GIT Workflow for COTS

Last updated on 5/6/2020 by Sungwook Kwon

Version: 1.3

# Contents

[Contents 2](#_Toc40963698)

[Introduction 4](#_Toc40963699)

[Architecture 4](#_Toc40963700)

[Git Repositories 4](#_Toc40963701)

[Local Repository 4](#_Toc40963702)

[user (Remote – OneDrive - State of Ohio) 4](#_Toc40963703)

[origin (Remote – https://odjfs.visualstudio.com) 4](#_Toc40963704)

[Deployment Manager (Local - Optional) 4](#_Toc40963705)

[Folder Structure 5](#_Toc40963706)

[GUI Client - Installation 5](#_Toc40963707)

[GIT Documentation – Pro Git 7](#_Toc40963708)

[Operation 7](#_Toc40963709)

[Set up user 7](#_Toc40963710)

[Setup Identity 7](#_Toc40963711)

[Set up remote repository 7](#_Toc40963712)

[Clone a remote repository to a local location 7](#_Toc40963713)

[Basic Operation 10](#_Toc40963714)

[.gitignore 10](#_Toc40963715)

[Fetch remote repository 10](#_Toc40963716)

[Adding files to index 11](#_Toc40963717)

[Renaming and Moving Files 12](#_Toc40963718)

[History 13](#_Toc40963719)

[Alias 13](#_Toc40963720)

[Create a local repository 14](#_Toc40963721)

[Create a remote repository 14](#_Toc40963722)

[Link the remote repository from the local repository 15](#_Toc40963723)

[Relating branches to the issues 15](#_Toc40963724)

[Add the files in local directory 15](#_Toc40963725)

[Push the changes to the remote repository 16](#_Toc40963726)

[Create a branch 17](#_Toc40963727)

[Commit to the new branch 17](#_Toc40963728)

[Delete a remote branch 18](#_Toc40963729)

# Introduction

This is the workflow for COTS team to maintain the version of FileNet/Kofax Projects using GIT.

# Architecture

## Git Repositories



### Local Repository

This is the repository that each user has in their local workstation. Once a user is participated in a project, the user will clone a whole project from the remote, COTS. Users will maintain their own version locally until the change is completely tested and ready for production. It is suggested to use an umbrella folder, “git” for all projects that are maintained by git under the folder.

### user (Remote – OneDrive - State of Ohio)

This is an optional repository that each user can have to make a backup of their local repository. Through this repository, users store any changes that they are working so that they will not be tied to a single workstation. If a change is small and it can be made quickly, users could use origin repository directly.

### origin (Remote – https://odjfs.visualstudio.com)

“origin” is the main repository for COTS team that is residing on https://odjfs.visualstudio.com. This is the repository that team members copy projects from and submit the changes that have been made, and completely tested. The source code and artifacts in this repository should be error-free and ready for deployment to production environment.

### Deployment Manager (Local - Optional)

Deployment Manager takes care of the deployment of all the changes that have been pushed to COTS repository. This repository should be on deployment server so that the change can easily be deployed to the designated environment. The manager pulling the most updated version from COTS repository and synchronize the changes with Dimension and perform the deployment.

## Folder Structure

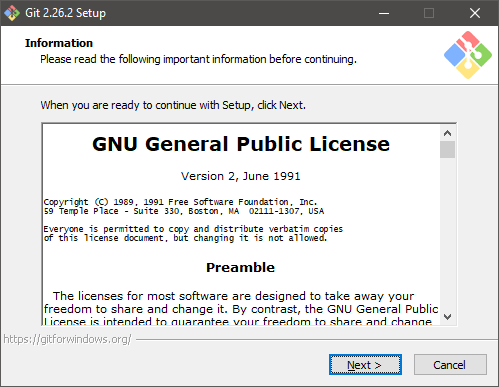
Each project will have its own root folder under GIT folder and the structure should be remained the same consistently with other project. New folders can be added as needed.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **COTS GIT Folder Structure**   |  |  |  |  | | --- | --- | --- | --- | | Git |  | | | | Proj-A | .git | Local Git Repository | | Documents | Folder for documents. | | Folders | Folders can be added as needed. | | Files | .gitignore  README.md | |

# GUI Client - Installation

The current version as of 5/6/2020 is 2.26.2. Get the code from the following site:  
<http://git-scm.com/downloads>

