Git 102

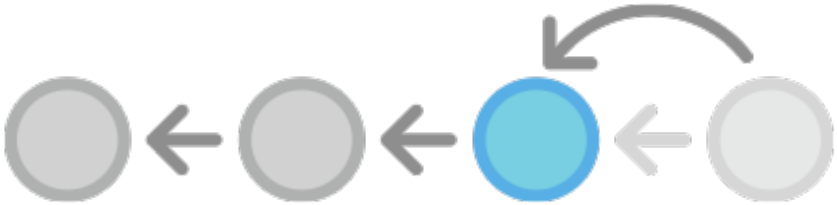
@somkiat

Topic

- Undo change
- Branch
- Rewrite git history

1. Undo Change

- git checkout
- git revert
- git reset
- git clean



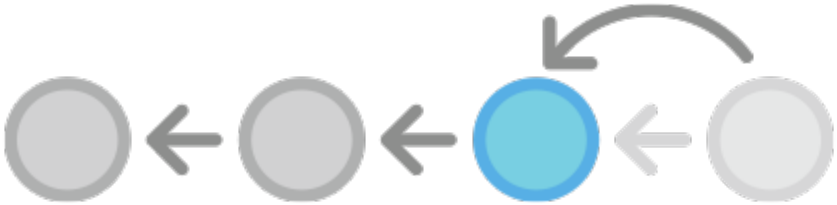
git checkout

- checkout files
- checkout commit
- checkout branch

>git checkout <commit>

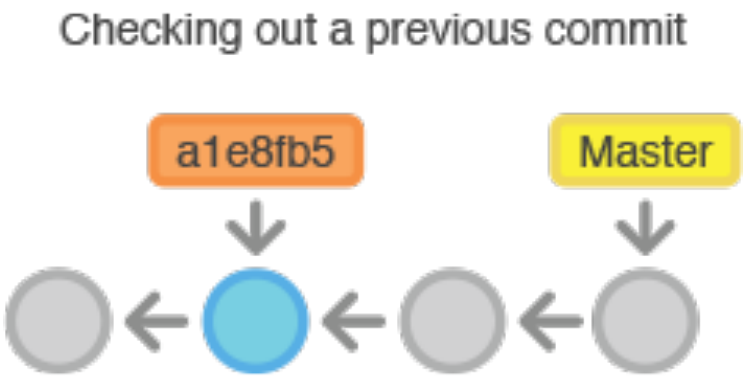
>git checkout <commit> <file>

>git checkout <branch name>



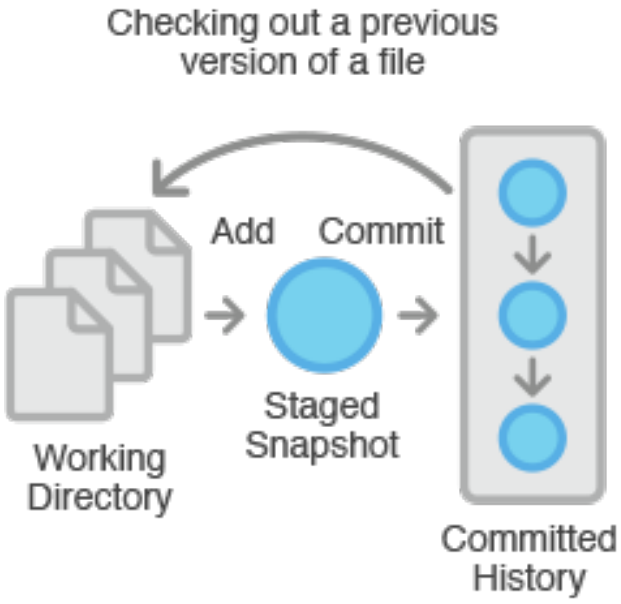
Demo :: git checkout

>git log --oneline
>git checkout <commit>
>git status
>git checkout master



Demo :: git checkout file

>git log --oneline
>git checkout <commit> <file>
>git status
>git checkout HEAD <file>



