Git – Part 1

Vineel Kovvuri

Senior SDE @ Microsoft

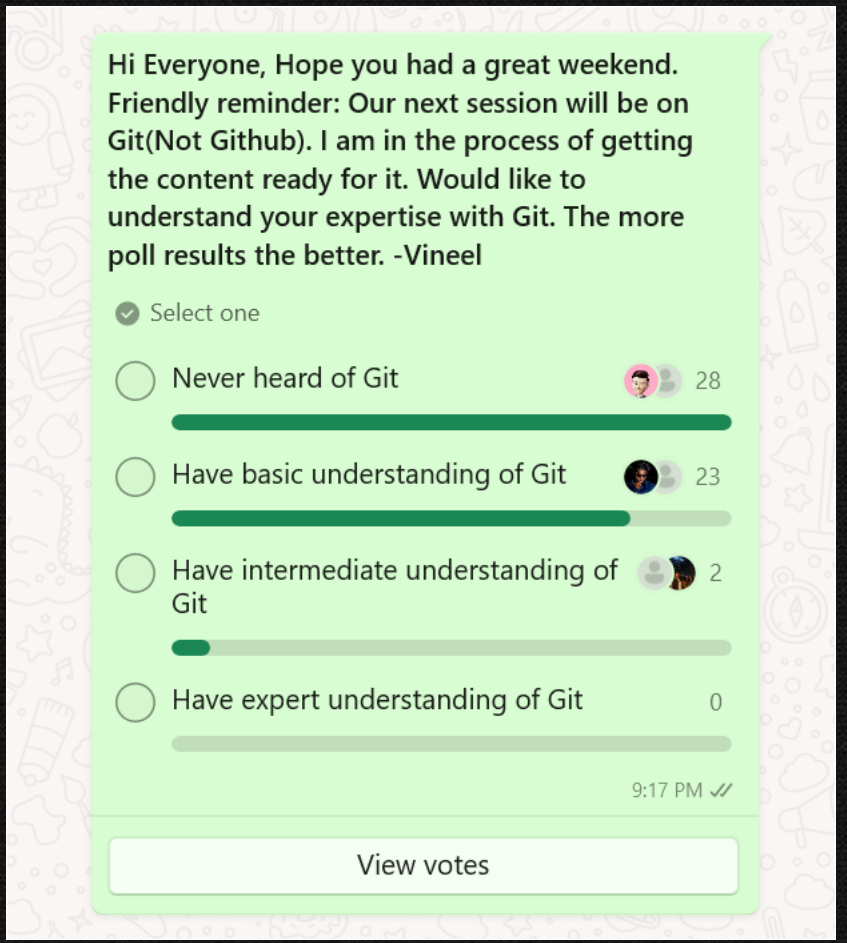
https://vineelkovvuri.github.io

Agenda

- What is Git and why should you care?
- Installation – Configuring Git
- How to initialize a git repository?
- Git Basics★
- What is HEAD in Git?
- Git log
- Git difftool
- Undoing changes in Git

Not in Agenda

- Branches
- Push/Pull
- Rebase
- Blame
- Bisect
- Github



What is Git? Why should you care?

Git is a software that keeps track of changes to your source code. Source Code Management software

Much like your bank account's passbook, which keeps track of every transaction, but for source code.

Passbook/Bank Statement

1. Debit
2. Credit

DATE	DESCRIPTION	WITHDRAWAL	DEPOSIT	BALANCE
	Previous balance			**27,584.38**
03/02	Internet Bill	75.99		27,508.39
03/05	Electric Bill	253.68		27,254.71
03/06	Check No. 4598 *Payment from Lisa Williams*		456.84	27,711.55
03/10	Deposit from Credit Card Processor		5,891.26	33,602.81
03/12	Payroll Run	3,894.75		29,708.06
03/16	Debit Transaction *Main Office Wholesale*	243.46		29,464.60
03/21	Rent Bill	750.00		28,714.60
03/21	Check No. 234 *Payment From Mark Moore*		268.84	28,983.44
03/26	Payroll Run	3,743.23		25,240.21
03/28	Deposit		3,656.45	28,896.66
03/29	Debit Transaction *ABC Business Supplies*	1,548.96		27,347.70
	Ending balance			**27,347.70**

Changes possible in source code

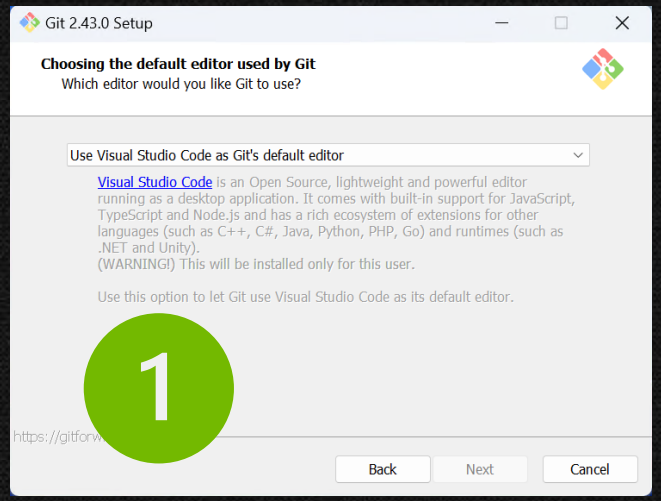
1. Modify existing files
2. Add new files
3. Delete existing files

```
commit d5b1a4f0958c860e825bfd5827cb985a4b86147d (HEAD -> master)
Author: Eric Biggers <ebiggers3@gmail.com>
Date:   Fri Oct 27 21:04:32 2023 -0700

    fuzz.sh: add --max-len option

tools/libFuzzer/fuzz.sh

commit 2da8c3dac2d66ededac860a1147e8794bf557a46
Author: Eric Biggers <ebiggers3@gmail.com>
Date:   Sat Sep 16 16:21:11 2023 -0700

    README.WINDOWS.md: use https URL instead of git

README.WINDOWS.md

commit 4baa0ab7b8ad5c7f0a109e22e9a6c8621a7edd42 (tag: v1.14.3)
Author: Eric Biggers <ebiggers3@gmail.com>
Date:   Mon Sep 4 11:41:23 2023 -0700

    v1.14.3

Makefile.am
NEWS.md
README.WINDOWS.md
README.md
doc/man1/mkwinpeimg.1
```

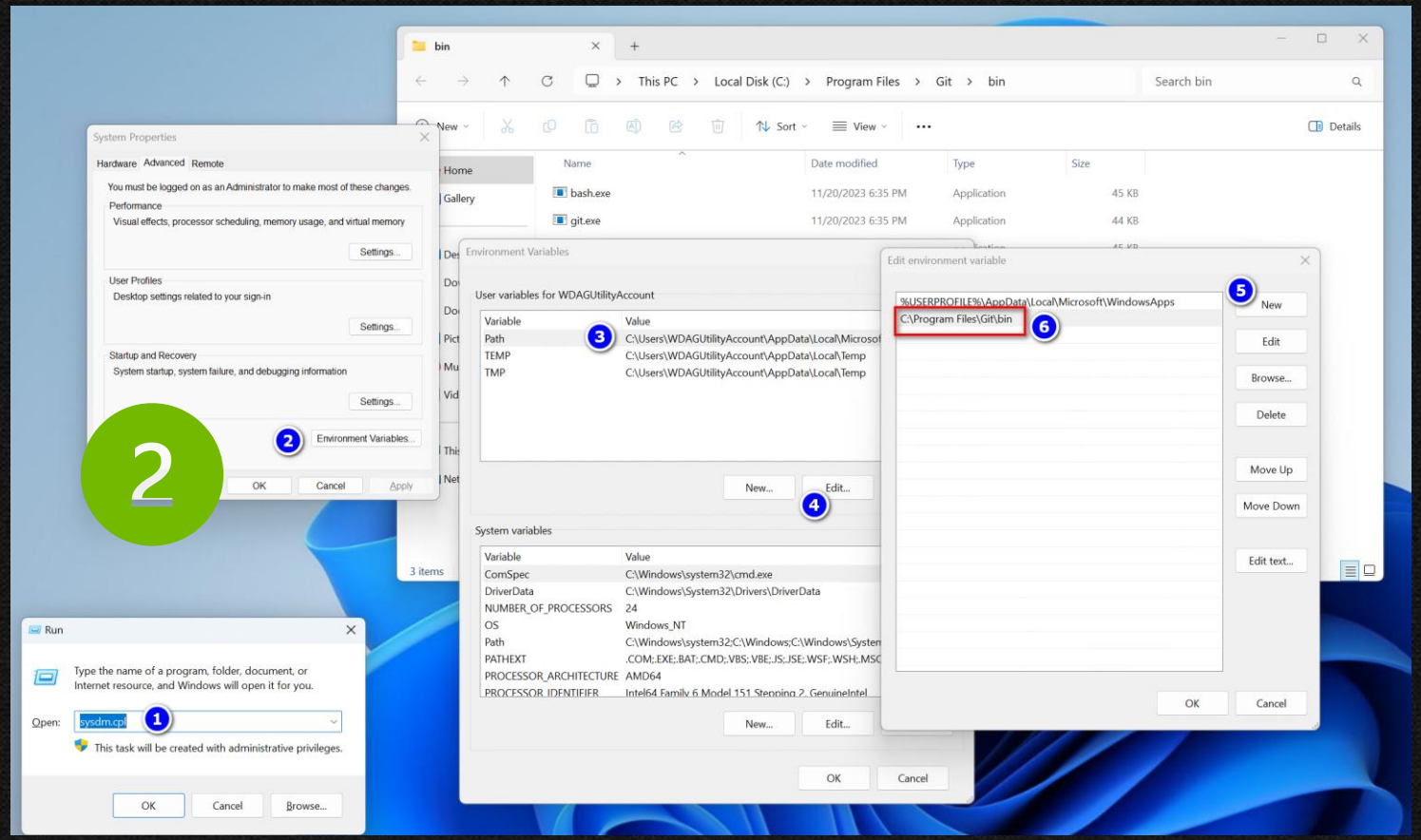
- Git is used by 99.99% of the projects

Installation – Configure Git

https://git-scm.com/



Add git.exe to Path Environment variable



Set name and email

```
C:\>git config --global user.name "Vineel Kovvuri"

C:\>git config --global user.email "vineel.kovvuri@gmail.com"

C:\>
```