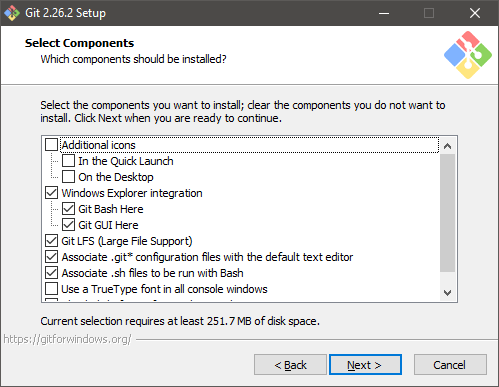
Download 61-bit Git for Windows Setup and follow the step for installation. Run the installer as Administrator.



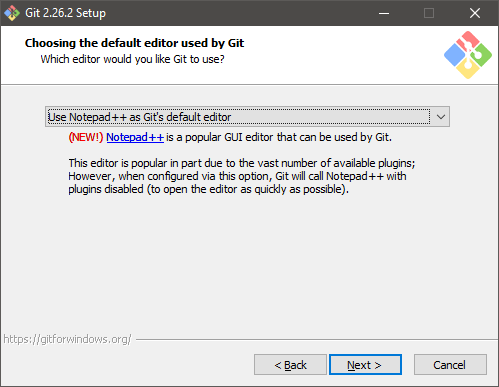
Accept the default installation location:

C:\Program Files\Git

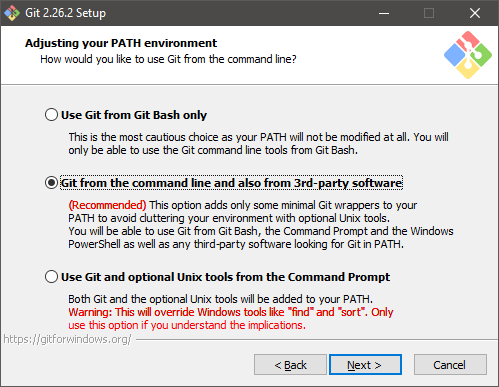
Select options. You can accept defaults for most cases.



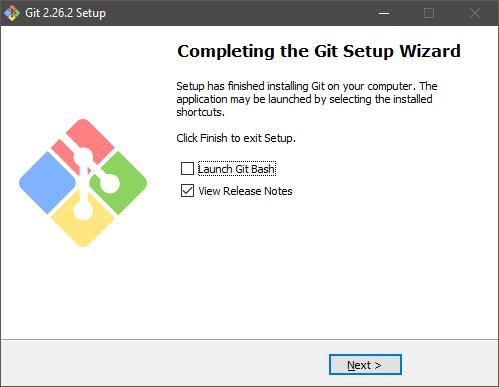
Change the default editor to your preferred editor.



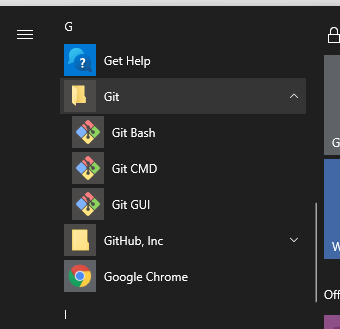
Accept recommendation for PATH.



Accept default selection for the rest of options.



Make sure new entries for GIT have been created in start menu.



# GIT Documentation – Pro Git

<https://git-scm.com/book/en/v2>

# Operation

## Set up user

This is one-time task that needs to be done to be able to use GIT. This will be recorded with every commit that you make.

|  |  |
| --- | --- |
| Setup Identity | |
|  | Open Git CMD from start menu.  $ git config --global user.name "Sungwook Kwon"  $ git config --global user.email [sungwook.kwon@jfs.ohio.gov](mailto:sungwook.kwon@jfs.ohio.gov)  $ git config --global --list  user.name=SungWook Kwon  user.email=sungwook.kwon@jfs.ohio.gov  … |
| Ref. | <https://git-scm.com/docs/git-config> |

