Capgemini

Introduction to Git

Lesson 01 : Getting Started with Git

Lesson Objective

- In this lesson, you will learn:
  - Introduction to Git
  - Creating Git repository
  - Understanding local and remote repository
  - Committing changes to the repository
  - Branching the repository
  - Merging the changes back to the primary repository.
  - Creating Azure DevOps account
  - Creating & Collaborating with Azure DevOps repository

Introduction

- What is Source Control?
  - Source control helps team manage changes to source code over time.
- What is Version Control?
  - The term version control is used interchangeably with source control.
- Difference between source control and version control:
  - Source control is specific to source code, where as version control also covers large binary files and digital assets.

Introduction

- What is source control management?
  - Source control management (SCM) refers to tools that helps software team manage changes to source code over time.
  - Version control software keeps track of every modification to the code in a special kind of database.
  - Software teams that do not use any form of version control often run into problems like not knowing which changes that have been made are available to users or the creation of incompatible changes between two unrelated pieces of work that must then be painstakingly untangled and reworked.

What is Git?

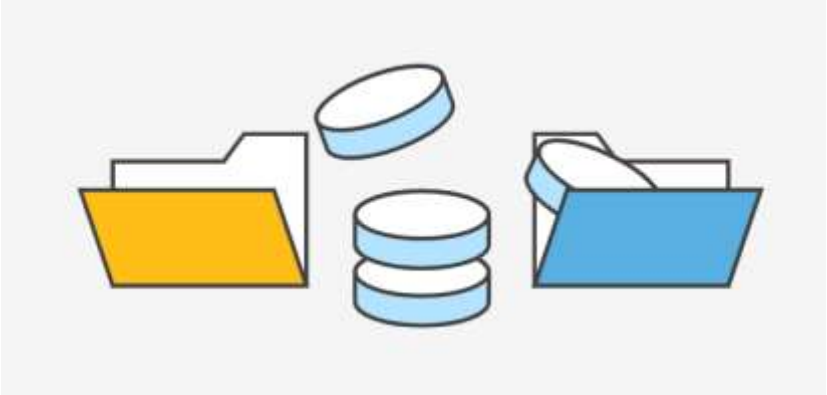
- By far, the most widely used modern version control system in the world today is Git.
- Git is a mature, actively maintained open source project originally developed in 2005 by Linus Torvalds, the famous creator of the Linux operating system kernel.
- Unlike CVS or SVN (Subversion), Git is a DVCS (Distributed Version Control System).
- In addition to being distributed, Git has been designed with performance, security and flexibility in mind.

Benefits of Git

- Complete long term change history of every file is maintained.
- Multiple independent streams of work can co-exist with branches.
- Distributed Version Control System (DVCS).
- Performance of Git is very strong when compared to many alternatives.
- Git repository are secured with a cryptographically secure hashing algorithm called SHA1.
- Open source

Installing Git

- Installing Git on Windows:
  - Download the latest Git from https://gitforwindows.org/
  - When you've successfully started the installer, you should see the **Git Setup** wizard screen. Follow the **Next** and **Finish** prompts to complete the installation. The default options are pretty sensible for most users.



Basic Commands

- Open a Command Prompt (or Git Bash if during installation you elected not to use Git from the Windows Command Prompt).
- Checking the installed version:

```
$ git --version
git version 2.27.0.windows.1
```

- Setting username and email:
- git config --global user.name “[name]”
- git config --global user.email “[email address]”

- Verifying username and email:

```
$ git config --global user.name
```

```
$ git config --global user.email
```

Basic Commands (Cont....)

- **git and get help**: Both commands are same.

```
$ git
usage: git [--version] [--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>]
           <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
   sparse-checkout   Initialize and modify the sparse-checkout
```

- **get help init**: Offers browser based help.

Creating new local git repository

- Step1. Open up your project folder, and navigate into the same.
  - For checking your current location:

```
$ pwd
/c/Users/mrajhans
```

  - Identify location of repository and navigate:

```
$ cd c:/study/GitRepos
```

  - Create new directory to make it as repository and navigate to it.

```
$ mkdir HelloWorld
```

```
$ cd HelloWorld/
```

  - Initialize directory as git repository:

```
$ git init
Initialized empty Git repository in C:/Study/GitRepos/HelloWorld/.git/
```

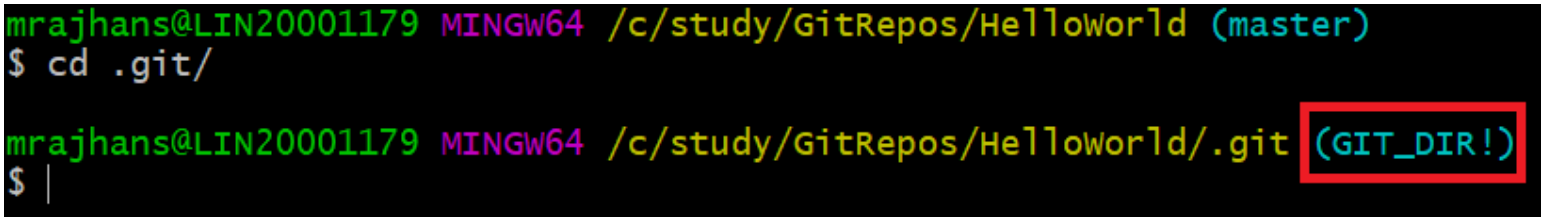
  - After repository is created, then observe that now prompt is showing it as master, as shown below:

```
mrajhans@LIN20001179 MINGW64 /c/study/GitRepos/HelloWorld (master)
$
```

.git folder in repository

- The . git folder contains all the information that is necessary for your project in version control and all the information about commits, remote repository address, etc. All of them are present in this folder. It also contains a log that stores your commit history so that you can roll back to history.