3

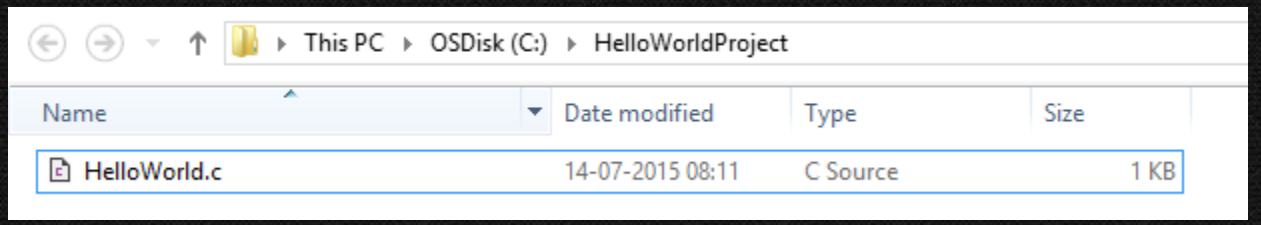
Please do not use any git GUI tools 😁

Git Commands

Demo Time

```
C:\HelloWorldProject>git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--config-env=<name>=<envvar>] <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone     Clone a repository into a new directory
   init      Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add       Add file contents to the index
   mv        Move or rename a file, a directory, or a symlink
   restore   Restore working tree files
   rm        Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect    Use binary search to find the commit that introduced a bug
   diff      Show changes between commits, commit and working tree, etc
   grep      Print lines matching a pattern
   log       Show commit logs
   show      Show various types of objects
   status    Show the working tree status

grow, mark and tweak your common history
   branch    List, create, or delete branches
   commit    Record changes to the repository
   merge     Join two or more development histories together
   rebase    Reapply commits on top of another base tip
   reset     Reset current HEAD to the specified state
   switch    Switch branches
   tag       Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch     Download objects and refs from another repository
   pull      Fetch from and integrate with another repository or a local branch
   push      Update remote refs along with associated objects
```

How to initialize a git repository?

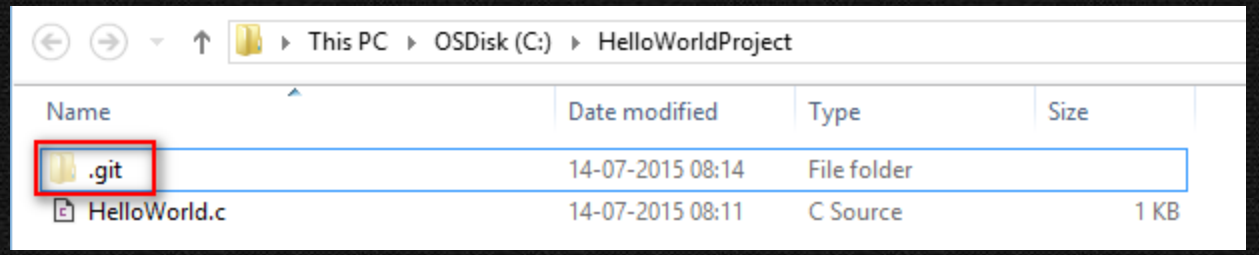



```
C:\HelloWorldProject>git status
fatal: Not a git repository (or any of the parent directories): .git
```

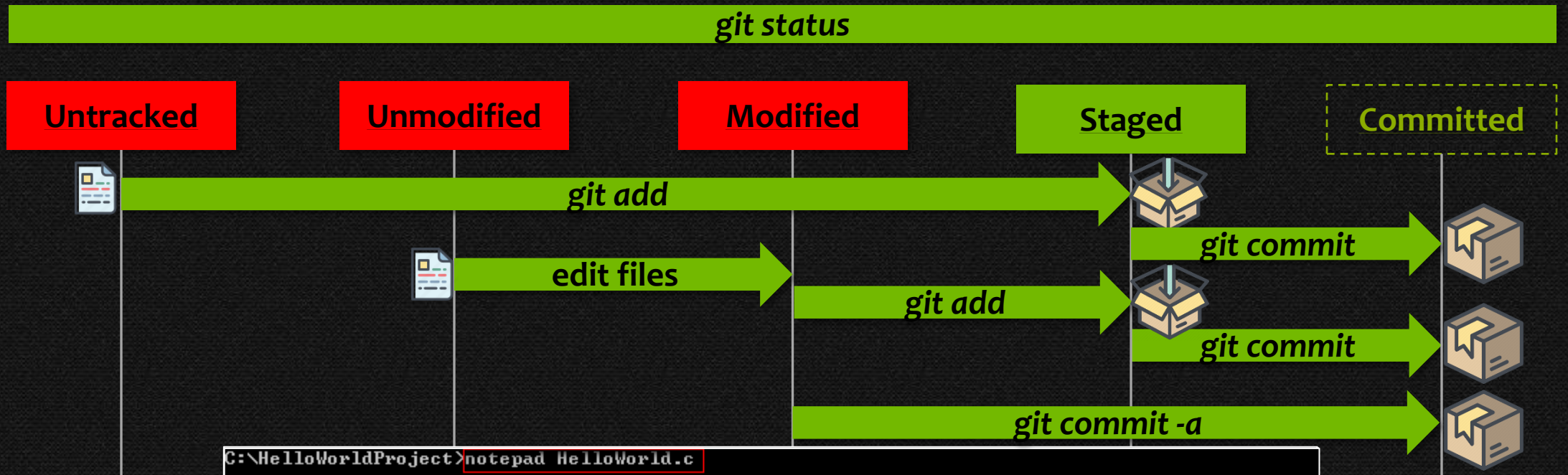
Demo Time

git init .

```
C:\HelloWorldProject>git init .
Initialized empty Git repository in C:/HelloWorldProject/.git/
```



Git Basics




```
C:\HelloWorldProject>notepad HelloWorld.c

C:\HelloWorldProject>git status
On branch 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:   HelloWorld.c

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

C:\HelloWorldProject>git commit -a
[master 5aea899] Comment Added
 1 file changed, 1 insertion(+), 1 deletion(-)

C:\HelloWorldProject>git status
On branch master
nothing to commit, working directory clean

C:\HelloWorldProject>
```

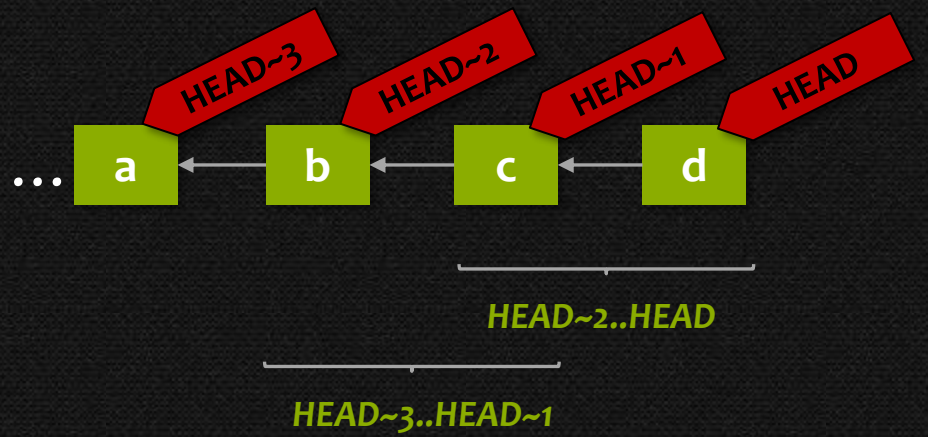
Git log

git log show history of commits(aka bank statement)


```
C:\MyProject>git log
commit 302dddb8c5886e28a861e804b2a6556c2f90fa92
Merge: d751102 f3f8a35
Author: Vineel Kumar Reddy Kovvuri <vineel.kovvuri@gmail.com>
Date:   Mon Jul 13 21:50:23 2015 +0530

    Merge branch 'opt_helloworld'

commit d7511025d2361e66626b943aea570736dc17dfd9
Author: Vineel Kumar Reddy Kovvuri <vineel.kovvuri@gmail.com>
Date:   Mon Jul 13 21:46:58 2015 +0530

    Comment added

commit f3f8a3518871e1a0e06605f79a5ab9e104161cc1
Author: Vineel Kumar Reddy Kovvuri <vineel.kovvuri@gmail.com>
Date:   Mon Jul 13 21:45:03 2015 +0530

    Optimised Hello World

commit ee8a73a41b5b1273720aef1f56f8a1aaa412c207
Author: Vineel Kumar Reddy Kovvuri <vineelko@microsoft.com>
Date:   Mon Jul 13 21:26:17 2015 +0530

    Adding .gitignore

commit 940a3a6bfd4460cedc1738b645d05fa4505d29ec
Author: Vineel Kumar Reddy Kovvuri <vineelko@microsoft.com>
Date:   Mon Jul 13 19:41:10 2015 +0530

    My First helloWorld commit

C:\MyProject>
```

