Introduction to Configuration & Source Management

Università di Modena e Reggio Emilia

Prof. Nicola Bicocchi (nicola.bicocchi@unimore.it)

ICT Company (large dimension)

- Multinational Company
- Up to 200 employees working together
- Multiple working location across the world
- Different spoken languages
- Flexible working hours
- Different approaches to work
- Well-defined hierarchical roles

Configuration Management

- Configuration Management (ITILv3): «The Process responsible for maintaining information about Configuration Items required to deliver an IT Service, including their Relationships.»
- Key part of the Agile manifesto
- Set of processes designed to manage and control objects of complex systems

Parallel between SCM and cooking

- **Source Code Management**: Accurately verify weigh and measure ingredients
- **Build Engineering**: shake ingredients and “make a cake”
- **Environment Configuration**: check the shop window is ready for sell the cake
- **Change Control**: choose when the cake is ready to be selled
- **Release engineering**: put the cake in the shop window so people can see it but they can’t buy it
- **Deployment**: deliver the cake to the customer

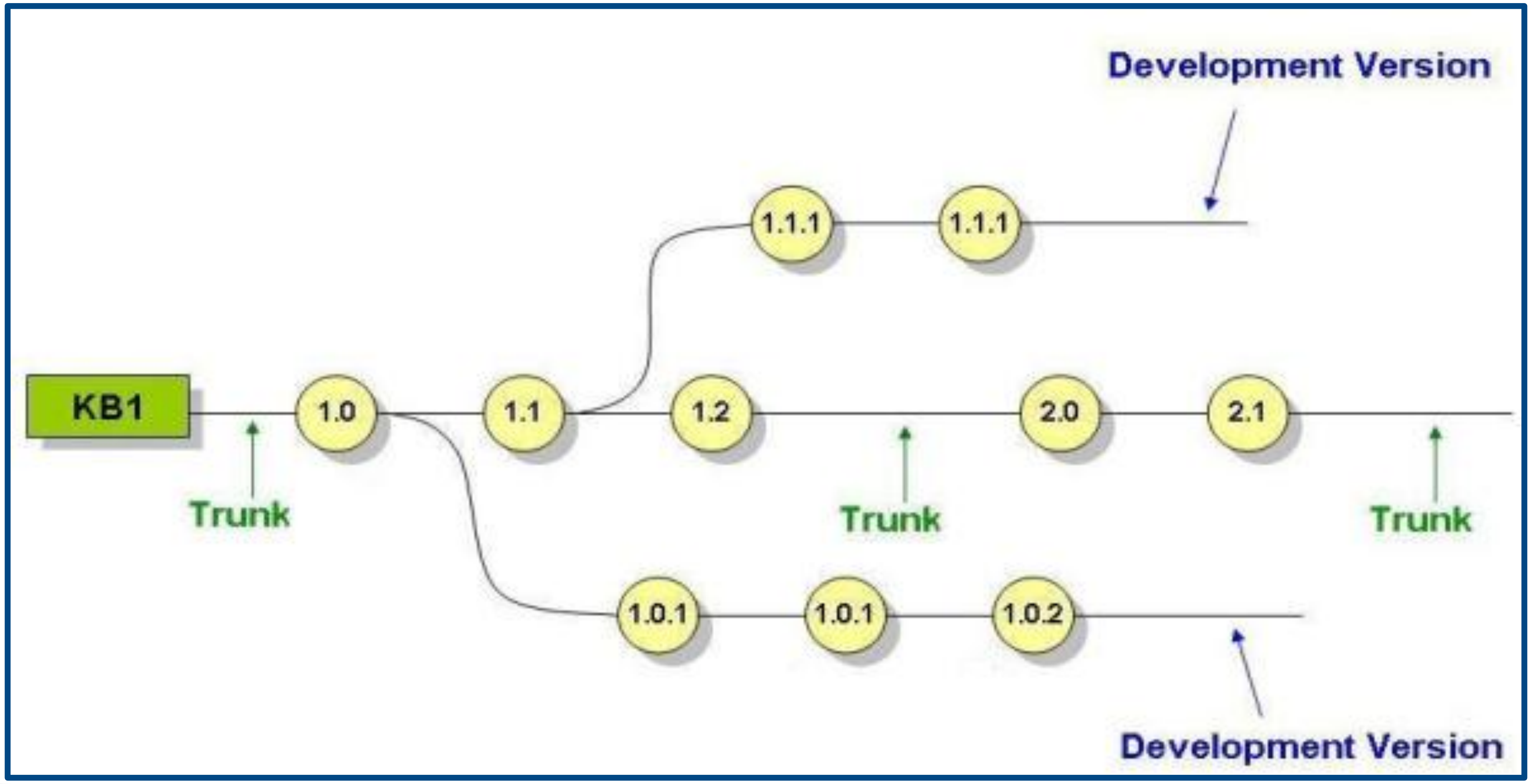
Source Management

- Precursor of the Configuration Management
- Affects the product quality and the team productivity
- Often understimated

Goals

- Create a vault for all source codes: **no source code mustn’t get lost**
- Increase productivity of the work teams (e.g., manage more than one development team)
- **Traceability**: Everyone know who change the code and when, if necessary, be able to rollback to a previous version of the source code

Basic Concepts



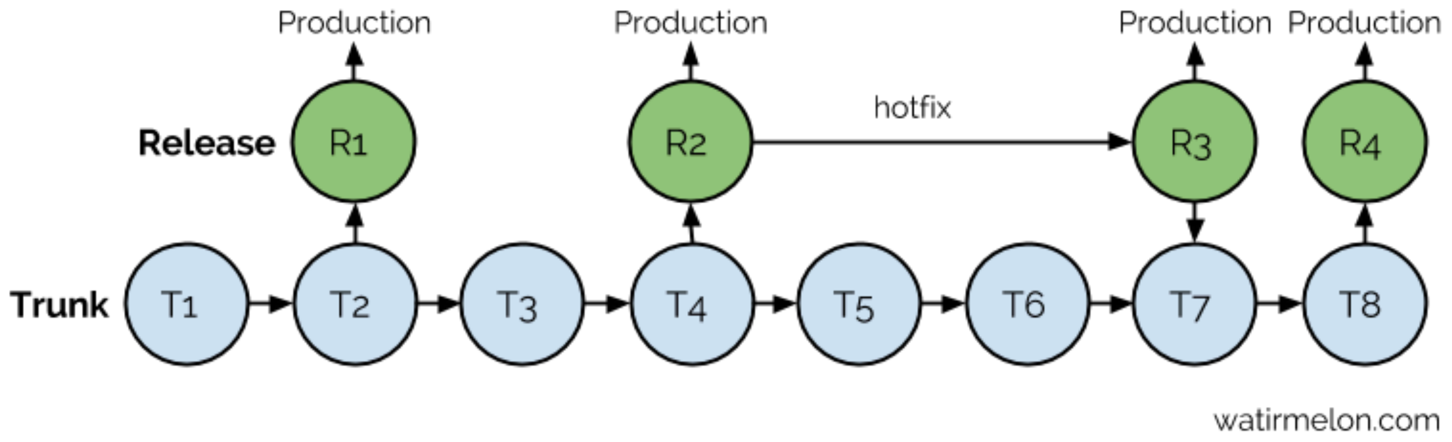
Basic Concepts(1)

- **Baseline**: Identify the exact version number of each source contained in a specific software release. Virtual time machine allows to move through versions

Basic Concepts(2)

