Quick Intro to Git version control

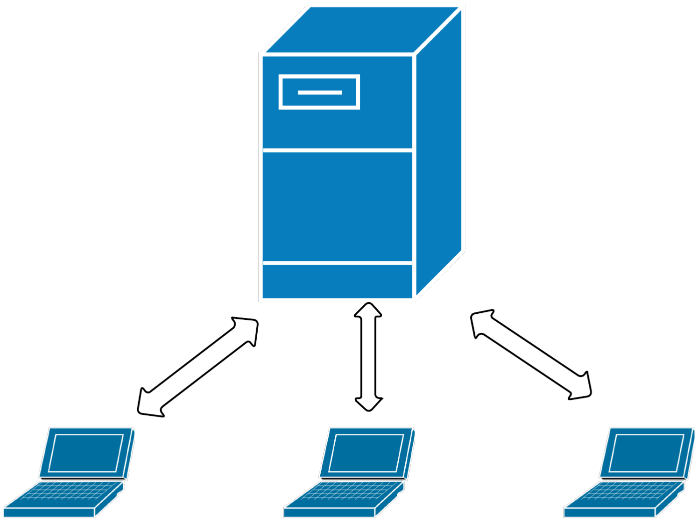
July 15, 2014

Contents

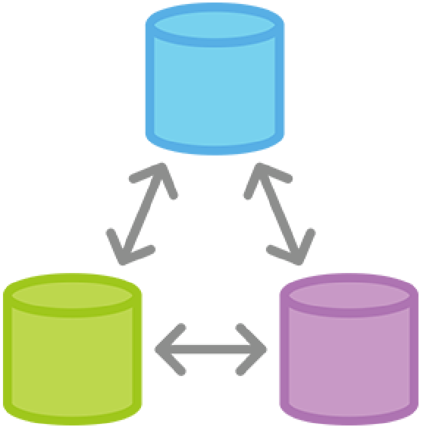
Intro

Git is an open source, **distributed** version control system designed for speed and efficiency.

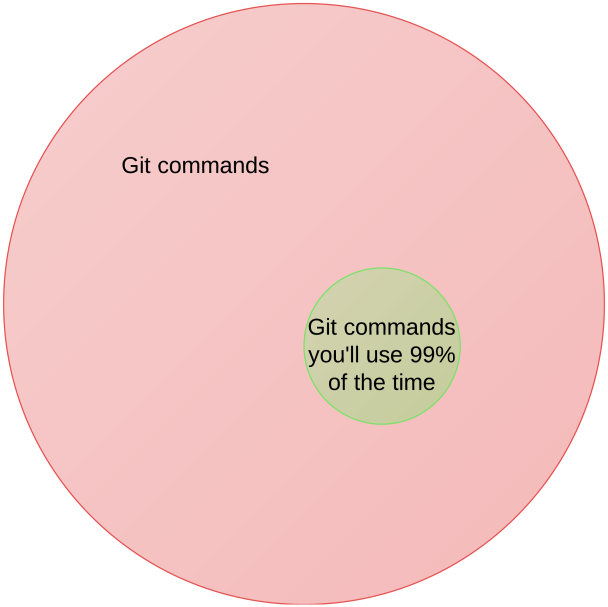
Centralized paradigm (CVS, SVN, Perforce)



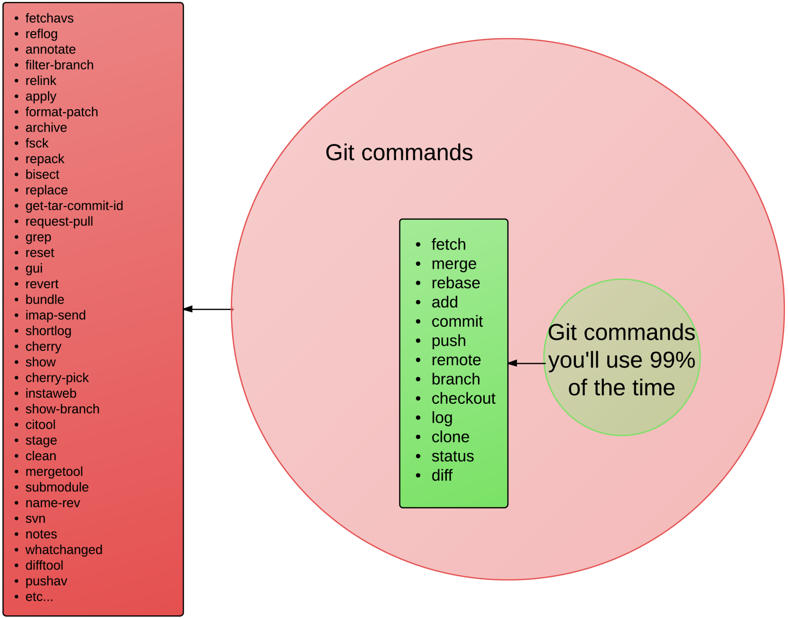
Distributed paradigm (Git, Mercurial)



