

git Branching model

# GIT Branching Model

[I. Overview 2](#_Toc369593962)

[II. Clone repository 3](#_Toc369593963)

[1. Command 3](#_Toc369593964)

[*2.* GUI *(Step by step)* 3](#_Toc369593965)

[III. Create master and develop branch 4](#_Toc369593966)

[1. Command 4](#_Toc369593967)

[*2.* GUI *(Step by step)* 4](#_Toc369593968)

[IV. Hotfix branch 6](#_Toc369593969)

[1. Command 6](#_Toc369593970)

[2. GUI 7](#_Toc369593971)

[a) Config Git Flow 7](#_Toc369593972)

[b) Create hotfix branch using “Git Flow” 7](#_Toc369593973)

[*c)* Finish hotfix (hotfix-1.2.10) branch *(Step by step)* 8](#_Toc369593974)

[3. Delete branch *(Step by step)* 10](#_Toc369593975)

[V. Release branch 12](#_Toc369593976)

[VI. Feature branch 12](#_Toc369593977)

[VII. Tagging 13](#_Toc369593978)

[1. Command 13](#_Toc369593979)

[*2.* GUI *(Step by step)* 13](#_Toc369593980)

[VIII. Setting up GIT repository on your local 15](#_Toc369593981)

[1. Create server repository. 15](#_Toc369593982)

[2. Checkout the code from repository 16](#_Toc369593983)

## Overview



**Note:** We use “Totoise Git” for demonstrating example GUI in this article

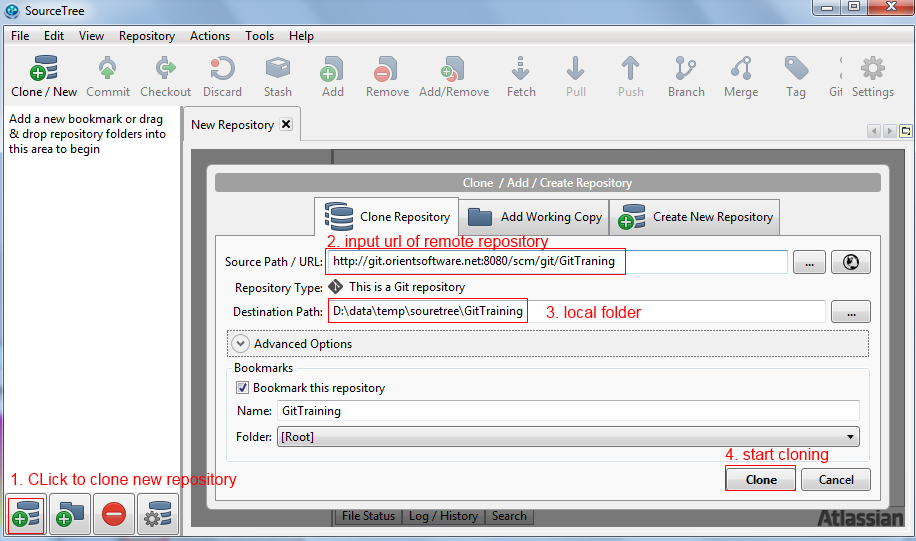
## Clone repository

### Command

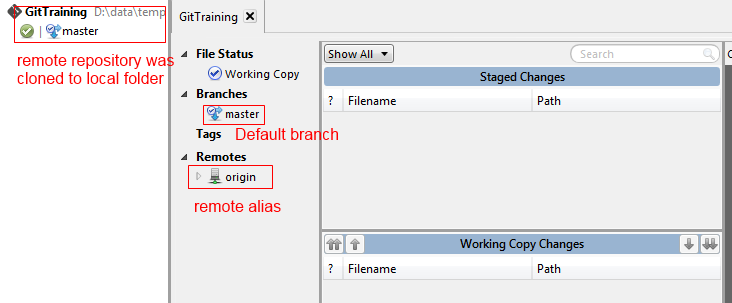
git clone http://git.orientsoftware.net:8080/scm/git/GitTraning

### GUI (Step by step)

* Run SourceTree program and follow instruction as in picture below:



* Progress was done.