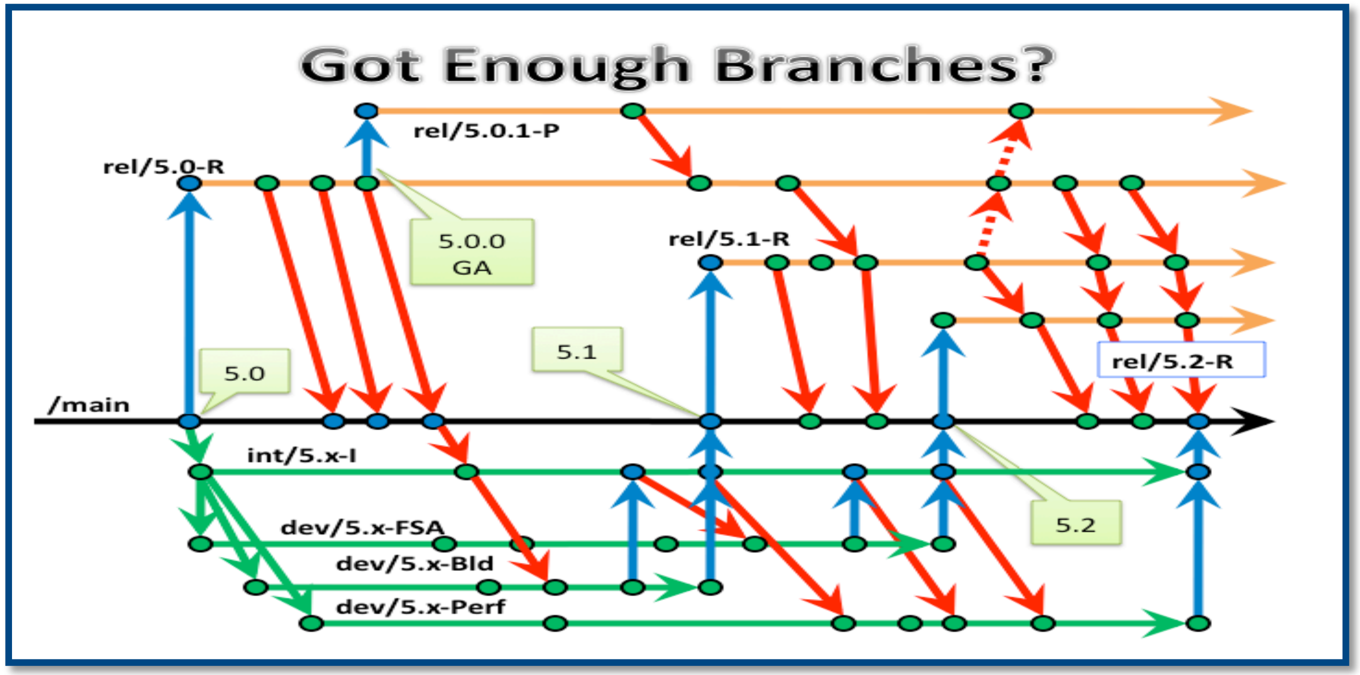
Trunk: Baseline of a software production.

Changes on trunk often reflects in a new software release (e.g., iOS7 –> iOS8)