Git commands



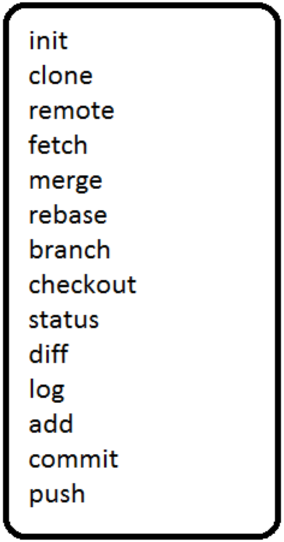
Git commands



- fetchavs
- reflog
- annotate
- filter-branch
- relink
- apply
- format-patch
- archive
- fsck
- repack
- bisect
- replace
- get-tar-commit-id
- request-pull
- grep
- reset
- gui
- revert
- bundle
- imap-send
- shortlog
- cherry
- show
- cherry-pick
- instaweb
- show-branch
- citool
- stage
- clean
- mergetool
- submodule
- name-rev
- svn
- notes
- whatchanged
- difftool
- pushav
- etc...

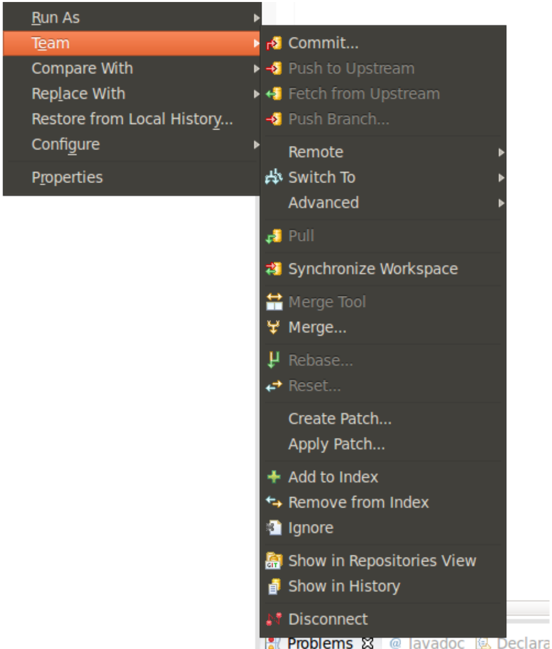
- fetch
- merge
- rebase
- add
- commit
- push
- remote
- branch
- checkout
- log
- clone
- status
- diff