- Viewing .git directory:
  - Change the directory to .git:



.git folder in repository (cont….)

- .git folder contains the following information. You can view the available file and folders with following command.

```
mrajhans@LIN20001179 MINGW64 /c/study/GitRepos/HelloWorld/.git (GIT_DIR!)
$ ls -al
total 11
drwxr-xr-x 1 mrajhans 1049089   0 Jun 29 17:04 ./
drwxr-xr-x 1 mrajhans 1049089   0 Jun 29 17:12 ../
-rw-r--r-- 1 mrajhans 1049089 130 Jun 29 17:04 config
-rw-r--r-- 1 mrajhans 1049089  73 Jun 29 17:04 description
-rw-r--r-- 1 mrajhans 1049089  23 Jun 29 17:04 HEAD
drwxr-xr-x 1 mrajhans 1049089   0 Jun 29 17:04 hooks/
drwxr-xr-x 1 mrajhans 1049089   0 Jun 29 17:04 info/
drwxr-xr-x 1 mrajhans 1049089   0 Jun 29 17:04 objects/
drwxr-xr-x 1 mrajhans 1049089   0 Jun 29 17:04 refs/
```

Adding new file to git repository

- Create file in the current folder / repo:

```
$ nano helloworld.js
```

- This will open the file in edit mode, as shown below. You can write anything into it.

- Press Ctrl+X to close and press Y to save changes.

Git status

- To check status of repository, type command, git status:

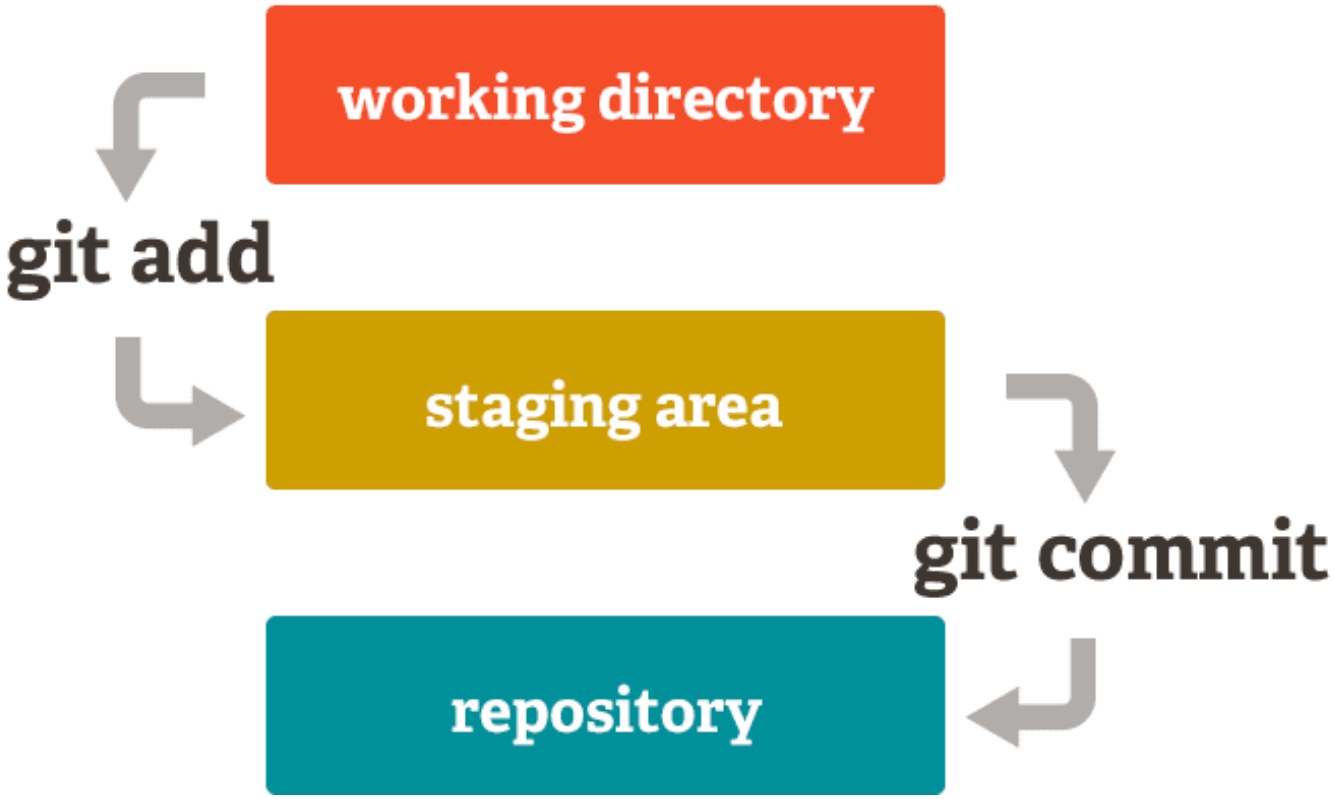
```
$ git status
On branch master

No commits yet

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

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

Working directory, staging area and repository



Adding file into staging area

- To add file from working directory to the staging area we can use git add command.