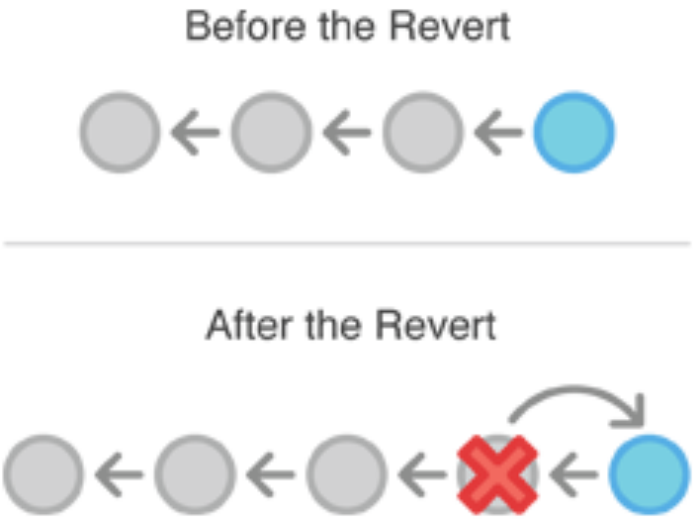
git revert

- Undo a committed and New commit
- Not remove committed in project history
- Safe way to undo change

>git revert <commit>

Demo :: git revert

>touch user.txt
>git commit -m "Add comment .."
>git revert HEAD



git reset

- Danger !!!
- Don't use in public !!!
- Remove committed
- Undo change in stage area + working dir

git reset command

>git reset
>git reset <file>
>git reset --hard
>git reset <commit>
>git reset --hard <commit>

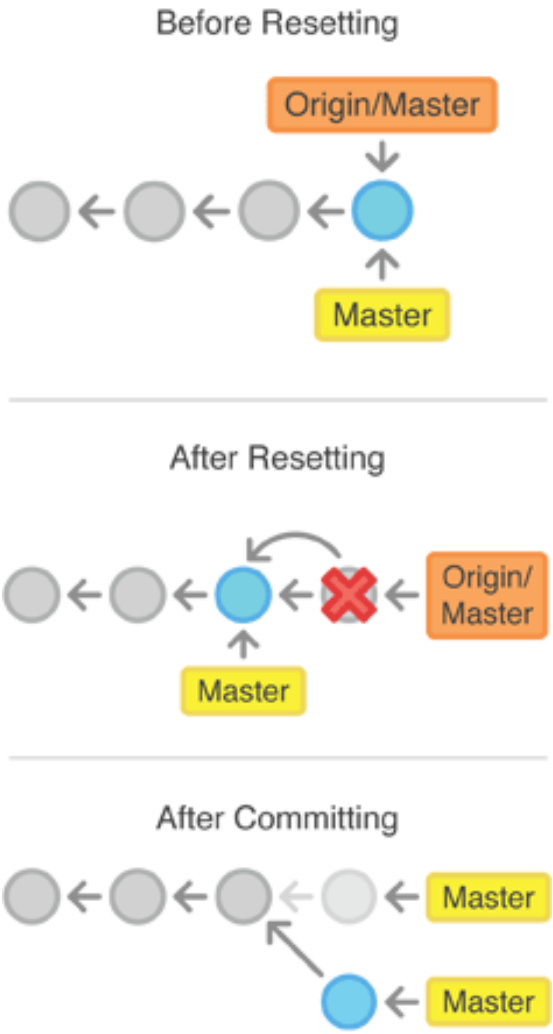
Demo :: git reset (Unstage file)

> edit user1.txt, user2.txt
>git add .
>git reset user2.txt #unstage
>git commit -m "Add user1.txt"
>git add user2.txt
>git commit -m "Add user2.txt"

Demo :: git reset (Remove local commit)

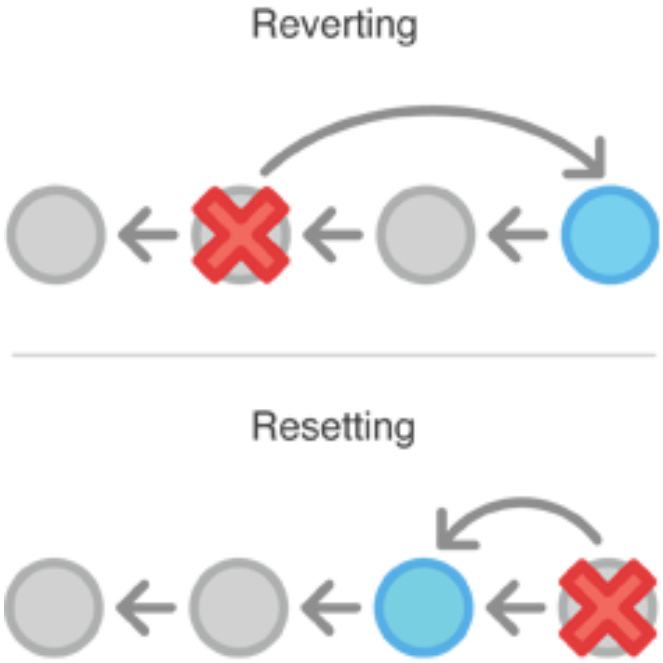
```
>touch foo.txt
>git add foo.txt
>git commit -m "Start feature 1"
>edit file foo.txt
>git commit -a -m "Continue feature 1"

>git reset --hard  HEAD~2  #back 2 commit
```

