UMKC

Spring-2017

Software Methods and Tools

Assignment-8

(GIT)

Submitted by:

Moulika Chadalavada

16234180

1. Objective

The main aim of this assignment is to get familiar with GIT workflow and its commands and operations.

2. Working Process

As part of this assignment I am designing a workflow that characterizes how GIT is used when multiple users working on application. Here I have considered 3 developers Bill, Mary, Ben. Initially Remote repository SnakeGameRemote.git is created. Further local repositories are also created for Bill, Mary, and Ben. First for Bill, BillLocalGit repository is created and pushed to remote initially. For Mary and Ben I am cloning remote repository SnakeGameRemote.git to create MaryLocalGit and BenLocalGit respectively. Each user has their respective Development Branch where developer makes changes and push the changes to master branch. Finally, from master branch changes are moved to remote repository SnakeGameRemote.git
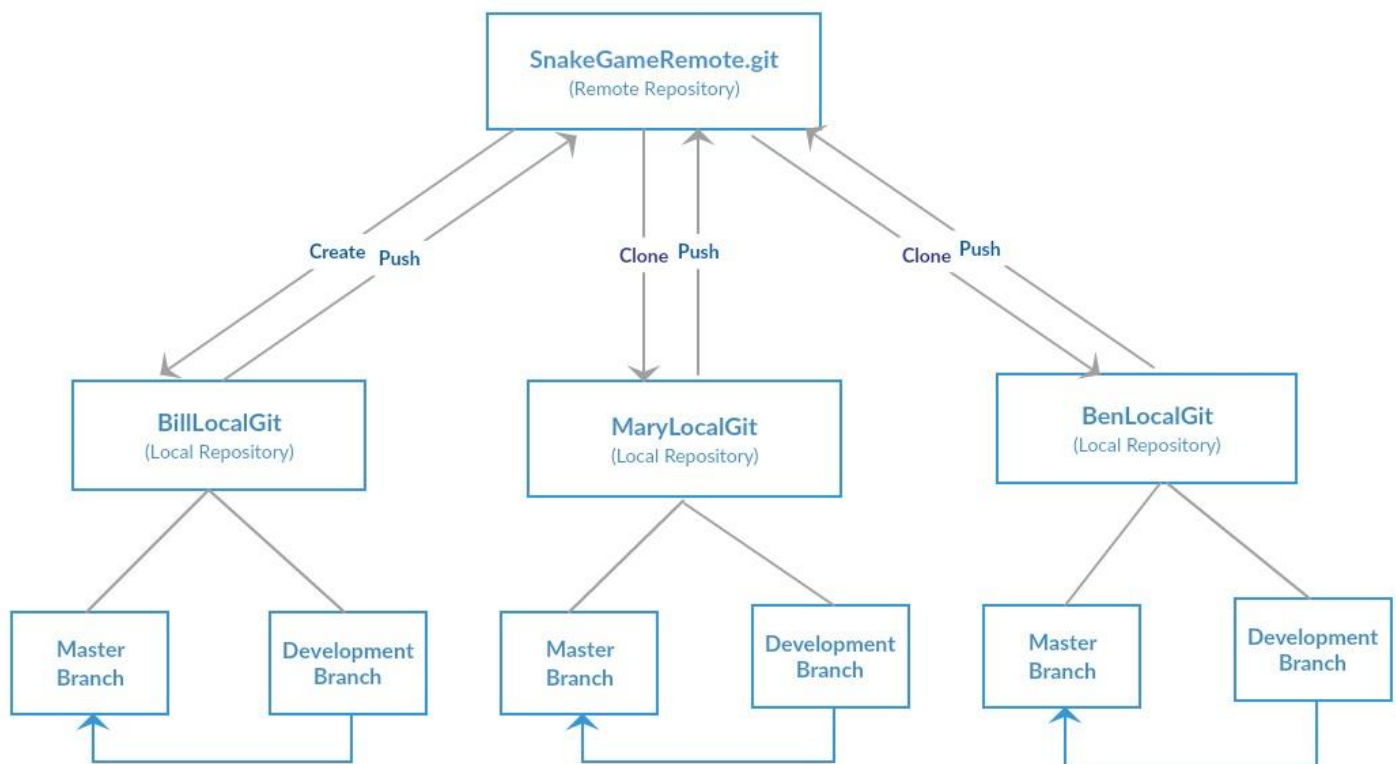
3. GIT Initial Setup

To proceed with the GIT workflow and commands initially GIT is installed. Then GIT Bash is opened and we proceed with executing commands for maintain project in GIT. So initially the Repository Name, default editor particulars are given as shown below

```
Dell@MOULIKA MINGW64 ~
$ git config --global user.name "Moulika"

Dell@MOULIKA MINGW64 ~
$ git config --global core.editor emacs
```

4. GIT Operations

GIT Working flow must involve all the important concepts/operations of GIT. These include: remote branch, tracking branch, branching, merging, merging conflicts, repository creation, tracking new files, modifying files, staging files, committing changes, fetching, pulling, and pushing.

