DZone

Top 20 Git Commands with Examples

by Sahiti Kappagantula MVB · Jul. 24, 18 · Open Source Zone · Presentation

In the previous blog, you got an understanding of What Git is. In this blog, I will talk about the Top 20 Git Commands that you will be using frequently while you are working with Git.

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**
- **g**
- **g**
- **git stash**

So, let's get started now!!

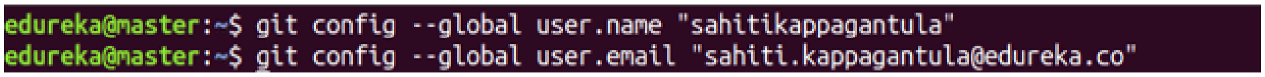
Git Commands

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.

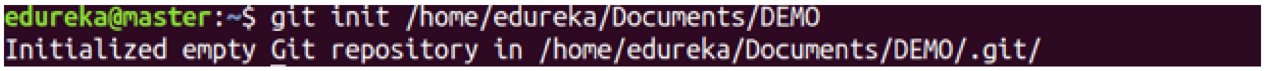


```
edureka@master:~$ git config --global user.name "sahitikappagantula"
edureka@master:~$ git config --global user.email "sahiti.kappagantula@edureka.co"
```

git init

Usage: `git init [repository name]`

This command is used to start a new repository.

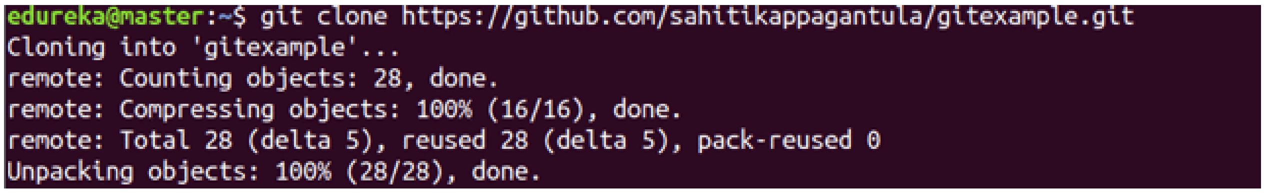


```
edureka@master:~$ git init /home/edureka/Documents/DEMO
Initialized empty Git repository in /home/edureka/Documents/DEMO/.git/
```

git clone

Usage: `git clone [url]`

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

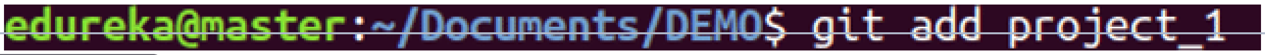


```
edureka@master:~$ git clone https://github.com/sahitikappagantula/gitexample.git
Cloning into 'gitexample'...
remote: Counting objects: 28, done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 28 (delta 5), reused 28 (delta 5), pack-reused 0
Unpacking objects: 100% (28/28), done.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

git add

Usage: `git add [file]`

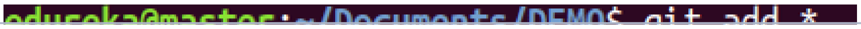
This command adds a file to the staging area.



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

Usage:

This co s one or more to the staging area.



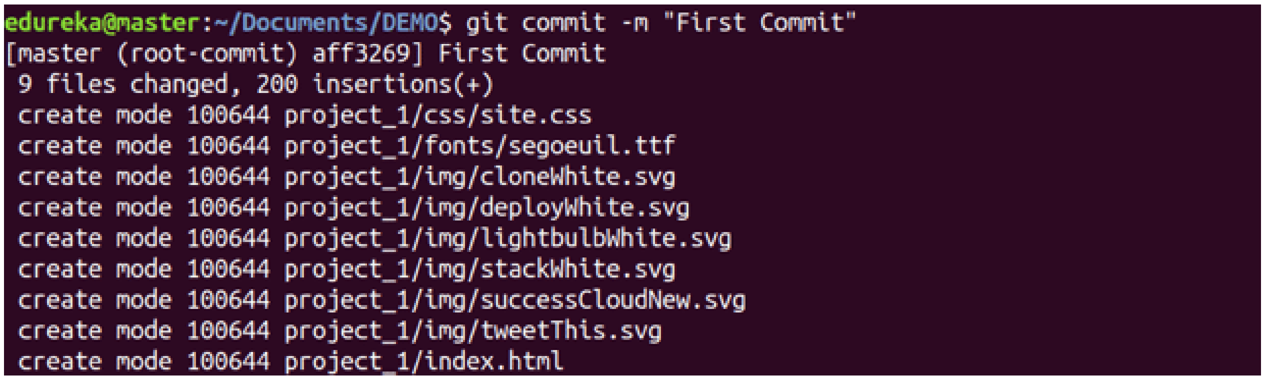
```
edureka@master:~/Documents/DEMO$ git add *
```

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

git commit

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

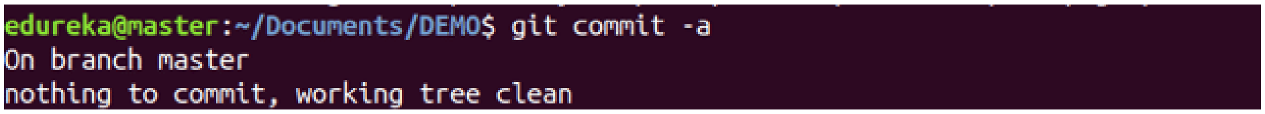
This command records or snapshots the file permanently in the version history.