## Set up remote repository

|  |  |
| --- | --- |
| Clone a remote repository to a local location | |
|  | Login Visual Studio repository with your state ID.  <http://odjfs.visualstudio.com>  All FileNet projects are found under “FileNet” project. Select “FileNet”.    Select the repository that needs to be copied (cloned) to the local copy (clone) of it.    To be able to clone the repository, you need a url for the repository.  Click “Clone” button to see and copy the url.      Open GIT GUI and select “Clone Existing Repository”.    Paste the copied URL to “Source Location” and provide a new folder for your own local repository    Verify local repository and link between local and remote repositories.   1. Files in the project have been copied to the local folder. 2. From Git GUI, Check the history of changes of the project   You should be able to see all the changes and commit message for each change. 3. Open Git Bash and run the following command to see the linked remote repository.  |  | | --- | | C:\Users\kwons>cd \git2\UIST C:\Git2\UIST>git remote -v origin https://odjfs.visualstudio.com/Filenet/\_git/UIST (fetch) origin https://odjfs.visualstudio.com/Filenet/\_git/UIST (push) C:\Git2\UIST> | |
| Ref. | <https://git-scm.com/docs/git-clone> |

## Basic Operation

|  |  |
| --- | --- |
| .gitignore | |
|  | .gitignore file is a plain text file where each line contains a pattern for files/directories to ignore. The following items are found useful in FileNet projects. This file is located under each project folder.   |  | | --- | | # General Executable  \*.tmp  \*.temp  \*.exe  \*.log  # Git  \*.orig  # FileNet Deployment Manager  FDM/Temp/\*  FDM/Samples/\*  FDM/Environments/\*\*/Assets/\*  # Microsoft Office – All temporary files  ~$\* | |
| Ref. | <https://git-scm.com/docs/gitignore> |

|  |  |
| --- | --- |
| Fetch remote repository | |
|  | Before making any changes to the local repository, it is strongly recommended to download objects and refs from origin repository to your local repository using fetch command.      In this case, there has been no changes made to remote repository since the user fetched the changes last time, therefore the user is good to go and make changes to the local repository.  While the user is making changes to a file, it is possible and allowed for other users to update the same file from their own local repository. In that case, user will need to resolve the conflict before the change can be committed and pushed back to the remote repository. Resolving conflict will be discussed in detail later. At this time, it would be much easier to avoid changing the same file simultaneously. |
| cmd | >git fetch origin master |
| Ref | <https://git-scm.com/docs/git-fetch> |

|  |  |
| --- | --- |
| Adding files to index | |
|  | Adding files to the local GIT repository to track the changes is two-step process: staging and commit.  All changes in the local repository directory or its subdirectories are tracked by GIT, but they will not automatically be included to repository unless they are explicitly staged to index. Once the files with changes are staged to the index, they will be included in every commit from that time on.  The following example shows two changes; one is made to the file that has been already added to the index and the other is made fresh new.  Once changes are made, user will need to decide which changes should be included coming  User is free to aBefore making any changes to the local repository, it is strongly recommended to synchronize your local repository with the remote repository by fetching the changes to local repository.      In this case, there has been no changes made to remote repository since the user fetched the changes last time, therefore the user is good to go and make changes to the local repository.  While the user is making changes to a file, it is possible and allowed for other users to update the same file from their own local repository. In that case, user will need to resolve the conflict before the change can be committed and pushed back to the remote repository. Resolving conflict will be discussed in detail later. At this time, it would be much easier to avoid changing the same file simultaneously. |
|  | <https://git-scm.com/docs/git-add> |

|  |  |
| --- | --- |
| Renaming and Moving Files | |
|  | KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ ls -al  total 8  drwxr-xr-x 1 KWONS 1049089 0 May 14 10:51 ./  drwxr-xr-x 1 KWONS 1049089 0 May 14 09:48 ../  -rw-r--r-- 1 KWONS 1049089 739 May 14 10:51 level1.txt  drwxr-xr-x 1 KWONS 1049089 0 May 14 10:52 level2/  KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ git mv level1.txt level1\_mod.txt    KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ git status  On branch master  Your branch is ahead of 'origin/master' by 6 commits.  (use "git push" to publish your local commits)  Changes to be committed:  (use "git restore --staged <file>..." to unstage)  renamed: level1.txt -> level1\_mod.txt  KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master) |
| Ref. | <https://git-scm.com/docs/git-mv> |