Don't reset public history



Revert vs Reset

- git revert => Public undo
- git reset => Local undo



git clean

- Permanent remove
- Remove untracked file
- Clean working directory after build

git clean command

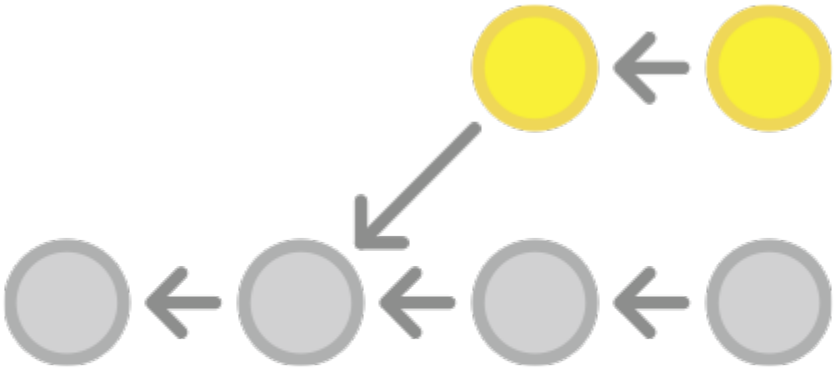
>git clean -n
>git clean -f #force delete file
>git clean -f <path>
>git clean -df #file and directory
>git clean xf

Demo :: git clean

```
>touch clean1.txt clean2.txt
>git add clean1.txt

#Undo changes in tracked files
>git reset --hard

#Remove untracked files
>git clean -df
```

2. Branch

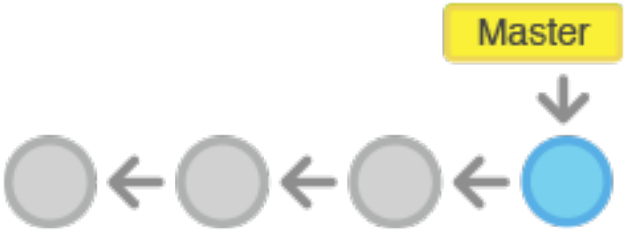
- git branch
- git checkout
- git merge



git branch

- Line of development

```
>git branch
>git branch <branch name>      #create
>git branch -d <branch name>  #delete
>git branch -m <branch name> #rename
```



git checkout

- Navigate between the branch
- Select your development line

>git checkout <branch name>

>git checkout -b <branch name>

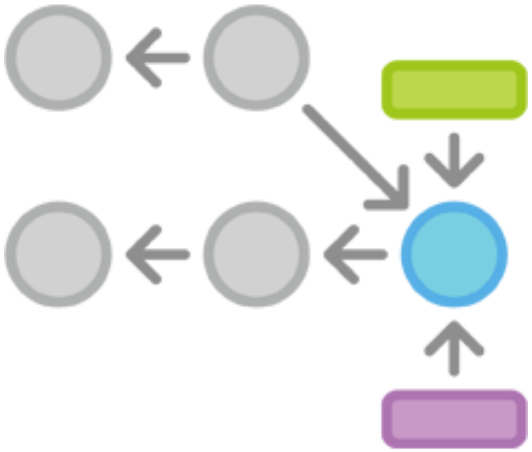
>git checkout -b <branch name> <existing branch>