Git commands

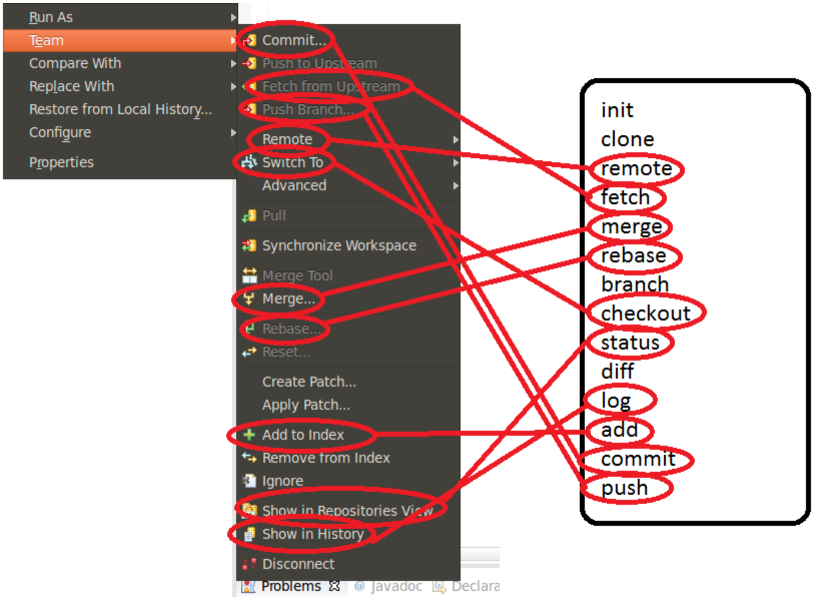


init
clone
remote
fetch
merge
rebase
branch
checkout
status
diff
log
add
commit
push

Eclipse

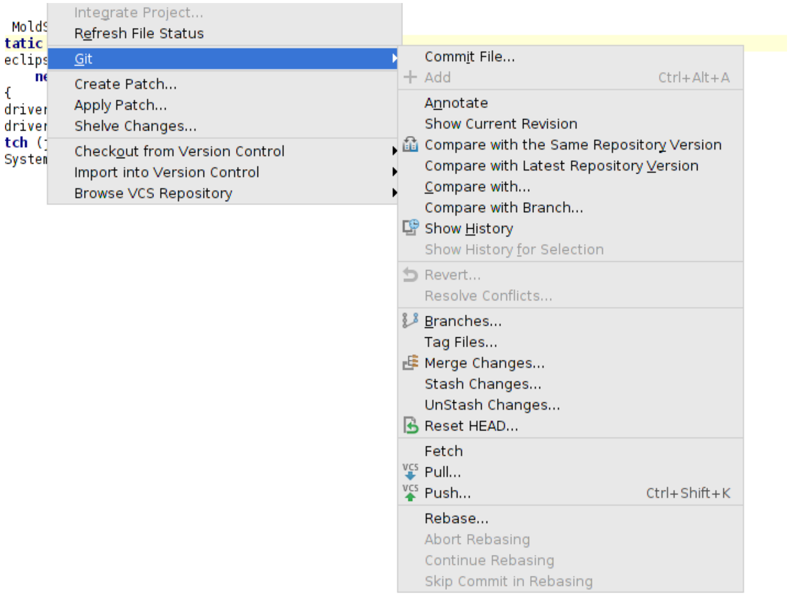

Run As
Team
Compare With
Replace With
Restore from Local History...
Configure
Properties
Commit...
Push to Upstream
Fetch from Upstream
Push Branch...
Remote
Switch To
Advanced
Pull
Synchronize Workspace
Merge Tool
Merge...
Rebase...
Reset...
Create Patch...
Apply Patch...
Add to Index
Remove from Index
Ignore
Show in Repositories View
Show in History
Disconnect
Problems
Javadoc
Declara

Eclipse



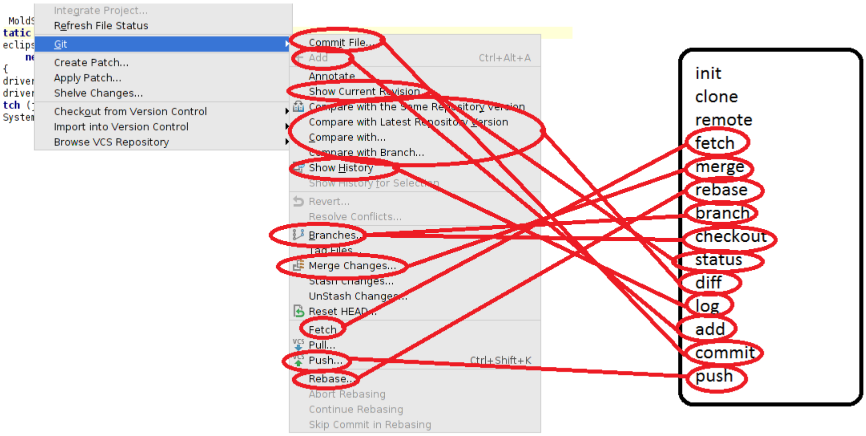
Run As
Team
Compare With
Replace With
Restore from Local History...
Configure
Properties
Commit...
Push to Upstream
Fetch from Upstream
Push Branch...
Remote
Switch To
Advanced
Pull
Synchronize Workspace
Merge Tool
Merge...
Rebase...
Reset...
Create Patch...
Apply Patch...
Add to Index
Remove from Index
Ignore
Show in Repositories View
Show in History
Disconnect
Problems
Javadoc
Declara
init
clone
remote
fetch
merge
rebase
branch
checkout
status
diff
log
add
commit
push

