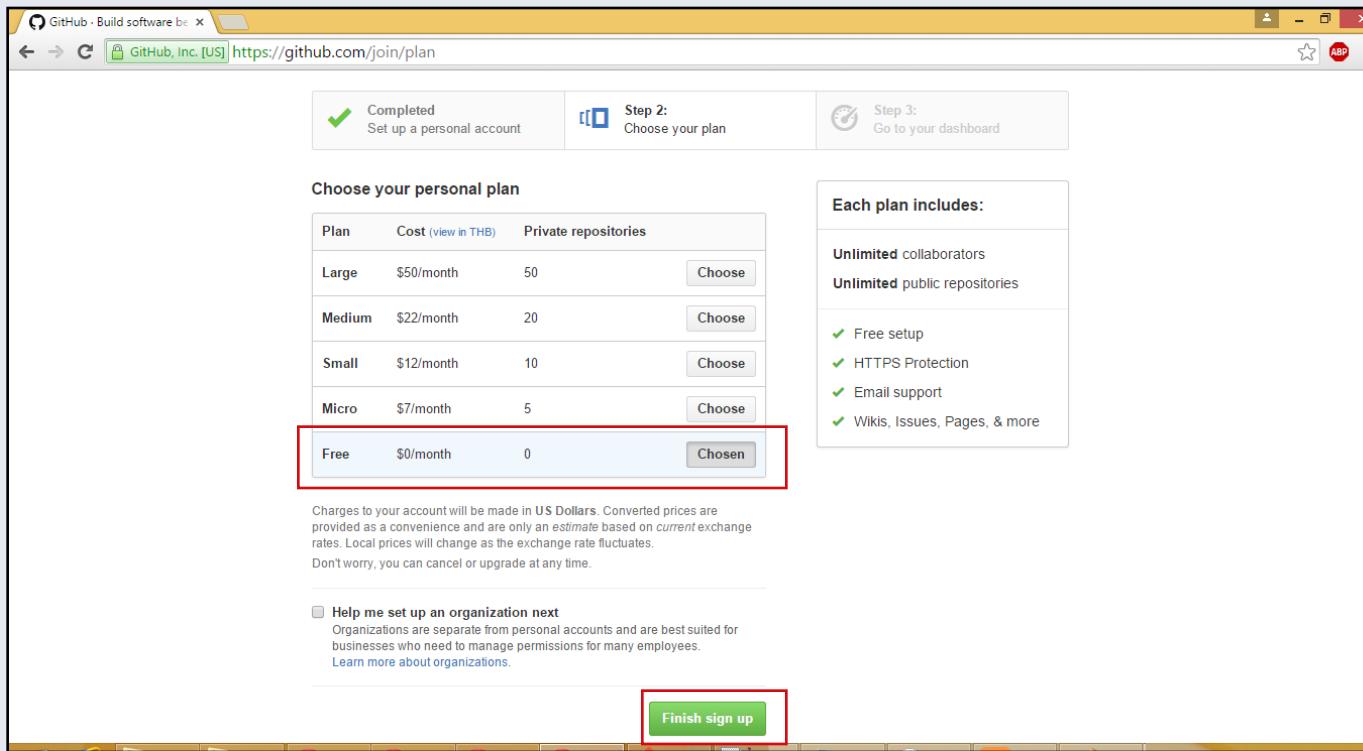
- Open "<https://github.com/join>" to create a new Github ID
- Insert these information and click "Create an account"



# 1. Preparing a Github ID

## 1-2 Create a Github ID

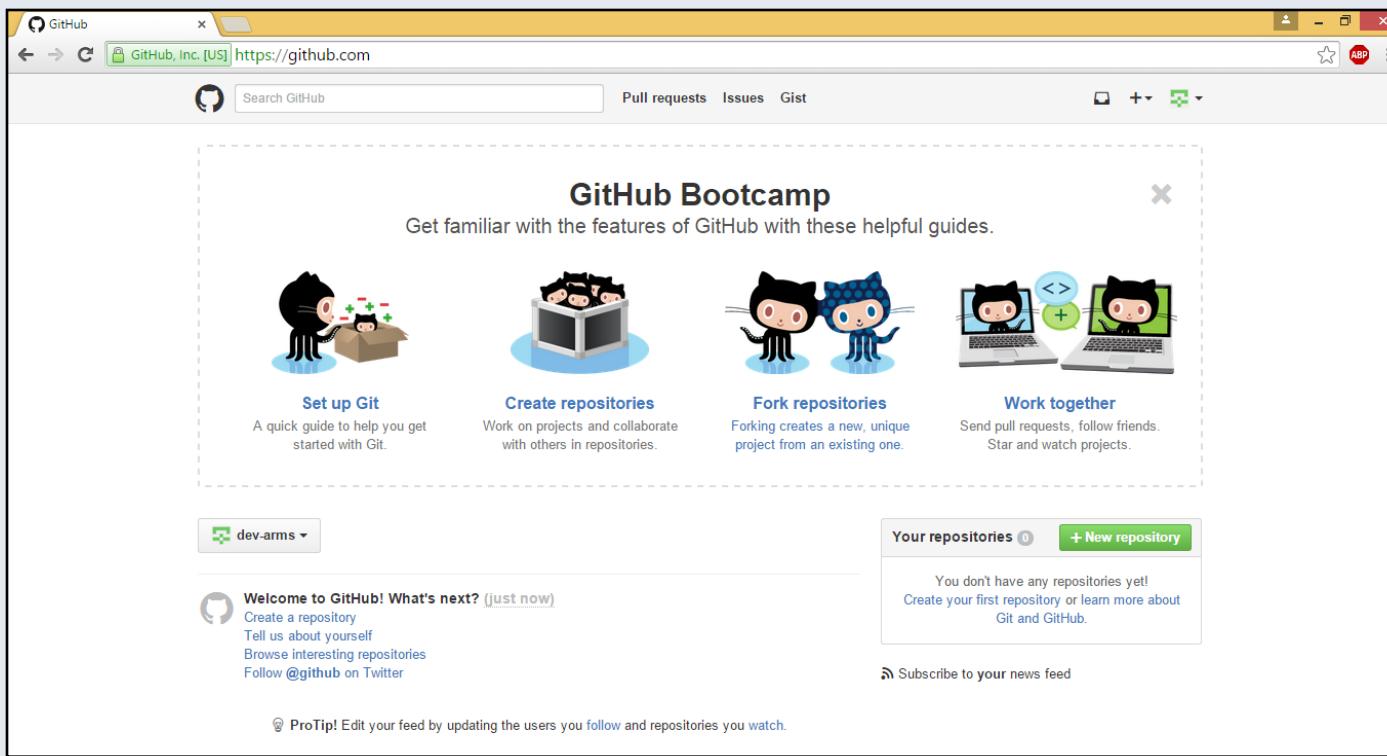
- Choose "Free" plan and click "Finish sign up"



# 1. Preparing a Github ID

## 1-2 Create a Github ID

- Now, you got a Github ID

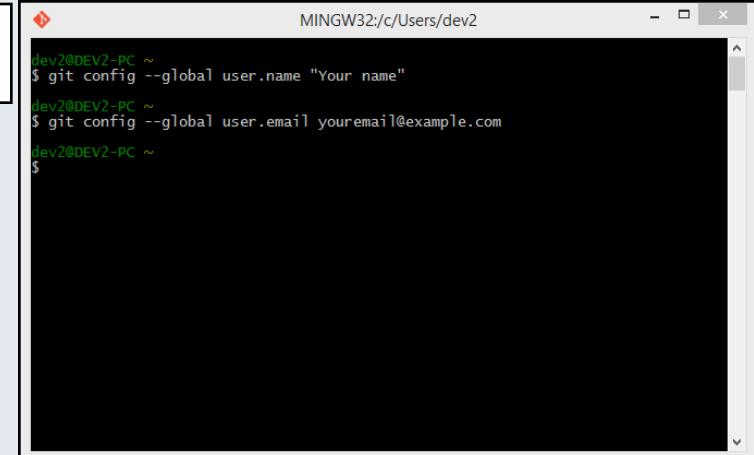


## 2. Preparing a Git

### 2-1 Configure

- The first thing you should do when Git installed is to set your user name and e-mail address
- This is important because Git uses this information when you commit
- Open Git Bash (there should be a shortcut on your desktop).

```
git config --global user.name "Your name"  
git config --global user.email youremail@example.com
```



A screenshot of a Git Bash terminal window titled 'MINGW32:c/Users/dev2'. The window shows the command line with the following text:  
dev2@DEV2-PC ~  
\$ git config --global user.name "Your name"  
dev2@DEV2-PC ~  
\$ git config --global user.email youremail@example.com  
dev2@DEV2-PC ~  
\$

## 2. Preparing a Git

### 2-1 Configure

- Check your settings
- Use the git config --list command to list all the settings

```
git config --list
```

- Make sure that **user.name** and **user.email** have been properly added to the setting

```
dev2@DEV2-PC ~
$ git config --global user.name "Your name"
dev2@DEV2-PC ~
$ git config --global user.email youremail@example.com
dev2@DEV2-PC ~
$ git config --list
core.symlinks=false
core.autocrlf=true
color.diff=auto
color.status=auto
color.branch=auto
color.interactive=true
pack.packsizelimit=2g
help.format=html
http.sslcainfo=/bin/curl-ca-bundle.crt
sendemail.smtpserver=/bin/msmtp.exe
diff.astextplain.textconv=astextplain
rebase.autosquash=true
user.name=Your name
user.email=youremail@example.com

dev2@DEV2-PC ~
$
```

You can check a specific key value like below

```
git config {key}
```

```
git config user.email
```

## 2. Preparing a Git

### 2-1 Configure

- If you set a wrong user.name or user.email, use the following commands to edit the wrong setting.

git config --global --edit

or

vi .gitconfig

To edit the .gitconfig, you have to use vi editor.

It has a command mode and edit mode,

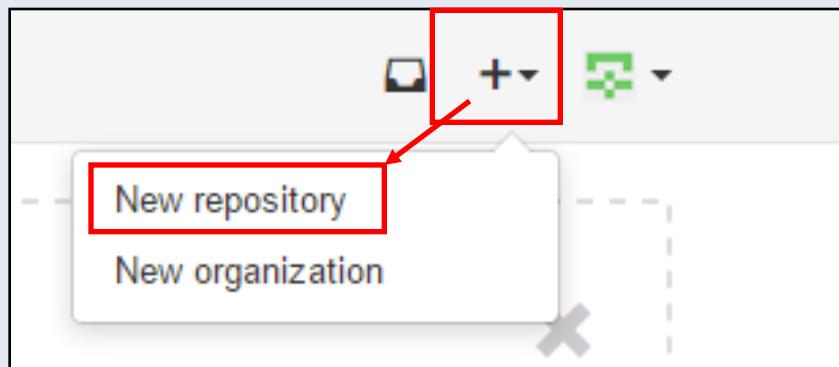
1. To enter into an edit mode, press either “**a**” “**i**” “**o**” key.
2. To finish the edit mode, press “**esc**” key.
3. Then type : wq (colon : **w**rite the change and **q**uit)

If you don't want to write the change,  
:q! (colon : **q**uit without saving)

### 3. Getting started with Git and GitHub

#### 3-1 Create a new repository on GitHub

- Open Github page, click + icon on the upper-right corner , and then click New repository



### 3. Getting started with Git and GitHub

#### 3-1 Create a new repository on GitHub

- Create Repository name  
**Repository name : "test-git-01"**
- Optionally, add a description of your repository.  
**Description : "test-git-01 repository on GitHub."**
- Choose "**Public**" repository
  - **Public repository** is a great choice for getting started! They're visible to any user on GitHub, so you can benefit from a collaborative community
  - **Private repository** requires a little more setup. They're only available to you, the repository owner, as well as any collaborators you choose to share with. Private repository is only available for a paid account.
- Select "**Initialize this repository with a README**"
- Click "**Create repository**"

### 3. Getting started with Git and GitHub

#### 3-1 Create a new repository on GitHub

Owner  Repository name  

Great repository names are short and memorable. Need inspiration? How about [ducking-octo-bear](#).