git merge

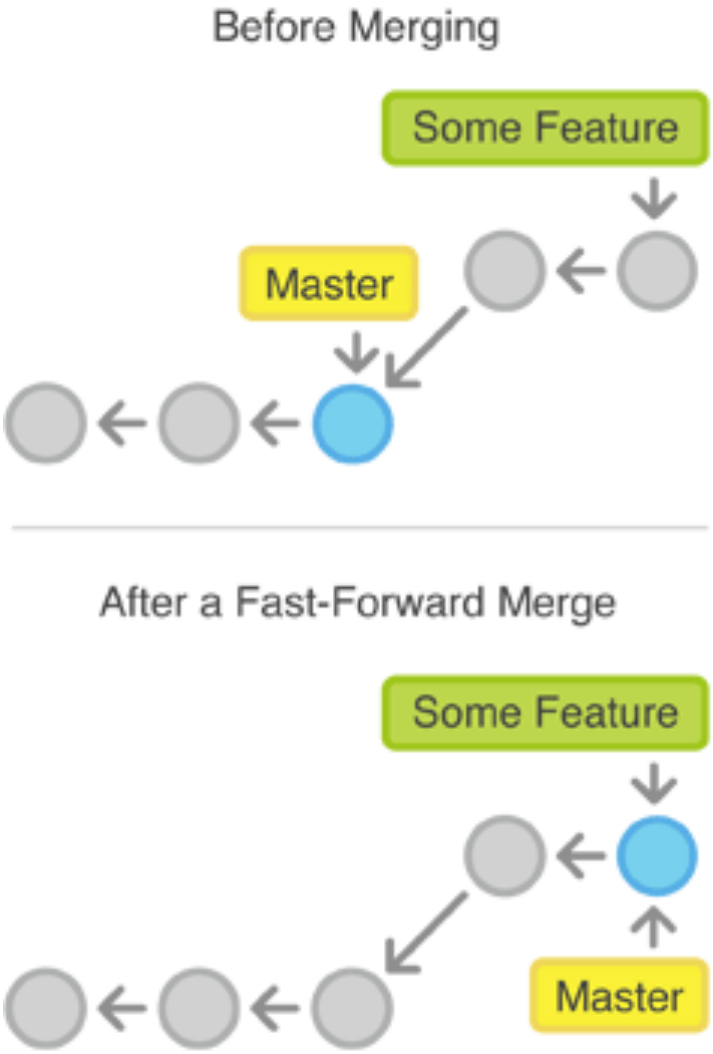
- Integrate changes from other branches

#Automatic merge
>git merge <branch name>

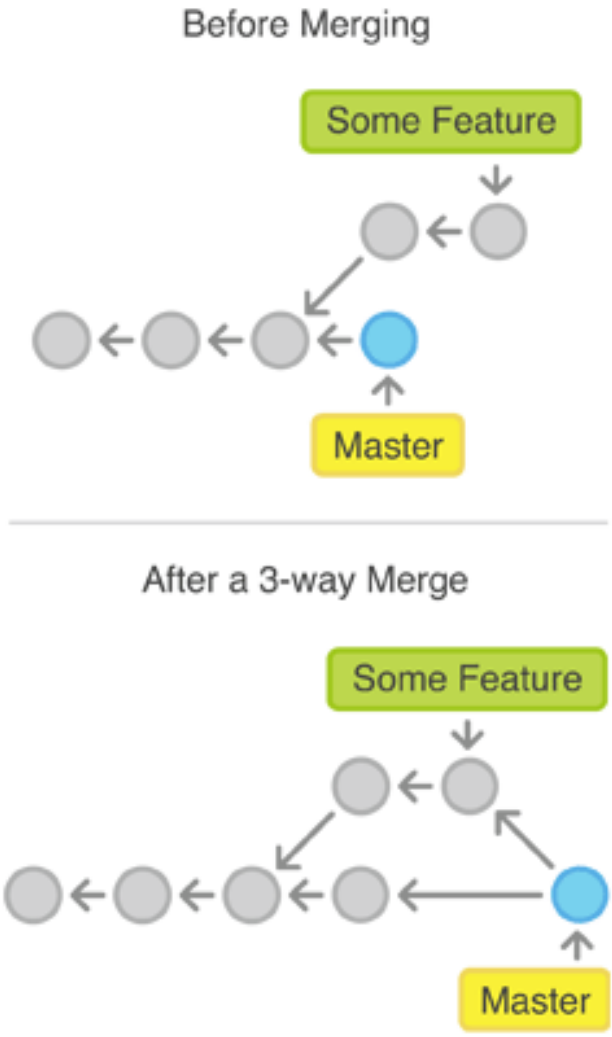
#Generate a merge commit
>git merge --no-ff <branch name>