Git HEAD

- HEAD always refers to the latest commit on the current branch
- HEAD~1 always refers to the commit one before the latest commit
- HEAD~2, HEAD~3, …



- .. Syntax(revision/range syntax) is used to refer a range of commits
- *HEAD~2..HEAD* means all commit b/w HEAD~2 and HEAD not including HEAD~2

Git log

git log show history of commits


```
C:\MyProject>git log
commit 302dddb8c5886e28a861e804b2a6556c2f90fa92
Merge: d751102 f3f8a35
Author: Vineel Kumar Reddy Kovvuri <vineel.kovvuri@gmail.com>
Date:   Mon Jul 13 21:50:23 2015 +0530

    Merge branch 'opt_helloworld'

commit d7511025d2361e66626b943aea570736dc17dfd9
Author: Vineel Kumar Reddy Kovvuri <vineel.kovvuri@gmail.com>
Date:   Mon Jul 13 21:46:58 2015 +0530

    Comment added

commit f3f8a3518871e1a0e06605f79a5ab9e104161cc1
Author: Vineel Kumar Reddy Kovvuri <vineel.kovvuri@gmail.com>
Date:   Mon Jul 13 21:45:03 2015 +0530

    Optimised Hello World

commit ee8a73a41b5b1273720aef1f56f8a1aaa412c207
Author: Vineel Kumar Reddy Kovvuri <vineelko@microsoft.com>
Date:   Mon Jul 13 21:26:17 2015 +0530

    Adding .gitignore

commit 940a3a6bfd4460cedc1738b645d05fa4505d29ec
Author: Vineel Kumar Reddy Kovvuri <vineelko@microsoft.com>
Date:   Mon Jul 13 19:41:10 2015 +0530

    My First helloWorld commit

C:\MyProject>
```

HEAD

HEAD~1

HEAD~2

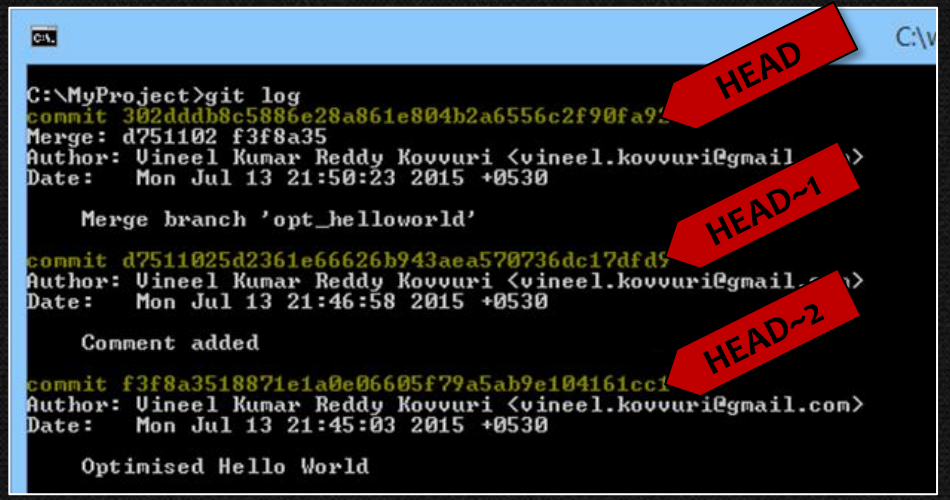
HEAD~3

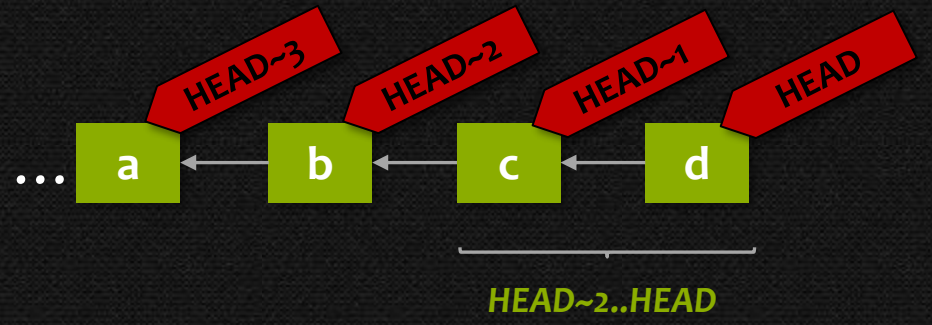
HEAD~4

Git difftool

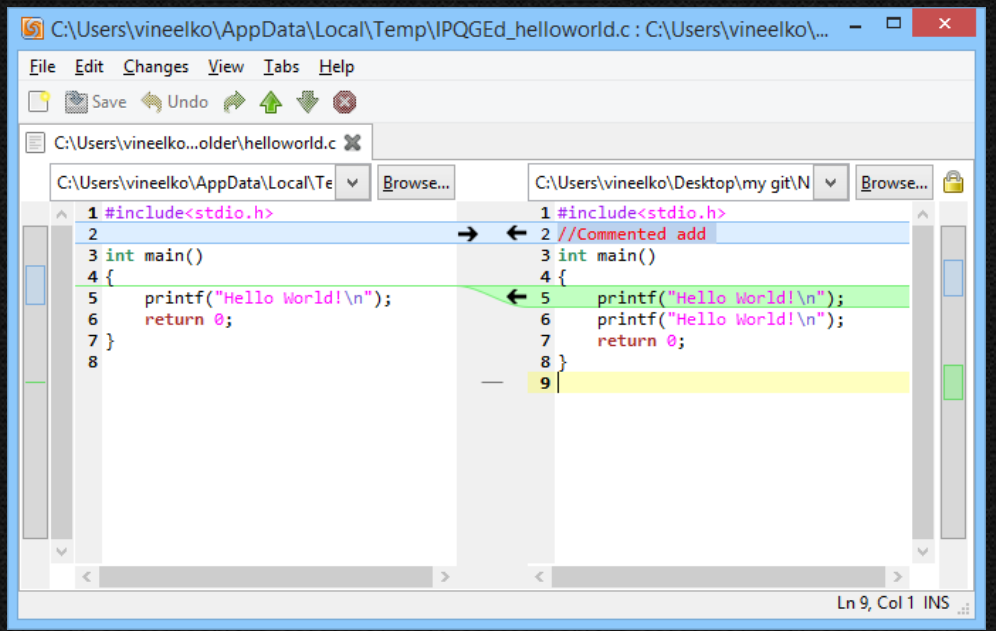
http://sourceforge.net/projects/meld-installer/

```
C:\repos>git config --global diff.tool meld

C:\repos>git config --global difftool.meld.path "C:\Program Files (x86)\Meld\Meld\Meld.exe"
```

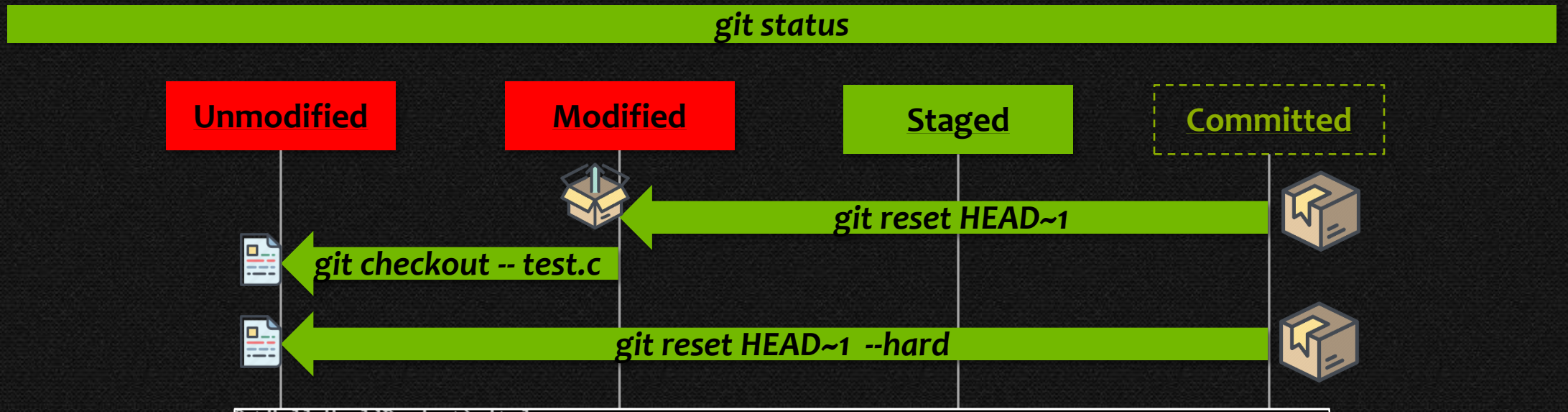




git difftool -d
git difftool -d HEAD~2..HEAD



Undo changes in Git




```
C:\HelloWorldProject>git log
commit cb3b5d203281e8203373528447a18bcea086a342
Author: Vineel Kumar Reddy Kovvuri <vineelko@microsoft.com>
Date:   Tue Jul 14 20:14:34 2015 +0530

    Added comments to the helloworld!

commit e00b227c5ce76721190f983f34958e3a1aab283b
Author: Vineel Kumar Reddy Kovvuri <vineelko@microsoft.com>
Date:   Tue Jul 14 20:08:39 2015 +0530

    First HelloWorld Program

C:\HelloWorldProject>git reset HEAD~1 --hard
HEAD is now at e00b227 First HelloWorld Program

C:\HelloWorldProject>git log
commit e00b227c5ce76721190f983f34958e3a1aab283b
Author: Vineel Kumar Reddy Kovvuri <vineelko@microsoft.com>
Date:   Tue Jul 14 20:08:39 2015 +0530

    First HelloWorld Program

C:\HelloWorldProject>
```

