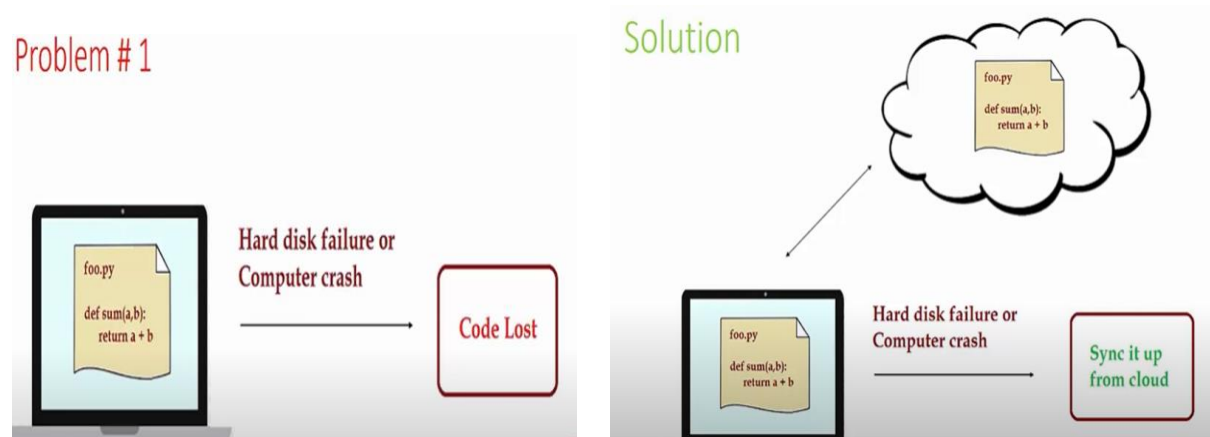


GIT: Problems without using code version tool:

Problem 1: If you machine crashes, then complete code will be lost.

Solution: You can write data to cloud.



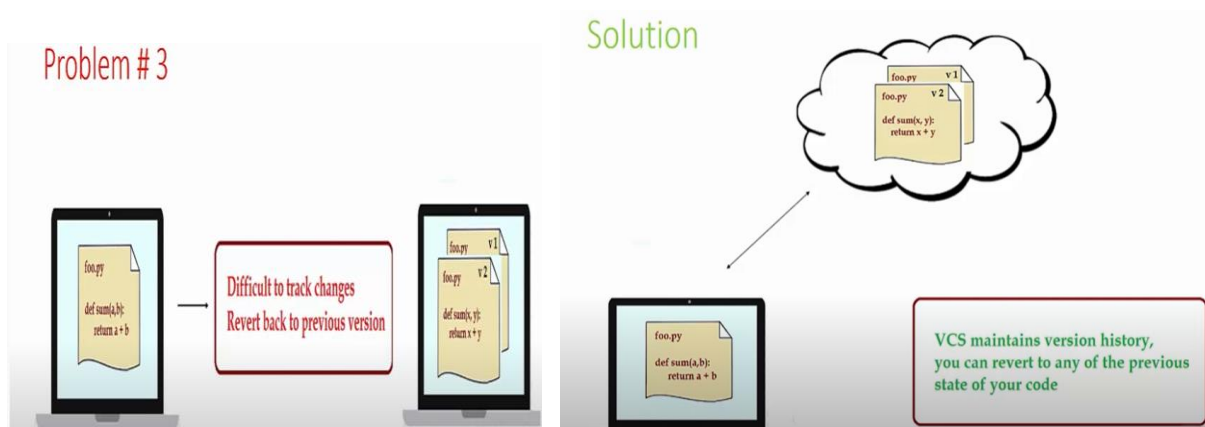
Problem 2: Only one person can work on same code.

Solution: Both person can update same code.