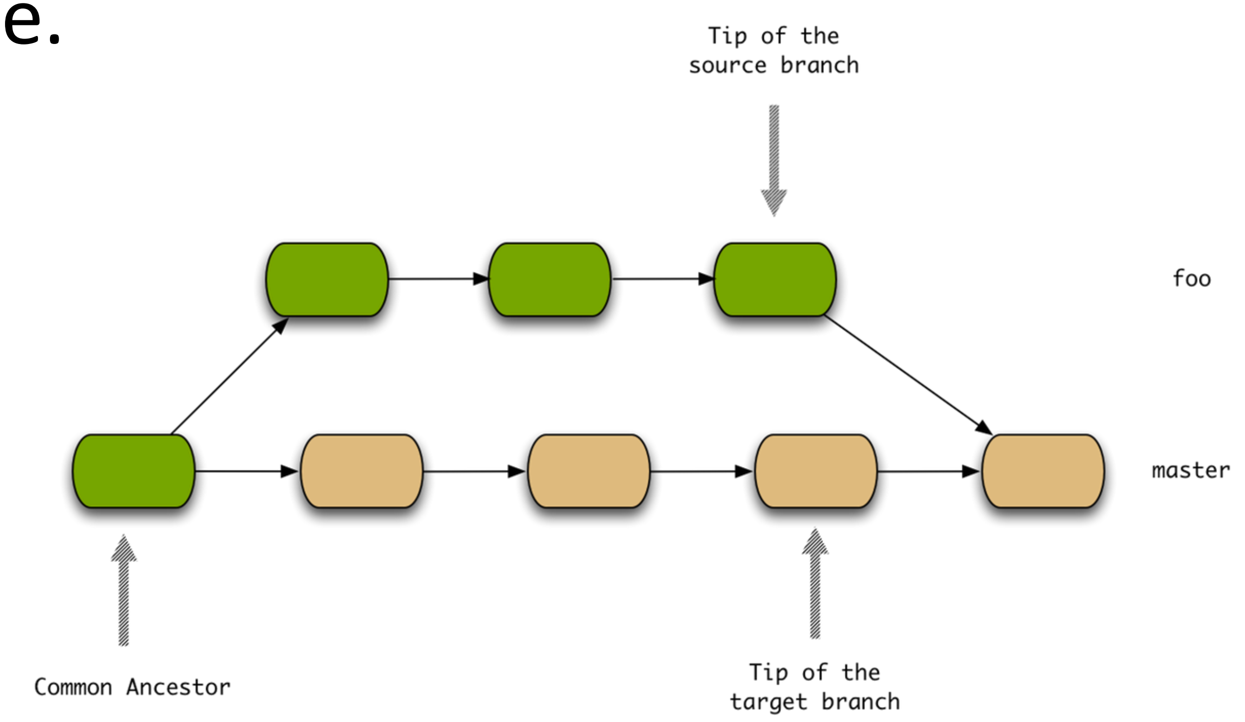
Basic Concepts(3)

Branch: parallel baseline to the trunk, of which it take some characteristic at a given instant of time. The branch is used for minor changes or product customization. (e.g., different versions of the same application Unix, Windows, Mac). Use with caution!



Basic Concepts(4)

Merging: inverse of branching. Can be used to merge changes from branch to trunk at specific instant of time. Usually this operation leads to a new release.

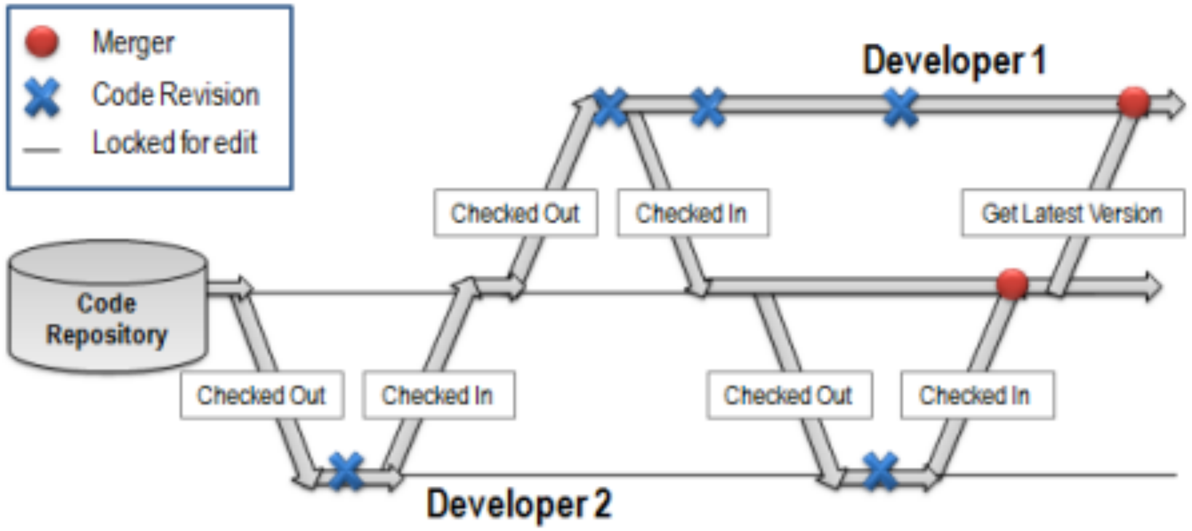


Basic Concepts(5)