Recap

Create Repo	git init	Initialize a repository
Inspect Repo	git status	Know the status of the repository
Create Commits	git add	Add files for staging
	git commit	Create commit of the staged files
Inspect Commits	git log	View the commit log
	git diff/difftool	See changes between the commits
Undo Commits	git reset	Undo commit(unpack the commit)
	git checkout	Discard the changes

Git – Part 2

Vineel Kovvuri

Senior SDE @ Microsoft

https://vineelkovvuri.github.io

Agenda

- Recap from part 1
- Branches
- Merging Branches
- Rebasing Branches
- Resolving Conflicts

Not in Agenda

- Push/Pull/Fetch
- Github

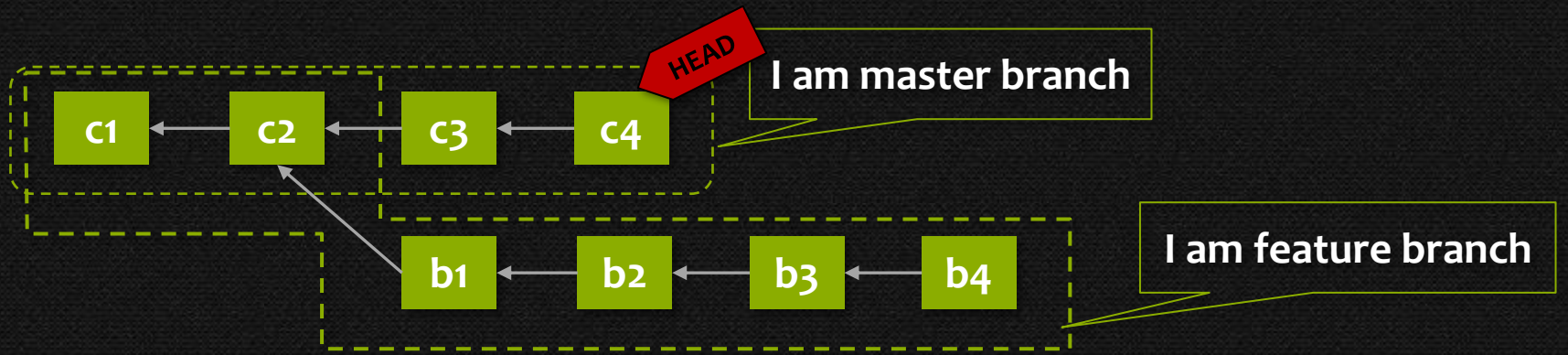
Recap

Create Repo	git init	Initialize a repository
Inspect Repo	git status	Know the status of the repository
Create Commits	git add	Add files for staging
	git commit	Create commit of the staged files
Inspect Commits	git log	View the commit log
	git diff/difftool	See changes between the commits
Undo Commits	git reset	Undo commit(unpack the commit)
	git checkout	Discard the changes

What are branches and why should I care?

- Branch is just a sequence of commits with a parent child relationship
- The default branch is always referred as *master* or *main*

- Branching helps in working with multiple features independently
- At any given point in time, There can be only one *active* branch in a repository

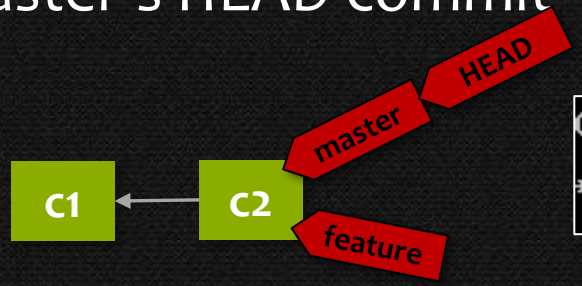


- The content of the file and folder structure of the repo is determined by the commits on current *active* branch

- *git branch* will show *all branches and highlights the current active branch

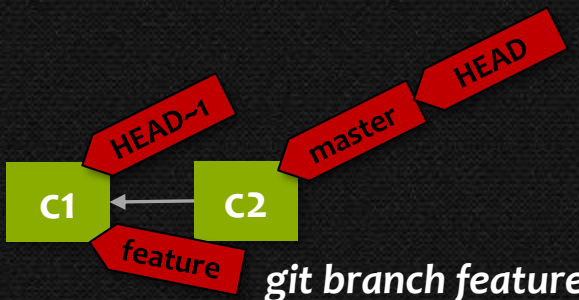
Branching

- *git branch feature master* will create a new branch named '*feature*' from master's HEAD commit



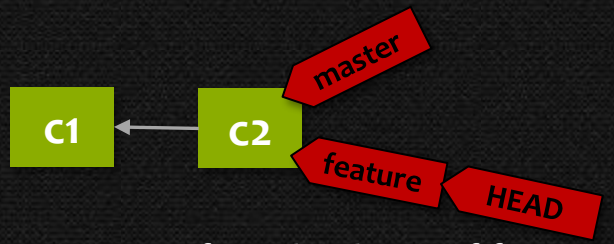
git branch feature HEAD

```
C:\MyProject>git branch
  feature
* master
```



git branch feature HEAD~1

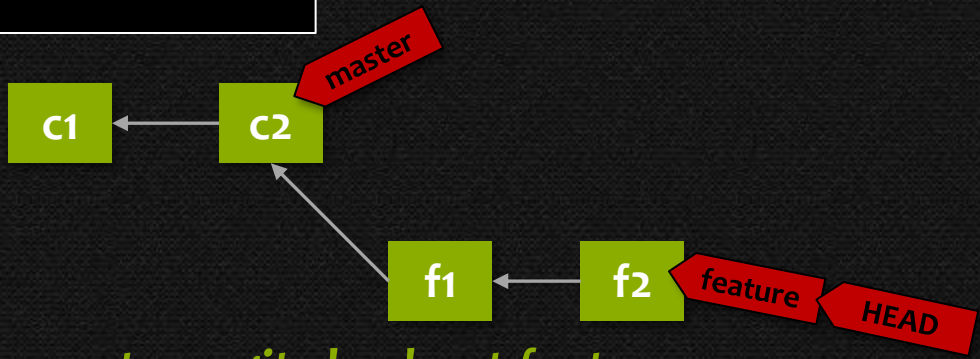
- *git checkout feature* is used to switch to the branch named '*feature*'



HEAD after checkout of feature

```
C:\MyProject>git checkout feature
Switched to branch 'feature'

C:\MyProject>git branch
* feature
  master
```

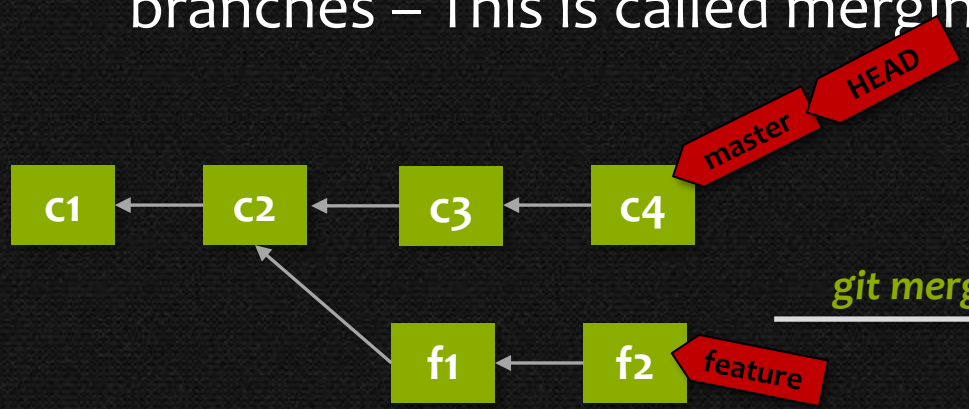
- With each commit on the feature branch, The HEAD moves forward on the feature branch



git checkout –b feature master = git branch feature master + git checkout feature

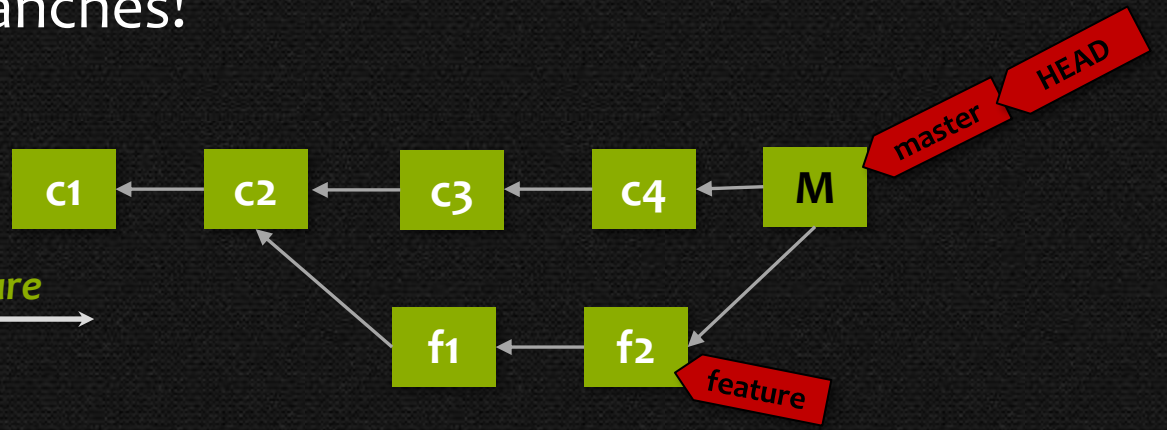
Merging

- *git merge* is used to create a merge commit between two or more branches – This is called merging branches!