IntelliJ



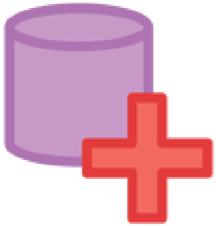
Integrate Project...
Refresh File Status
Git
Create Patch...
Apply Patch...
Shelve Changes...
Checkout from Version Control
Import into Version Control
Browse VCS Repository
Commit File...
Add Ctrl+Alt+A
Annotate
Show Current Revision
Compare with the Same Repository Version
Compare with Latest Repository Version
Compare with...
Compare with Branch...
Show History
Show History for Selection
Revert...
Resolve Conflicts...
Branches...
Tag Files...
Merge Changes...
Stash Changes...
UnStash Changes...
Reset HEAD...
Fetch
Pull...
Push... Ctrl+Shift+K
Rebase...
Abort Rebasing
Continue Rebasing
Skip Commit in Rebasing

IntelliJ



Integrate Project...
Refresh File Status
Git
Create Patch...
Apply Patch...
Shelve Changes...
Checkout from Version Control
Import into Version Control
Browse VCS Repository
Commit File...
Add Ctrl+Alt+A
Annotate
Show Current Revision
Compare with the Same Repository Version
Compare with Latest Repository Version
Compare with...
Compare with Branch...
Show History
Show History for Selection
Revert...
Resolve Conflicts...
Branches...
Tag Files...
Merge Changes...
Stash Changes...
UnStash Changes...
Reset HEAD...
Fetch
Pull...
Push... Ctrl+Shift+K
Rebase...
Abort Rebasing
Continue Rebasing
Skip Commit in Rebasing
init
clone
remote
fetch
merge
rebase
branch
checkout
status
diff
log
add
commit
push

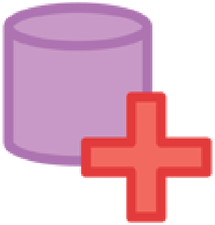
$ git init



```
$ git init
```

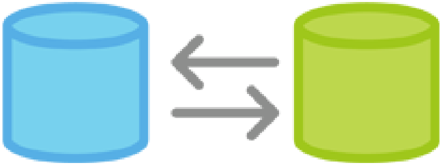
makes the current working directory a Git repository

$ git init



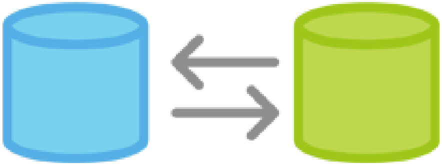
a hidden directory .git is inserted in the root directory of the Git repository. Unlike CVS or SVN, no .git directory is inserted in each subdirectories

$ git remote



A Git remote is best thought as an alias for a URL. It's an address to a remote Git repository from which you can fetch or push source code.

$ git remote



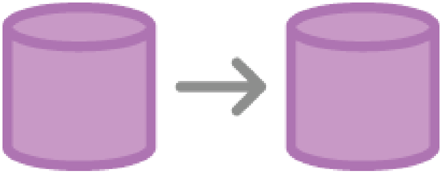
Create a new remote:

```
$ git remote add remote_name remote_url
```

List remotes:

```
$ git remote -v
```

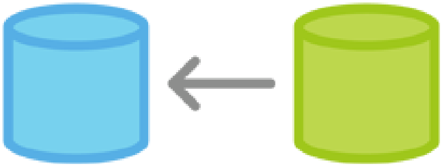
$ git clone



```
$ git clone remote_url
```

create a local copy of the Git repository hosted at remote_url

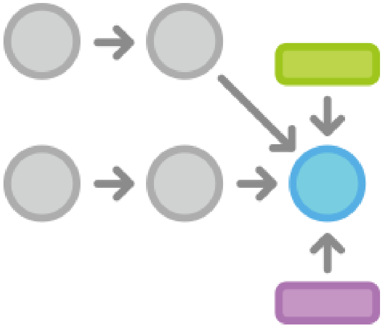
$ git fetch



```
$ git fetch remote_name_or_url
```