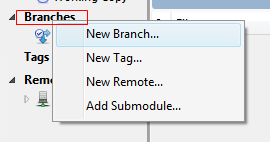
## Create master and develop branch

### Command

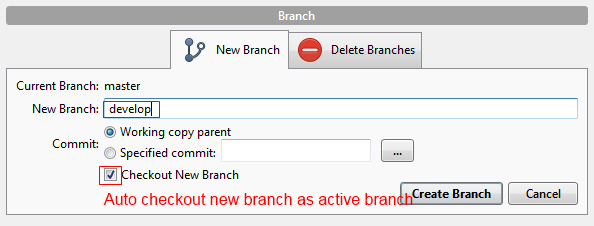
//Master branch was created by default. So we do not run the command here  
  
//create develop branch  
git branch develop  
git checkout develop  
git commit -m "Create develop branch"  
git push origin develop

### GUI (Step by step)

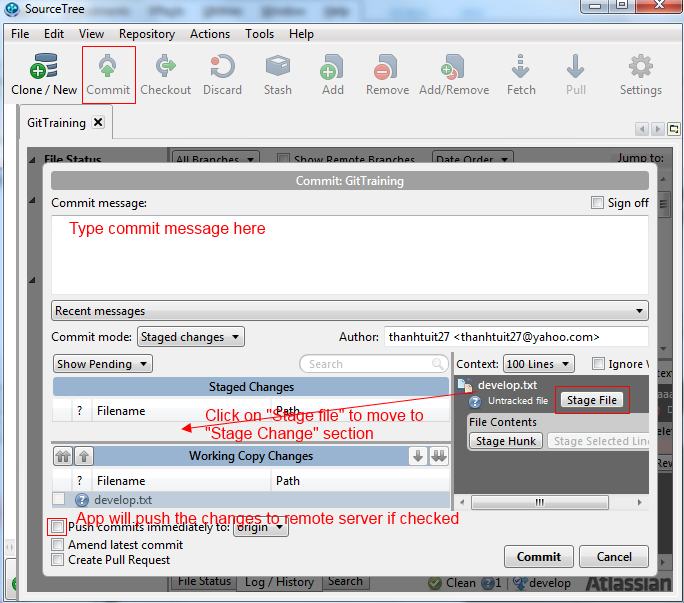
* From main editor of SoureTree, right click on the right panel and select “New Branch” from context menu.



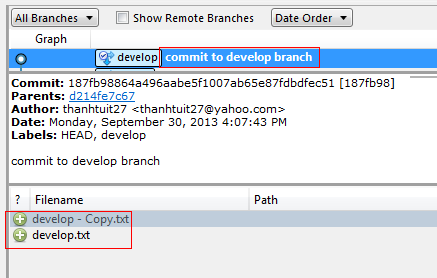
* Input “develop” into “New Branch” textbox and click on “Create Branch” to create new branch



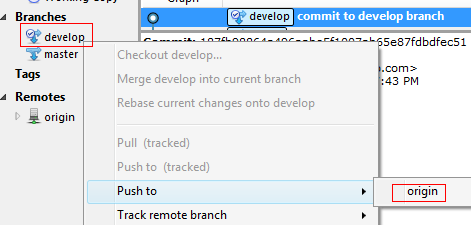
* Currently, develop branch was created on local only. So we can make change and commit to local branch before pushing to remote server by click on commit button as picture below:



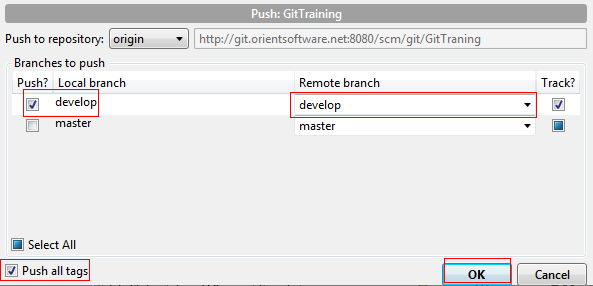
* New files were committed to local branch (develop branch)



* To push develop branch to remote server. Select “Push To” from context menu when click on develop branch in main window



* Check on appropriated branch that you want to push to remote server and click on ok button.