HEAD on master branch after checkout of master

git merge feature



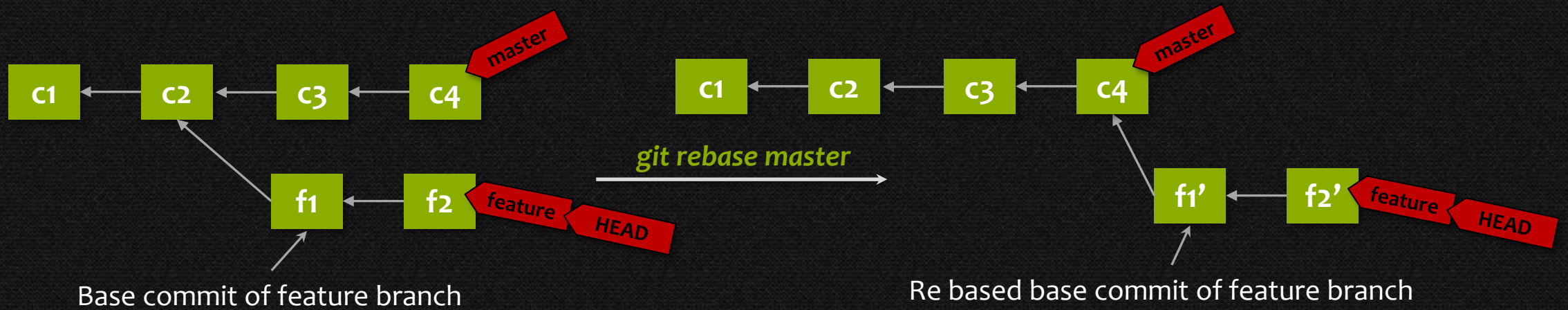
Merge commit created on master after git merge command

```
C:\MyProject>git log
commit a8a5250f3ee66af7e4a4afdfb2a5a0a32bbb97d3
Merge: d751102 f3f8a35
Author: Vineel Kumar Reddy Kovvuri <vineel.kovvuri@gmail.com>
Date:   Tue Jul 14 19:05:02 2015 +0530

    Merge branch 'feature'
```

```
C:\MyProject>git log --graph --oneline --decorate --all
*   a8a5250 (HEAD, master) Merge branch 'feature'
|\
| * f3f8a35 (feature) Optimised Hello World
* | d751102 Comment added
|/
* ee8a73a Adding .gitignore
* 940a3a6 My First helloWorld commit
```

Rebasing

- *git rebase* realigns the base commit of the current branch with other branch



Feature branch before git rebase

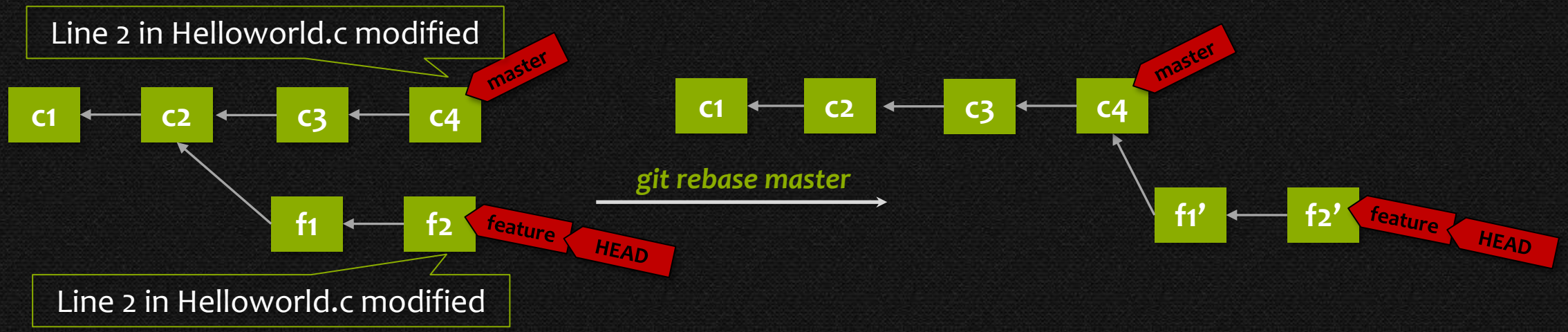
Feature branch after git rebase

f1 Contains changes made before rebase

f1' May not contain the same changes as f1 because of merge conflicts

Resolving conflicts manually in Git

- *git merge* and *git rebase* can sometime lead to merge conflicts




```
C:\HelloWorldProject>git rebase master
First, rewinding head to replay your work on top of it...
Applying: Comment updated in feature
Using index info to reconstruct a base tree...
M       HelloWorld.c
Falling back to patching base and 3-way merge...
Auto-merging HelloWorld.c
CONFLICT (content): Merge conflict in HelloWorld.c
Failed to merge in the changes.
Patch failed at 0001 Comment updated in feature
The copy of the patch that failed is found in:
   c:/HelloWorldProject/.git/rebase-apply/patch

When you have resolved this problem, run "git rebase --continue".
If you prefer to skip this patch, run "git rebase --skip" instead.
To check out the original branch and stop rebasing, run "git rebase --abort".
```

HelloWorld.c

```
1   #include<stdio.h>
2   <<<<<<< HEAD
3   //Comments add in master
4   =======
5   //Comments add feature branch
6   >>>>>>> Comment updated in feature
7   int main()
8   {
9       printf("Hello World!\n");
10      return 0;
11  }
12
```

Recap

Branching Commands	git branch	List all branches
	git branch <new> <existing>	Create <new> branch from <existing> branch
	git checkout <branch>	Switch to <branch>
	git checkout –b <new> <existing>	Create a new branch and switch to that branch
Merge Command	git merge <feature>	Merge current branch with <feature> branch
Rebase Command	git rebase <feature>	Rebase current branch with <feature> branch

Git – Part 3

Vineel Kovvuri

Senior SDE @ Microsoft

https://vineelkovvuri.github.io

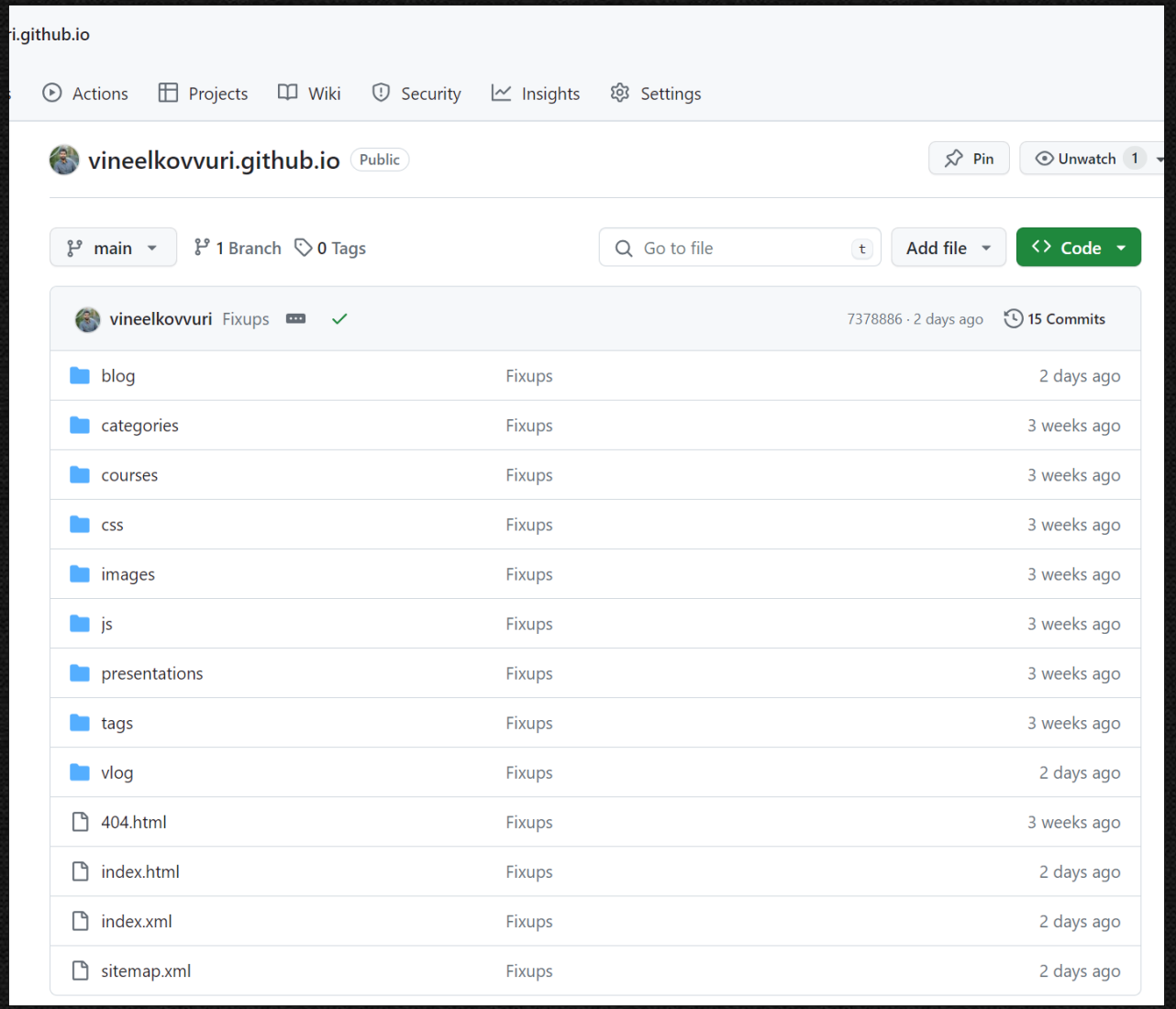
Agenda

- Github
- Remote
- Push
- Clone
- Fetch
- Pull

Recap

Create Repo	git init	Initialize a repository	Part - 1
Inspect Repo	git status	Know the status of the repository	
Create Commits	git add	Add files for staging	
	git commit	Create commit of the staged files	
Inspect Commits	git log	View the commit log	
	git diff/difftool	See changes between the commits	
Undo Commits	git reset	Undo commit(unpack the commit)	
	git checkout	Discard the changes	
Branching Commands	git branch	List all branches	Part - 2
	git branch <new> <existing>	Create <new> branch from <existing> branch	
	git checkout <branch>	Switch to <branch>	
	git checkout –b <new> <existing>	Create a new branch and switch to that branch	
Merge Command	git merge <feature>	Merge current branch with <feature> branch	
Rebase Command	git rebase <feature>	Rebase current branch with <feature> branch	