fetches (but does not apply) new commits from the remote Git repository

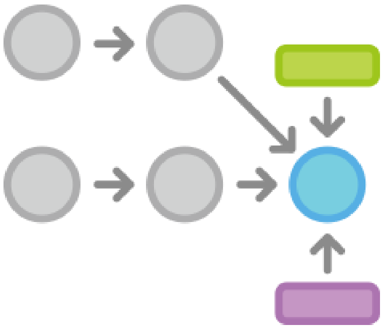
$ git merge



```
$ git merge [remote_name/]branch_name
```

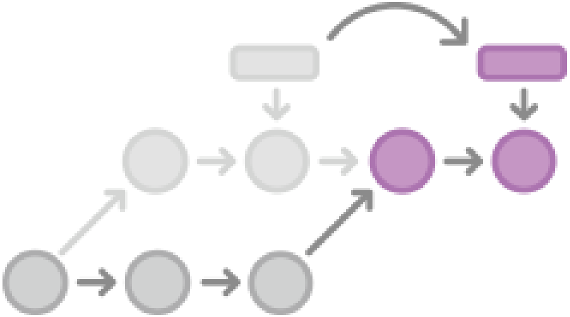
is used to merge a branch into the current one.

$ git merge



One can merge two branches of a local repository or (more often) two branches across repositories

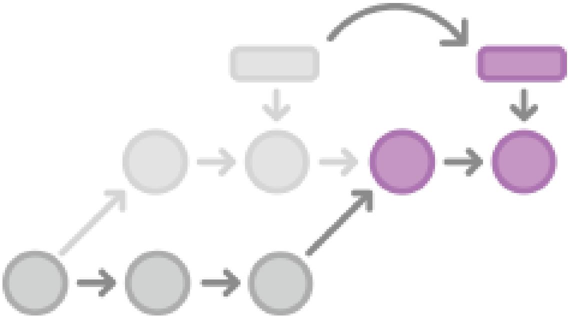
$ git rebase



```
$ git rebase [remote_name/]branch_name
```

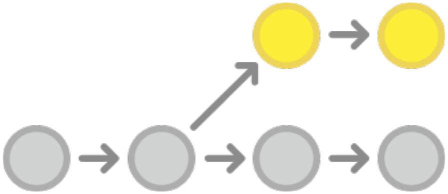
is used to reorder local commits to appear after remote commits

$ git rebase



rebase should ONLY be used if local commits have NEVER been shared

$ git branch



```
$ git branch new_branch_name
```

is used to create a new branch from the current branch

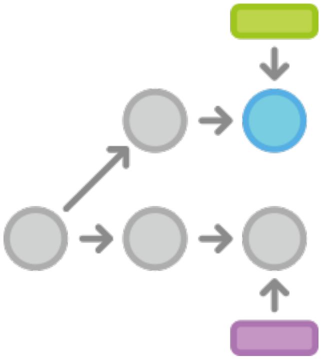
$ git branch

Without a branch name, it will list all the branch in the local repository:

```
$ git branch
  develop
* featureJIRA12
  master
```

the asterisk shows the current branch

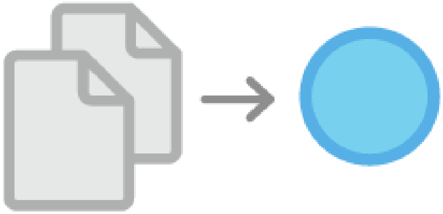
$ git checkout



```
$ git checkout branch_name
```

is used to switch between branches

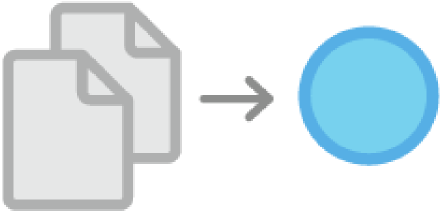
$ git add



```
$ git add [filename1 [filename2]...]
```

is used to add files to staging area

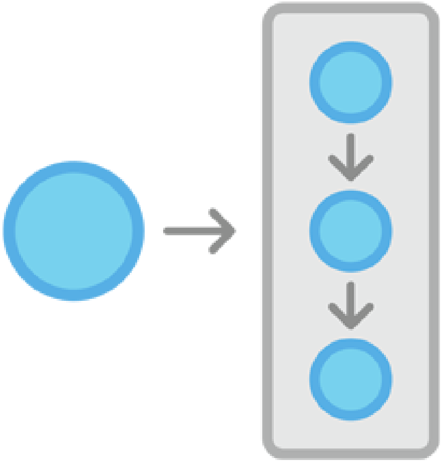
$ git add



```
$ git add [filename1 [filename2]...]
```