```
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/segoeuil.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.

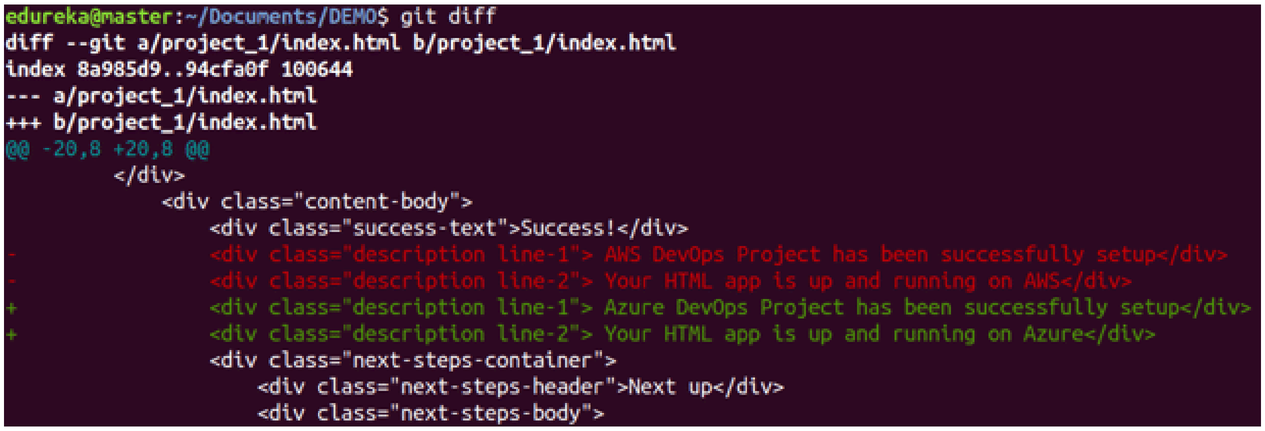


```
edureka@master:~/Documents/DEMO$ git commit -a
On branch master
nothing to commit, working tree clean
```

git diff

Usage: `git diff`

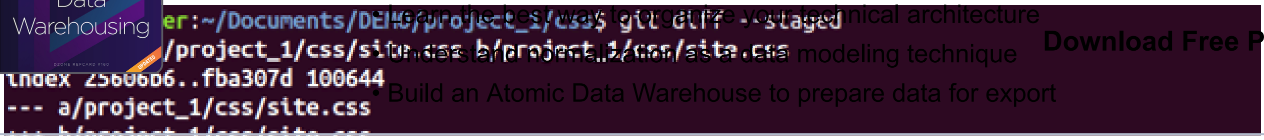
This command shows the file differences which are not yet staged.



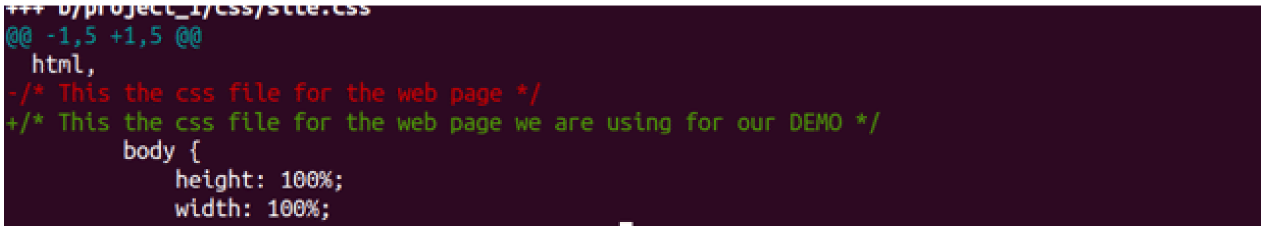
```
edureka@master:~/Documents/DEMO$ git diff
diff --git a/project_1/index.html b/project_1/index.html
index 8a985d9..94cfa0f 100644
--- a/project_1/index.html
+++ b/project_1/index.html
@@ -20,8 +20,8 @@
         </div>
             <div class="content-body">
                 <div class="success-text">Success!</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 co ws the differences between the files in the staging area and the latest version present.



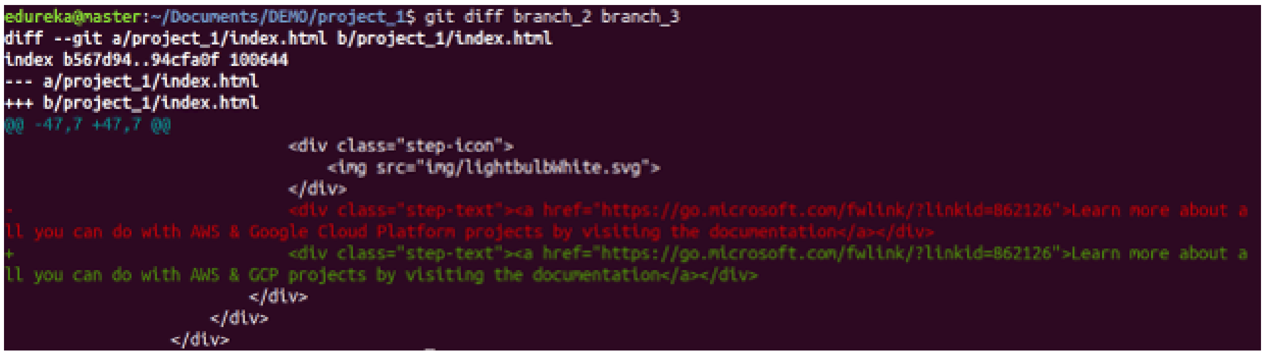
```
er:~/Documents/DEMO/project_1/css$ git diff --staged
/project_1/css/site.css b/project_1/css/site.css
index 25606b6..fba307d 100644
--- a/project_1/css/site.css
```