Github – Walk through

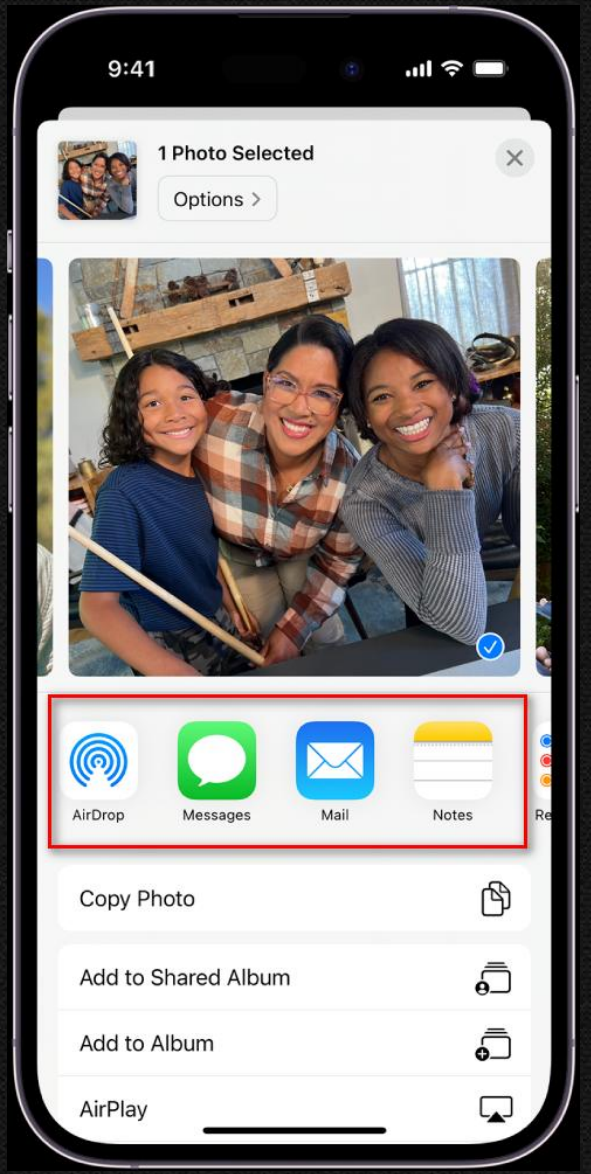


Remote

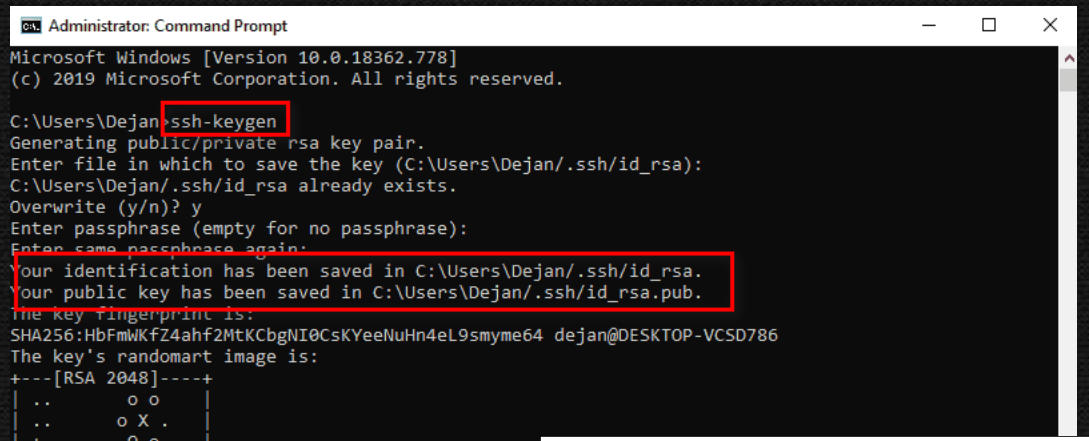
- Remote is a place where you can upload your git source code. Github is one such place
- There can be more than one remote for a given repository

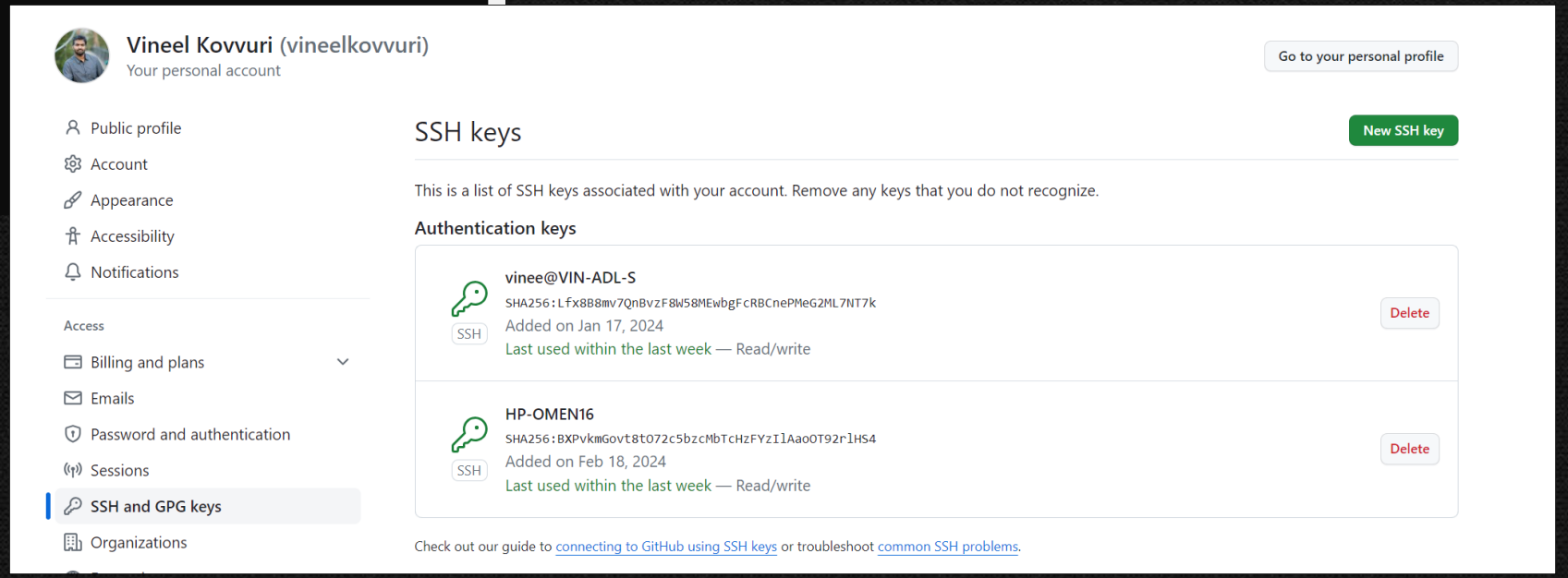
```
C:\repos\wimlib>git remote -v
origin  https://github.com/ebiggers/wimlib.git (fetch)
origin  https://github.com/ebiggers/wimlib.git (push)
```

