

Module 4: Git Workflows

Demo1: Demo on difference between git and git flow.

Problem Statement:

In this demo we will Show commands to see difference between git and git flow.

Solution Steps:

1. Initialize 'GIT BASH' in two different directories, one for 'GIT' and other for 'GIT FLOW'.
In GIT directory only you need to CLONE any GITHUB repository which you have created in previous module demo using 'GIT CLONE' command. Now you will learn difference between GIT and GIT FLOW commands.

GIT

Command to initialize GIT:

git init

```
MINGW64:/c:/GIT DEMO/GitDemo
bhask@LAPTOP-93NE4NT6 MINGW64 /c:/GIT DEMO/GitDemo (main)
$ git init
Reinitialized existing Git repository in C:/GIT DEMO/GitDemo/.git/
bhask@LAPTOP-93NE4NT6 MINGW64 /c:/GIT DEMO/GitDemo (main)
$ |
```

GIT FLOW

Command to initialize GIT FLOW:

git flow init

```
bhask@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (master)
$ git flow init
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/Git Flow Demo/.git/hooks]

bhask@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (develop)
$ |
```

2. Command to create **FEATURE** branch.

GIT

git branch feature

```
MINGW64:/c/GIT DEMO/GitDemo
bhas@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ git branch feature

bhas@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ git branch
feature
* main

bhas@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ |
```

Command to switch from **MAIN** branch to **FEATURE** branch in **GIT**:

git checkout feature

```
MINGW64:/c/GIT DEMO/GitDemo
bhas@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ git checkout feature
Switched to branch 'feature'

bhas@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ |
```

GITFLOW

Command to create and switch to **FEATURE** branch in **GIT FLOW**:

git flow feature start feature_demo

```
MINGW64:/d/Git Flow Demo
bhas@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (develop)
$ git flow feature start feature_demo
Switched to a new branch 'feature/feature_demo'

Summary of actions:
- A new branch 'feature/feature_demo' was created, based on 'develop'
- You are now on branch 'feature/feature_demo'

Now, start committing on your feature. When done, use:

    git flow feature finish feature_demo

bhas@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (feature/feature_demo)
$ |
```

3. Commands to navigate back to **MASTER/MAIN** branch and merge the changes to **MASTER/MAIN** branch.

GIT

Now create text file in '**FEATURE**' Branch in **GIT** and **GIT FLOW** projects, respectively.
Then **ADD** the text file in staging area and perform the **COMMIT**. Refer the below screenshots.

```
MINGW64:/c/GIT DEMO/GitDemo
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ echo 'GIT DEMO' >demo.txt

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ ls
README.md  demo.txt

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ cat demo.txt
GIT DEMO
```

```
MINGW64:/c/GIT DEMO/GitDemo
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ git add demo.txt
warning: LF will be replaced by CRLF in demo.txt.
The file will have its original line endings in your working directory
```

```
MINGW64:/c/GIT DEMO/GitDemo
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ git status
On branch feature
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   demo.txt

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ git commit -m "initial commit"
[feature c1335e2] initial commit
1 file changed, 1 insertion(+)
create mode 100644 demo.txt

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ |
```

```
MINGW64:/c/GIT DEMO/GitDemo
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ ls
README.md  demo.txt

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ |
```

4. Now checkout to '**MAIN**' branch using command:

git checkout main

```
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (feature)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ ls
README.md

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ |
```

5. To add the **FEATURE BRANCH** files into **MAIN BRANCH** use **MERGE** command in **GIT**:

git merge feature

```
MINGW64:/c/GIT DEMO/GitDemo

bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ git merge feature
Updating 342b06d..c1335e2
Fast-forward
 demo.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 demo.txt
```

```
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ |
```

```
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ ls
README.md  demo.txt
```

```
bhask@LAPTOP-93NE4NT6 MINGW64 /c/GIT DEMO/GitDemo (main)
$ |
```

6. GITFLOW

Use the same command mentioned above to create **text file**. **ADD** it in stage area and **COMMIT** the changes.

Then finish the **FEATURE** branch to move the changes from **FEATURE** branch to **DEVELOP** branch using command:

```
git flow feature finish feature_demo
```

```
MINGW64:/d/Git Flow Demo
bhasK@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (feature/feature_demo)
$ git flow feature finish feature_demo
Switched to branch 'develop'
Updating 1dec74d..74d864d
Fast-forward
 demo.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 demo.txt
Deleted branch feature/feature_demo (was 74d864d).

Summary of actions:
- The feature branch 'feature/feature_demo' was merged into 'develop'
- Feature branch 'feature/feature_demo' has been locally deleted
- You are now on branch 'develop'

bhasK@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (develop)
$ |
```

7. Start the **release process** based on your '**DEVELOP**' branch using command:

```
git flow release start release1.0
```

```
MINGW64:/d/Git Flow Demo
bhasK@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (develop)
$ git flow release start release1.0
Switched to a new branch 'release/release1.0'

Summary of actions:
- A new branch 'release/release1.0' was created, based on 'develop'
- You are now on branch 'release/release1.0'

Follow-up actions:
- Bump the version number now!
- Start committing last-minute fixes in preparing your release
- When done, run:

    git flow release finish 'release1.0'

bhasK@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (release/release1.0)
$ |
```

8. Finish the **'RELEASE1.0'** branch to merge your files to **'MASTER'** branch using command:

git flow release finish 'release1.0'

```
MINGW64:/d/Git Flow Demo
bhas@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (release/release1.0)
$ git flow release finish 'release1.0'
Switched to branch 'master'
Merge made by the 'recursive' strategy.
 demo.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 demo.txt
Already on 'master'
fatal: no tag message?
Fatal: Tagging failed. Please run finish again to retry.

bhas@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (master)
$ |
```

9. You will be navigated back to **'Master'** branch from **'Develop'** branch. Your changes are also copied to **MASTER** branch.

```
bhas@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (master)
$ ls
demo.txt

bhas@LAPTOP-93NE4NT6 MINGW64 /d/Git Flow Demo (master)
$ |
```