```
+++ 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.

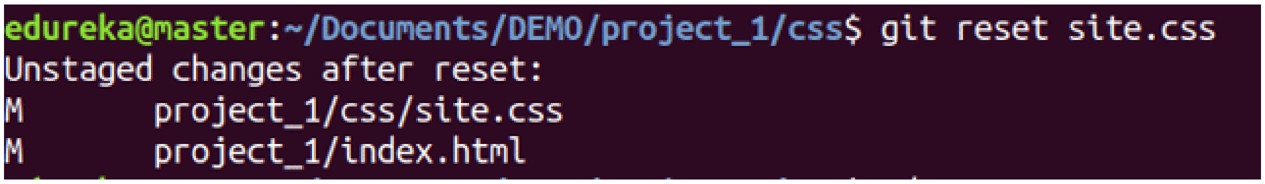


```
edureka@master:~/Documents/DEMO/project_1$ git diff branch_2 branch_3
diff --git a/project_1/index.html b/project_1/index.html
index b567d94..94cfa0f 100644
--- a/project_1/index.html
+++ b/project_1/index.html
@@ -47,7 +47,7 @@
                        <div class="step-icon">
                            <img src="img/lightbulbWhite.svg">
                        </div>
-                       <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]`

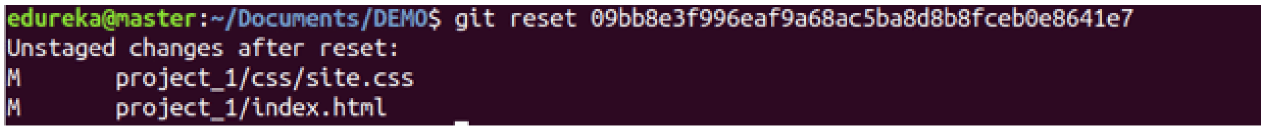
This command unstages the file, but it preserves the file contents.



```
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
```

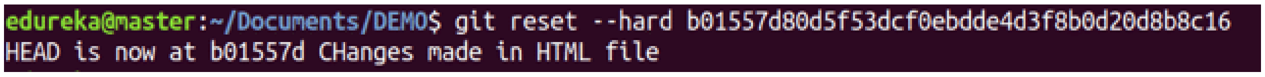
Usage: `git reset [commit]`

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.

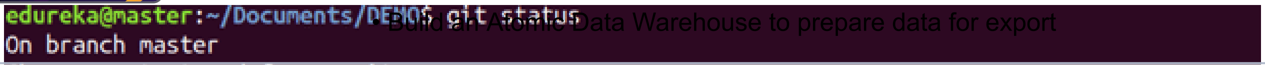


```
edureka@master:~/Documents/DEMO$ git reset --hard b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
HEAD is now at b01557d CHanges made in HTML file
```

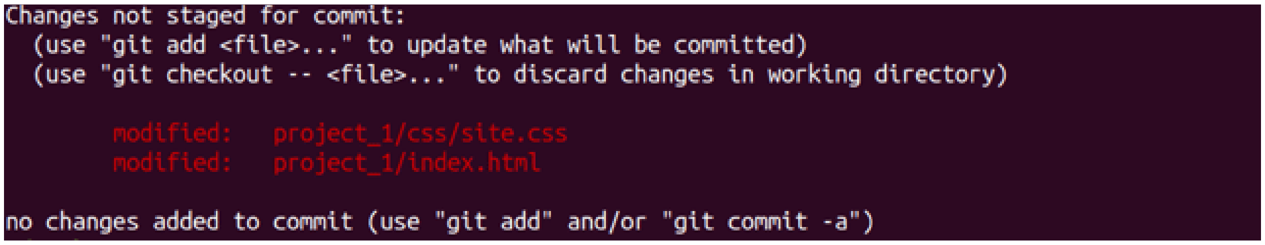
git status

Usage:

This co all the files that have to be committed.



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

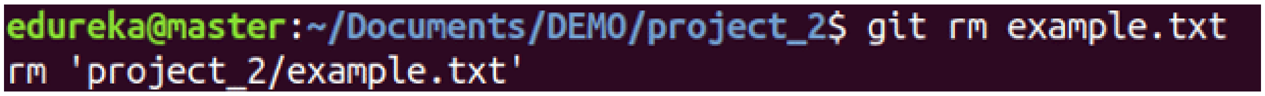


```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        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.