## Hotfix branch

* This is temporary branch for fixing issues on production.
* Hotfix branch was branched from master branch and merged back to master and develop branches
* Please have a look at “[Create master and develop branch](#_Create_master_and)” section for more information how to create branch (command line)
* We will focus on how to merge from hotfix branch to master branch and develop branch (command line)

### Command

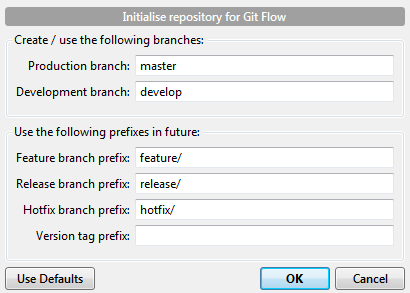
//merge to master branch  
git checkout master  
git merge --no-ff hotfix-1.2.1.0  
git tag -a 1.2.1.0  
  
//merge to develop branch  
git checkout develop  
git merge --no-ff hotfix-1.2.1.0  
  
//delete hotfix branch  
git branch -d hotfix-1.2.1.0

### GUI

* Name of hotfix branch should prefix by “hotfix-” and followed by version number. For example: hotfix-1.2.10

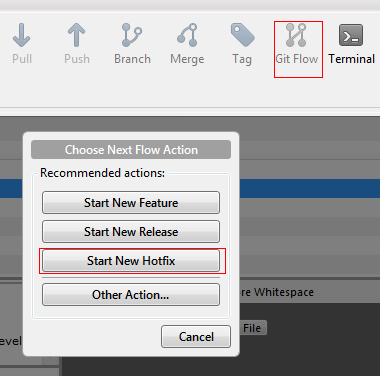
#### Config Git Flow

Click on “Git flow” menu. There is a configuration dialog as below. Leave everything as default and click ok:

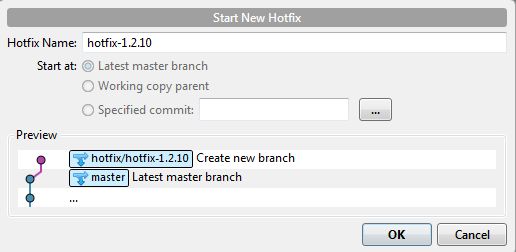


#### Create hotfix branch using “Git Flow”

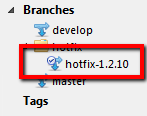
* Click on “Git Flow” menu and select “Start New Hotfix” button:



* Input name of the hotfix and click ok:



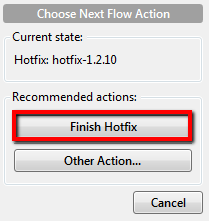
* The new hotfix branch (named hotfix/hotfix-1.2.10) was created on local:



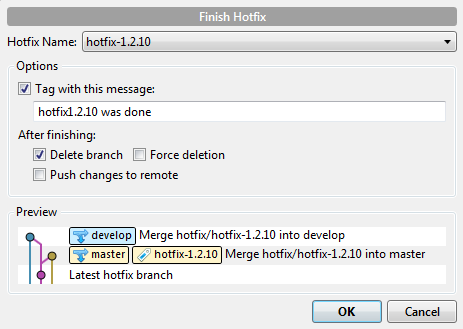
From now, we can work on newly hotfix-1.2.04 branch as any others branch.

#### Finish hotfix (hotfix-1.2.10) branch (Step by step)

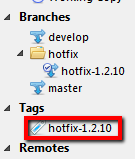
* Make sure the hotfix branch is active branch and click on “Git Flow” menu:



* Click on “Finish Hotfix” and input appropriated information as below:

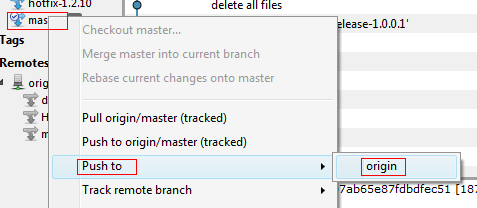


* At this time, your code in hotfix branch was merged to develop branch, master branch and new tag (the same name as hotfix name) was created too.