1. Creating Remote Repository
- Created directory SnakeGameRemote.git
- Initialize folder as GIT remote repository using ***git init --bare***

```
Dell@MOULIKA MINGW64 ~
$ cd F:

Dell@MOULIKA MINGW64 /f
$ cd GIT

Dell@MOULIKA MINGW64 /f/GIT
$ mkdir SnakeGameRemote.git

Dell@MOULIKA MINGW64 /f/GIT
$ cd SnakeGameRemote.git

Dell@MOULIKA MINGW64 /f/GIT/SnakeGameRemote.git
$ git init --bare
Initialized empty Git repository in F:/GIT/SnakeGameRemote.git/

Dell@MOULIKA MINGW64 /f/GIT/SnakeGameRemote.git (BARE:master)
$ ls
config  description  HEAD  hooks/  info/  objects/  refs/
```

2. Creating Local Repository and Tracking New Files
- Create new local repository BillLocalGit
- Initialize local repository using ***git init***
- Copy code to BillLocalGit folder
- Add tacked files to staging using ***git add ****
- Finally commit staging files to local repository using ***git commit -m "Bill Initial Commit"***

```
Dell@MOULIKA MINGW64 /f/GIT
$ mkdir BillLocalGit

Dell@MOULIKA MINGW64 /f/GIT
$ cd BillLocalGit

Dell@MOULIKA MINGW64 /f/GIT/BillLocalGit
$ git init
Initialized empty Git repository in F:/GIT/BillLocalGit/.git/

Dell@MOULIKA MINGW64 /f/GIT/BillLocalGit (master)
$ git add *
warning: LF will be replaced by CRLF in org/psnbtech/BoardPanel.java.
The file will have its original line endings in your working directory.
warning: LF will be replaced by CRLF in org/psnbtech/Clock.java.
The file will have its original line endings in your working directory.
warning: LF will be replaced by CRLF in org/psnbtech/Direction.java.
The file will have its original line endings in your working directory.
warning: LF will be replaced by CRLF in org/psnbtech/SidePanel.java.
The file will have its original line endings in your working directory.
warning: LF will be replaced by CRLF in org/psnbtech/SnakeGame.java.
The file will have its original line endings in your working directory.
warning: LF will be replaced by CRLF in org/psnbtech/TileType.java.
The file will have its original line endings in your working directory.

Dell@MOULIKA MINGW64 /f/GIT/BillLocalGit (master)
$ git commit -m "Bill initial commit"
[master (root-commit) 9107404] Bill initial commit
 6 files changed, 1159 insertions(+)
 create mode 100644 org/psnbtech/BoardPanel.java
 create mode 100644 org/psnbtech/Clock.java
 create mode 100644 org/psnbtech/Direction.java
 create mode 100644 org/psnbtech/SidePanel.java
 create mode 100644 org/psnbtech/SnakeGame.java
 create mode 100644 org/psnbtech/TileType.java
```
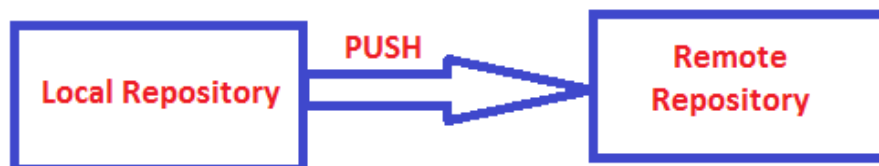
3. Pushing to Remote Branch
   - Next add the remote repository into local repositories list of remote servers using
     git remote add origin /f/GIT/SnakeGameRemote.git
   - Finally push local repository to remote repository using
     git push origin master

4. Cloning Remote Repository

- Create new Local repository MaryLocalGit but just cloning the remote repository SnakeGameRemote.git using ***git clone file:///f/GIT/SnakeGameRemote.git MaryLocalGit***
- Once it is successfully created, all files in remote repository will be copied to Local Repository.