|  |  |
| --- | --- |
| History | |
|  | KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ git log --oneline --graph --decorate  \* ff61e6a (HEAD -> master) renames level3.txt  \* 631b9f2 adds all files  \* c7f096c removed the first two characters from level1.txt.  \* a7f8688 adds multi-level folders and files.  \* 45521b1 adds newfile.txt  \* a90f018 adds more ipsum text  \* 4e5eb88 (origin/master, origin/HEAD) initial commit  \* 4beb7f0 Merge pull request #6 from jasongtaylor/feature-readme  |\  | \* e73f914 Adding Purpose section to README  | \* 34f563b Adding README file  |/  \* 5c05047 Copying files from initializr project zip file and then creating simple.html as basis for super simple pages  KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $  KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ git log --since="3 days ago" --oneline  ff61e6a (HEAD -> master) renames level3.txt  631b9f2 adds all files  c7f096c removed the first two characters from level1.txt.  a7f8688 adds multi-level folders and files.  45521b1 adds newfile.txt  a90f018 adds more ipsum text  4e5eb88 (origin/master, origin/HEAD) initial commit  $ git log --since="yesterday" –oneline  $ git log --since="1/1/2020" --oneline |
| Ref. | <https://git-scm.com/docs/git-log> |

|  |  |
| --- | --- |
| Alias | |
|  | KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ git config --global alias.hist "log --all --graph --decorate --oneline"  KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ git hist  \* ff61e6a (HEAD -> master) renames level3.txt  \* 631b9f2 adds all files  \* c7f096c removed the first two characters from level1.txt.  \* a7f8688 adds multi-level folders and files.  \* 45521b1 adds newfile.txt  \* a90f018 adds more ipsum text  \* 4e5eb88 (origin/master, origin/HEAD) initial commit  \* 4beb7f0 Merge pull request #6 from jasongtaylor/feature-readme  |\  | \* e73f914 Adding Purpose section to README  | \* 34f563b Adding README file  |/  \* 5c05047 Copying files from initializr project zip file and then creating simple.html as basis for super simple pages  KWONS@JFSLT427170 MINGW64 ~/projects/starter-web/level1 (master)  $ vi ~/.gitconfig |
| Ref. | <https://git-scm.com/book/en/v2/Git-Basics-Git-Aliases> |

|  |  |
| --- | --- |
| Create a local repository | |
|  | Move to the root folder of Git projects  $ cd c:\git  KWONS@JFS398323 /c/git  $ mkdir OFC-ICPCDMS  KWONS@JFS398323 /c/git  $ cd OFC-ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS  $ git init  Initialized empty Git repository in c:/git/OFC-ICPCDMS/.git/  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ |

|  |  |
| --- | --- |
| Create a remote repository | |
|  | (remote repository can be different that P: drive)  $ cd p:\git  KWONS@JFS398323 ~/git  $ mkdir OFC-ICPCDMS  KWONS@JFS398323 ~/git  $ cd OFC-ICPCDMS  KWONS@JFS398323 ~/git/OFC-ICPCDMS  $ git init --bare  Initialized empty Git repository in p:/git/OFC-ICPCDMS/  KWONS@JFS398323 ~/git/OFC-ICPCDMS (BARE:master)  $ |

|  |  |
| --- | --- |
| Link the remote repository from the local repository | |
|  | KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git remote add KWONS file://p:/git/OFC-ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git remote add COTS file://ms-kofax-pa10/git/OFC-ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git remote -v  COTS file://ms-kofax-pa10/git/OFC-ICPCDMS (fetch)  COTS file://ms-kofax-pa10/git/OFC-ICPCDMS (push)  KWONS file://p:/git/OFC-ICPCDMS (fetch)  KWONS file://p:/git/OFC-ICPCDMS (push) |

|  |  |
| --- | --- |
| Relating branches to the issues | |
|  | * Git is not tied into any particular issue-tracking system. * The main should be “Remedy” to log the issues.   $ git checkout -b "INC000000742322"  Switched to a new branch 'INC000000742322'  KWONS@JFS398323 /c/git/OFC-ICPCDMS (INC000000742322)  $ git branch -a  \* INC000000742322  ` master  remotes/KWONS/ACL  remotes/KWONS/master  KWONS@JFS398323 /c/git/OFC-ICPCDMS (INC000000742322)  $ |