```
mrajhans@LIN20001179 MINGW64 /c/study/GitRepos/HelloWorld (master)
$ git add helloworld.js
warning: LF will be replaced by CRLF in helloworld.js.
The file will have its original line endings in your working directory
```

- Checking status after adding file in staging area:

```
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   helloworld.js
```

Removing file from staging area

- To remove file from staging area and put it back into the working directory, use git reset command.
- `$ git reset`

```
$ git status
On branch master

No commits yet

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

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

- To add the file again in staging area, use git add command.

Committing changes to the local repository

- When commit command is used all the tracked changes will be committed/ reflected to the repository.
- To commit the files from staging area to local repository, use git commit command.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/HelloWorld (master)
$ git commit -m "First Commit"
[master (root-commit) 83edd88] First Commit
 1 file changed, 1 insertion(+)
 create mode 100644 helloworld.js
```

- Status after commit:

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/HelloWorld (master)
$ git status
On branch master
nothing to commit, working tree clean
```

Creating repository in the existing working directory

- You can download the sample repository from shared material.
- Copy the folder and navigate to it.

```
mrajhans@LIN20001179 MINGW64 ~
$ cd C:/Study/GitRepos/SampleRepository
```

- Now, initialize the directory as a git repository.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository
$ git init
Initialized empty Git repository in C:/Study/GitRepos/SampleRepository/.git/
```

- add all the existing files from the working directory into staging area with git add command.

Creating repository in the existing working directory (Cont....)

- Now, check the status of created repository.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        404.html
        apple-touch-icon.png
        browserconfig.xml
        css/
        favicon.ico
        humans.txt
        index.html
        js/
        tile-wide.png
        tile.png

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

- add all the existing files from the working directory into staging area with git add command.

Creating repository in the existing working directory (Cont....)

- Now add all the existing files from the working directory into staging area with git add command.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git add .
warning: LF will be replaced by CRLF in 404.html.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in browserconfig.xml.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in css/main.css.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in css/normalize.css.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in humans.txt.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in index.html.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in js/main.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in js/vendor/jquery-1.11.2.min.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in js/vendor/modernizr-2.8.3-respond-1.4.2.min.js.
The file will have its original line endings in your working directory
```

Creating repository in the existing working directory (Cont....)

- After checking status, it shows the status as files are added in staging area .

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   404.html
        new file:   apple-touch-icon.png
        new file:   browserconfig.xml
        new file:   css/main.css
        new file:   css/normalize.css
        new file:   css/normalize.min.css
        new file:   favicon.ico
        new file:   humans.txt
        new file:   index.html
        new file:   js/main.js
        new file:   js/vendor/jquery-1.11.2.min.js
        new file:   js/vendor/modernizr-2.8.3-respond-1.4.2.min.js
        new file:   tile-wide.png
        new file:   tile.png
```

Creating repository in the existing working directory (Cont.…)

- Once files are added in staging area, commit the files into local repository.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git commit -m "First Commit"
[master (root-commit) 1bd1941] First Commit
 14 files changed, 992 insertions(+)
 create mode 100644 404.html
 create mode 100644 apple-touch-icon.png
 create mode 100644 browserconfig.xml
 create mode 100644 css/main.css
 create mode 100644 css/normalize.css
 create mode 100644 css/normalize.min.css
 create mode 100644 favicon.ico
 create mode 100644 humans.txt
 create mode 100644 index.html
 create mode 100644 js/main.js
 create mode 100644 js/vendor/jquery-1.11.2.min.js
 create mode 100644 js/vendor/modernizr-2.8.3-respond-1.4.2.min.js
 create mode 100644 tile-wide.png
 create mode 100644 tile.png
```

Creating repository in the existing working directory (Cont.…)

- Now all the files in working directory are in synch with files from working directory. Check the status of the same.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git status
On branch master
nothing to commit, working tree clean
```

Modifying the files in the local repository

- Modify file index.html.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ nano index.html
```

- Check the status, now it shows the status of index file as modified.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

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

Modifying the files in the local repository (Cont....)

- Add the updated index file into staging area..

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git add .
warning: LF will be replaced by CRLF in index.html.
The file will have its original line endings in your working directory
```

- Now commit the files.

```
mrajhans@LIN20001179 MINGW64 /c/Study/GitRepos/SampleRepository (master)
$ git commit -m "Updated index file commit"
[master d514e91] Updated index file commit
 1 file changed, 1 insertion(+), 1 deletion(-)
```