Description (optional)

 **Public**  
Anyone can see this repository. You choose who can commit.

 **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with a README**  
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

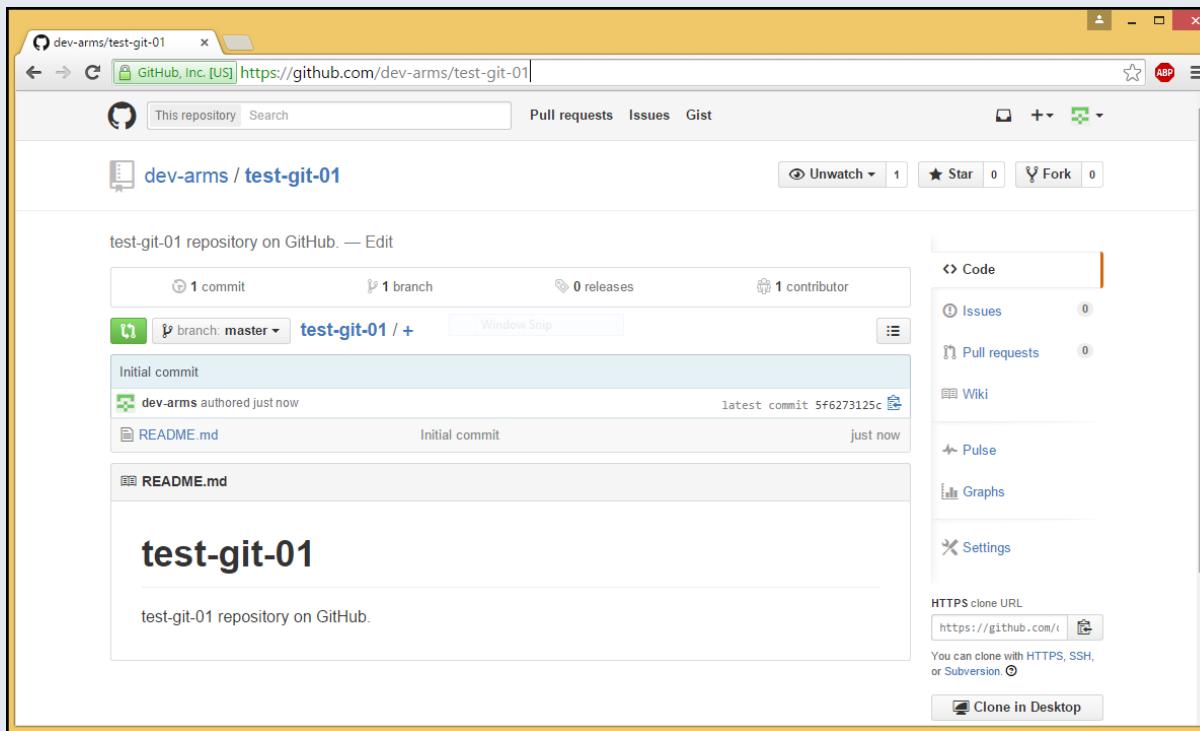
Add .gitignore: **None** | Add a license: **None** 

**Create repository**

### 3. Getting started with Git and GitHub

#### 3-1 Create a new repository on GitHub

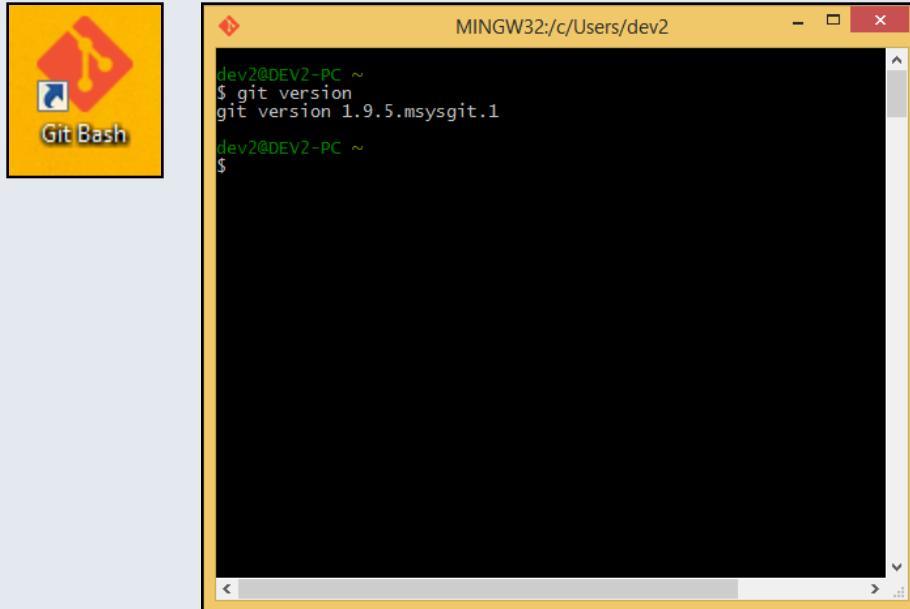
- Now, you can see your repository.



## 3. Getting started with Git and GitHub

### 3-2 Get local repository from Github

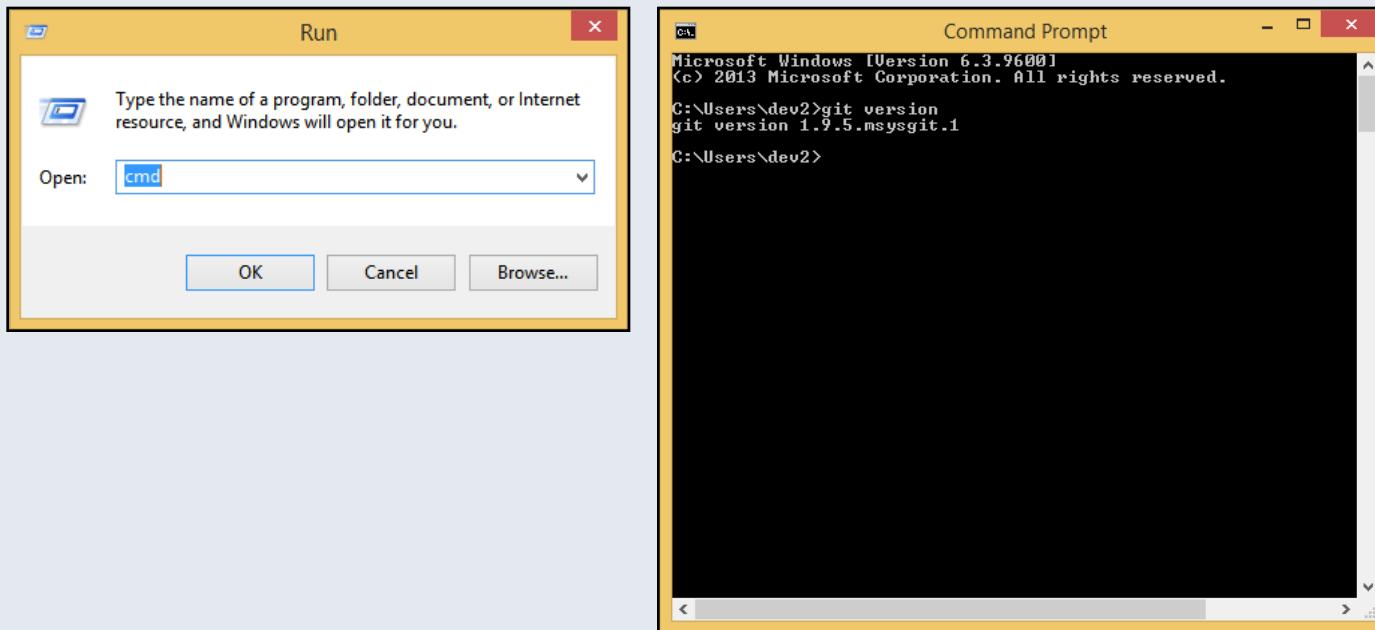
- You can use Git via Git Bash from desktop shortcut or via Windows Command Prompt



### 3. Getting started with Git and GitHub

#### 3-2 Get local repository from Github

- You can use git via Windows Command Prompt ("Windows Key + r" and type "cmd")



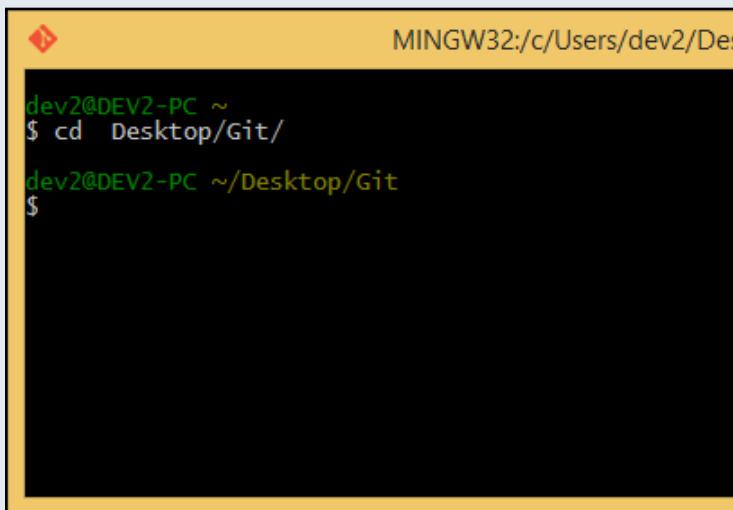
### 3. Getting started with Git and GitHub

#### 3-2 Get local repository from Github

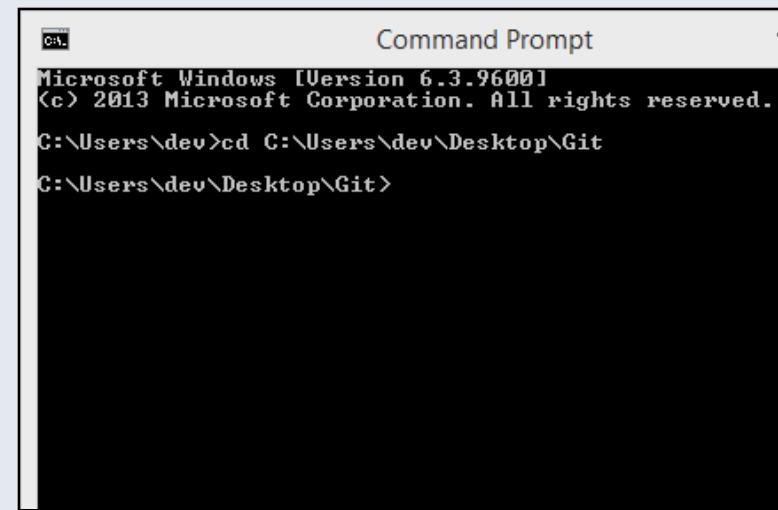
- Open your git bash or cmd and run "cd [path to directory]" to go to directory that you want to place this repository. For example

```
via Git bash use : cd Desktop/Git  
via cmd use      : cd C:\Users\dev\Desktop\Git
```

\* The above directories are just an example, please create a directory such as Git by using "mkdir" command.



```
MINGW32:c/Users/dev2/Desktop  
  
dev2@DEV2-PC ~  
$ cd Desktop/Git/  
  
dev2@DEV2-PC ~/Desktop/Git  
$
```