5. New Branch Creation

- Branch is just a movable pointer
- New Branch development is created under MaryLocalGit using ***git checkout -b development***
- This command automatically switches to newly created Branch
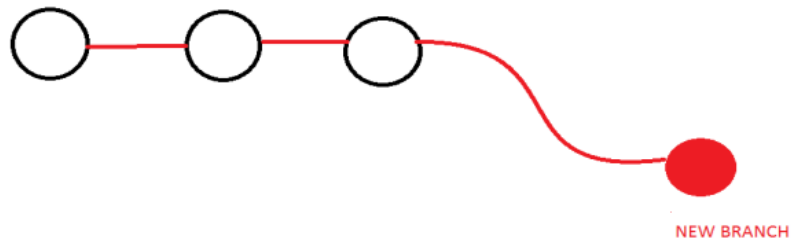
NEW BRANCH

```
Dell@MOULIKA MINGW64 /f/GIT/MaryLocalGit (master)
$ git checkout -b development
Switched to a new branch 'development'

Dell@MOULIKA MINGW64 /f/GIT/MaryLocalGit (development)
```

6. Staging Files

- After we change a file that is already being tracked, we need to explicitly stage it into our next commit.
- Initially made changes in SnakeGame.java in MaryLocalRepository. Using **git status** we can know what files are changes in that repository.


```
Dell@MOULIKA MINGW64 /f/GIT/MaryLocalGit (development)
$ git status
On branch development
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:    org/psnbtech/SnakeGame.java

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

- Further these modified files has to be staged before committing using *git add SnakeGame.java*
- Then changes has to be committed to local repository using
  git commit -m "Mary Development Changes"
- Finally these changes are pushed to Server using *git push origin development*

```
Dell@MOULIKA MINGW64 /f/GIT/MaryLocalGit (development)
$ cd org/psnbtech

Dell@MOULIKA MINGW64 /f/GIT/MaryLocalGit/org/psnbtech (development)
$ git add SnakeGame.java

Dell@MOULIKA MINGW64 /f/GIT/MaryLocalGit/org/psnbtech (development)
$ git commit -m "Mary Development changes"
[development 9298da8] Mary Development changes
 1 file changed, 1 insertion(+), 1 deletion(-)

Dell@MOULIKA MINGW64 /f/GIT/MaryLocalGit/org/psnbtech (development)
$ git push origin development
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 416 bytes | 0 bytes/s, done.
Total 5 (delta 2), reused 0 (delta 0)
To file:///f/GIT/SnakeGameRemote.git
 * [new branch]      development -> development
```

7. Cloning and Fetching Repository

- Let's create another local repository BenLocalGit by cloning from remote repository using
  git clone file:///f/GIT/SnakeGameRemote.git MaryLocalGit

```
Dell@MOULIKA MINGW64 /f/GIT
$ git clone file:///f/GIT/SnakeGameRemote.git BenLocalGit
Cloning into 'BenLocalGit'...
remote: Counting objects: 15, done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 15 (delta 2), reused 0 (delta 0)
Receiving objects: 100% (15/15), 9.95 KiB | 0 bytes/s, done.
Resolving deltas: 100% (2/2), done.

Dell@MOULIKA MINGW64 /f/GIT
$ cd BenLocalGit
```

- Further we inspect remote i.e. it shows more information about remote and particularly about branches. We do this using *git remote show origin.*
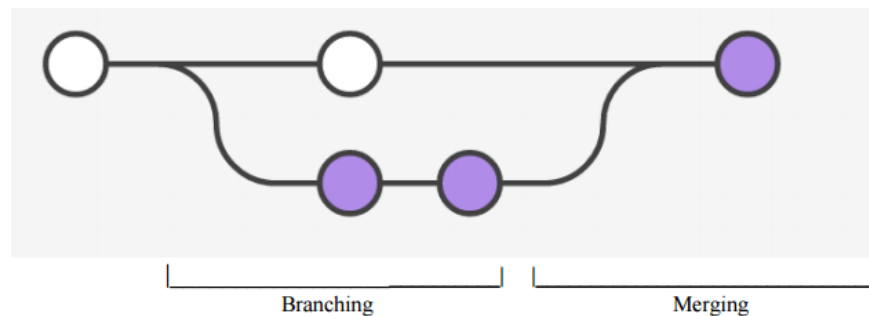- Next we pulls down data from remote project to local repository using *git fetch origin*

8. Tracking Branch and Merging

- Tracking branch is nothing but checking out a local branch from a remote branch.
- This can be done using *git checkout -b development origin/development*
- Further the branch is switched to master using *git checkout master*
- Finally, we are merging the development with master using *git merge development*
- As we merged the development in master the changes made by Mary in SnakeGame.java will also be reflected.

9. Conflicts and its Resolution

- Initially switched Mary's Local Repository branch to Development from Master, then changes are made in SnakeGame.java and committed the changes in MaryLocalGit repository. ([Same as steps followed in Staging Files](#))

- Commands used are shown below
  git add SnakeGame.java
  git commit -m "Mary Development Code Commit"
  git push origin development