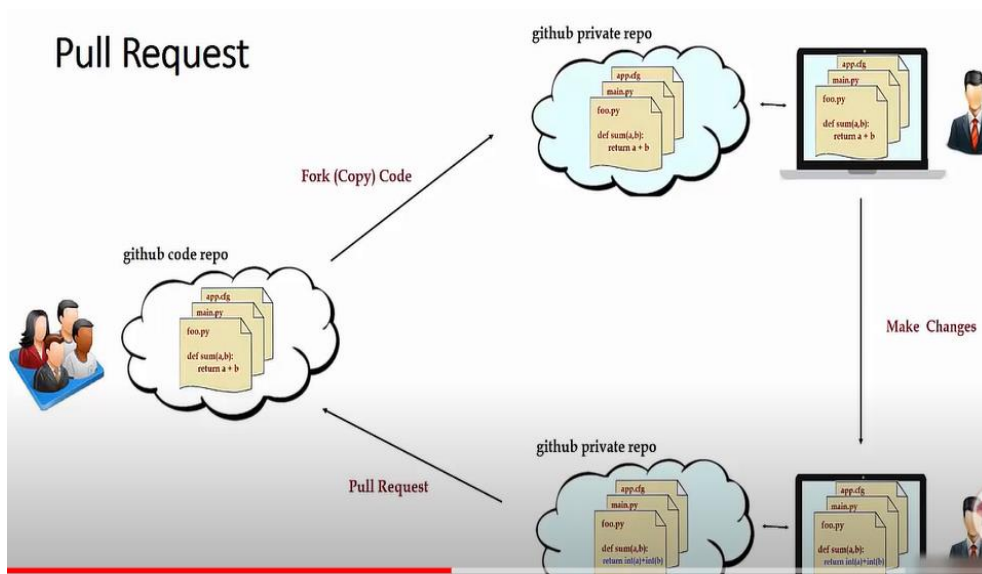


Problem 3: Difficult to track the changes.

Solution: Different versions of code is available.



Pull request concept: One can copy code from git hub code repo and will do changes. Person can send request to pull changes for review. After review, pull request will be accepted and code will be merged.



Git Basics:

Create new repository: Login to GitHub and create new repository.

Copy repository means clone the repository.

```
git clone https://github.com/ashukla2019/test.git
```

Check status of repository: git status

```
z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

Create new file to repository: Add new file to repository.

Now do git status, it will show you untracked file.

```
z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

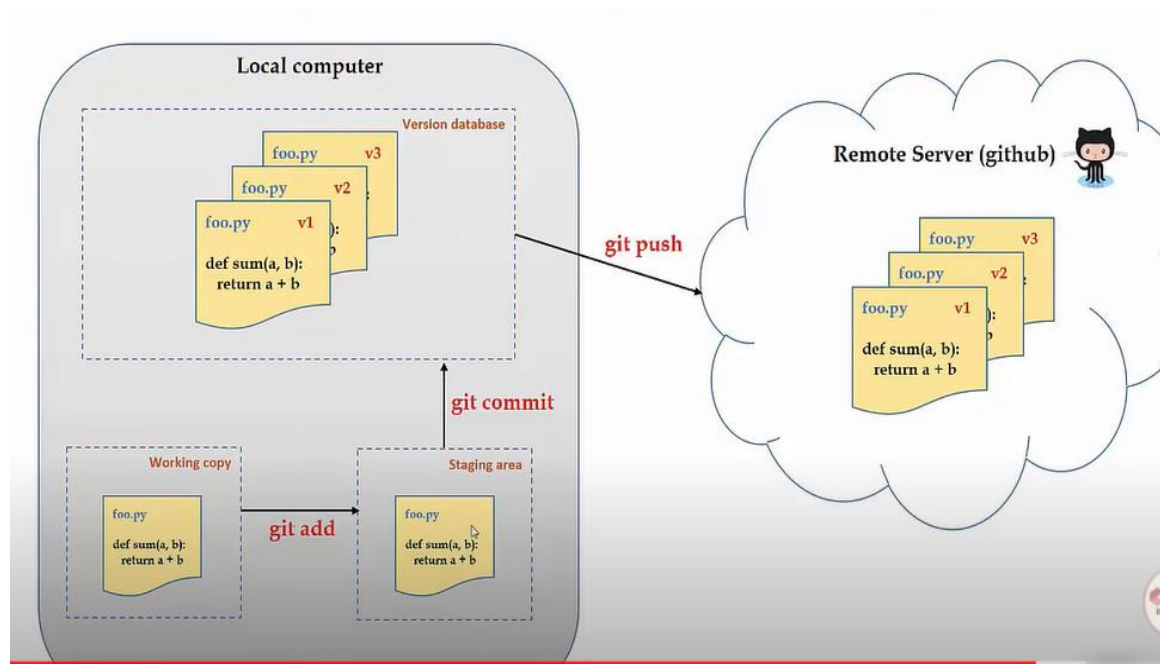
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        update_correct_timestamp_baseline.py

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

When we add file, we require to add it in **Staging Area** (means file is ready for commit now) and can commit file. Now file is committed to local version control system.

```
z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (main)
$ git commit -m "first commit"
[main b233ed2] first commit
```

Git log: This command will show you log of commit.