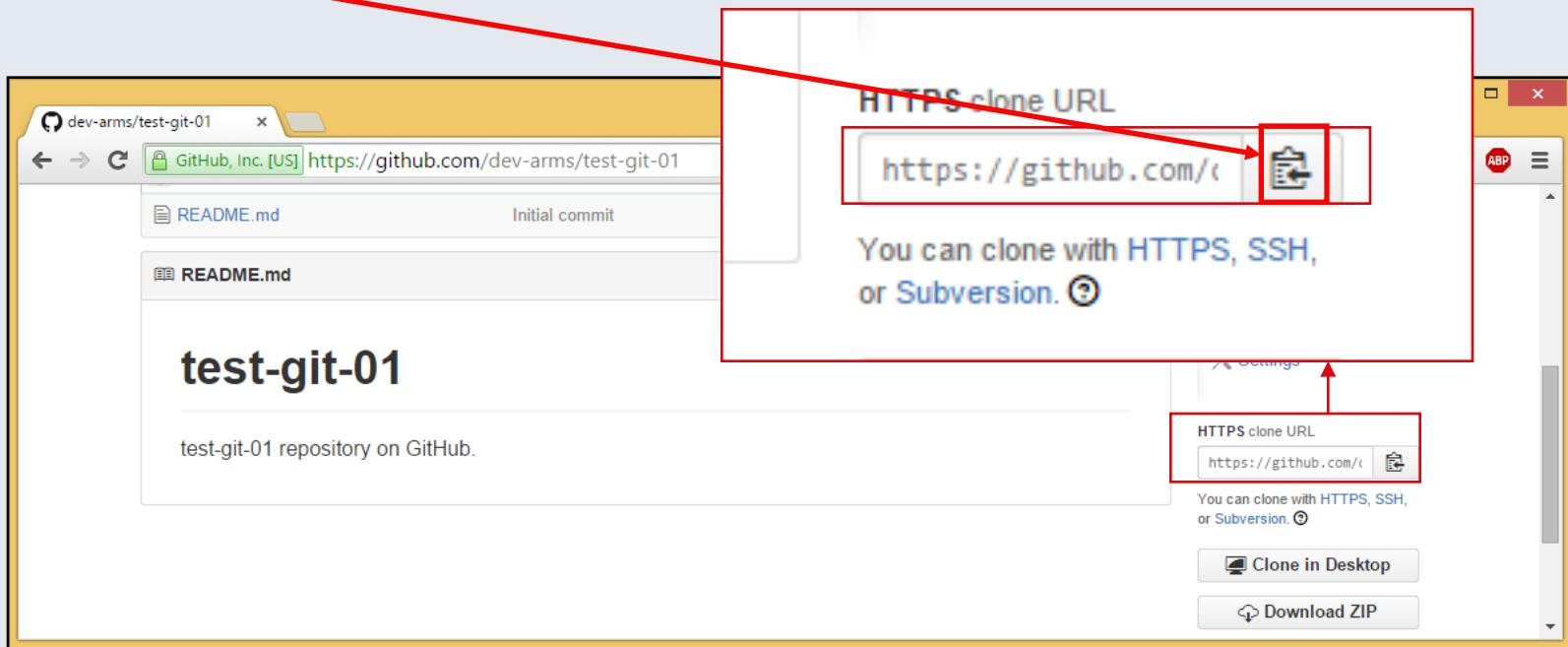


```
Command Prompt  
  
Microsoft Windows [Version 6.3.9600]  
(c) 2013 Microsoft Corporation. All rights reserved.  
C:\Users\dev>cd C:\Users\dev\Desktop\Git  
C:\Users\dev\Desktop\Git>
```

### 3. Getting started with Git and GitHub

#### 3-2 Get local repository from Github

- Use "git clone [url]" to clone Github repository to local. You can get repository [url] from your Github repository page. Click green “Clone or download” button. Copy URL



### 3. Getting started with Git and GitHub

#### 3-2 Get local repository from Github

- Example

\* URL from Github repository

```
git clone https://github.com/dev-arms/test-git-01.git
```



The screenshot shows a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git". The command \$ git clone https://github.com/dev-arms/test-git-01.git is entered and executed. The output shows the cloning process: counting objects (3), unpacking objects (100% 3/3), and checking connectivity. The terminal prompt \$ is visible at the end.

```
dev2@DEV2-PC ~
$ cd Desktop/Git

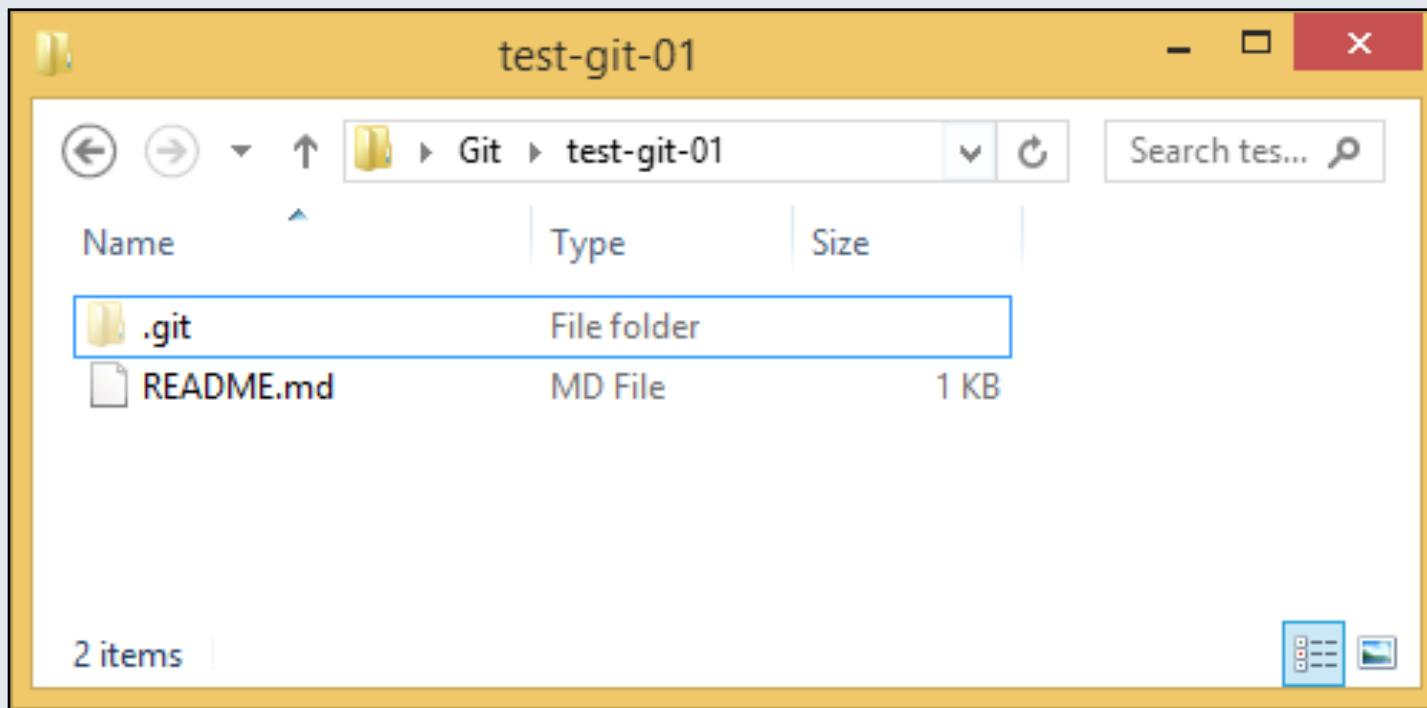
dev2@DEV2-PC ~/Desktop/Git
$ git clone https://github.com/dev-arms/test-git-01.git
Cloning into 'test-git-01'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
Checking connectivity... done.

dev2@DEV2-PC ~/Desktop/Git
$
```

### 3. Getting started with Git and GitHub

#### 3-2 Get local repository from Github

- Now, you can see "test-git-01" on your PC



### 3. Getting started with Git and GitHub

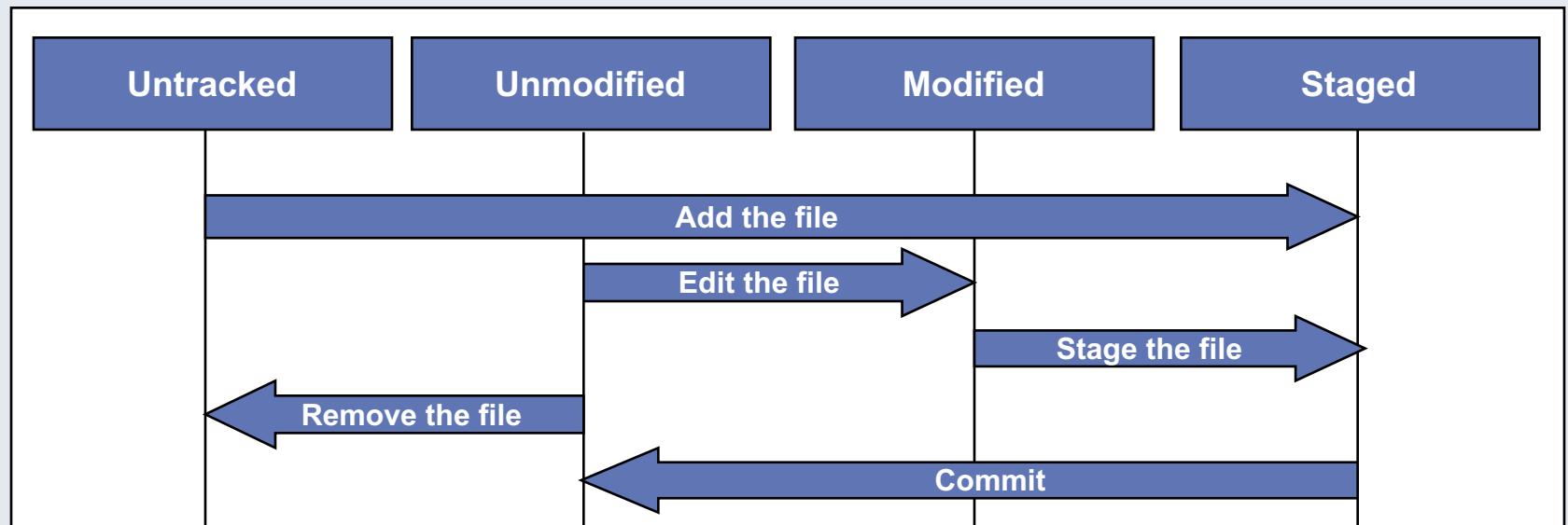
#### 3-3 Recording Changes to the Repository

- Remember that each file in your working directory can be in one of two states: tracked or untracked
- Tracked files are files that were in the last snapshot. They can be unmodified, modified, or staged
- Untracked files are everything else – any files in your working directory that were not in your last snapshot and are not in your staging area. When you first clone a repository, all of your files will be tracked and unmodified because you just checked them out and haven't edited anything

### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

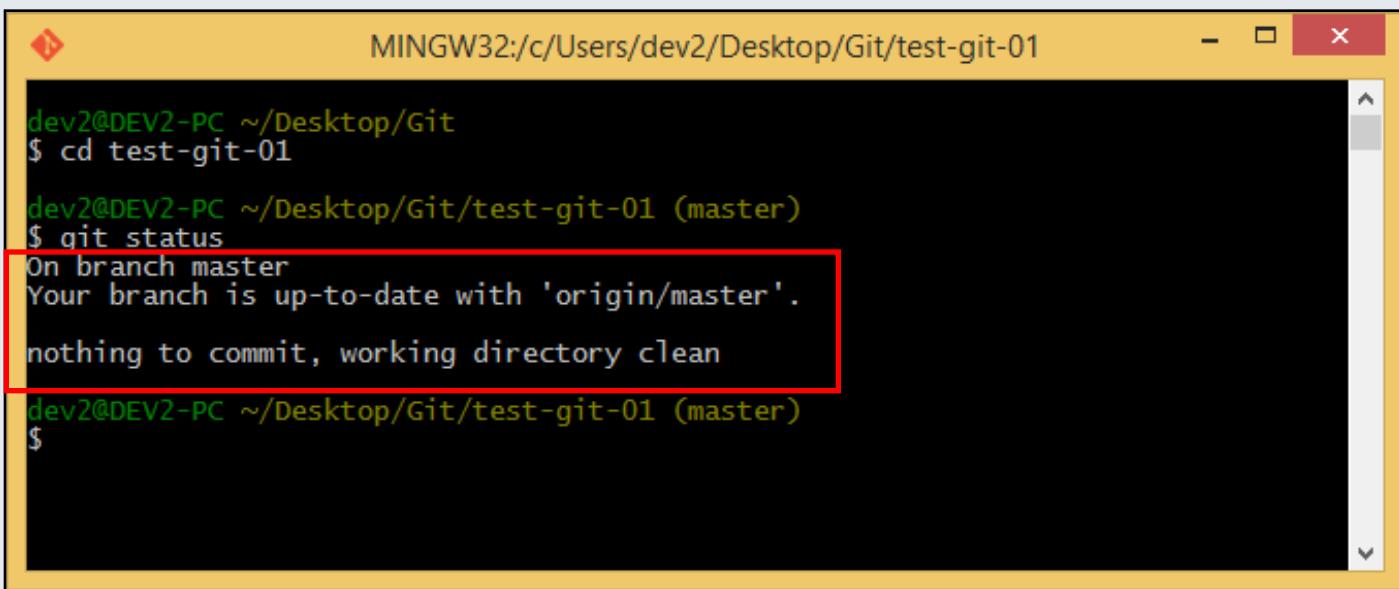
- The lifecycle of your file status



### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

- The main command you use to determine which files are in which state is "**git status**" command
- If you go to local git repository and run this command right after cloning a repository, you will see the screen below.



