**Step 1: Environment Preparation**

**Details:**

* Git is fully set up and configured on the machine
* P4Merge is installed for visual conflict resolution
* Local repository is linked to the GitLab remote

**Step 2: Confirm Master Branch is Clean**

git checkout master

git status

* Expected message: *“working tree clean”*

**Step 3: Create Feature Branch and Add XML File**

git branch GitWork

git checkout GitWork

echo "<message>Branch says hello</message>" > hello.xml

git add hello.xml

**Step 4: Commit Changes to Branch**

git commit -m "Created hello.xml in GitWork branch"

**Step 5: Switch to Master and Make Different Update**

git checkout master

echo "<message>Master says hello</message>" > hello.xml

git add hello.xml

git commit -m "Created hello.xml in master with alternate text"

**Step 6: Review Commit History**

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

**Step 7: Compare Master and Branch**

**Using CLI:**

git diff master GitWork

**Using P4Merge:**

git mergetool

**Step 8: Merge and Cause Conflict**

git merge GitWork

* Git will indicate a conflict in hello.xml

**Step 9: Resolve Conflict via 3-Way Merge**

git mergetool

* In P4Merge, decide whether to keep changes from master, branch, or merge both

**Step 10: Commit After Resolving**

git add hello.xml

git commit -m "Merge conflict resolved in hello.xml"

**Step 11: Add Backup Files to .gitignore**

echo "\*.orig" >> .gitignore

git add .gitignore

git commit -m "Added .orig files to .gitignore"

**Step 12: Remove the Feature Branch**

git branch -d GitWork

**Step 13: Final Log Check**

git log --oneline --graph --decorate