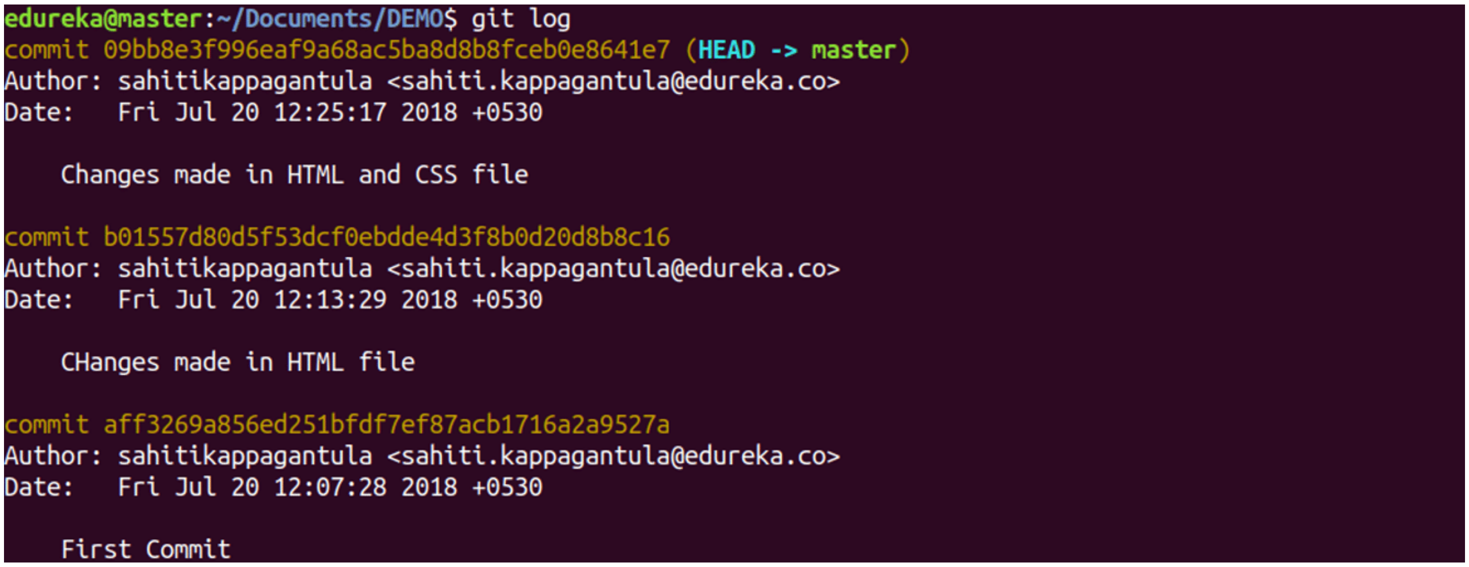


```
edureka@master:~/Documents/DEMO/project_2$ git rm example.txt
rm 'project_2/example.txt'
```

git log

Usage: `git log`

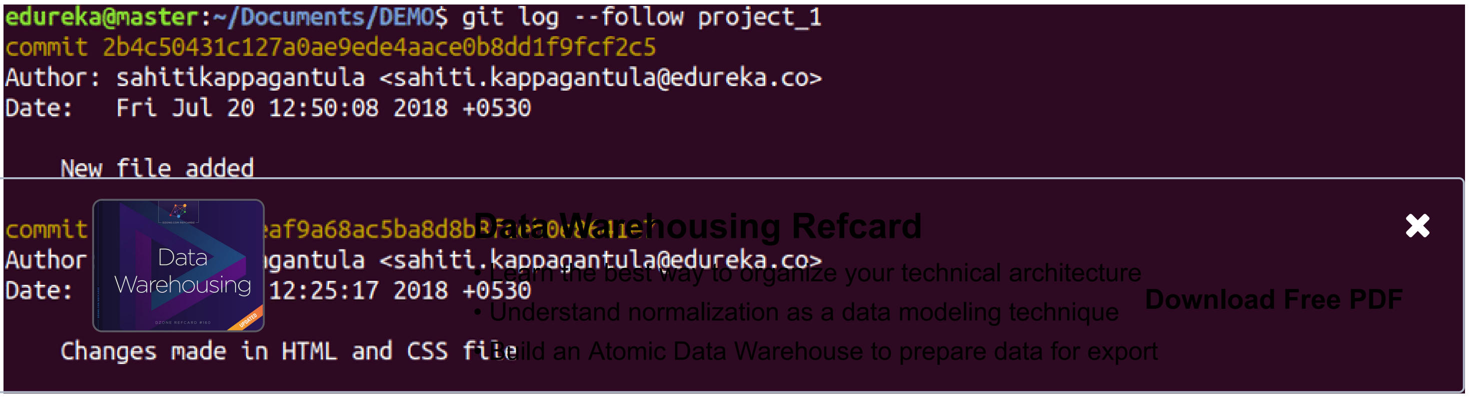
This command is used to list the version history for the current branch.