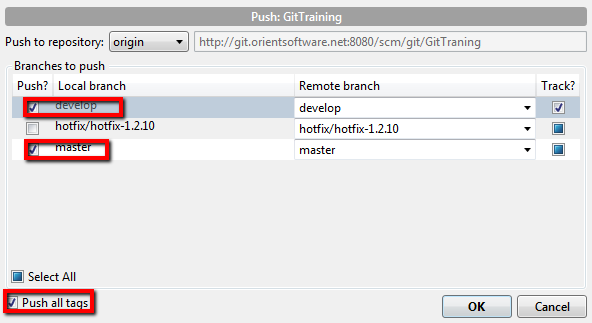


*Note: the hotfix branch was not removed as it is active branch.*

* The hotfix-1.2.10 branch was merged to master branch on local. We need to push the changed to remote server. So other members in team can see the changes. Right click on master branch and select origin as below:



* Check on master branch, develop branch, check “Push all tags” checkbox and click on to push the changes from local to remote branch on remote server.



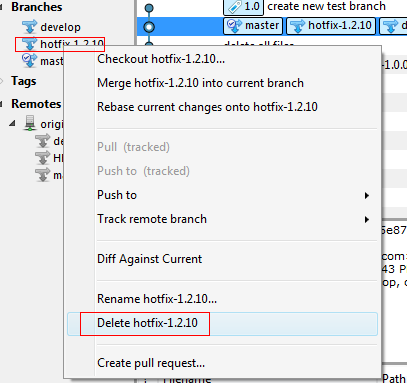
***Note: There is another way to finish a hotfix by:***

* *Checkout other branch (any branch except the hotfix is going to be finished).*
* *Go to “Git Flow”> Other Actions > Finish hotfix.*
* *Select name of hotfix from dropdown list and appropriated other options. Click OK.*

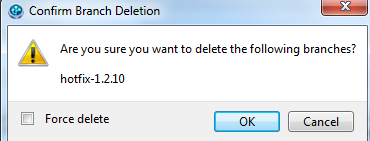
### Delete branch (Step by step)

After the issues were fixed and the code deployed to production (and merged back to master and develop branches too). We do not need the hotfix branch in the future. So we can delete it.

* Right click on hotfix-1.2.10 and select “delete hotfix-1.2.10” from context menu as picture:



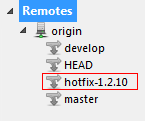
* Click ok on confirmation dialog



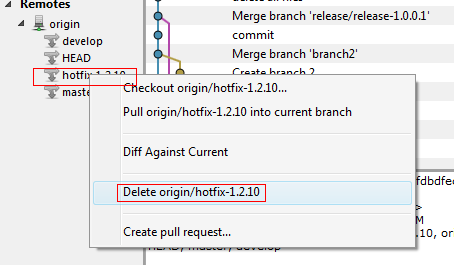
* Hotfix branch was deleted on your local now.



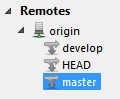
* If appropriated hotfix branch exists on remote server. We need to delete remote branch too.



* Right click on hotfix-1.2.10 in remotes section. And click on “delete hotfix-1.2.10” the same as delete branch on local.



* Remote branch was deleted on remote server

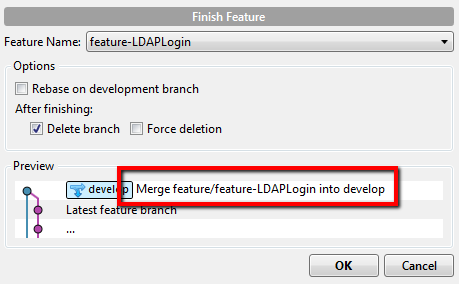


## Release branch

* Release branch was branched from develop branch and merge back to develop and master branches
* See [hotfix branch](#_Hotfix_branch) for more information for more information about how to create and merge release branch back to master branch and develop branches.
* Name of release branch should prefix by “release-” and followed by release number. For example: release-2.1.30

## Feature branch

* Feature branch was branched from develop branch and merged back to develop branch if any.
* We create new feature branch for each feature will be developed (for next release).
* See “[hotfix branch](#_Hotfix_branch)” for more information for more information about how to create and merge feature branch back develop branch.
* Name of feature branch should prefix by “feature-” and followed by the short descriptive name of feature. The short name should be meaningful as possible. This will help us easy to manage our repository.
* When finish a feature branch, the branch will be merged to develop branch only and there is not new tag was created as in hotfix branch and release branch.