Changes can be reverted. We check the log and then we will get commit id. Use git revert -n "**commit_id**". Then commit changes.

Branches: Create branch, merge branch, and delete branch

List all the branches: git branch

Create new branch: git branch xyz

```
z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (main)
$ git branch
* main
  xyz
```

Here main branch is active and if you want to perform operation to xyz branch, should make it active. For making branch active:

Git checkout xyz -> will switch to xyz branch

```
z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (main)
$ git checkout xyz
Switched to branch 'xyz'

z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (xyz)
$ git branch
  main
* xyz
```

Now modify into new xyz branch:

Do changes in file and then add and commit it.

```
z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (xyz)
$ git add update_correct_timestamp_baseline.py

z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (xyz)
$ git status
On branch xyz
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   update_correct_timestamp_baseline.py
```

```

z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (xyz)
$ git commit -m "modified -1"
[xyz 3449ec0] modified -1
Committer: Shukla <ankitshukla@siemens.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 3 insertions(+), 1 deletion(-)

```

When switched back to main branch, no modification is present:

```

z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (main)
$ ls
README.md  update_correct_timestamp_baseline.py

```

Merge xyz branch to main branch:

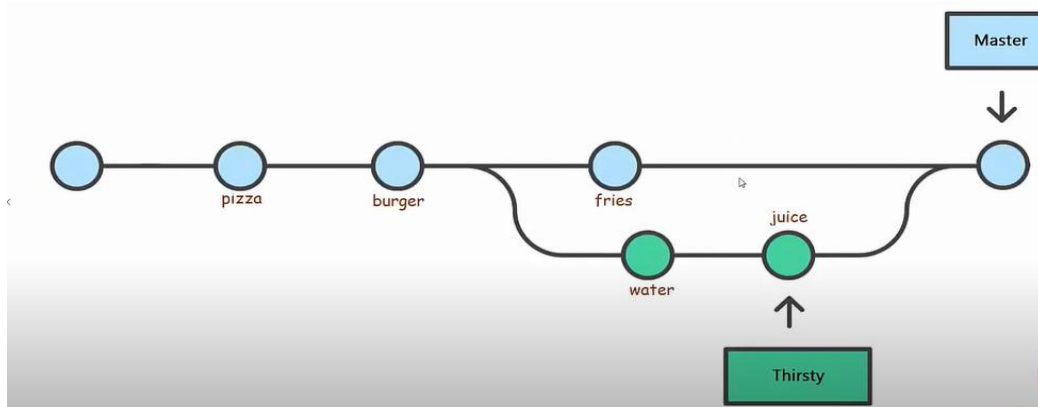
Switch to main branch and run merge:

```

z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (xyz)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

z00427jp@md2amvjc MINGW64 /d/docs/notes_c_c++/test (main)
$ git merge xyz
Updating b233ed2..e4d218b
Fast-forward
 update_correct_timestamp_baseline.py | 4 +++-
 1 file changed, 3 insertions(+), 1 deletion(-)

```



Delete branch: `git branch -d <branch-name>`

Here are the Git commands which are being covered:

- **git config**
- **git init**
- **git clone**
- **git add**
- **git commit**
- **git diff**
- **git reset**
- **git status**
- **git rm**
- **git log**
- **git show**
- **git tag**
- **git branch**
- **git checkout**
- **git merge**
- **git remote**
- **git push**
- **git pull**
- **git stash**

So, let's get started!

Git Commands

git config

Usage: `git config -global user.name "[name]"`

Usage: `git config -global user.email "[email address]"`

This command sets the author name and email address respectively to be used with your commits.

```
edureka@master:~$ git config --global user.name "sahitikappagantula"  
edureka@master:~$ git config --global user.email "sahiti.kappagantula@edureka.co"
```

git init

Usage: `git init [repository name]`

This command is used to start a new repository.

```
edureka@master:~$ git init /home/edureka/Documents/DEMO
Initialized empty Git repository in /home/edureka/Documents/DEMO/.git/
```

git clone

Usage: `git clone [url]`

This command is used to obtain a repository from an existing URL.

```
edureka@master:~$ git clone https://github.com/sahitikappagantula/gitexample.git
Cloning into 'gitexample'...
remote: Counting objects: 28, done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 28 (delta 5), reused 28 (delta 5), pack-reused 0
Unpacking objects: 100% (28/28), done.
```

git add

Usage: `git add [file]`

This command adds a file to the staging area.

```
edureka@master:~/Documents/DEMO$ git add project_1
```