```
Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit (master)
$ git checkout development
Switched to branch 'development'
Your branch is up-to-date with 'origin/development'.

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit (development)
$ git status
On branch development
Your branch is up-to-date with 'origin/development'.
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:   org/psnbtech/SnakeGame.java

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

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit (development)
$ cd /org/psnbtech
bash: cd: /org/psnbtech: No such file or directory

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit (development)
$ cd org/psnbtech

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (development)
$ git add SnakeGame.java

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (development)
$ git commit -m "Mary Development Code Commit"
[development c098b9b] Mary Development Code Commit
 1 file changed, 1 insertion(+), 1 deletion(-)

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (development)
$ git push origin development
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 425 bytes | 0 bytes/s, done.
Total 5 (delta 2), reused 0 (delta 0)
To file:///f/GIT/SnakeGameRemote.git
   9298da8..c098b9b  development -> development
```

- Once the changes are committed in development branch, now branch is shifted to master. And changes are made in SnakeGame.java at same line which we did in development branch. As done in previous step we commit the changes in master branch

```
Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (development)
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (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:   SnakeGame.java

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

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (master)
$ git add SnakeGame.java

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (master)
$ git commit -m "Mary Master Code Commit"
[master f23862c] Mary Master Code Commit
 1 file changed, 1 insertion(+), 1 deletion(-)
```

- So finally, when we try to merge development branch changes in master branch using *git merge development* conflicts arise as shown below.

```
Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (master)
$ git merge development
Auto-merging org/psnbtech/SnakeGame.java
CONFLICT (content): Merge conflict in org/psnbtech/SnakeGame.java
Automatic merge failed; fix conflicts and then commit the result.
```

- To resolve this conflict, we need to manually edit the file SnakeGame.java as shown below

```
    /**
     * The number of milliseconds that should pass between each frame.
     */
    private static final long FRAME_TIME = 1000L / 50L;

    /**
     * The minimum length of the snake. This allows the snake to grow
     * right when the game starts, so that we're not just a head moving
     * around on the board.
     */
<<<<<<< HEAD
    private static final int MIN_SNAKE_LENGTH = 10; //changed value to 10
=======
    private static final int MIN_SNAKE_LENGTH = 9; //Change Snake Length from 5 to 9
>>>>>>> development

    /**
     * The maximum number of directions that we can have polled in the
     * direction list.
     */
```

- Once the file is edited manually, again this file has to be moved to staged state and it has to be committed in local repository before merging with development. (Same as steps followed in Staging Files)
- Finally, when we merge development branch with master branch, no conflicts arise and everything is up-to-date.

```
Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (master|MERGING)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both modified:   SnakeGame.java

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

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (master|MERGING)
$ git add SnakeGame.java

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (master|MERGING)
$ git commit -m "Mary Master Code Commit after Conflict Resolve"
[master 3277fde] Mary Master Code Commit after Conflict Resolve

Dell@MOULIKA MINGW64 /f/GIT/BenLocalGit/org/psnbtech (master)
$ git merge development
Already up-to-date.
```

10. PULL Command

- Pull command fetches data from the server, and automatically tries to merge it into the code we are currently working on.
- Now we are pulling the changes from remote serve to BillLocalGit repository using **git pull**

```
Dell@MOULIKA MINGW64 /f/GIT
$ cd BillLocalGit

Dell@MOULIKA MINGW64 /f/GIT/BillLocalGit (master)
$ git pull
remote: Counting objects: 10, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 10 (delta 4), reused 0 (delta 0)
Unpacking objects: 100% (10/10), done.
From F:/GIT/SnakeGameRemote
 * [new branch]      development -> origin/development
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=origin/<branch> master
```