Fast-forward merge



3-way merge


Before Merging
Some Feature
Master
After a 3-way Merge
Some Feature
Master

Demo :: Fast-forward merge

>git checkout -b branch1 master
>touch demo.txt
>git commit -m "Commit of branch1"

>git checkout master
>git merge branch1

>git branch -d branch1

Demo :: 3-way merge

>git checkout -b branch1 master
>touch demo.txt
>git add demo.txt
>git commit -m "Commit of branch1"

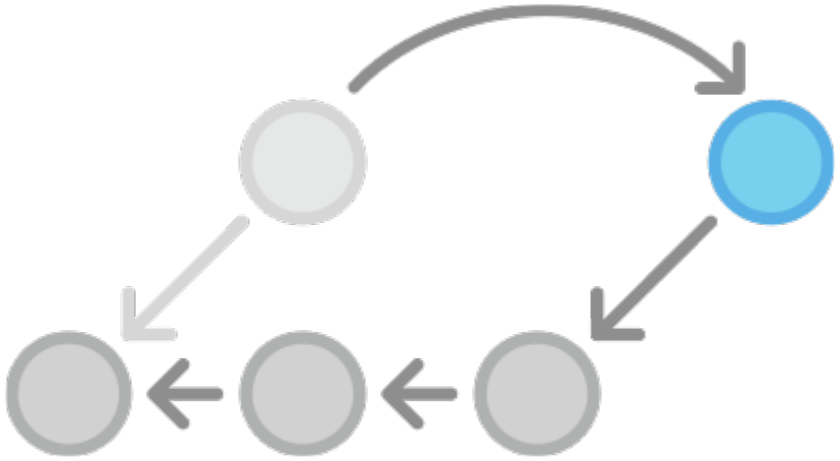
Demo :: 3-way merge

>git checkout master
>touch main.txt
>git add main.txt
>git commit -m "Add main.txt"

>git merge branch1
>git branch -d branch1

3. Rewrite Git History

- git commit -amend
- git rebase
- git rebase -i



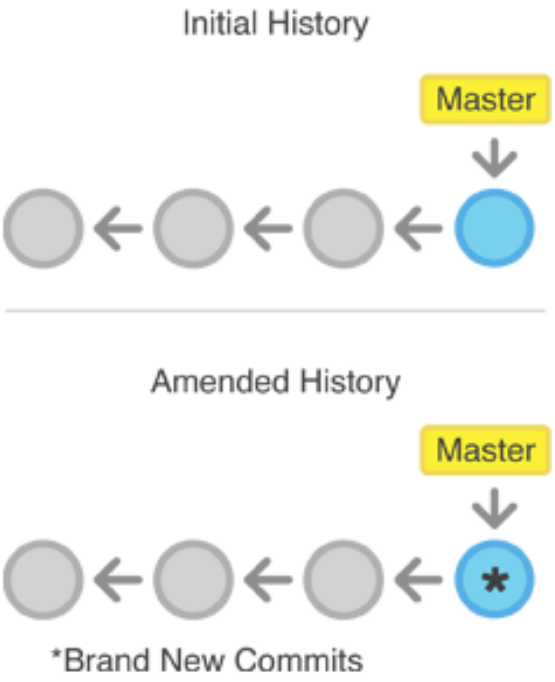
git commit --amend

- Easy way to fix the most recent commit
- Don't use for public commit

>git commit --amend

Demo :: git commit --amend

```
>touch a1.txt a2.txt
>git add a1.txt
>git commit -m "Add feature xxx"

#Forgot some file
>git add a2.txt
>git commit --amend --no-edit
```