## Tagging

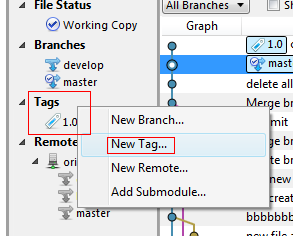
* Tag help us hold a snapshot of branch (usually master branch) at specifies of time.
* Tag was tagged on specified branch
* Tag is similar to branch except source code of tag can’t be modified.
* We should create new tag when deploy new code to production environment (such as: new release, hotfix …).

### Command

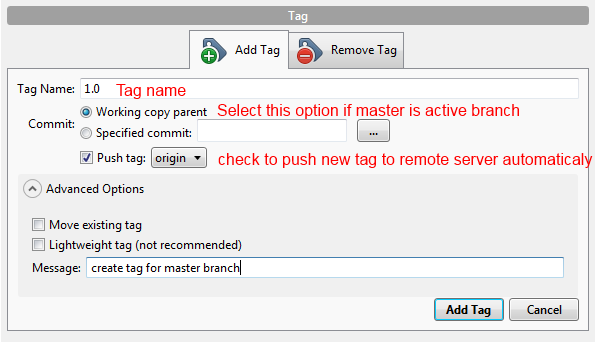
git tag -a v1.0 -m message

### GUI (Step by step)

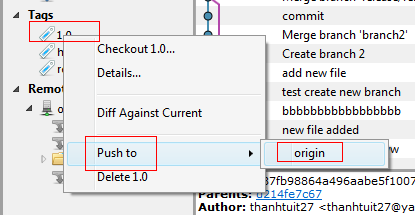
* Right click anywhere in the left panel and select “New Tag” as below:



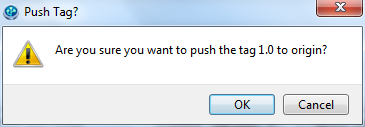
* Select appropriated options and click “Add Tag” button



* We need to push the change to remote server.



* Select ok on confirmation dialog:

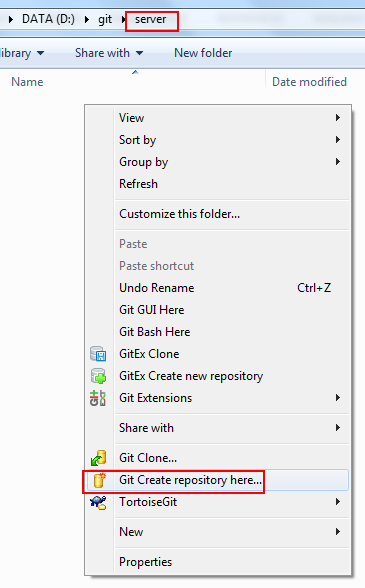


## Setting up GIT repository on your local

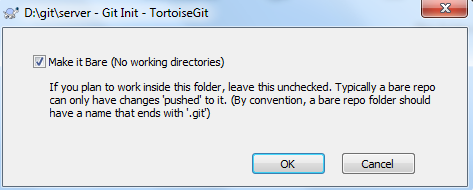
Assump that repository folder on server is at “d:\git\server” where we can checkout the code from.

### Create server repository.

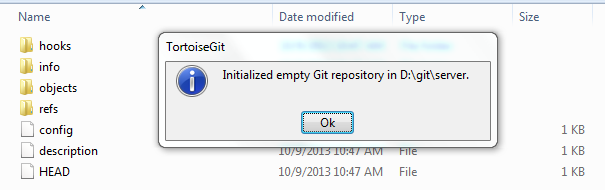
* Go to “d:\git\server”.