The screenshot shows a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git/test-git-01". The terminal output is as follows:

```
dev2@DEV2-PC ~/Desktop/Git
$ cd test-git-01

dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

nothing to commit, working directory clean

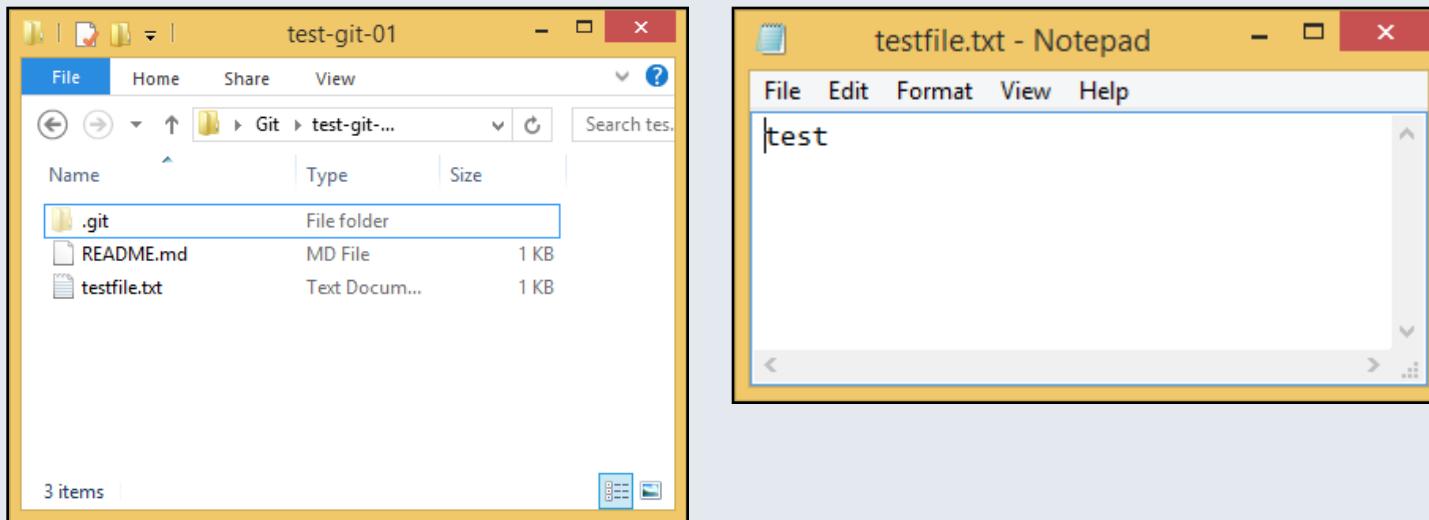
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$
```

A red box highlights the text "On branch master" through "nothing to commit, working directory clean". The entire terminal window is enclosed in a yellow border.

### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

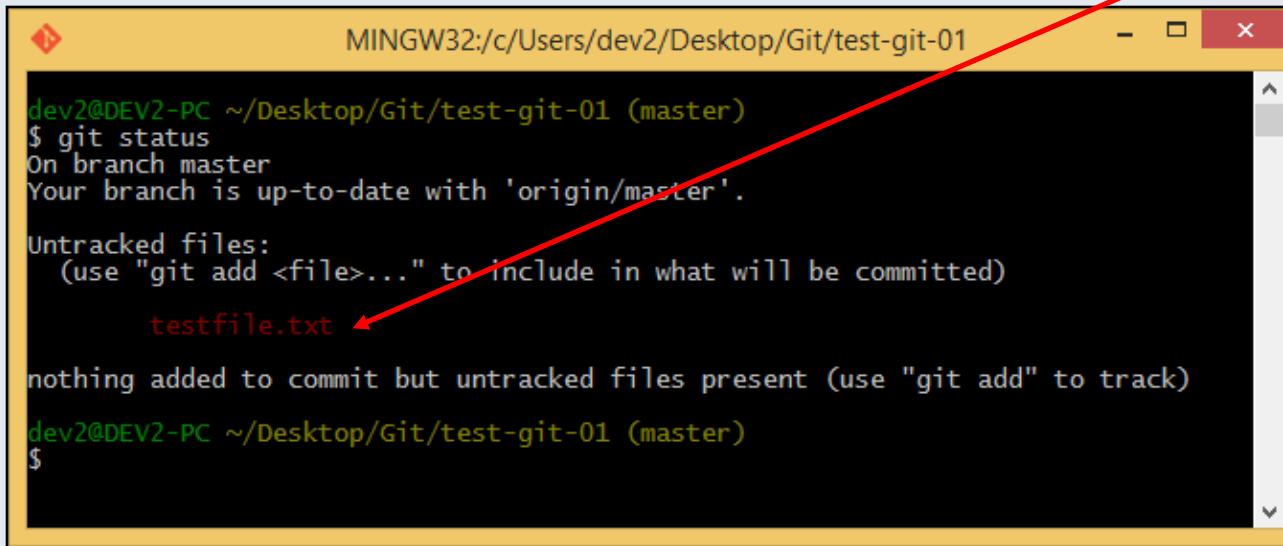
- This means you have a clean working directory. In other words, there are no tracked and modified files.
- Try to create a new file to your project.
- For example, create "**testfile.txt**" file that included some text in your project



### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

- After creating a new file, and run "**git status**", you will see an untracked file.



The screenshot shows a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git/test-git-01". The command \$ git status is run, showing the following output:

```
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (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)

    testfile.txt

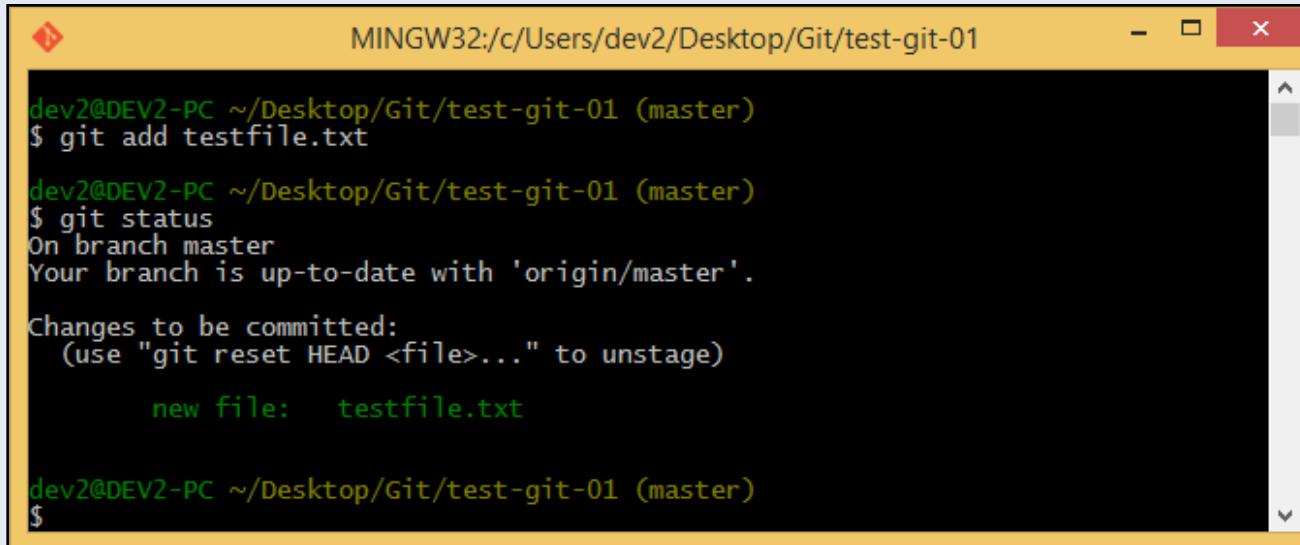
nothing added to commit but untracked files present (use "git add" to track)
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$
```

A red arrow points from the text "testfile.txt" in the "Untracked files:" section to the file itself in the terminal window.

### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

- To begin tracking a new file, use "git add [filename]" (Or "git add \*" to track all files) . **git add testfile.txt** and run "**git status**" again



The screenshot shows a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git/test-git-01". The command history and output are as follows:

```
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$ git add testfile.txt

dev2@DEV2-PC ~/Desktop/Git/test-git-01 (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:   testfile.txt

dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$
```

- You see your "testfile.txt" file tracked and staged to be committed.

### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

- Now that your staging area is set up the way you want it, you can commit your changes (*If you want to unstage file because you add a wrong file, you can use "git reset HEAD [filename]"*)
- The simplest way to commit is to run "**git commit -m 'commit message'**" and run "git status" again.

```
git commit -m 'Story 001 : Add testfile.txt'  
git status
```

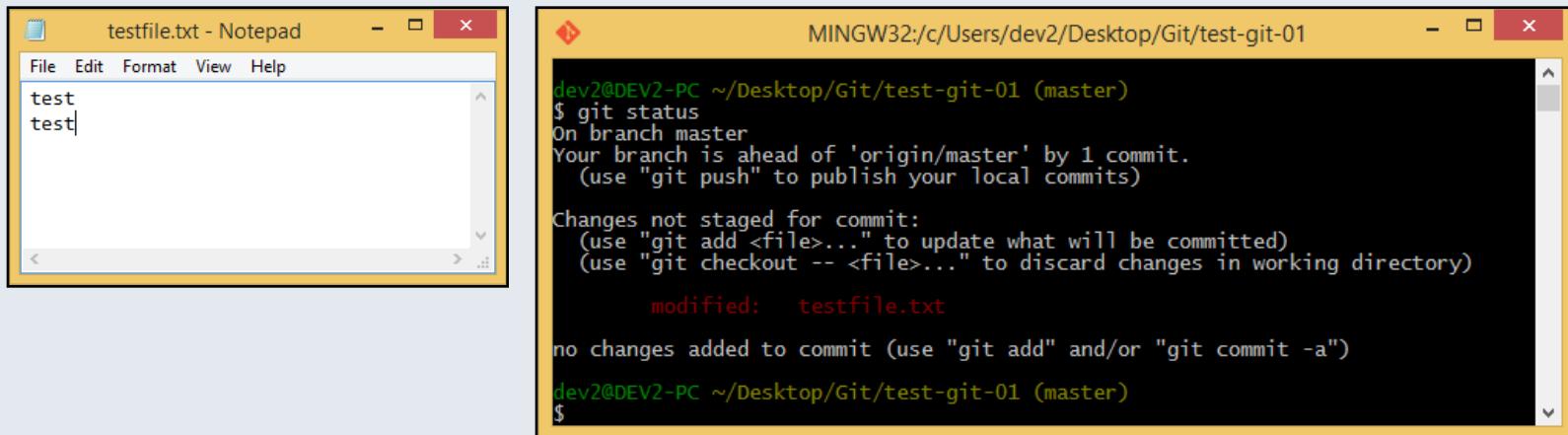


```
MINGW32:/c/Users/dev2/Desktop/Git/test-git-01  
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)  
$ git commit -m 'Story 001 : Add testfile.txt'  
[master 6505eef] Story 001 : Add testfile.txt  
1 file changed, 1 insertion(+)  
create mode 100644 testfile.txt  
  
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (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 directory clean  
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)  
$
```

### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

- Try to modify text in "testfile.txt" and run "git status"



The screenshot shows a Windows desktop environment. On the left, a Notepad window titled "testfile.txt - Notepad" displays the text "test" on two lines. On the right, a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git/test-git-01" shows the following command-line output:

```
dev2@DEV2-PC ~~/Desktop/Git/test-git-01 (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

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:   testfile.txt

no changes added to commit (use "git add" and/or "git commit -a")
dev2@DEV2-PC ~~/Desktop/Git/test-git-01 (master)
$
```