Usage: `git add *`

This command adds one or more to the staging area.

```
edureka@master:~/Documents/DEMO$ git add *
```

git commit

Usage: `git commit -m "[Type in the commit message]"`

This command records or snapshots the file permanently in the version history.


```
edureka@master:~/Documents/DEMO$ git commit -m "First Commit"
[master (root-commit) aff3269] First Commit
9 files changed, 200 insertions(+)
create mode 100644 project_1/css/site.css
create mode 100644 project_1/fonts/segoeui.ttf
create mode 100644 project_1/img/clonewhite.svg
create mode 100644 project_1/img/deployWhite.svg
create mode 100644 project_1/img/lightbulbWhite.svg
create mode 100644 project_1/img/stackWhite.svg
create mode 100644 project_1/img/successCloudNew.svg
create mode 100644 project_1/img/tweetThis.svg
create mode 100644 project_1/index.html
```

Usage: `git commit -a`

This command commits any files you've added with the `git add` command and also commits any files you've changed since then.

```
edureka@master:~/Documents/DEMO$ git commit -a
On branch master
nothing to commit, working tree clean
```

git diff

Usage: `git diff`

This command shows the file differences which are not yet staged.

```
edureka@master:~/Documents/DEMO$ git diff
diff --git a/project_1/index.html b/project_1/index.html
index 8a985d9..94cfa0f 100644
--- a/project_1/index.html
+++ b/project_1/index.html
@@ -20,8 +20,8 @@
     </div>
     <div class="content-body">
       <div class="success-text">Success!</div>
-      <div class="description line-1"> AWS DevOps Project has been successfully setup</div>
-      <div class="description line-2"> Your HTML app is up and running on AWS</div>
+      <div class="description line-1"> Azure DevOps Project has been successfully setup</div>
+      <div class="description line-2"> Your HTML app is up and running on Azure</div>
       <div class="next-steps-container">
         <div class="next-steps-header">Next up</div>
         <div class="next-steps-body">
```

Usage: `git diff -staged`

This command shows the differences between the files in the staging area and the latest version present.


```
edureka@master:~/Documents/DEMO/project_1/css$ git diff --staged
diff --git a/project_1/css/site.css b/project_1/css/site.css
index 25606b6..fba307d 100644
--- a/project_1/css/site.css
+++ b/project_1/css/site.css
@@ -1,5 +1,5 @@
    html,
-/* This the css file for the web page */
+/* This the css file for the web page we are using for our DEMO */
    body {
        height: 100%;
        width: 100%;
```

Usage: `git diff [first branch] [second branch]`

This command shows the differences between the two branches mentioned.

```
edureka@master:~/Documents/DEMO/project_1$ git diff branch_2 branch_3
diff --git a/project_1/index.html b/project_1/index.html
index b567d94..94cfa0f 100644
--- a/project_1/index.html
+++ b/project_1/index.html
@@ -47,7 +47,7 @@
        <div class="step-icon">
            
        </div>
-        <div class="step-text"><a href="https://go.microsoft.com/fwlink/?linkid=862126">Learn more about a
+        <div class="step-text"><a href="https://go.microsoft.com/fwlink/?linkid=862126">Learn more about a
ll you can do with AWS & Google Cloud Platform projects by visiting the documentation</a></div>
+        <div class="step-text"><a href="https://go.microsoft.com/fwlink/?linkid=862126">Learn more about a
ll you can do with AWS & GCP projects by visiting the documentation</a></div>
        </div>
    </div>
</div>
```

git reset

Usage: `git reset [file]`

This command unstages the file, but it preserves the file contents.

```
edureka@master:~/Documents/DEMO/project_1/css$ git reset site.css
Unstaged changes after reset:
M    project_1/css/site.css
M    project_1/index.html
```

Usage: `git reset [commit]`

This command undoes all the commits after the specified commit and preserves the changes locally.

```
edureka@master:~/Documents/DEMO$ git reset 09bb8e3f996eaf9a68ac5ba8d8b8fceb0e8641e7
Unstaged changes after reset:
M    project_1/css/site.css
M    project_1/index.html
```

Usage: `git reset -hard [commit]` This command discards all history and goes back to the specified commit.

```
edureka@master:~/Documents/DEMO$ git reset --hard b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
HEAD is now at b01557d Changes made in HTML file
```

git status

Usage: `git status`

This command lists all the files that have to be committed.

```
edureka@master:~/Documents/DEMO$ 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:   project_1/css/site.css
        modified:   project_1/index.html

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

git rm

Usage: `git rm [file]`

This command deletes the file from your working directory and stages the deletion.

