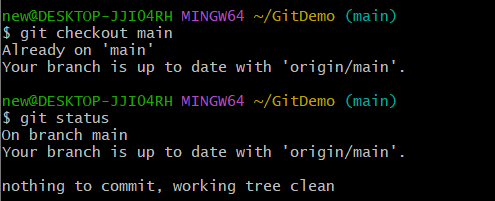
**GIT HANDSON-4**

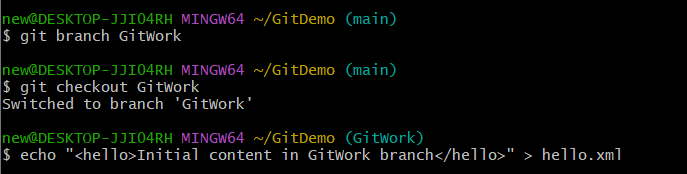
**Gitlab:** [**https://gitlab.com/syamprasad-111-group/gitdemo**](https://gitlab.com/syamprasad-111-group/gitdemo)

When a merge conflict occurs, Git pauses the merge and marks the conflicting files. You resolve it by opening the conflicted files (or using a merge tool like P4Merge), deciding which changes to keep or how to combine them, then saving the resolved files. After that, you stage the resolved files with git add and complete the merge by committing with git commit.

**1. Verify if master is in clean state**



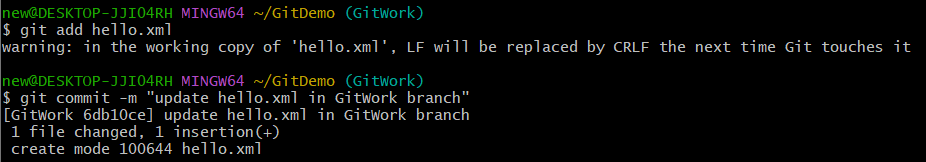
**2. Create a branch “GitWork” and add a file hello.xml**



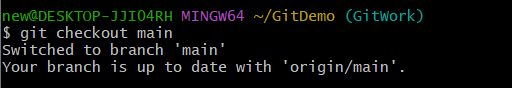
**3. Update the content of hello.xml and observe status**



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



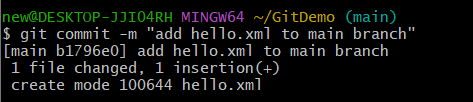
**5. Switch to master**



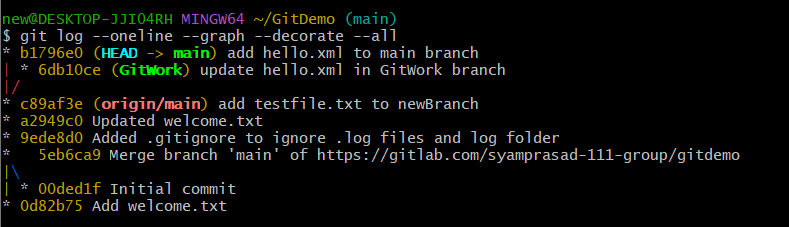
**6. Add a file hello.xml to master with different content**



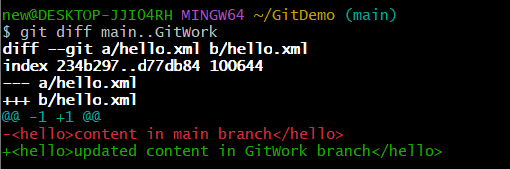
**7. Commit the changes to master**



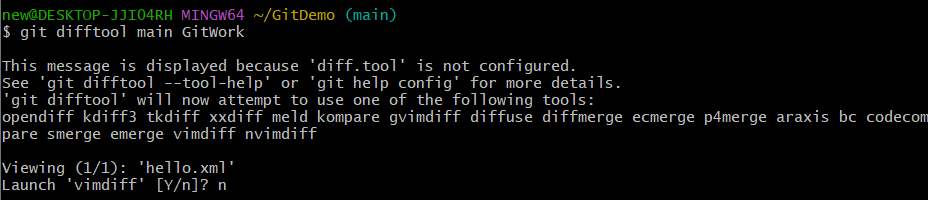
**8. Observe the log**



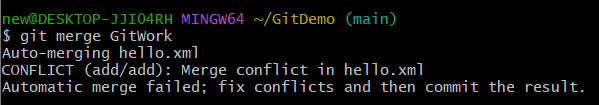
**9. Check the differences with Git diff tool**



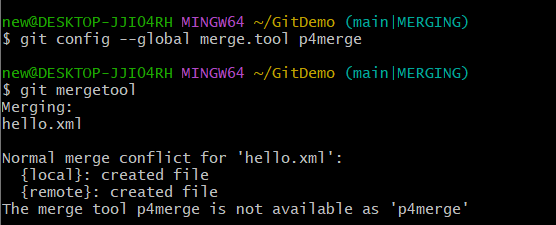
**10. Use P4Merge tool to visualize differences**



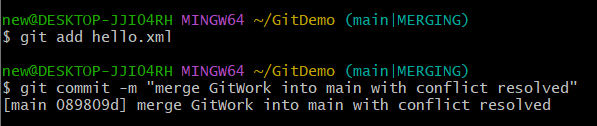
**11. Merge the branch to master**



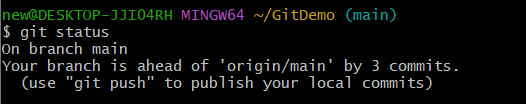
**13. Use 3-way merge tool (P4Merge) to resolve conflicts**

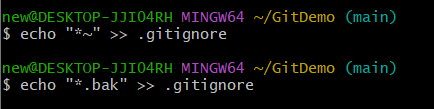


**14. Commit the changes once conflicts are resolved**

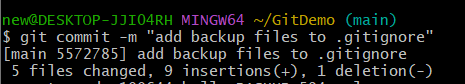


**15. Observe git status and add backup files to .gitignore**

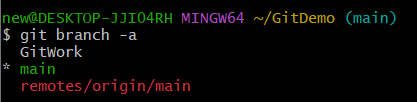
****



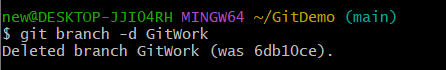
**16. Commit the changes to .gitignore**



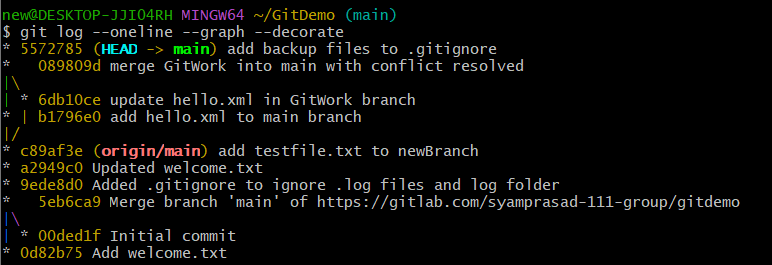
**17. List all available branches**

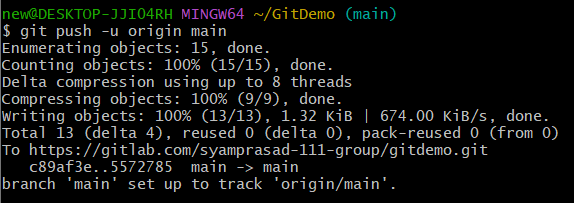


**18. Delete the branch merged to master**

****

**19. Observe the log**

**20. push**

****

**Final GitDemo repository:**