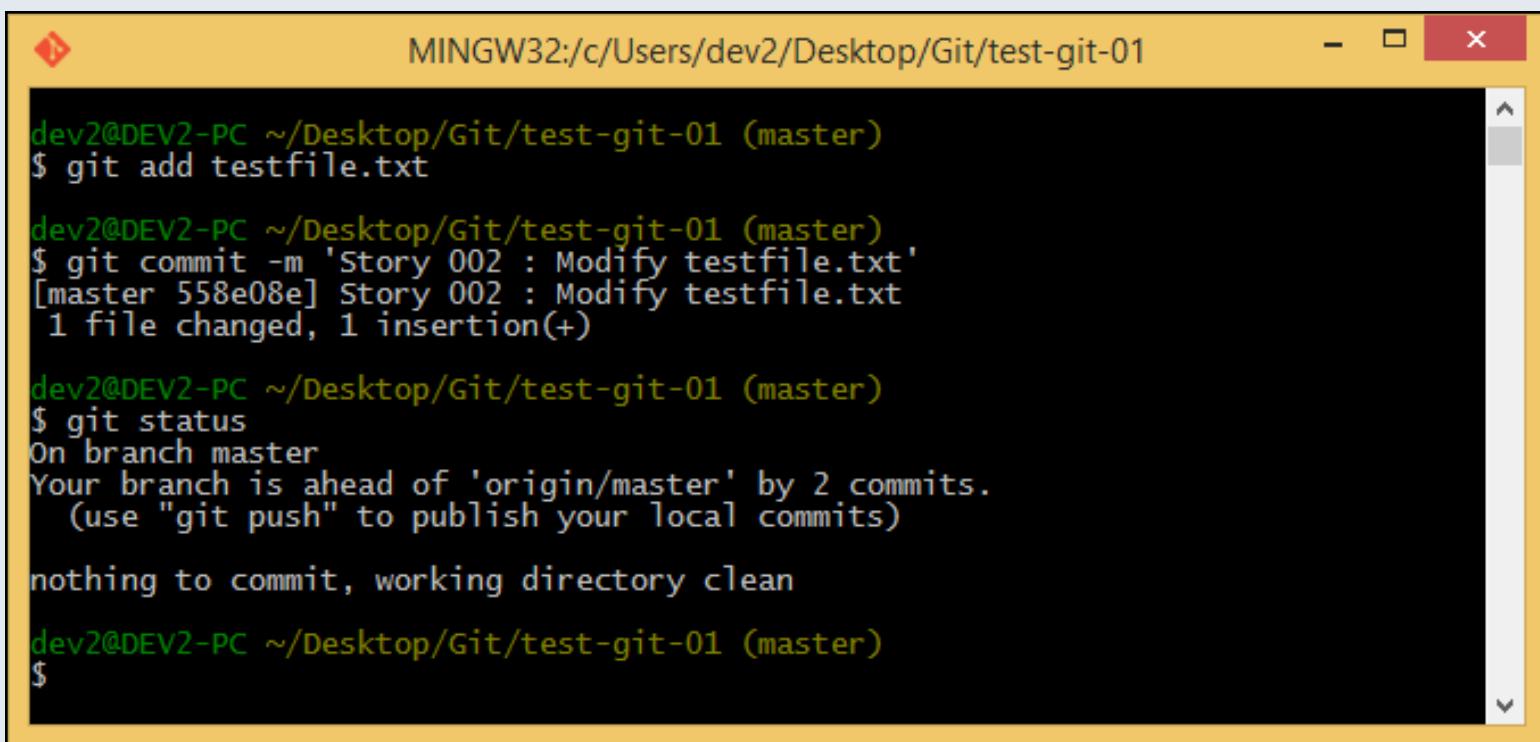
- It's similar to tracking a new file. You must run "git add [filename]" and "git commit -m 'commit message'" again. For example

```
git add testfile.txt
git commit -m 'Story 002 : Modify testfile.txt'
git status
```

### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

- You can see this result



The screenshot shows a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git/test-git-01". The terminal output is as follows:

```
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$ git add testfile.txt

dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$ git commit -m 'Story 002 : Modify testfile.txt'
[master 558e08e] Story 002 : Modify testfile.txt
 1 file changed, 1 insertion(+)

dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)

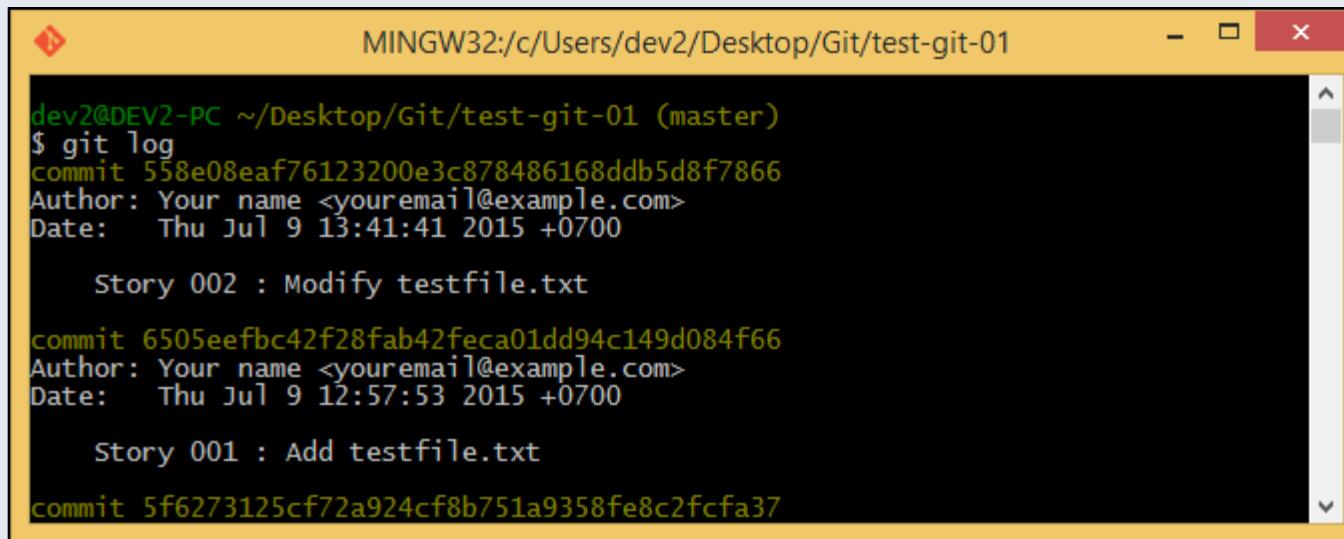
nothing to commit, working directory clean

dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$
```

### 3. Getting started with Git and GitHub

#### 3-3 Recording Changes to the Repository

- Remember that when you create, modify, delete, rename or something that makes a change to a file and you want to track the file, you have to run "git add [filename]" and "git commit 'commit message'" everytime changes are made.
- If you want to see commit history. You can run "**git log**" to see the commit history.



The screenshot shows a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git/test-git-01". The window contains the following command and its output:

```
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$ git log
commit 558e08eaf76123200e3c878486168ddb5d8f7866
Author: Your name <youremail@example.com>
Date: Thu Jul 9 13:41:41 2015 +0700

    Story 002 : Modify testfile.txt

commit 6505eefbc42f28fab42feca01dd94c149d084f66
Author: Your name <youremail@example.com>
Date: Thu Jul 9 12:57:53 2015 +0700

    Story 001 : Add testfile.txt

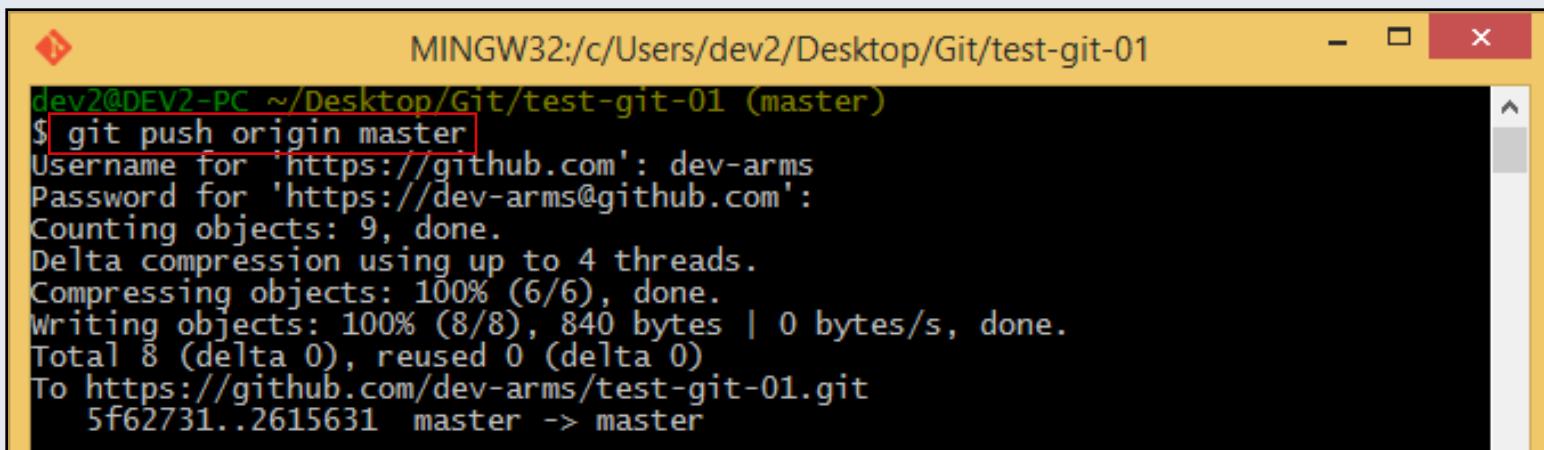
commit 5f6273125cf72a924cf8b751a9358fe8c2fcfa37
```

### 3. Getting started with Git and GitHub

#### 3-4 Pushing to a remote

- Pushing is to put your commits on local to your origin repository (on GitHub)
- Run "git push origin [branch name]" to push to your origin repository
- When you run this command, it will ask username and password. Type your Github's username and password.

```
git push origin master
```



A screenshot of a terminal window titled "MINGW32:/c/Users/dev2/Desktop/Git/test-git-01". The window shows the command \$ git push origin master being run. The output of the command is displayed below, showing the process of pushing changes to a GitHub repository named "dev-arms/test-git-01". The terminal window has a yellow header bar and a black body. The command and its output are highlighted with a red box.

```
dev2@DEV2-PC ~/Desktop/Git/test-git-01 (master)
$ git push origin master
Username for 'https://github.com': dev-arms
Password for 'https://dev-arms@github.com':
Counting objects: 9, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (8/8), 840 bytes | 0 bytes/s, done.
Total 8 (delta 0), reused 0 (delta 0)
To https://github.com/dev-arms/test-git-01.git
  5f62731..2615631  master -> master
```

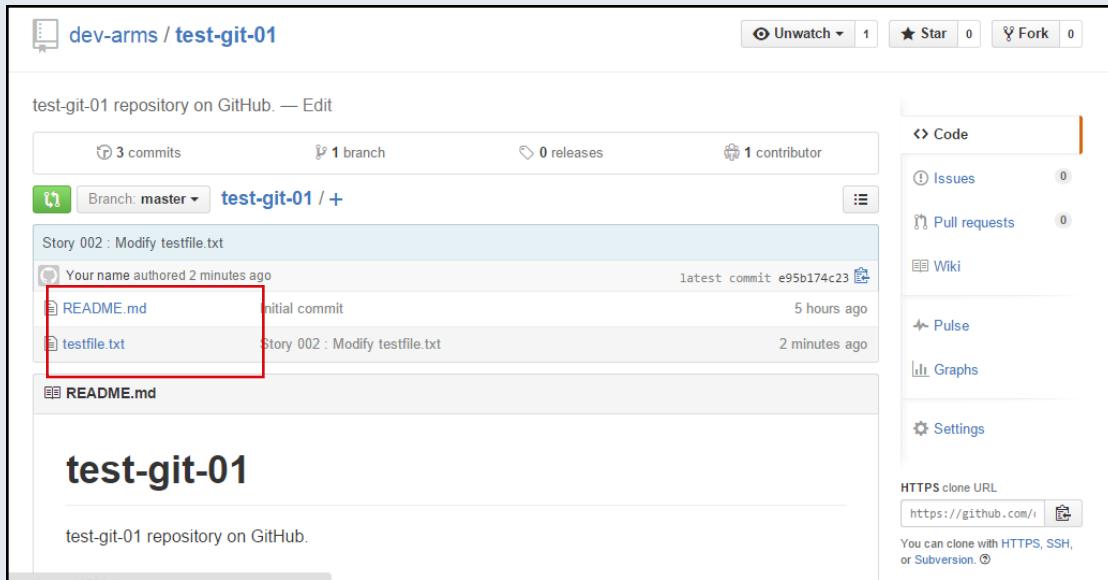
### 3. Getting started with Git and GitHub

#### 3-4 Pushing to a remote

- Go to your repository on Github website

`https://github.com/your_username/your_reponame`

- Now you can see new files(testfile.txt) pushed from your local.



