Objectives

- Explain how to resolve the conflict during merge.

In this hands-on lab, you will learn how to:

- Implement conflict resolution when multiple users are updating the trunk (or master) in such a way that it results into a conflict with the branch's modification.

Prerequisites

The following are the pre-requisites to complete this hands-on lab:

- Hands-on ID: **"Git-T03-HOL_001"**

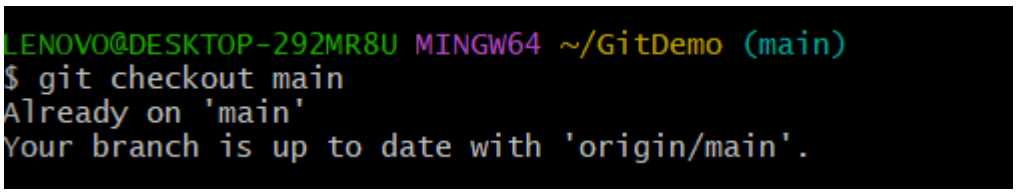
Please follow the instructions to complete the hands-on. Each instruction expect a command for the Git Bash.

1. Verify if master is in clean state.
2. Create a branch **"GitWork".** Add a file "hello.xml".
3. Update the content of "hello.xml" and observe the status
4. Commit the changes to reflect in the branch
5. Switch to master.
6. Add a file **"hello.xml"** to the master and add some different content than previous.
7. Commit the changes to the master
8. Observe the log by executing **"git log –oneline –graph –decorate –all"**
9. Check the differences with Git diff tool
10. For better visualization, use P4Merge tool to list out all the differences between master and branch
11. Merge the bran to the master
12. Observe the git mark up.
13. Use 3-way merge tool to resolve the conflict
14. Commit the changes to the master, once done with conflict
15. Observe the git status and add backup file to the .gitignore file.
16. Commit the changes to the .gitignore
17. List out all the available branches
18. Delete the branch, which merge to master.
19. Observe the log by executing **"git log –oneline –graph –decorate"**

Step-by-step Conflict Resolution in Git

Verify master is clean

```
git checkout main       # or 'master' if that's your default
git status              # should say "working tree clean"
```



Create branch GitWork and add hello.xml
git checkout -b GitWork
echo "<message>Hello from GitWork branch</message>" > hello.xml
git add hello.xml
git status

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git checkout -b GitWork
Switched to a new branch 'GitWork'
```

Commit in branch
git commit -m "Add hello.xml in GitWork branch"

Switch to master
git checkout main

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (GitWork)
$ echo "<message>Hello from GitWork branch</message>" > hello.xml
git add hello.xml
git status
warning: in the working copy of 'hello.xml', LF will be replaced by CRLF the next time Git touches it
On branch GitWork
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   hello.xml


LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (GitWork)
$ git commit -m "Add hello.xml in GitWork branch"
[GitWork 0f8046e] Add hello.xml in GitWork branch
 1 file changed, 1 insertion(+)
 create mode 100644 hello.xml

LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (GitWork)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Add a different hello.xml in master
echo "<message>Hello from master branch</message>" > hello.xml
git add hello.xml
git commit -m "Add hello.xml in master branch with different content"

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ echo "<message>Hello from master branch</message>" > hello.xml
git add hello.xml
git commit -m "Add hello.xml in master branch with different content"
warning: in the working copy of 'hello.xml', LF will be replaced by CRLF the next time Git touches it
[main 4288c3f] Add hello.xml in master branch with different content
 1 file changed, 1 insertion(+)
 create mode 100644 hello.xml
```

See log of all branches
git log --oneline --graph --decorate –all

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git log --oneline --graph --decorate --all
* 4288c3f (HEAD -> main) Add hello.xml in master branch with different content
| * 0f8046e (GitWork) Add hello.xml in GitWork branch
|/
* fccbb92 (origin/main) Add new line in welcome.txt from GitNewBranch branch
* 46bf5cd Add .gitignore to ignore .log files and log folder
* e2e88a9 Hello all...Now I am working on GIT Hands-on week8
```

Compare master and GitWork

Text diff:

git diff main GitWork

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git diff main GitWork
diff --git a/hello.xml b/hello.xml
index 82868a7..0bf4c19 100644
--- a/hello.xml
+++ b/hello.xml
@@ -1 +1 @@
-<message>Hello from master branch</message>
+<message>Hello from GitWork branch</message>
```

Visual diff (P4Merge if set):

git difftool main GitWork

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git difftool main GitWork

This message is displayed because 'diff.tool' is not configured.
See 'git difftool --tool-help' or 'git help config' for more details.
'git difftool' will now attempt to use one of the following tools:
opendiff kdiff3 tkdiff xxdiff meld kompare gvimdiff diffuse diffmerge ecmerge p4merge araxis bc codecom
pare smerge emerge vimdiff nvimdiff

Viewing (1/1): 'hello.xml'
Launch 'vimdiff' [Y/n]? y
2 files to edit
```

Merge and trigger conflict
git merge GitWork

You will get:

CONFLICT (content): Merge conflict in hello.xml
Automatic merge failed; fix conflicts and then commit the result.

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git merge GitWork
Auto-merging hello.xml
CONFLICT (add/add): Merge conflict in hello.xml
Automatic merge failed; fix conflicts and then commit the result.
```

Resolve the conflict

Open hello.xml in your editor (Notepad, Notepad++, or P4Merge if configured).
You'll see something like:

<<<<<<< HEAD
<message>Hello from master branch</message>
=======
<message>Hello from GitWork branch</message>
>>>>>>> GitWork

Manually edit it to the final version you want, e.g.:

<message>Hello from both branches</message>

Save and close.

Mark as resolved and commit
git add hello.xml
git commit -m "Merge GitWork into master and resolve conflict"

Ignore backup files

If a merge tool created backup files (like hello.xml.orig), create a .gitignore:

echo "*.orig" >> .gitignore
git add .gitignore
git commit -m "Ignore merge backup files"

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main|MERGING)
$ git add hello.xml
git commit -m "Merge GitWork into master and resolve conflict"
[main 5bfac66] Merge GitWork into master and resolve conflict

LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ echo "*.orig" >> .gitignore
git add .gitignore
git commit -m "Ignore merge backup files"
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
[main 35a75f5] Ignore merge backup files
 1 file changed, 1 insertion(+)
```

Delete merged branch
git branch -d GitWork

View final merge history
git log --oneline --graph –decorate

```
LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git branch -d GitWork
Deleted branch GitWork (was 0f8046e).

LENOVO@DESKTOP-292MR8U MINGW64 ~/GitDemo (main)
$ git log --oneline --graph --decorate
* 35a75f5 (HEAD -> main) Ignore merge backup files
*   5bfac66 Merge GitWork into master and resolve conflict
|\
| * 0f8046e Add hello.xml in GitWork branch
* | 4288c3f Add hello.xml in master branch with different content
|/
* fccbb92 (origin/main) Add new line in welcome.txt from GitNewBranch branch
* 46bf5cd Add .gitignore to ignore .log files and log folder
* e2e88a9 Hello all...Now I am working on GIT Hands-on week8
```