- origin is the name given to the default remote

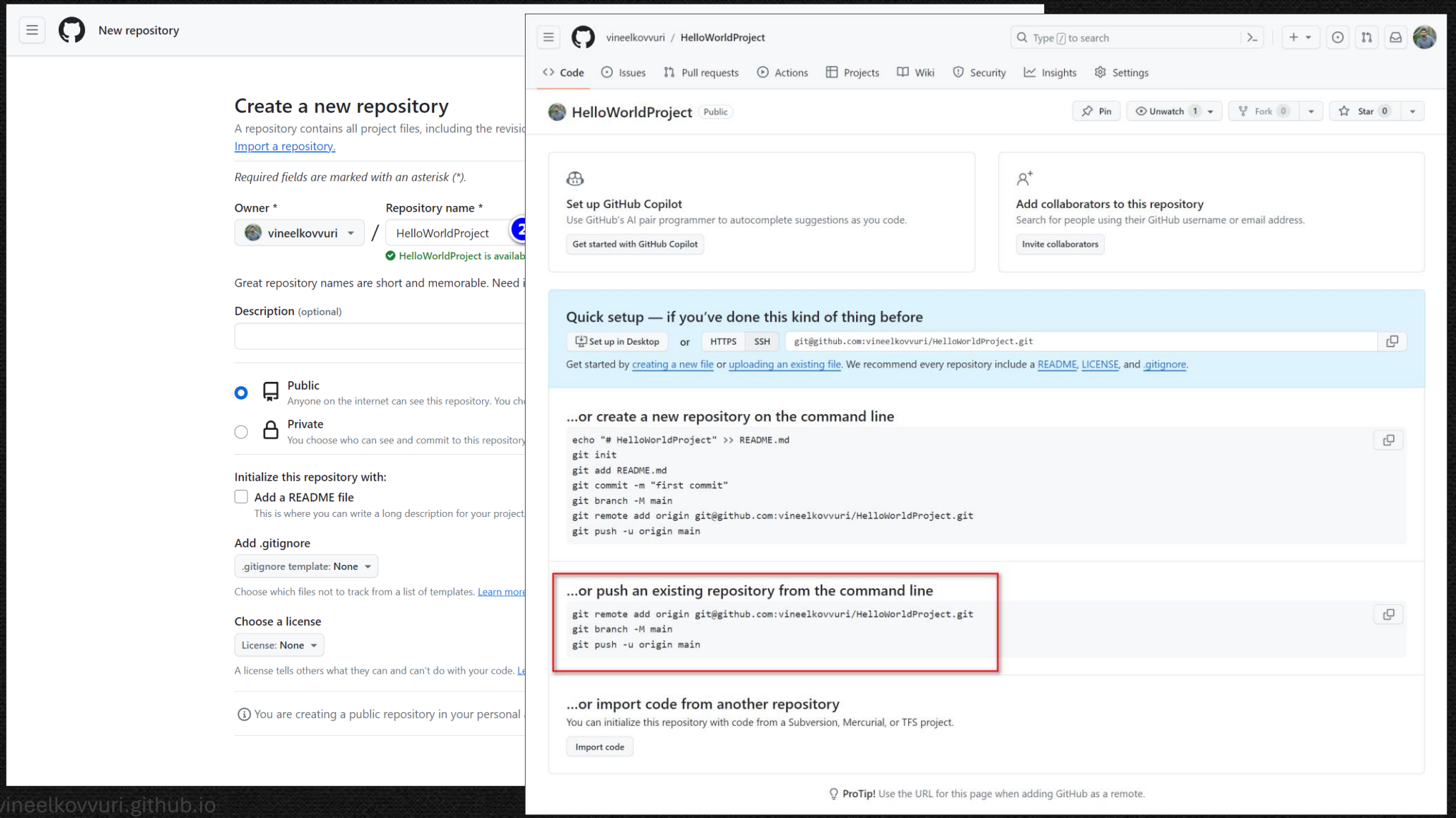


Github – Configure ssh



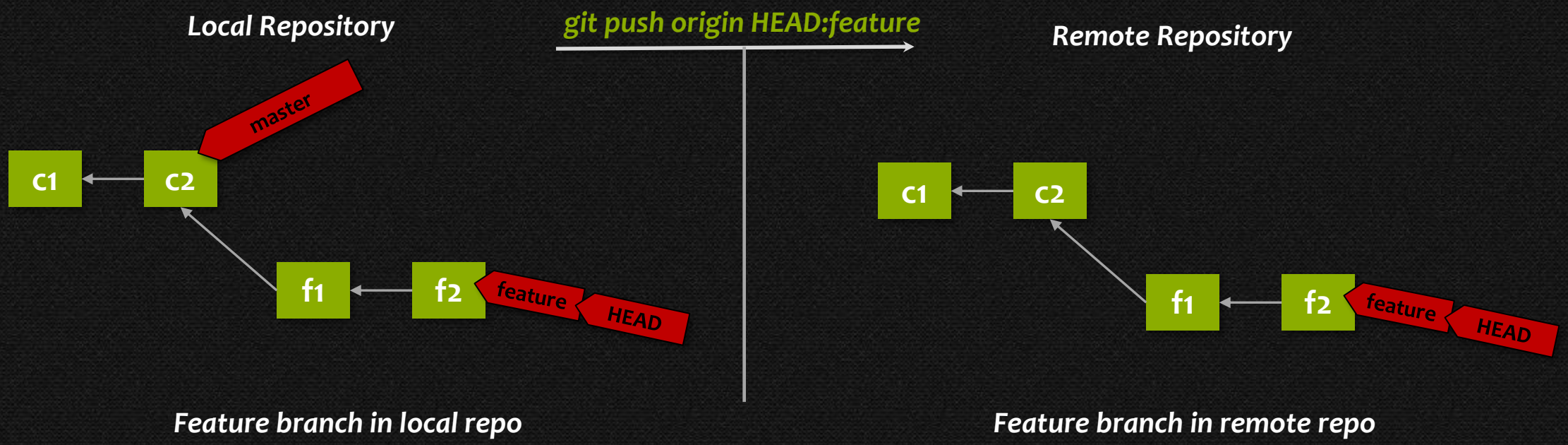


Github – How to Create a new repo?



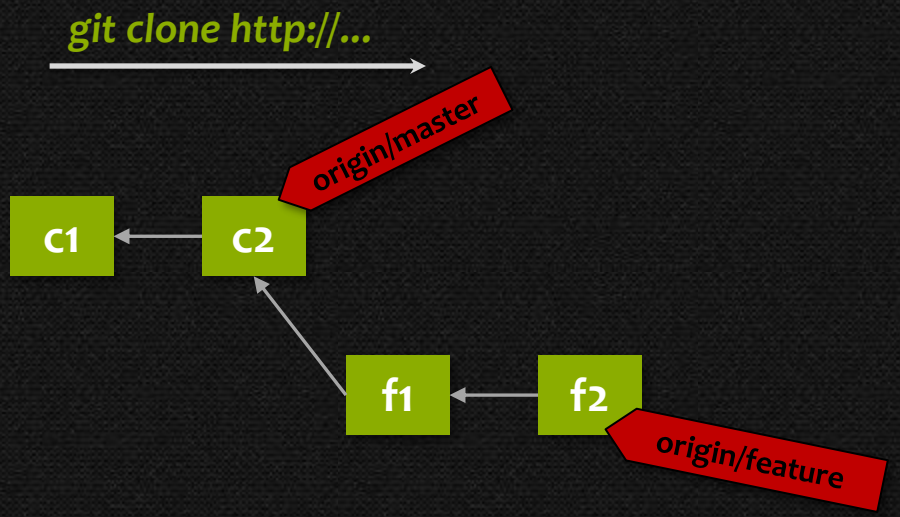
How to push to a new remote repository?

- *git push origin HEAD:<branchname>*
  - Push the local branch to remote branch with the name *<branchname>*



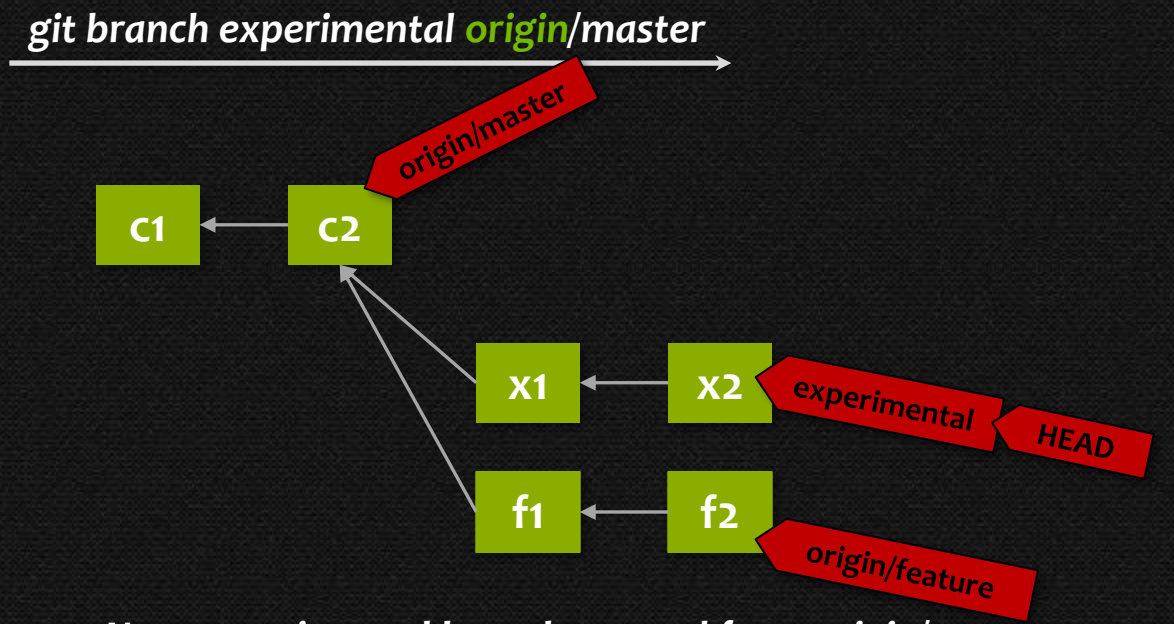
Cloning an existing remote repository

- *git clone* is used to create a new copy of remote repository in local machine
- Git clone completely copies all the branches from the remote repository
- By default, git will add the cloned remote as *origin*



All branches from remote repository are Cloned into local repo after a git clone

Create local branch with remote branch reference



New experimental branch created from origin/master

Listing local and remote branches

- *git branch –r* can be used to list only remote branches

```
C:\RemoteHelloWorld>git branch -r
  origin/feature
  origin/master

C:\RemoteHelloWorld>git branch -r -vv
  origin/feature 119aaed Added help file to use multiply function
  origin/master  6ec5b63 Converted int to long to fix overflow

C:\RemoteHelloWorld>
```