### 3. Getting started with Git and GitHub

#### 3-4 Pushing to a remote

- When you pushed something new to a remote repository on Github and you have collaborators that had cloned your repository before you pushed.
  - Those collaborators can run "**git pull**" to merge changes into their local repository, that way they can also have the same code as yours.
- 
- Important point to remember, with Git, you can also manage even Word files and other documents, so you can use Git for even managing your documents.

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Try team collaboration using “your” repository
- To try team collaboration, you have to add collaborators for your repository.

Settings tab

Collaborators

Search and Add collaborators

The screenshot shows the GitHub repository settings interface. At the top, there are navigation links: Code, Issues 0, Pull requests 0, Wiki, Pulse, Graphs, and Settings. The Settings link is highlighted with a red box. On the left, a sidebar menu has 'Options' at the top, followed by 'Collaborators' which is also highlighted with a red box. Below that are 'Branches', 'Webhooks & services', and 'Deploy keys'. The main content area is titled 'Collaborators' and includes a sub-header 'Push access to the repository'. It states, 'This repository doesn't have any collaborators yet. Use the form below to add a collaborator.' Below this is a search bar with the placeholder 'Search by username, full name or email address' and an 'Add collaborator' button. Both the search bar and the 'Add collaborator' button are also highlighted with a red box.

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Try to do something to “your repository”. For example, add a new file
- First, Create a new Issue (An Issue is a note on a repository about something that needs attention. It could be a bug, a feature request, a question or lots of other things)
  - Click the Issues tab from the sidebar on your "bu-arms-xx" repository
  - Click New Issue
  - Give your Issue a title and description. For example
    - title : **[your name] : I cannot run your Project**
    - description : **[your name] : I cannot run your Project, Why ???**

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

The screenshot shows the GitHub Issues interface. On the left, there's a sidebar with links: Code, Issues (which is selected and highlighted with a red box), Pull requests, Wiki, Pulse, and Graphs. A large black arrow points from the 'Issues' link in the sidebar to the 'Issues' tab in the main header. The main area shows an issue titled "team-arms : I cannot run your Project" with a detailed description below it: "team-arms : I cannot run your Project. Why ???". There are 'Write' and 'Preview' tabs at the top of the issue editor. Below the issue title, there's a note: "Attach images by dragging & dropping, selecting them, or pasting from the clipboard." At the bottom right of the editor, there's a green button labeled "Submit new issue".

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- You can see your issue and others' issue

The screenshot shows a GitHub repository page for 'dev-arms / bu-arms-00'. At the top, there are buttons for 'Issues', 'Pull requests', 'Labels', and 'Milestones'. A search bar contains the query 'is:issue is:open'. Below the search bar, it says '2 Open' and '0 Closed'. There are two issues listed:

- dev-arms : File not Found, Why ???**  
#2 opened 3 minutes ago by dev-arms
- team-arms : I cannot run your Project**  
#1 opened 4 minutes ago by team-arms

#### team-arms : I cannot run your Project #1

**Open** team-arms opened this issue 10 minutes ago · 0 comments



team-arms commented 10 minutes ago

Collaborator

team-arms : I cannot run your Project. Why ???

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- you can share and discuss each issue like below.

## dev-arms : File not Found, Why ??? #2

Open dev-arms opened this issue 8 minutes ago · 1 comment

 dev-arms commented 8 minutes ago Owner edit

dev-arms : File not Found, Why ???

 team-arms commented just now Collaborator edit x

i think you're using old version, plz update!!

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

team-arms : I cannot run your Project #1

**Open** team-arms opened this issue 16 minutes ago · 2 comments

team-arms commented 16 minutes ago

team-arms : I cannot run your Project, Why ???

Collaborator

team-arms commented 2 minutes ago

please copy your log and paste here

Owner

team-arms commented just now

[debug] application - Attempting risky calculation.  
[error] application - Exception with riskyCalculation

Collaborator

Labels None yet

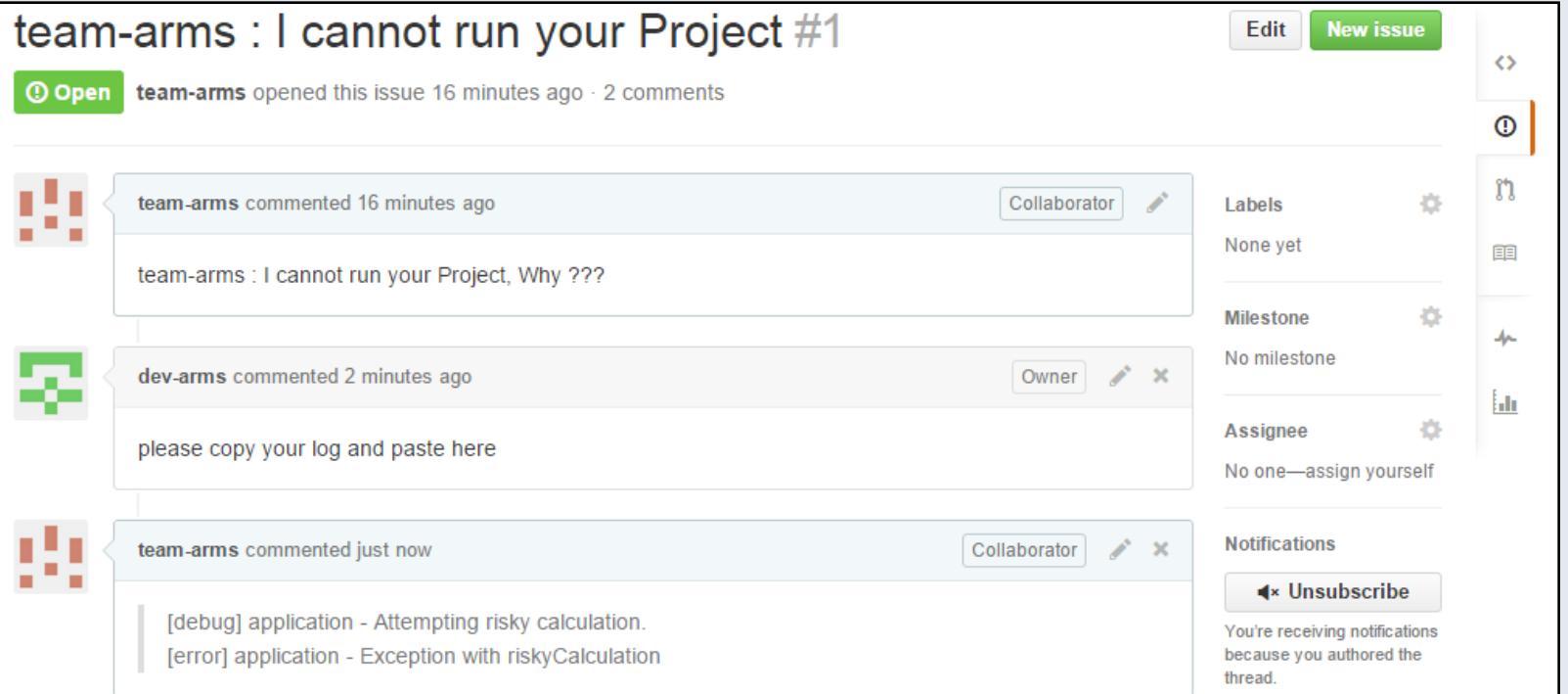
Milestone No milestone

Assignee No one—assign yourself

Notifications

Unsubscribe

You're receiving notifications because you authored the thread.



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- For example....., if someone found how to solve an issue. You can clone repository, modify, commit, and push, then use pull request to merge.
- Try, Clone your team repository

You can copy URL

Clone or download ▾

```
git clone https://github.com/username/repositoryname.git
```



A screenshot of a terminal window titled "MINGW32:c/Users/dev/Desktop/Git". The window shows the command \$ git clone https://github.com/dev-arms/bu-arms-00.git being run, followed by the output of the cloning process. The output includes: Cloning into 'bu-arms-00'..., remote: Counting objects: 3, done., remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0, Unpacking objects: 100% (3/3), done., Checking connectivity... done.

```
dev@DEV-PC ~/Desktop/Git
$ git clone https://github.com/dev-arms/bu-arms-00.git
Cloning into 'bu-arms-00'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
Checking connectivity... done.

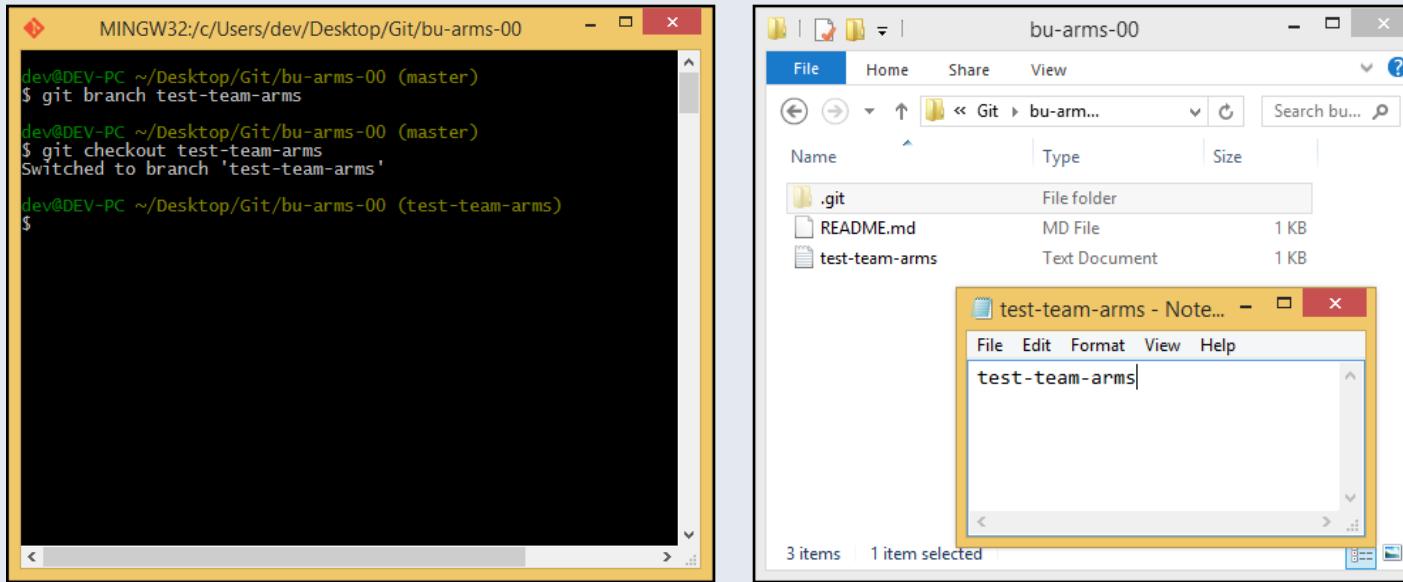
dev@DEV-PC ~/Desktop/Git
$
```

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Create branch "test-[your name]",
- check out to the branch and create a new file "test-[your name].txt"

```
git branch test-[your name]  
git checkout test-[yourname]
```



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Add "test-[your name].txt" file to commit, Commit and push to Github using your id

```
git add test-[your name].txt
git commit -m "Add a new text file by [your name]"
git push origin test-[your name]
```



A screenshot of a Windows terminal window titled "MINGW32:/c/Users/dev/Desktop/Git/bu-arms-00". The window contains the following command-line session:

```
dev@DEV-PC ~/Desktop/Git/bu-arms-00 (test-team-arms)
$ git add test-team-arms.txt
dev@DEV-PC ~/Desktop/Git/bu-arms-00 (test-team-arms)
$ git commit -m "Add a new text file by team-arms"
[team-arms ec7f4e2] Add a new text file by team-arms
 1 file changed, 1 insertion(+)
 create mode 100644 test-team-arms.txt

dev@DEV-PC ~/Desktop/Git/bu-arms-00 (test-team-arms)
$ git push origin test-team-arms
Username for 'https://github.com': team-arms
Password for 'https://team-arms@github.com':
Counting objects: 4, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 314 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/dev-arms/bu-arms-00.git
 * [new branch]      test-team-arms -> test-team-arms

dev@DEV-PC ~/Desktop/Git/bu-arms-00 (test-team-arms)
$
```