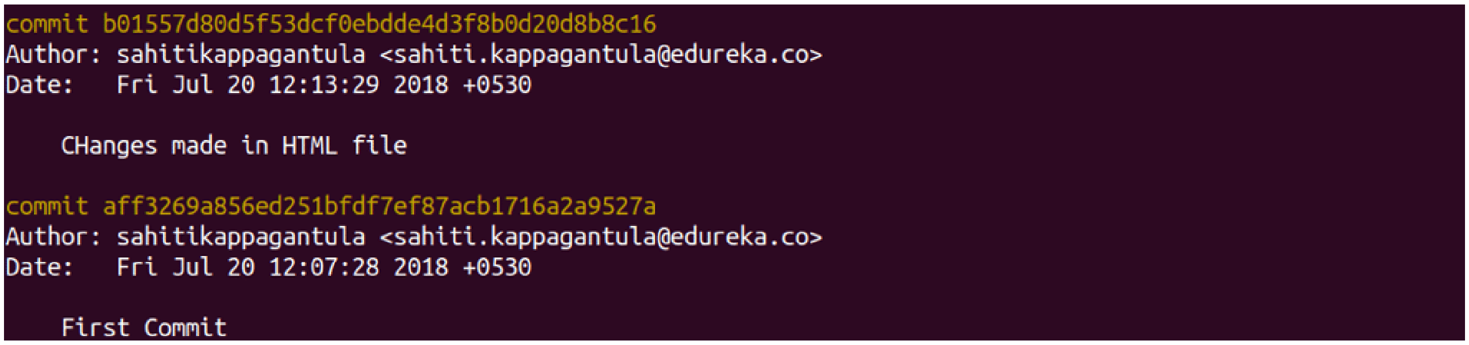
```
edureka@master:~/Documents/DEMO$ git log
commit 09bb8e3f996eaf9a68ac5ba8d8b8fceb0e8641e7 (HEAD -> master)
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:25:17 2018 +0530

    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 aff3269a856ed251bfdf7ef87acb1716a2a9527a
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         af9a68ac5ba8d8b
Author         gantula <sahiti.kappagantula@edureka.co>
Date:          12:25:17 2018 +0530

    Changes made in HTML and CSS fi
```

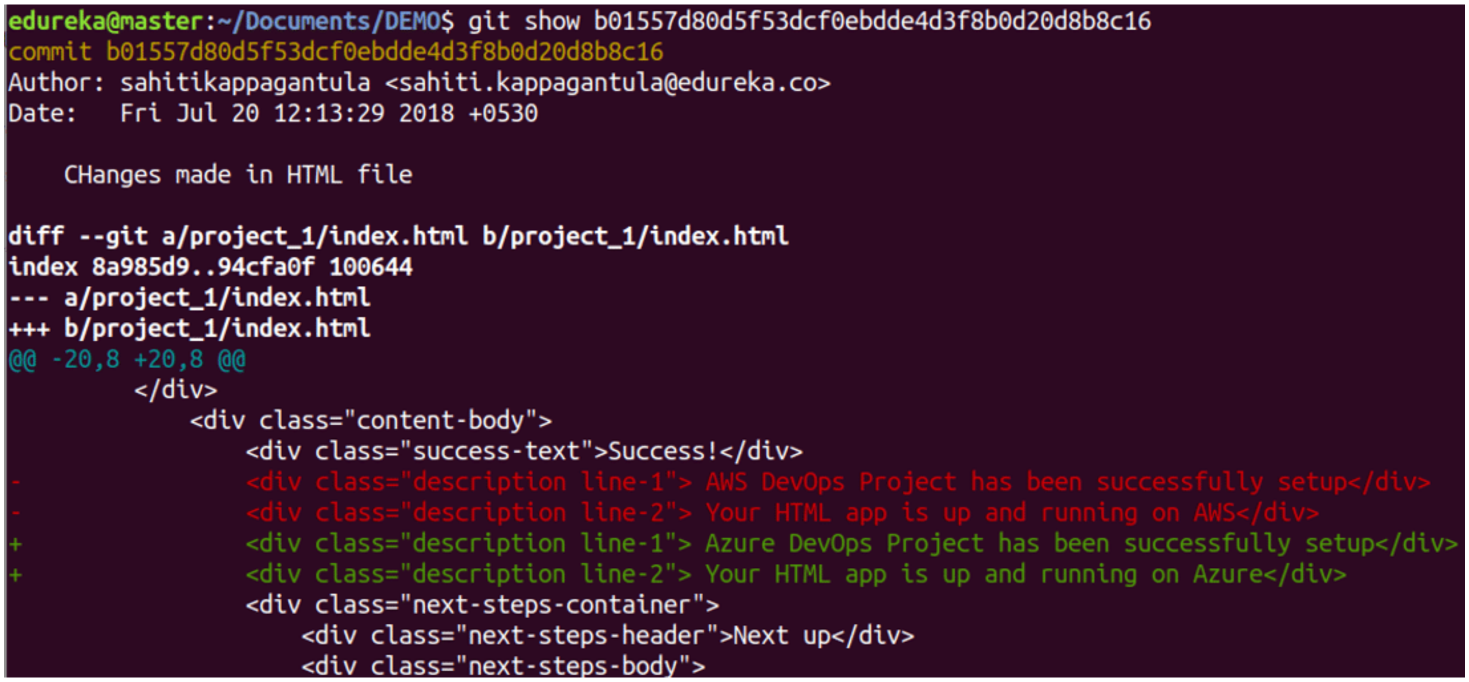


```
commit b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:13:29 2018 +0530

    CHanges made in HTML file

commit aff3269a856ed251bfdf7ef87acb1716a2a9527a
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.

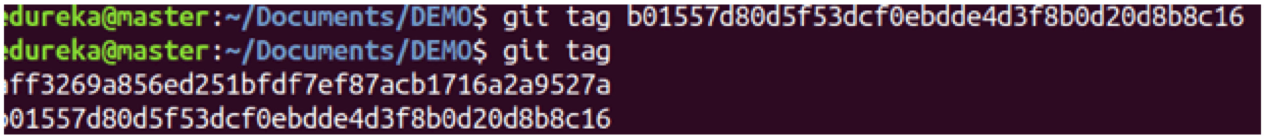