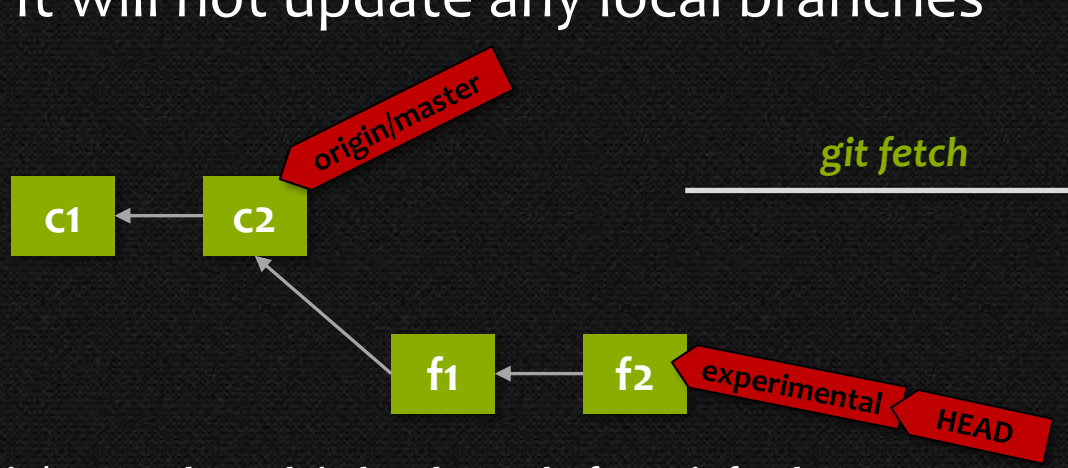
- *git branch –a –vv* list all(-a) branches(both local and remote) with tracking information(-vv)

```
C:\RemoteHelloWorld>git checkout -b experimental origin/master
Branch experimental set up to track remote branch master from origin.
Switched to a new branch 'experimental'

C:\RemoteHelloWorld>git branch -a -vv
* experimental             95f2e83 [origin/master] Initial multiplication commit
  master                   95f2e83 [origin/master] Initial multiplication commit
  remotes/origin/master    95f2e83 Initial multiplication commit

C:\RemoteHelloWorld>
```

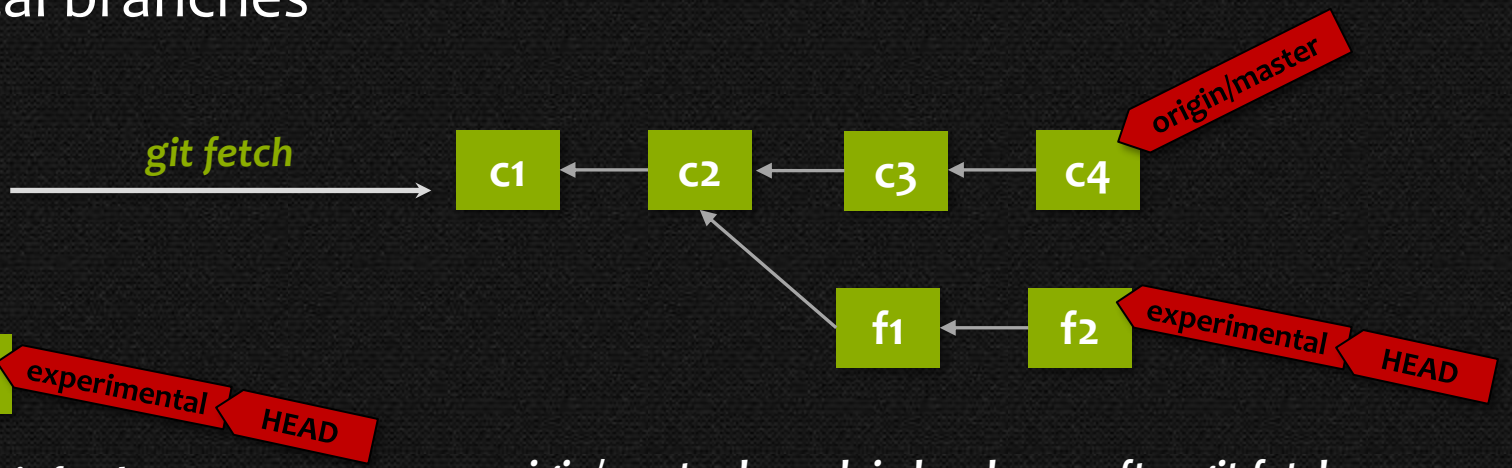
origin/master

Fetching

- *git fetch* gets and updates all the remote branches
- It will not update any local branches



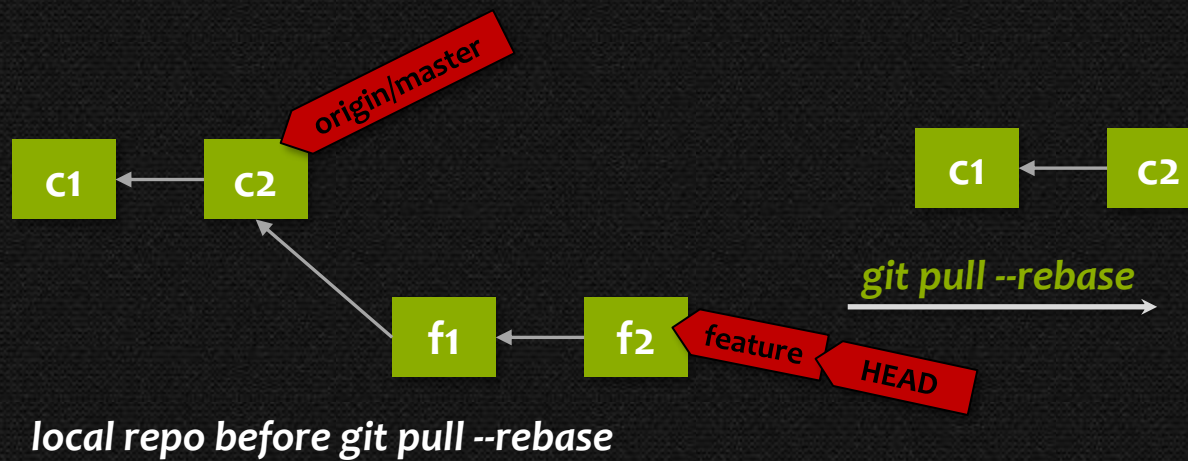
origin/master branch in local repo before git fetch



origin/master branch in local repo after git fetch

Pull

- *git pull –-rebase* fetches and also rebases the current branch with the origin/master



local repo before git pull --rebase

local repo after git pull --rebase

***git pull –-rebase = git fetch + git rebase**(current branch)*

Recap

Clone Command	git clone <url>	Clone a git repository
Branch Command	git branch –r	Show only remote branches
Push Command	git push origin HEAD:<branch>	Push current branch as <branch> to origin
Fetch Command	git fetch	Update all locally cloned remote branches(aka origin/…) with any updates from origin
Pull Command	git pull –-rebase	Update all locally cloned remote branches(aka origin/…) with any updates from origin and also rebases the current local branch

Git Cheat Sheet

Create Repo	git init .	Initialize a git repository in the current directory	Part - 1
Inspect Repo	git status	Know the status of the repository	
Create Commits Commands	git add	Add files for staging	
	git commit	Create commit of the staged files	
Inspect Commits Commands	git log	View the commit log	
	git diff/difftool	See changes between the commits	
Undo Commits Commands	git reset	Undo commit(unpack the commit). But retains the changes	
	git checkout	Discard the changes	
Branch Commands	git branch or git branch -r	List local branches. List only remote branches(aka origin/… branches)	Part - 2
	git branch <new> <existing>	Create <new> branch from <existing> branch	
	git checkout <branch>	Switch to <branch>	
	git checkout –b <new> <existing>	Create a new branch and switch to that branch	
Merge Command	git merge <feature>	Merge current branch with <feature> branch	
Rebase Command	git rebase <feature>	Rebase current branch with <feature> branch	
Clone Command	git clone <url>	Clone a git repository	Part - 3
Fetch Command	git fetch	Update all locally cloned remote branches(aka origin/…) with any updates from origin	
Pull Command	git pull --rebase	Update all locally cloned remote branches(aka origin/…) with any updates from origin but also rebases the current local branch	
Push Command	git push origin HEAD:<branch>	Push current branch as <branch> to origin	

Git Cheat Sheet

Create Repo	git init .	Initialize a git repository in the current directory	Part - 1
Inspect Repo	git status	Know the status of the repository	
Create Commits Commands	git add	Add files for staging	
	git commit	Create commit of the staged files	
Inspect Commits Commands	git log	View the commit log	
	git diff/difftool	See changes between the commits	
Undo Commits Commands	git reset	Undo commit(unpack the commit). But retains the changes	
	git checkout	Discard the changes	
Branch Commands	git branch or git branch -r	List local branches. List only remote branches(aka origin/… branches)	Part - 2
	git branch <new> <existing>	Create <new> branch from <existing> branch	
	git checkout <branch>	Switch to <branch>	
	git checkout –b <new> <existing>	Create a new branch and switch to that branch	
Merge Command	git merge <feature>	Merge current branch with <feature> branch	
Rebase Command	git rebase <feature>	Rebase current branch with <feature> branch	
Clone Command	git clone <url>	Clone a git repository	Part - 3
Fetch Command	git fetch	Update all locally cloned remote branches(aka origin/…) with any updates from origin	
Pull Command	git pull --rebase	Update all locally cloned remote branches(aka origin/…) with any updates from origin but also rebases the current local branch	
Push Command	git push origin HEAD:<branch>	Push current branch as <branch> to origin	

Thank You