### 3. Getting started with Git and GitHub

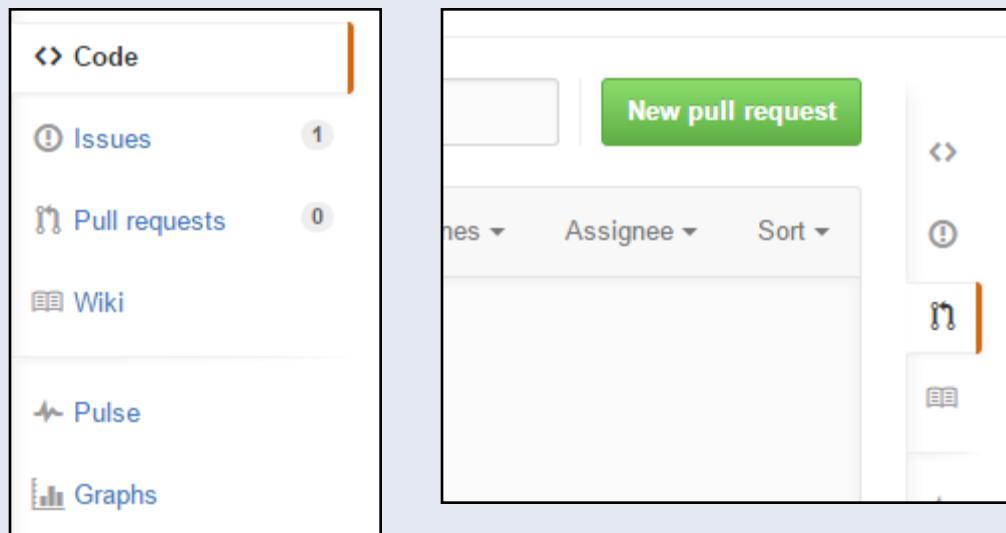
#### 3-5 Team collaboration

- We will use "Pull request"
- **Pull Requests are the heart of collaboration on GitHub**
- When you make a pull request, you're proposing changes in source code and requesting collaborators (and merge them into their branch)
- GitHub's Pull Request feature allows you to compare the content on two branches. The changes, additions and subtractions, are shown in green and red and called diffs (differences)
- People use Pull Requests to discuss commits (code review) even before the code is finished, that way you can get feedback as you go or help when you're stuck

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Create a Pull Request
- Click the Pull Request icon on the sidebar and click "New pull request" on your repository



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Select the branch(test-[your name]) to compare with master
- Look over your changes in the diffs, make sure they're what you want to submit

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base: master ... compare: test-team-arms Able to merge. These branches can be automatically merged.

[Create pull request](#) Discuss and review the changes in this comparison with others. [?](#)

1 commit 1 file changed 0 commit comments 1 contributor

Commits on Jul 23, 2015

- Your name Add a new text file by team-arms ec7f4e2

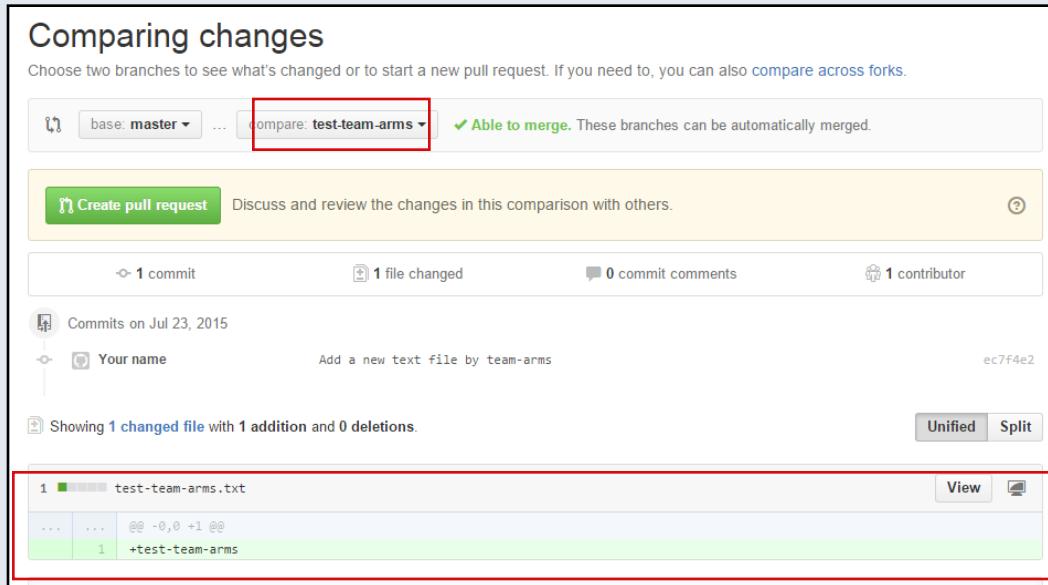
Showing 1 changed file with 1 addition and 0 deletions.

Unified Split

test-team-arms.txt

1	... ...	@@ -0,0 +1 @@ +test-team-arms
---	------------	----------------------------------

View



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- When you're satisfied with changes, click "Create Pull Request"

**Comparing changes**

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare against a commit, tag, or pull request.

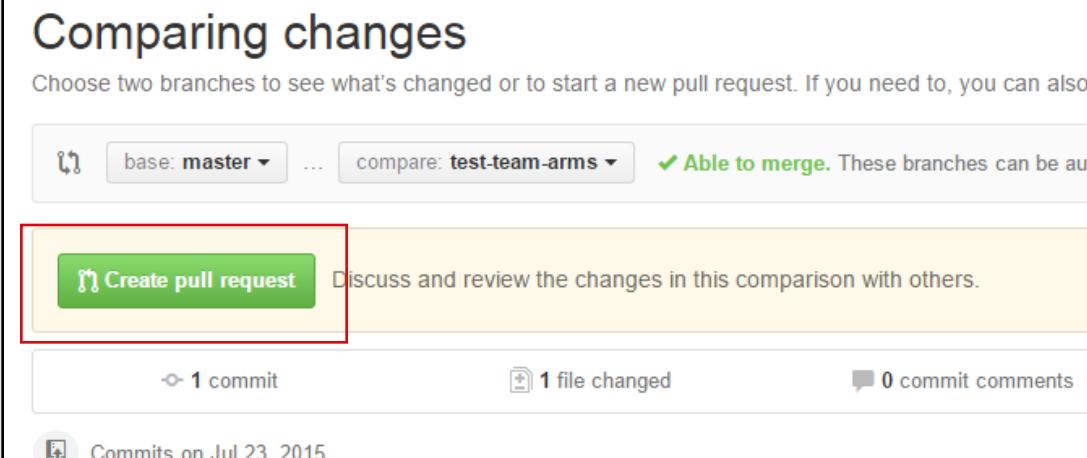
base: **master** ▾ ... compare: **test-team-arms** ▾ **Able to merge**. These branches can be automatically merged.

**Create pull request**

Discuss and review the changes in this comparison with others.

1 commit 1 file changed 0 commit comments

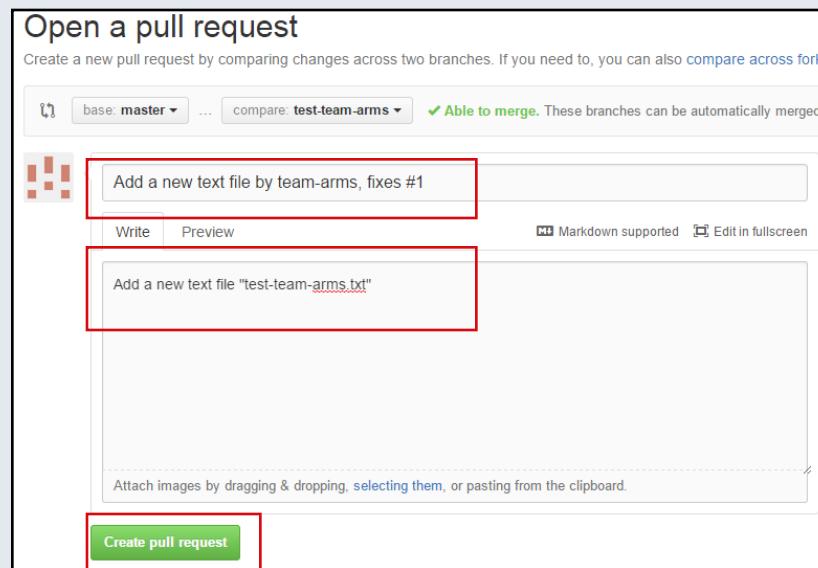
Commits on Jul 23, 2015

A screenshot of a GitHub interface showing a comparison between 'master' and 'test-team-arms'. The 'Create pull request' button is highlighted with a red box. The interface shows 1 commit, 1 file changed, and 0 commit comments. A note at the bottom says 'Commits on Jul 23, 2015'.

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

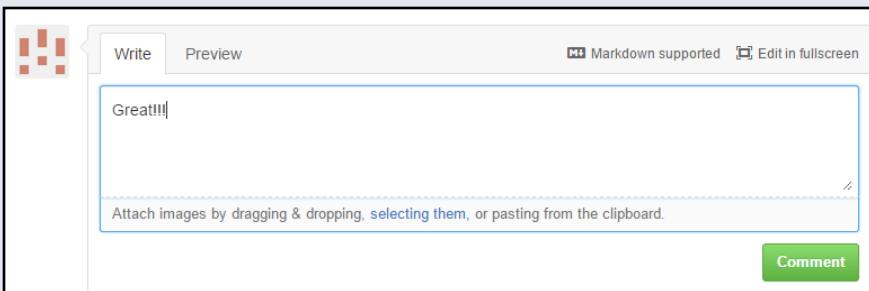
- Give your pull request a title that relates directly to an issue, include “fixes #” and the issue number in the title. Write a description of your changes
  - title : **Add a new text file by [your name], fixes #1**
  - description : **Add a new text file "test-[your names].txt"**



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- You can comment for suggest, blame, tell the bug, and etc to this Pull request



A screenshot of a GitHub pull request interface. The title of the pull request is 'Add a new text file by team-arms, fixes #1 #3'. It shows an 'Open' button and a message from 'team-arms' wanting to merge 1 commit into 'master' from 'test-team-arms'. Below this, there are three comments:

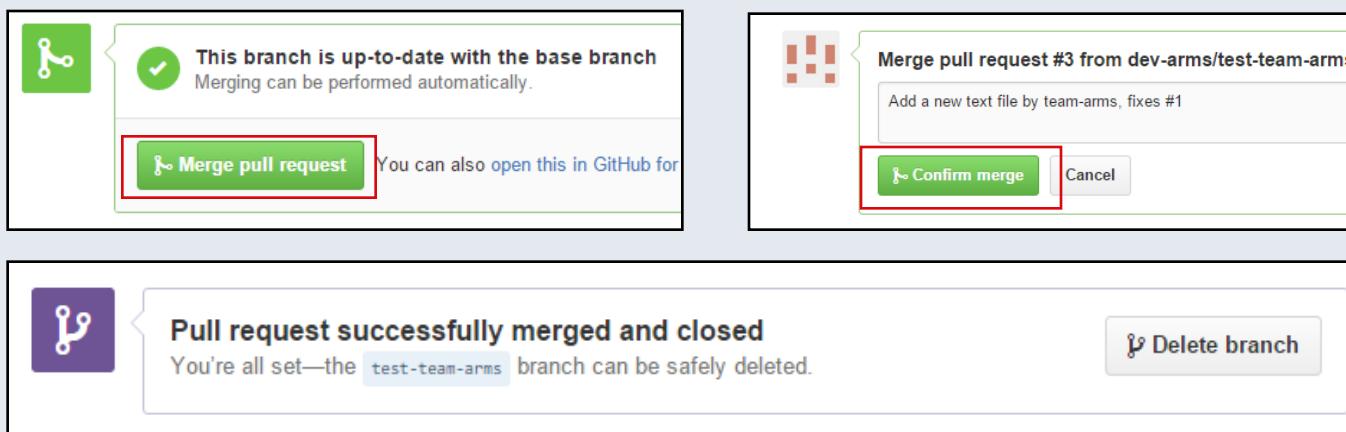
- 'team-arms commented 2 minutes ago': 'Add a new text file "test-team-arms.txt"' (Collaborator)
- 'dev-arms commented 16 seconds ago': 'Great!!!' (Owner)

The commit hash 'ec7f4e2' is also visible.