```
edureka@master:~/Documents/DEMO$ git show b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
commit b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
Author: sahitikappagantula <sahiti.kappagantula@edureka.co>
Date:   Fri Jul 20 12:13:29 2018 +0530

    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.



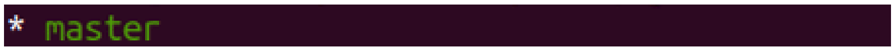
```
dureka@master:~/Documents/DEMO$ git tag b01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
dureka@master:~/Documents/DEMO$ git tag
ff3269a856ed251bfdf7ef87acb1716a2a9527a
01557d80d5f53dcf0ebdde4d3f8b0d20d8b8c16
```

git branch

Usage:

This co all the local branches in the current repository.

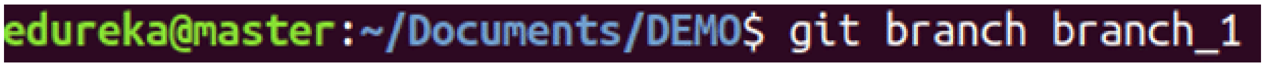
```
edureka@master:~/Documents/DEMO$ git branch
```



```
* master
```

Usage: `git branch [branch name]`

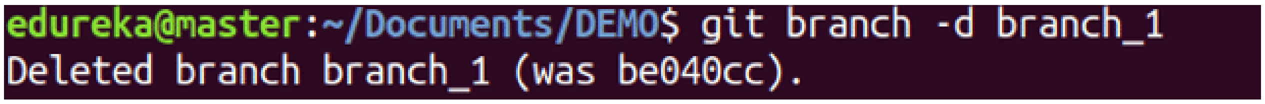
This command creates a new branch.



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

Usage: `git branch -d [branch name]`

This command deletes the feature branch.

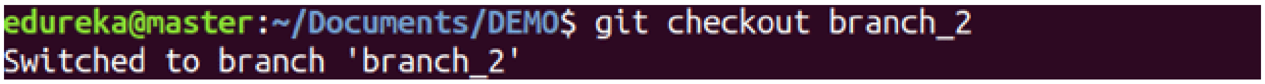


```
edureka@master:~/Documents/DEMO$ git branch -d branch_1
Deleted branch branch_1 (was be040cc).
```

git checkout

Usage: `git checkout [branch name]`

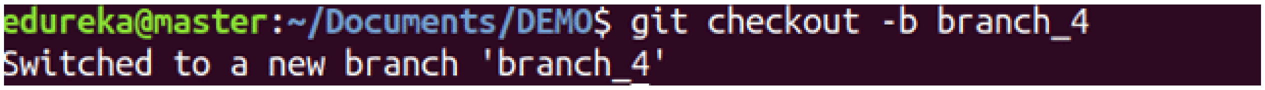
This command is used to switch from one branch to another.



```
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.

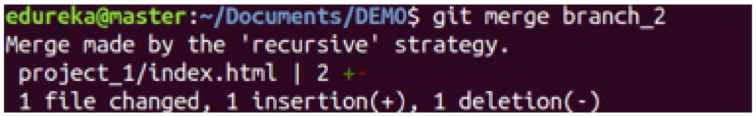