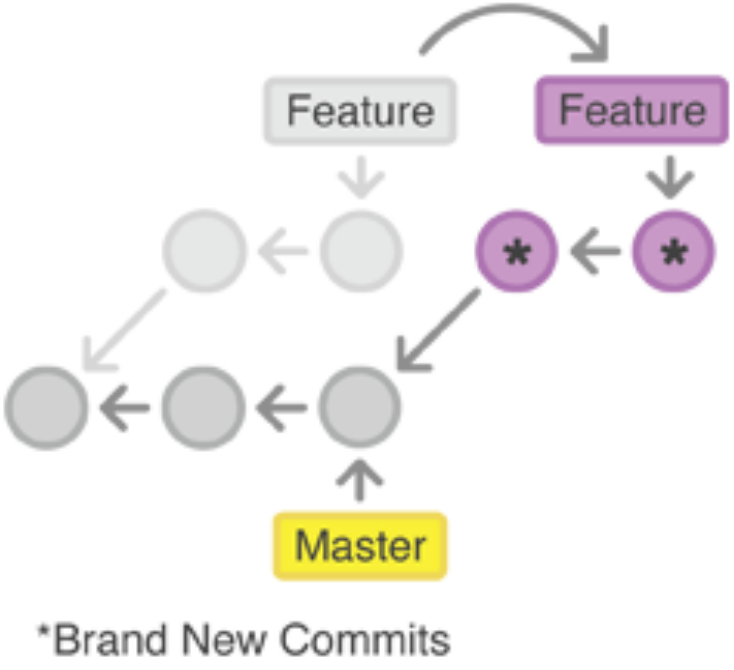


git rebase

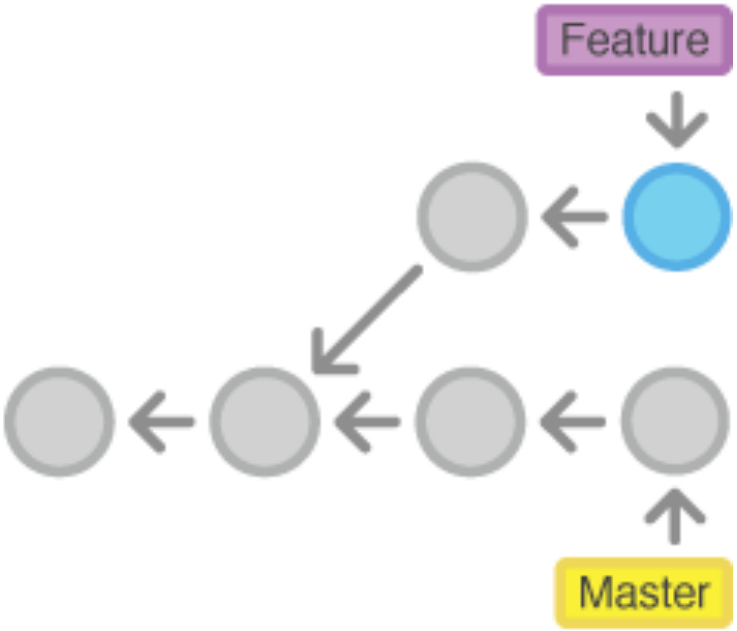
- Move branch to new base commit
- Maintain a linear project history
- Don't use for public commit

>git rebase <base>

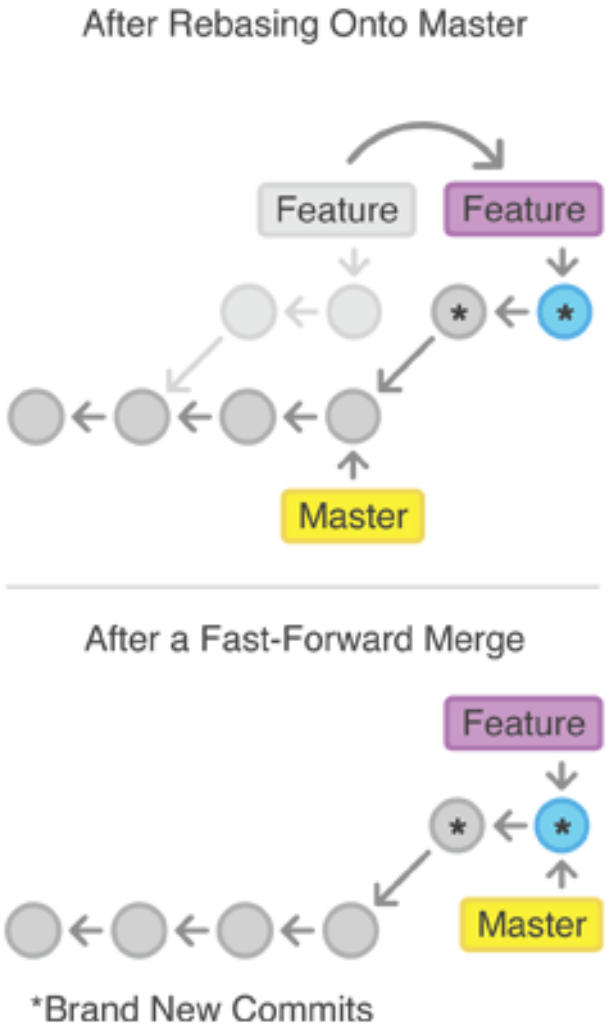
<base>
=> id, branch, tag, HEAD



How to integrate Feature to Master ?

