Summary

Git commands

Configuring user information used across all local repositories

1) git config --global user.name "firstname"
→ sets a name that is identifiable for credit when review version history.

2) git config --global user.email "[valid-email]"
→ sets email address that will be associated with each history marker.

3) git config --global color.ui auto
→ sets automatic command line coloring for git for easy reviewing.

Setup & initialize

1) git init
initialize an existing directory as a git repository.

2) git clone [url]
retrieve an entire repository from a hosted location via url.

Working with snapshots & git staging area.

1) git status
• show modified files in working directory, staged for your next commit.

2) git add :
- add the files as it looks now to your next commit.

3) git commit -m ["Message"]
- commit your staged content as a new commit snapshot.

Branch & Merge

1) git branch
list your branches. a * will appear next to the currently active branch.

2) git branch [branch-name]
- create a new branch at the current commit

3) git checkout
- switch to another branch.

4) git merge :
- merge the specified branch's history into the current one

5) git log
- shows all commits in the current branch's history.

6) git log -1
- shows the first commit

7) git show [SHA].
- show any object in git in human readable form

8) git push [alias] [branch].
- transmit local branch commits to the remote repository branch.

9) git pull
- fetch & merge any commits from the tracking remote branch

10) git rm [file]
- deletes the file from the project & stage the removal for commit.

11) git log --oneline
- condense each commit to a single line

Git Ignoring
We need to track only source code files & configuration files & not the environment files as they will be different for different person.

To remove files from git but to still keep them on local machine, we use
- git rm --cached *.classpath
Cached & to keep file in work filename directory but remove from git staging area.

- git rm -r --cached bin/
All files i.e cached & bin are removed from staging area.

=> To avoid untracked files to be committed again, we create .gitignore file & add the files here.

1) touch .gitignore // creates file
2) nano .gitignore // nano is text editor
   On nano file write
   *.class
   bin/
   .classpath
   Now save above file.
3) git status
4) git add .
5) git commit -m "Added .gitignore & removed bin/ from staging".

Commit history

1) git log
   prints commit history of current branch (most recent ones shown up)
2) git log -2
   shows first two commits (recent ones)
3) git log --pretty=online
   shows hash value & a message in single line.

Amend last commit

1) git commit --amend
   ammends the commit message (last commit message).

DATE
PAGE No.

To change the editor

git config --global core.editor "vi"

- set editor as "vi" editor.

Git hosting & remote repository

git repositories

1) Github
2) Atlassian bitbucket (integrated with jira, confluence)
3) gitlab

a) git remote -v

→ to check that git repository is not connected with any other local repository yet.

b) git remote add origin [git URL]

c) git push origin -u master

-u : upstream (linking our repo to central repo).

README.md file : people use this file to put their repository description

Git stash :

When to use stash

- Temporarily put work aside
- Apply the same changes on multiple branches
- Create a new branch focused on current changes later

a) git stash
saved working directory

b) git stash apply
=> to apply changes made in file.

c) git stash list
=> list of stash changes.

Git Reflog

At times, we can accidently loose a commit.
Git won't be able to restore the files that were never in the staging area.

git reflog :- reflog silently records the head pointer & how does it look like, everytime we commit changes, reflog is updated.

-> git log -g
-> git reflog --since="1 hour"
(shows last 1 hour commit)