* Tick on the “make it bare” checkbox

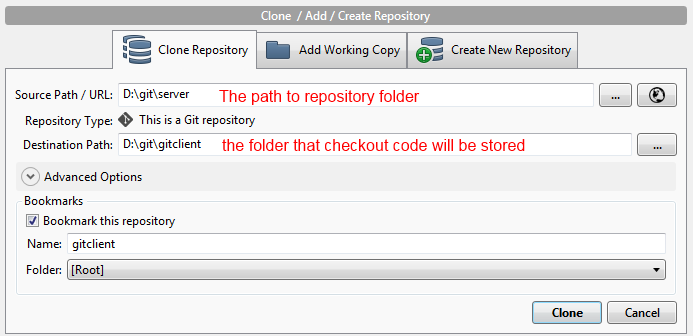


* Repository was created

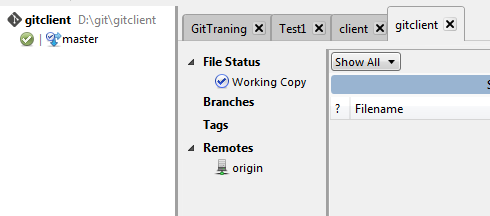


### Checkout the code from repository

* Open sourcetree tool and click on “Clone/New” botton at the left-top conor. Fill in appropriated information and click on “Clone” button



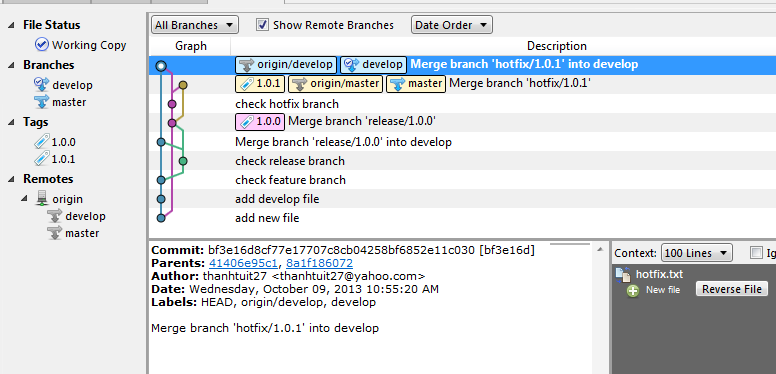
* The code was checked out



By default, there is not branch by default and we can not create any new branch.

Open folder of repository (checked out code) and make any change on this (create new file, …) and commit the code to local. Git will create master branch for us by default. Push master branch to remote.

* From now, we can continue our work as normal.



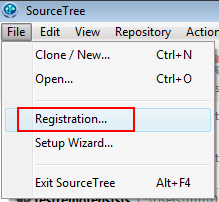
## SourceTree Registration

As informed from Atlassian (http://sourcetreeapp.com/faq), SourceTree is free but we need to register.

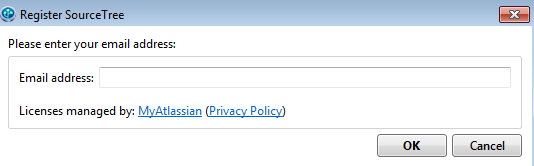
If you see the dialog as below. Follow the instructions to register an account.

*“To be updated”*

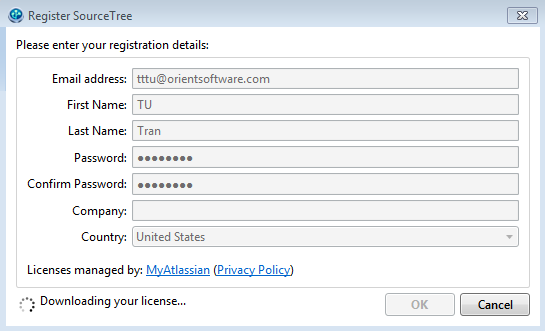
* From file menu, click on Registration menu item.



* Type your email into the textbox



* Fill in more information and click ok button



* We are done now