|  |  |
| --- | --- |
| Add the files in local directory | |
|  | $ git add \*  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git status  On branch master  Initial commit  Changes to be committed:  (use "git rm --cached <file>..." to unstage)  new file: DOCS/ICPC\_RetentionSchedule.pdf  new file: DOCS/OFC ICPCDMS\_ACL.xlsx  new file: DOCS/OFC ICPCDMS\_Configuration.xlsx  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git commit -m "Initial Commit for ICPCDMS"  [master (root-commit) d0fb245] Initial Commit for ICPCDMS  3 files changed, 0 insertions(+), 0 deletions(-)  create mode 100644 DOCS/ICPC\_RetentionSchedule.pdf  create mode 100644 DOCS/OFC ICPCDMS\_ACL.xlsx  create mode 100644 DOCS/OFC ICPCDMS\_Configuration.xlsx  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git status  On branch master  nothing to commit, working directory clean  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git log  commit d0fb2450d7961d295fdd158d1550ad8b9cbacf5b  Author: Sungwook Kwon <swooki@gmail.com>  Date: Wed Apr 30 11:40:36 2014 -0400  Initial Commit for ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ |

|  |  |
| --- | --- |
| Push the changes to the remote repository | |
|  | $ git log  commit d0fb2450d7961d295fdd158d1550ad8b9cbacf5b  Author: Sungwook Kwon <swooki@gmail.com>  Date: Wed Apr 30 11:40:36 2014 -0400  Initial Commit for ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git push KWONS master  Counting objects: 6, done.  Delta compression using up to 4 threads.  Compressing objects: 100% (5/5), done.  Writing objects: 100% (6/6), 284.86 KiB | 0 bytes/s, done.  Total 6 (delta 0), reused 0 (delta 0)  To file://p:/git/OFC-ICPCDMS  \* [new branch] master -> master  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master) |

|  |  |
| --- | --- |
| Create a branch | |
|  | $ cd c:/git/OFC-ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git branch ACL  KWONS@JFS398323 /c/git/OFC-ICPCDMS (master)  $ git checkout ACL  M DOCS/OFC ICPCDMS\_ACL.xlsx  Switched to branch 'ACL'  KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ git branch -a  \* ACL  master  remotes/KWONS/master |

|  |  |
| --- | --- |
| Commit to the new branch Make a change to one of the file and commit the change to the branch. | |
|  | KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ git status  On branch ACL  Changes not staged for commit:  (use "git add <file>..." to update what will be committed)  (use "git checkout -- <file>..." to discard changes in working directory)  modified: DOCS/OFC ICPCDMS\_ACL.xlsx  no changes added to commit (use "git add" and/or "git commit -a")  KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ git add \*  KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ git status  On branch ACL  Changes to be committed:  (use "git reset HEAD <file>..." to unstage)  modified: DOCS/OFC ICPCDMS\_ACL.xlsx  KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ git log  commit d0fb2450d7961d295fdd158d1550ad8b9cbacf5b  Author: Sungwook Kwon <swooki@gmail.com>  Date: Wed Apr 30 11:40:36 2014 -0400  Initial Commit for ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ git commit -m "ACL file has been cleaned"  [ACL a56fbde] ACL file has been cleaned  1 file changed, 0 insertions(+), 0 deletions(-)  KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ git log  commit a56fbde4aad276f8d5c2289060b5bcd9344e0e5e  Author: Sungwook Kwon <swooki@gmail.com>  Date: Wed Apr 30 13:37:16 2014 -0400  ACL file has been cleaned  commit d0fb2450d7961d295fdd158d1550ad8b9cbacf5b  Author: Sungwook Kwon <swooki@gmail.com>  Date: Wed Apr 30 11:40:36 2014 -0400  Initial Commit for ICPCDMS  KWONS@JFS398323 /c/git/OFC-ICPCDMS (ACL)  $ |

|  |  |
| --- | --- |
| Delete a remote branch | |
|  | $ git push COTS --delete <branch name> |