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Bring changes together – merge your "test-[your name]" branch into the "master" branch
  - Click "Merge pull request" to merge the changes into master
  - Click "Confirm merge"
  - Go ahead and delete the branch with the "Delete branch", (since its changes have been incorporated)



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- the issue is closed

The screenshot shows a GitHub repository interface for 'dev-arms / bu-arms-00'. The top navigation bar includes 'Unwatch' (2), 'Star' (0), and 'Fork' (0) buttons. Below the bar, there are tabs for 'Issues' (selected), 'Pull requests', 'Labels', and 'Milestones'. A search bar contains the query 'is:issue is:closed'. A green 'New issue' button is visible. A red box highlights the '1 Closed' status under the filter dropdown. The main area displays one closed issue from 'team-arms': 'team-arms : I cannot run your Project' (commented 2 hours ago by team-arms). A 'ProTip!' message at the bottom suggests following discussions with more than 50 comments.

dev-arms / bu-arms-00

Issues Pull requests Labels Milestones

Filters ▾ is:issue is:closed New issue

Clear current search query, filters, and sorts

1 Open ✓ 1 Closed

team-arms : I cannot run your Project  
#1 opened an hour ago by team-arms

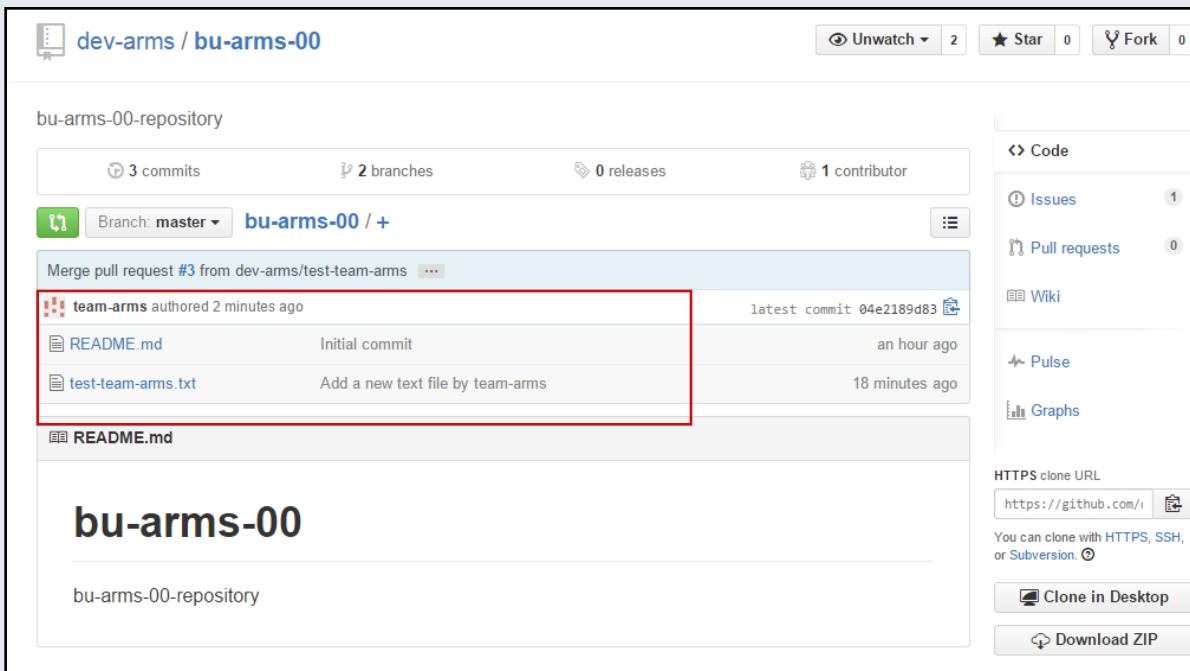
Author ▾ Labels ▾ Milestones ▾ Assignee ▾ Sort ▾

ProTip! Follow long discussions with comments:>50.

### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

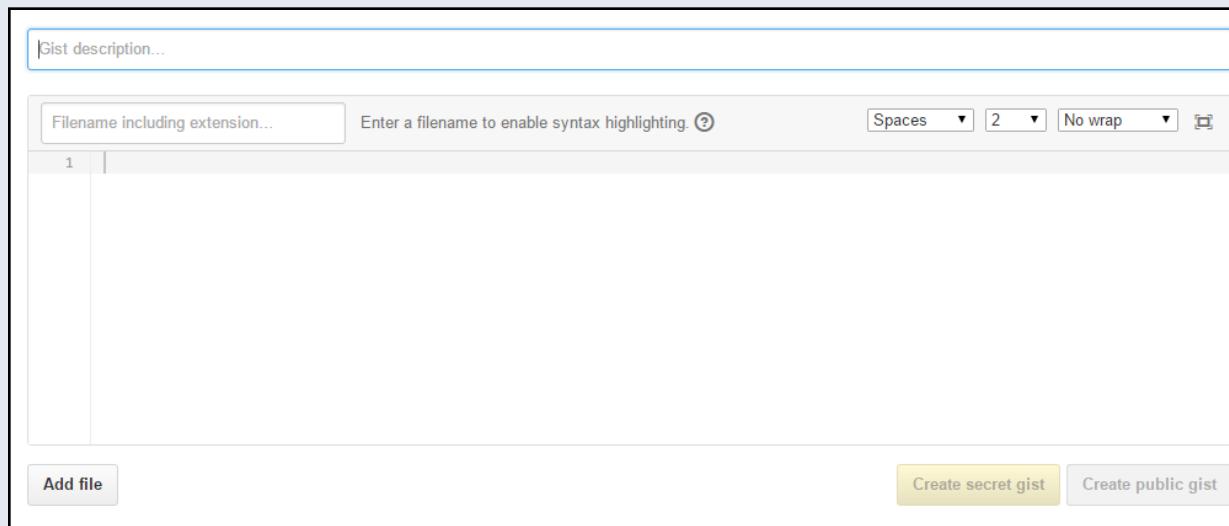
- You can see a new file or a lot of files (depends on the number of your members) on master branch because it's merged by pull request



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- For improving team collaboration, you can use Gist on Github
- Gist is a simple way to share snippets and pastes with others. All gists are Git repositories, so they are automatically versioned, forkable and usable from Git
- Try to create Gist, Open "<https://gist.github.com/>"



### 3. Getting started with Git and GitHub

#### 3-5 Team collaboration

- Try to give snippet. For example, give Spring MVC Controller example

The screenshot shows the GitHub Gist creation interface at <https://gist.github.com>. A red box highlights the text "\* Snippet is simple code that you want to share. you can give any code for trying". The interface includes fields for "Gist description..." and "File name" (HelloController.java). The code editor contains the following Java code:

```
1 package com.arms.app.hello;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Controller;
5 import org.springframework.web.bind.annotation.RequestMapping;
6 import org.springframework.web.servlet.ModelAndView;
7
8 import com.arms.repository.HelloRepository;
9
10 @Controller
11 @RequestMapping("/")
12 public class HelloController {
13
14     @Autowired
15     HelloRepository helloRepository;
16
17     @RequestMapping("")
18     public ModelAndView index(ModelAndView modelAndView){
19         modelAndView.addObject("list", helloRepository.findAll());
20         modelAndView.setViewName("hello/list");
21         return modelAndView;
22     }
}
```

Buttons for "Create secret gist" and "Create public gist" are visible at the bottom.

- Click "Create secret gist" or "Create public gist" depends on your need
- Secret gists are hidden from search engines but visible to anyone you give the URL

# 3. Getting started with Git and GitHub

## 3-7 Team collaboration

- Now you can share snippet with Gist to your team

The screenshot shows a GitHub Gist page. At the top, there's a search bar, navigation links for 'All gists' and 'GitHub', and a 'New gist' button. Below that, a user profile icon and the title 'mk30715 / HelloController.java Secret' are displayed, along with a note 'Created 5 minutes ago'. There are buttons for 'Edit', 'Delete', 'Star', and a copy icon. A tab bar shows 'Code' (which is selected) and 'Revisions 1'. Below the tabs are buttons for 'Embed', 'Raw', and 'Download ZIP'. The main area contains the Java code for 'HelloController.java' with line numbers from 1 to 22.

```
1 package com.armz.app.hello;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Controller;
5 import org.springframework.web.bind.annotation.RequestMapping;
6 import org.springframework.web.servlet.ModelAndView;
7
8 import com.armz.repository.HelloRepository;
9
10 @Controller
11 @RequestMapping("/")
12 public class HelloController {
13
14     @Autowired
15     HelloRepository helloRepository;
16
17     @RequestMapping("")
18     public ModelAndView index(ModelAndView modelAndView){
19         modelAndView.addObject("list", helloRepository.findAll());
20         modelAndView.setViewName("hello/list");
21         return modelAndView;
22     }
}
```

## 4. Git commands

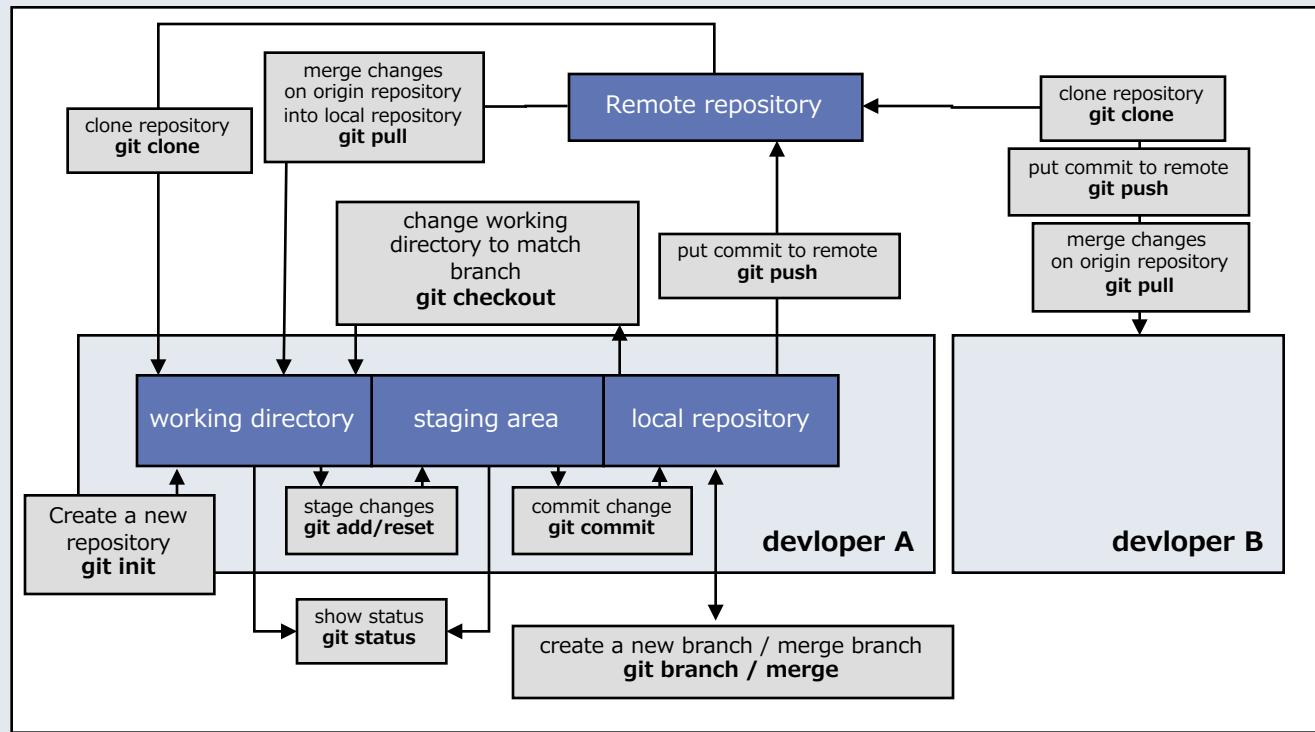
### 4-1 Git commands

•git init	Create a new Git local repository
•git clone	Clone a Git repository into a new directory
•git status	Show the working directory status
•git add	Add files to the staging area for tracking and committing
•git reset	Remove the specified file from the staging area
•git commit	Record changes to the repository
•git push	Put your commits on local to your origin repository
•git pull	Merge changes on origin repository into local repository
•git checkout	Updates the files in the working directory to match the version stored in that branch
•git branch	Create, list, rename, and delete branches
•git merge	Merge the specified branch into the current branch
•git log	Show commit history

# 4. Git commands

## 4-1 Git commands

- Simple command cycle





# ARMS

*Your Idea Leads Your Ideals*

homepage: <http://arms-asia.com/>

facebook: <https://www.facebook.com/arms.asia?fref=ts>