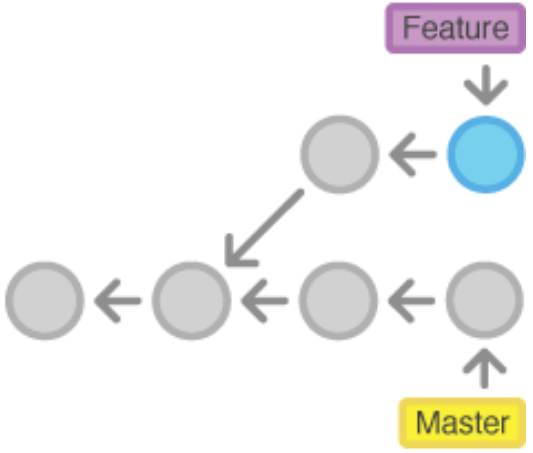


Solution :: Rebase and Merge



Demo :: git rebase

#start develop new feature
>git checkout -b new-feature master
>touch new.txt
>git add <file>
>git commit -m "Start new feature"



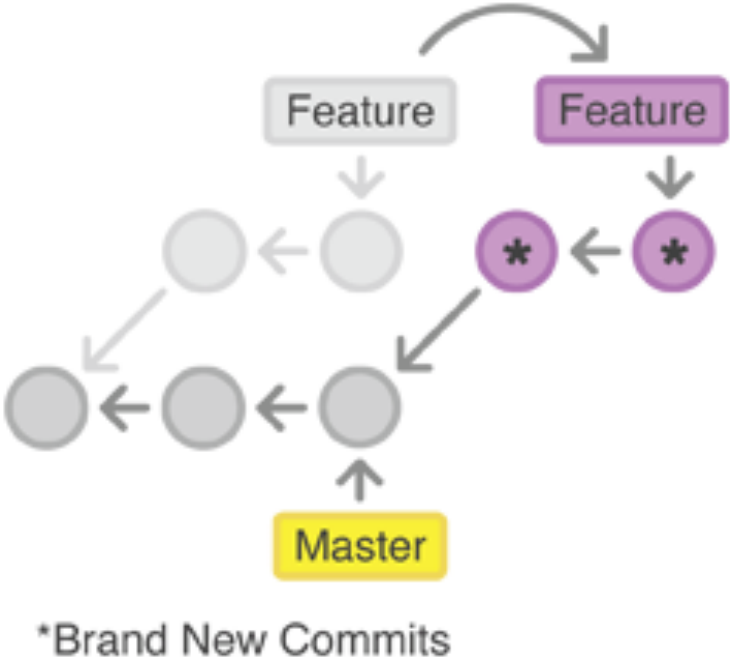
Demo :: git rebase

#Fix incident or bug
>git checkout -b hotfix master
>edit some file
>git add <file>
>git commit -m "Start hotfix"

>git checkout master
>git merge hotfix
>git branch -d hotfix

Demo :: git rebase

#Rebase new-feature to master
>git checkout new-feature
>git rebase master



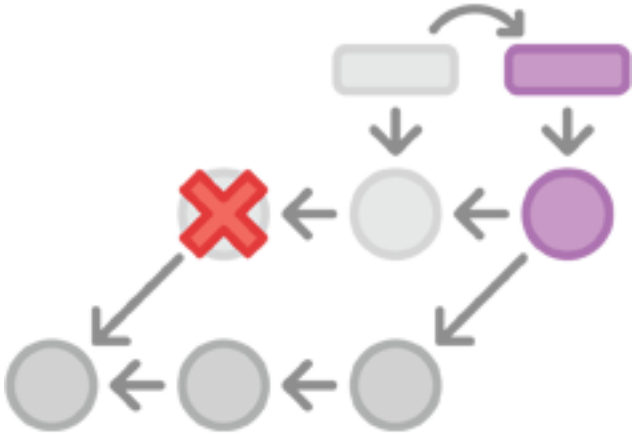
Demo :: git rebase

#Merge new-feature to master
>git checkout master
>git merge new-feature

git rebase -i

- Interactive rebase

>git rebase -i <base>



Reference Website

- http://www.atlassian.com/git/tutorial/