```
edureka@master:~/Documents/DEMO$ git checkout -b branch_4
Switched to a new branch 'branch_4'
```

git merge

Usage: `git merge [branch name]`

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

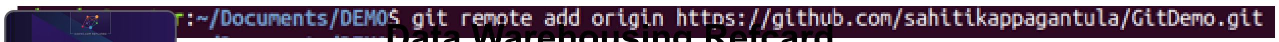


```
edureka@master:~/Documents/DEMO$ git merge branch_2
Merge made by the 'recursive' strategy.
 project_1/index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

git remote

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

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

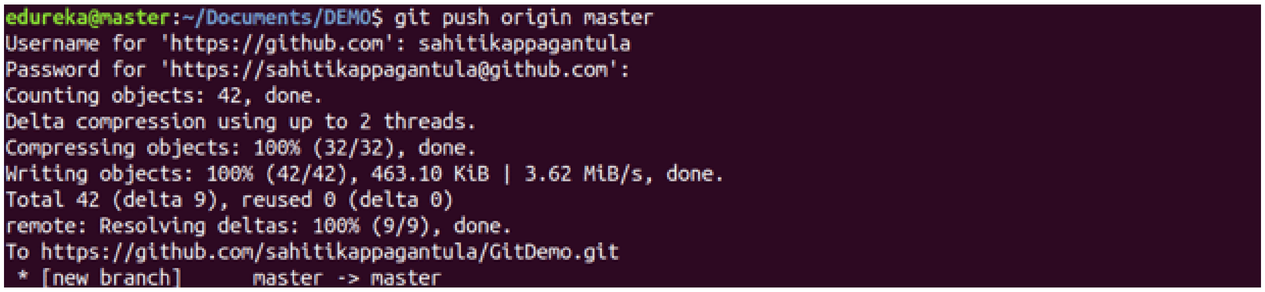


```
r:~/Documents/DEMO$ git remote add origin https://github.com/sahitikappagantula/GitDemo.git
```

git p

Usage: `git push [variable name] master`

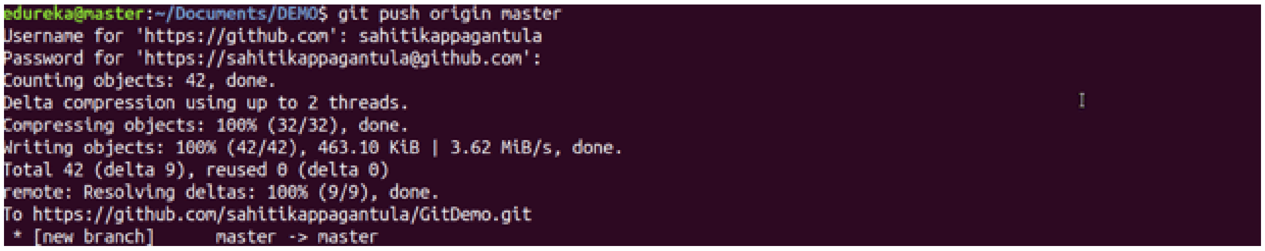
This command sends the committed changes of master branch to your remote repository.



```
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]`

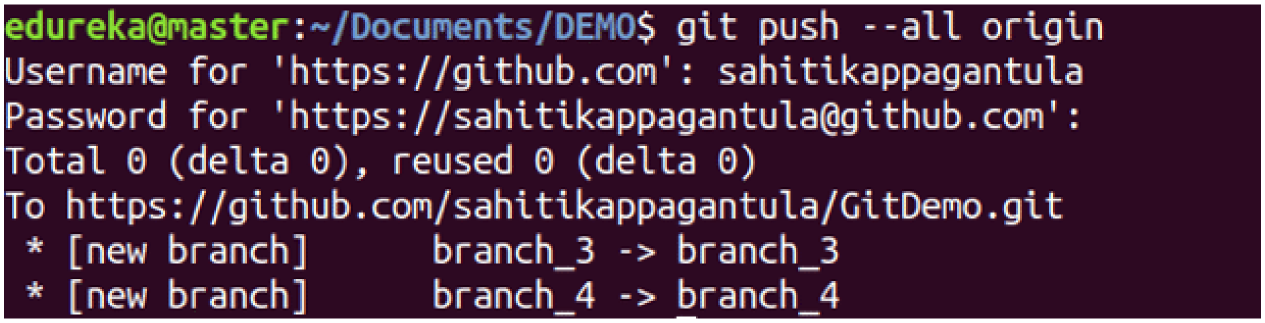
This command sends the branch commits to your remote repository.