If no file is specified, all modified files on the current branch are added to staging area

$ git commit



```
$ git commit -m "commit msg JIRA-XXX"
```

commits the changes previously added to staging area

$ git status

```
$ git status
```

shows which files on the current branch are untracked, modified and staged for commit

$ git diff

```
$ git diff [filename1 [filename2]...]
```

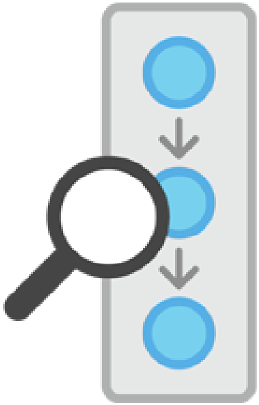
shows lines that have changed since latest commit on branch

$ git diff

```
$ git diff [filename1 [filename2]...]
```

If no file is specified, show diff for all modified files on the current branch

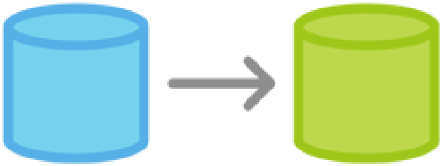
$ git log



```
$ git log [-n] [branch_name]
```

shows latest n (or all) commits for branch 'branch_name' (default to current branch)

$ git push



```
$ git push remote_name_or_url branch_name
```

pushes local commits to remote branch

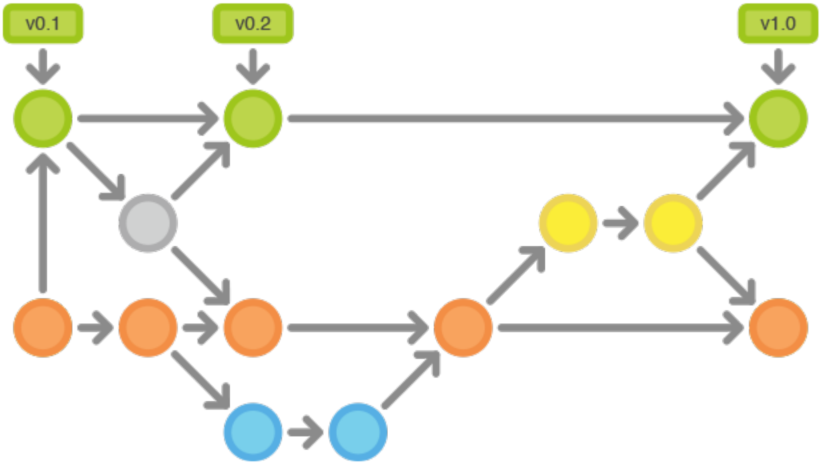
Workflows

- Centralized workflow
- Feature branch workflow
- Gitflow workflow
- Forking workflow

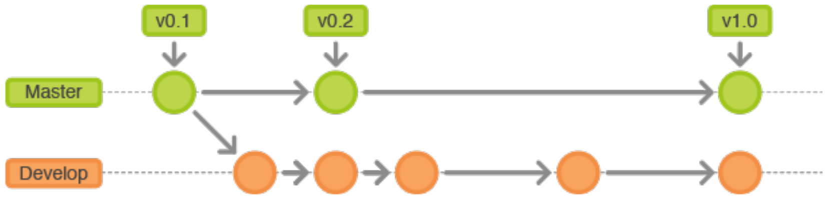
Workflows

- Centralized workflow
- Feature branch workflow
- Gitflow workflow
- Forking workflow