```
edureka@master:~/Documents/DEMO/project_2$ git rm example.txt
rm 'project_2/example.txt'
```

git log

Usage: `git log`

This command is used to list the version history for the current branch.

```

edureka@master:~/Documents/DEMO$ git log
commit 09bb8e3f996eaf9a68ac5ba8d8b8fceb0e8641e7 (HEAD -> master)
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:25:17 2018 +0530

    Changes made in HTML and CSS file

commit b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:13:29 2018 +0530

    CHanges made in HTML file

commit aff3269a856ed251bdfd7ef87acb1716a2a9527a
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:07:28 2018 +0530

    First Commit

```

Usage: `git log -follow[file]`

This command lists version history for a file, including the renaming of files also.

```

edureka@master:~/Documents/DEMO$ git log --follow project_1
commit 2b4c50431c127a0ae9ede4aace0b8dd1f9fcf2c5
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:50:08 2018 +0530

    New file added

commit 09bb8e3f996eaf9a68ac5ba8d8b8fceb0e8641e7
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:25:17 2018 +0530

    Changes made in HTML and CSS file

commit b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:13:29 2018 +0530

    CHanges made in HTML file

commit aff3269a856ed251bdfd7ef87acb1716a2a9527a
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:07:28 2018 +0530

    First Commit

```

git show

Usage: `git show [commit]`

This command shows the metadata and content changes of the specified commit.

Unrestricted

```
edureka@master:~/Documents/DEMO$ git show b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
commit b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:13:29 2018 +0530

    CHanges made in HTML file

diff --git a/project_1/index.html b/project_1/index.html
index 8a985d9..94cfa0f 100644
--- a/project_1/index.html
+++ b/project_1/index.html
@@ -20,8 +20,8 @@
     </div>
     <div class="content-body">
       <div class="success-text">Success!</div>
-       <div class="description line-1"> AWS DevOps Project has been successfully setup</div>
-       <div class="description line-2"> Your HTML app is up and running on AWS</div>
+       <div class="description line-1"> Azure DevOps Project has been successfully setup</div>
+       <div class="description line-2"> Your HTML app is up and running on Azure</div>
       <div class="next-steps-container">
         <div class="next-steps-header">Next up</div>
         <div class="next-steps-body">
```

git tag

Usage: git tag [commitID]

This command is used to give tags to the specified commit.

```
edureka@master:~/Documents/DEMO$ git tag b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
edureka@master:~/Documents/DEMO$ git tag
ff3269a856ed251bdfd7ef87acb1716a2a9527a
b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
```

git branch

Usage: git branch

This command lists all the local branches in the current repository.

```
edureka@master:~/Documents/DEMO$ git branch
* master
```

Usage: git branch [branch name]

This command creates a new branch.

```
edureka@master:~/Documents/DEMO$ git branch branch_1
```

Usage: git branch -d [branch name]

This command deletes the feature branch.

Unrestricted

```
edureka@master:~/Documents/DEMO$ git branch -d branch_1
Deleted branch branch_1 (was be040cc).
```

git checkout

Usage: `git checkout [branch name]`

This command is used to switch from one branch to another.

```
edureka@master:~/Documents/DEMO$ git checkout branch_2
Switched to branch 'branch_2'
```

Usage: `git checkout -b [branch name]`

This command creates a new branch and also switches to it.

```
edureka@master:~/Documents/DEMO$ git checkout -b branch_4
Switched to a new branch 'branch_4'
```

git merge

Usage: `git merge [branch name]`

This command merges the specified branch's history into the current branch.

