## **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”**

Notes\*:

|  |
| --- |
| Please follow the below steps for creating a free account in GitHub.  Do not use cognizant credentials to login to GitHub. |

Estimated time to complete this lab: **30 minutes.**

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”**

**Solution :**

1. Verify if master is in clean state

git checkout master

git status

2. Create a branch GitWork and add hello.xml

git branch GitWork

git checkout GitWork

echo "<message>Hello from GitWork branch</message>" > hello.xml

git add hello.xml

**3. Update content of hello.xml and observe status**

echo "<message>Updated content in GitWork branch</message>" > hello.xml

git status

**4. Commit the changes to the branch**

git commit -am "Added and updated hello.xml in GitWork branch"

**5. Switch to master**

git checkout master

**6. Add hello.xml to master with different content**

echo "<message>Hello from master branch</message>" > hello.xml

git add hello.xml

**7. Commit the changes to master**

git commit -m "Added hello.xml in master branch with different content"

**8. Observe the log**

git log --oneline --graph --decorate --all

**9. Check differences with Git diff tool**

git diff master GitWork

**10. Visual differences using P4Merge**

(Assumes P4Merge is installed & configured)

git difftool master GitWork

**11. Merge branch into master**

git merge GitWork

**12. Observe git markup (conflict markers)**

cat hello.xml

**13. Use 3-way merge tool to resolve conflict**

git mergetool

Select the correct content in P4Merge (or your configured merge tool), save & close.

**14. Commit changes after conflict resolution**

git commit -am "Merged GitWork into master and resolved conflict"

**15. Observe git status & add backup file to .gitignore**

git status

echo "\*.orig" >> .gitignore

**16. Commit .gitignore changes**

git add .gitignore

git commit -m "Added .gitignore to exclude backup files"

**17. List all branches**

git branch

**18. Delete merged branch**

git branch -d GitWork

**19. Observe the log**

git log --oneline --graph --decorate