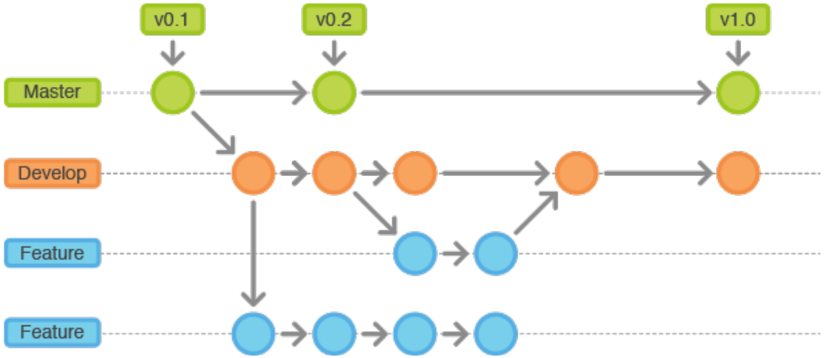
Gitflow



Gitflow



Gitflow



Gitflow

master branch
: stores the official release history. It is only changed when the Bamboo users merge the develop branch into it

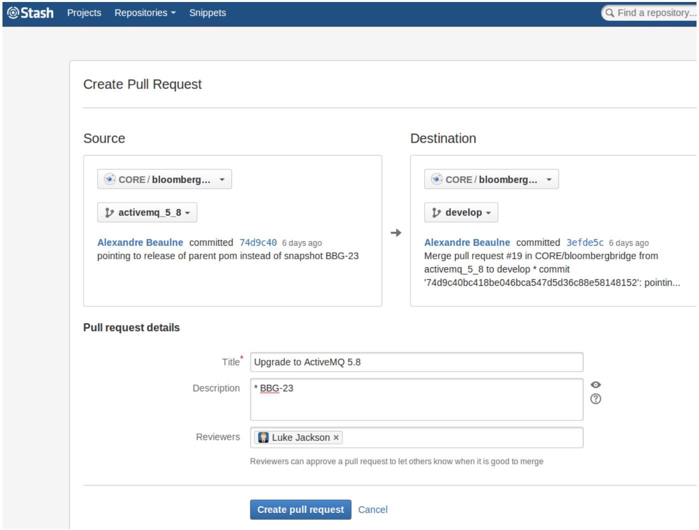
develop branch
: serves as an integration branch for features. It is only changed when feature branches are merged into it (via pull request, no direct push)

feature branches
: are where new features reside. Features branches are branched from the develop branch, and their changes are incorporated in the canonical repo via pull requests to the develop branch

Pull requests

- Not part of Git per se
- More a feature of Git hosting solutions (Stash, Github, etc)
- Great for code reviews

Pull requests



Create Pull Request

Source

CORE / bloomberg...

activemq_5_8

Alexandre Beaulne committed 74d9c40 6 days ago
pointing to release of parent pom instead of snapshot BBG-23

Destination

CORE / bloomberg...

develop

Alexandre Beaulne committed 3efde5c 6 days ago
Merge pull request #19 in CORE/bloombergbridge from activemq_5_8 to develop * commit '74d9c40bc418be046bca547d5d36c88e58148152': pointin...

Pull request details

Title*	Upgrade to ActiveMQ 5.8
Description	* BBG-23
Reviewers	Luke Jackson ×

Reviewers can approve a pull request to let others know when it is good to merge

Create pull request Cancel

Ressources

- These slides are on Confluence (`http://confluence/display/~abeaulne/Intro+to+Git+presentation`)
- Atlassian has a great straightforward tutorial at `https://www.atlassian.com/git/`

Homework

- fetch canonical ‘practice’ repository at `http://stash/scm/core/practice.git`
- create a feature branch (branched out of develop branch) with your name
- add your name to the README.txt
- commit your change locally
- push your commit to remote feature branch at canonical ‘practice’ repo
- create a pull request, adding me (Alex) and one of your colleagues/superiors as reviewer

Solution

```
~$ git clone http://stash/scm/core/practice.git
~$ cd practice/
~/practice$ git branch develop
~/practice$ git checkout develop
~/practice$ git rebase origin/develop
~/practice$ git branch alexb
~/practice$ git checkout alexb
~/practice$ vim README.txt
~/practice$ git add README.txt
~/practice$ git commit -m "added my name"
~/practice$ git push origin alexb
```

Finally go to
`http://stash/projects/CORE/repos/practice/browse` to
create pull request