```
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]`

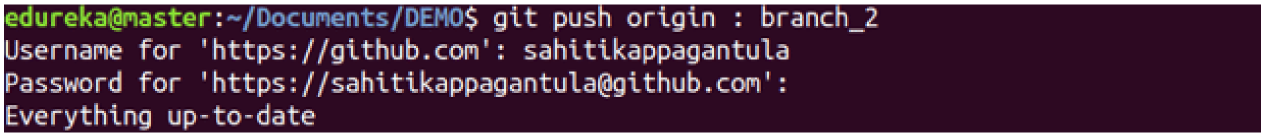
This command pushes all branches to your remote repository.



```
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.

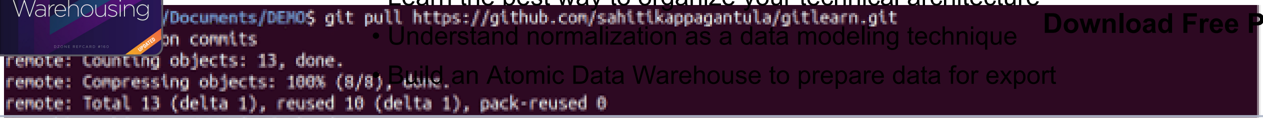


```
edureka@master:~/Documents/DEMO$ git push origin : branch_2
Username for 'https://github.com': sahitikappagantula
Password for 'https://sahitikappagantula@github.com':
Everything up-to-date
```

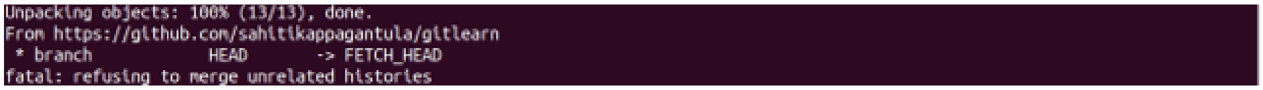
git pull

Usage: `git pull [Repository Link]`

This co[illegible]es and merges changes on the remote server to your working directory.



```
[illegible]/Documents/DEMO$ git pull https://github.com/sahitikappagantula/gitlearn.git
[illegible]on 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
```

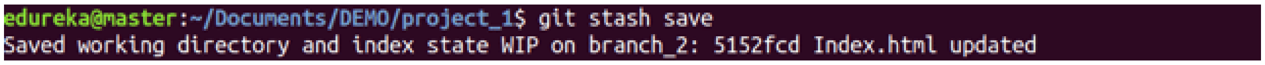


```
Unpacking objects: 100% (13/13), done.
From https://github.com/sahitikappagantula/gitlearn
 * branch            HEAD       -> FETCH_HEAD
fatal: refusing to merge unrelated histories
```

git stash

Usage: `git stash save`

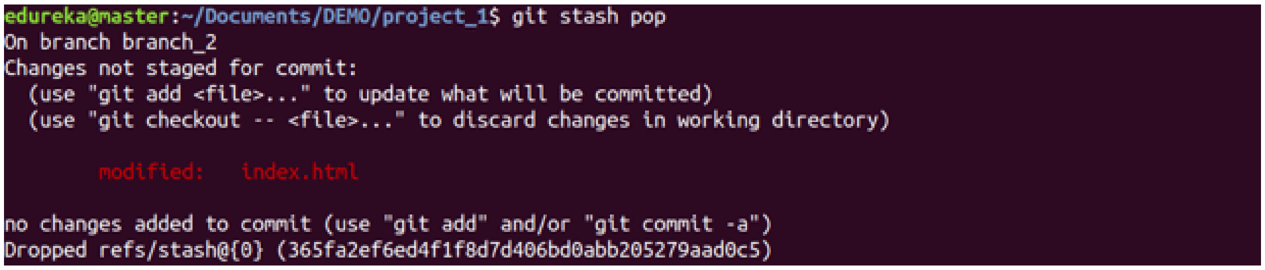
This command temporarily stores all the modified tracked files.



```
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`

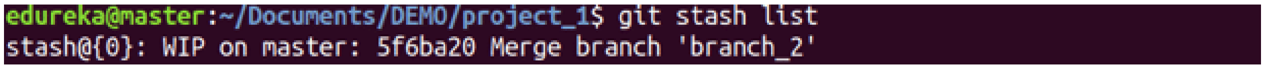
This command restores the most recently stashed files.



```
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`

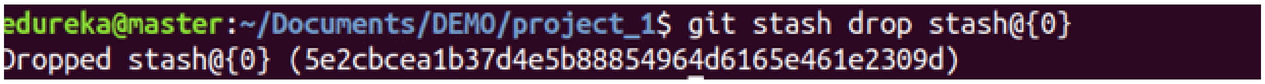
This command lists all stashed changesets.



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

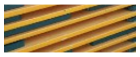
Usage: `git stash drop`

This command discards the most recently stashed changeset.



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

Want to learn more about git commands? Here is a Git Tutorial to get you started. Alternatively, you can take a top-down approach and start with this DevOps Tutorial.



Getting Started with Git

Topics: OPEN SOURCE , GIT , GIT COMMANDS , COMMAND EXAMPLES

Published at DZone with permission of Sahiti Kappagantula , DZone MVB. See the original article here.

Opinions expressed by DZone contributors are their own.

)pen Source Partner Resources

›w Third-Party and Open Source Components Build Hidden Risk Into Software
›racode
|
›w to Handle Insecure Open Source Components
›racode
|
'ographic: Reining In Software Component Risk
›racode
|
›timizing Your Approach to Securing Software Components
›racode
|