|  |  |
| --- | --- |
| Tagging | |
|  | **Simple Tagging Example / Lightweight Tags**  Simple example with *lightweight* tags and some basic tag commands.  **Command Listing**  pwd  cd projects/starter-web/  git status  git log --oneline --decorate --graph –all (hist)  git tag myTag  git log --oneline --decorate --graph --all  git tag --list  git show myTag  git tag --list  git tag --delete myTag  git tag --list  git log --oneline --decorate --graph --all  **Annotated Tags (“-a”)**  Annotated tags are tags with more information, like comments, associated with them.  **Command Listing**  pwd  git status  clear  git tag -a v-1.0  git tag --list  git log --oneline --decorate --graph --all  git show v-1.0  **Comparing Tags**  Tags mark important milestones in a project, so they are great way to compare what has happened between those milestones.  **Command Listing**  git tag -a v-1.1  mate simple.html  git commit -am "Updating for tag 1.1"  git commit --amend  git tag v-1.2 -m "Release 1.2"  git tag --list  git log --oneline --decorate --graph --all  git diff v-1.0 v-1.2  git difftool v-1.0 v-1.2  **Tagging a Specific Comment**  If you forget to tag a commit in the past, you can simply pass in the commit id while creating the tag.  **Command Listing**  pwd  git status  git log --oneline --decorate --graph --all  git tag -a v-0.9-beta 96ef75b  git log --oneline --decorate --graph --all  git tag -a v-0.8-alpha ab0d621  git log --oneline --decorate --graph --all  **Updating a Tag**  Sometimes mistakes happen and you'll need to update a tag.  **Command Listing**  pwd  git status  git log --oneline --decorate --graph --all  git tag -a v-0.8-alpha -f bd35d46  git log --oneline --decorate --graph --all  **Remote Tagging (GitHub)**  Working with remote tags on GitHub.  **Command Listing**  pwd  git status  git tag --list  git log --oneline --decorate --graph --all  git push origin v-0.9-beta  git push origin v-1.1  clear  git push origin master --tags  git push origin :v-0.8-alpha (remove tag from remote repository-github) |

|  |  |
| --- | --- |
| Reset and Reflog | |
|  | $git reset HEAD^1  $ git reflog  1e68130 (HEAD -> master, tag: v1.0.2, origin/master, origin/HEAD) HEAD@{0}: pull origin master: Fast-forward  9cffe78 (tag: v1.0.1) HEAD@{1}: reset: moving to HEAD^1  1e68130 (HEAD -> master, tag: v1.0.2, origin/master, origin/HEAD) HEAD@{2}: commit (amend): Release 1.0.2  c1d7ca2 HEAD@{3}: commit: v1.0.1  9cffe78 (tag: v1.0.1) HEAD@{4}: commit: updates index.html  ecb1e10 (tag: v1.0.0) HEAD@{5}: merge new-changes: Fast-forward  1f282bd HEAD@{6}: checkout: moving from new-changes to master  ecb1e10 (tag: v1.0.0) HEAD@{7}: commit: commit into a new branch  1f282bd HEAD@{8}: reset: moving to HEAD  1f282bd HEAD@{9}: reset: moving to HEAD  1f282bd HEAD@{10}: reset: moving to HEAD  Use reset to a commit in reflog to undo what you have performed to GIT |

|  |  |
| --- | --- |
| **How to resolve a binary file conflict with Git** | |
|  | Auto-merging somefile.dll  CONFLICT (content): Merge conflict in somefile.dll  Automatic merge failed; fix conflicts and then commit the result.  In this scenario, somefile.dll is a binary file that has been modified in both the current branch, and the branch you are attempting to merge in to the current branch. Since the file cannot be textually merged, you need to make a decision: do you keep the version of the file in your current branch, or the version in the other branch.  ## Resolve using mine  The file in your working copy is still the copy from your current branch – in other words, it was not modified by the merge attempt. To resolve the conflict and keep this file:  git add somefile.dll  git commit –m “My commit message for the merge”  ## Resolve using theirs  If you prefer to resolve the conflict using their copy, you need to get the version of the file from the branch you were trying to merge in:  git checkout otherbranch somefile.dll  Now that you have the correct version of the file in your working copy, you can mark it as resolved (by adding it), and commit:  git add somefile.dll  git commit –m “My commit message for the merge”  Note that in place of otherbranch, you can use any name (treeish) that refers to a branch: a local branch name (otherbranch), a remote branch name (origin/master), a specific commit SHA (980e3cc), etc. For example, if you were merging in from your remote when you received the conflict, and you wanted to resolve using the remote version, you would retrieve that copy of the file using:  git checkout origin/master somefile.dll |