```
edureka@master:~/Documents/DEMO$ git merge branch_2
Merge made by the 'recursive' strategy.
 project_1/index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

git remote

Usage: `git remote add [variable name] [Remote Server Link]`

This command is used to connect your local repository to the remote server.

```
edureka@master:~/Documents/DEMO$ git remote add origin https://github.com/sahitikappagantula/GitDemo.git
```

git push

Usage: `git push [variable name] master`

This command sends the committed changes of master branch to your remote repository.

```
edureka@master:~/Documents/DEMO$ git push origin master
Username for 'https://github.com': sahitikappagantula
Password for 'https://sahitikappagantula@github.com':
Counting objects: 42, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (32/32), done.
Writing objects: 100% (42/42), 463.10 KiB | 3.62 MiB/s, done.
Total 42 (delta 9), reused 0 (delta 0)
remote: Resolving deltas: 100% (9/9), done.
To https://github.com/sahitikappagantula/GitDemo.git
 * [new branch]      master -> master
```

Usage: `git push [variable name] [branch]`

This command sends the branch commits to your remote repository.

```
edureka@master:~/Documents/DEMO$ git push origin master
Username for 'https://github.com': sahitikappagantula
Password for 'https://sahitikappagantula@github.com':
Counting objects: 42, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (32/32), done.
Writing objects: 100% (42/42), 463.10 KiB | 3.62 MiB/s, done.
Total 42 (delta 9), reused 0 (delta 0)
remote: Resolving deltas: 100% (9/9), done.
To https://github.com/sahitikappagantula/GitDemo.git
 * [new branch]      master -> master
```

Usage: `git push -all [variable name]`

This command pushes all branches to your remote repository.

```
edureka@master:~/Documents/DEMO$ git push --all origin
Username for 'https://github.com': sahitikappagantula
Password for 'https://sahitikappagantula@github.com':
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/sahitikappagantula/GitDemo.git
 * [new branch]      branch_3 -> branch_3
 * [new branch]      branch_4 -> branch_4
```

Usage: `git push [variable name] :[branch name]`

This command deletes a branch on your remote repository.

```
edureka@master:~/Documents/DEMO$ git push origin : branch_2
Username for 'https://github.com': sahitikappagantula
Password for 'https://sahitikappagantula@github.com':
Everything up-to-date
```

git pull

Unrestricted

Usage: `git pull [Repository Link]`

This command fetches and merges changes on the remote server to your working directory.

```
edureka@master:~/Documents/DEMO$ git pull https://github.com/sahitkappagantula/gitlearn.git
warning: no common commits
remote: Counting objects: 13, done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 13 (delta 1), reused 10 (delta 1), pack-reused 0
Unpacking objects: 100% (13/13), done.
From https://github.com/sahitkappagantula/gitlearn
 * branch      HEAD       -> FETCH_HEAD
fatal: refusing to merge unrelated histories
```

git stash

Usage: `git stash save`

This command temporarily stores all the modified tracked files.

```
edureka@master:~/Documents/DEMO/project_1$ git stash save
Saved working directory and index state WIP on branch_2: 5152fcd Index.html updated
```

Usage: `git stash pop`

This command restores the most recently stashed files.

```
edureka@master:~/Documents/DEMO/project_1$ git stash pop
On branch branch_2
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")
Dropped refs/stash@{0} (365fa2ef6ed4f1f8d7d406bd0abb205279aad0c5)
```

Usage: `git stash list`

This command lists all stashed changesets.

```
edureka@master:~/Documents/DEMO/project_1$ git stash list
stash@{0}: WIP on master: 5f6ba20 Merge branch 'branch_2'
```

Usage: `git stash drop`

This command discards the most recently stashed changeset.

```
edureka@master:~/Documents/DEMO/project_1$ git stash drop stash@{0}
Dropped stash@{0} (5e2cbcea1b37d4e5b88854964d6165e461e2309d)
```


Add tag to release

- 1) git checkout release/a.b.c branch
- 2) git -a <tag_name> -m message(git -a a.b.c_x -m "created tag a.b.c_x")
- 3) git push origin : <tag_name>(git push origin a.b.c_x)

Rebase release a.b.c to master

- 1) Git clone master.
- 2) git checkout release/a.b.c
- 3) Create new branch from a.b.c
- 4) git rebase master

Squash commits from my branch

- 1) Checkout your branch
- 2) checkout target branch to local system
- 3) git rebase -i target_branch(release\x.y.z)
- 4) will show change log in VI/VIM editor
- 5) Change n-1 pick to s or squah, then save it
- 6) return back from VI editor screen
- 7) git push -f origin my_branch