Workspace: clone of the global repository stored in a private local directory. The workspace allows each user to work on a local and private copy

Basic Concepts(6)

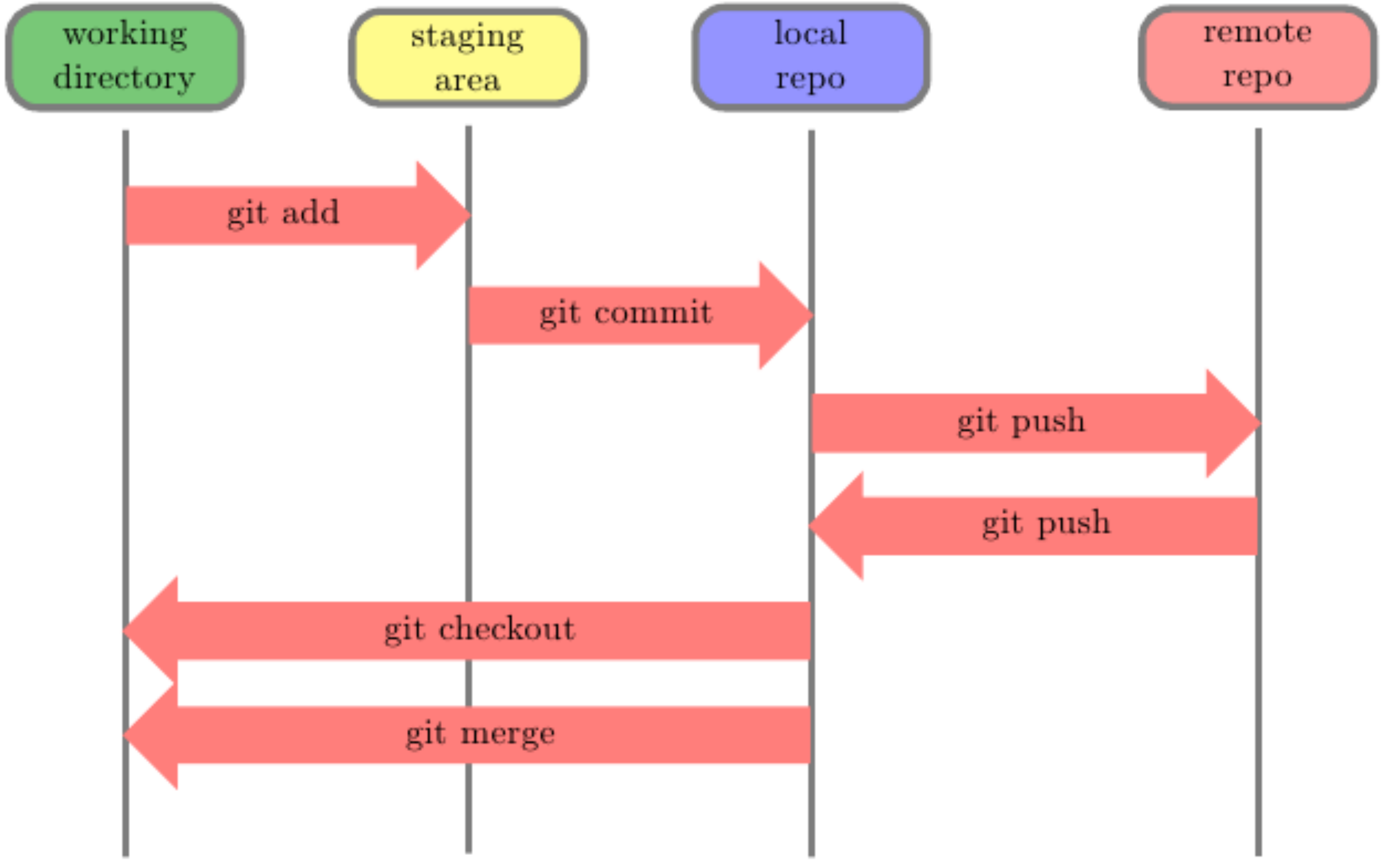
Check in / Check out: download/upload of sources from/to a Source Management



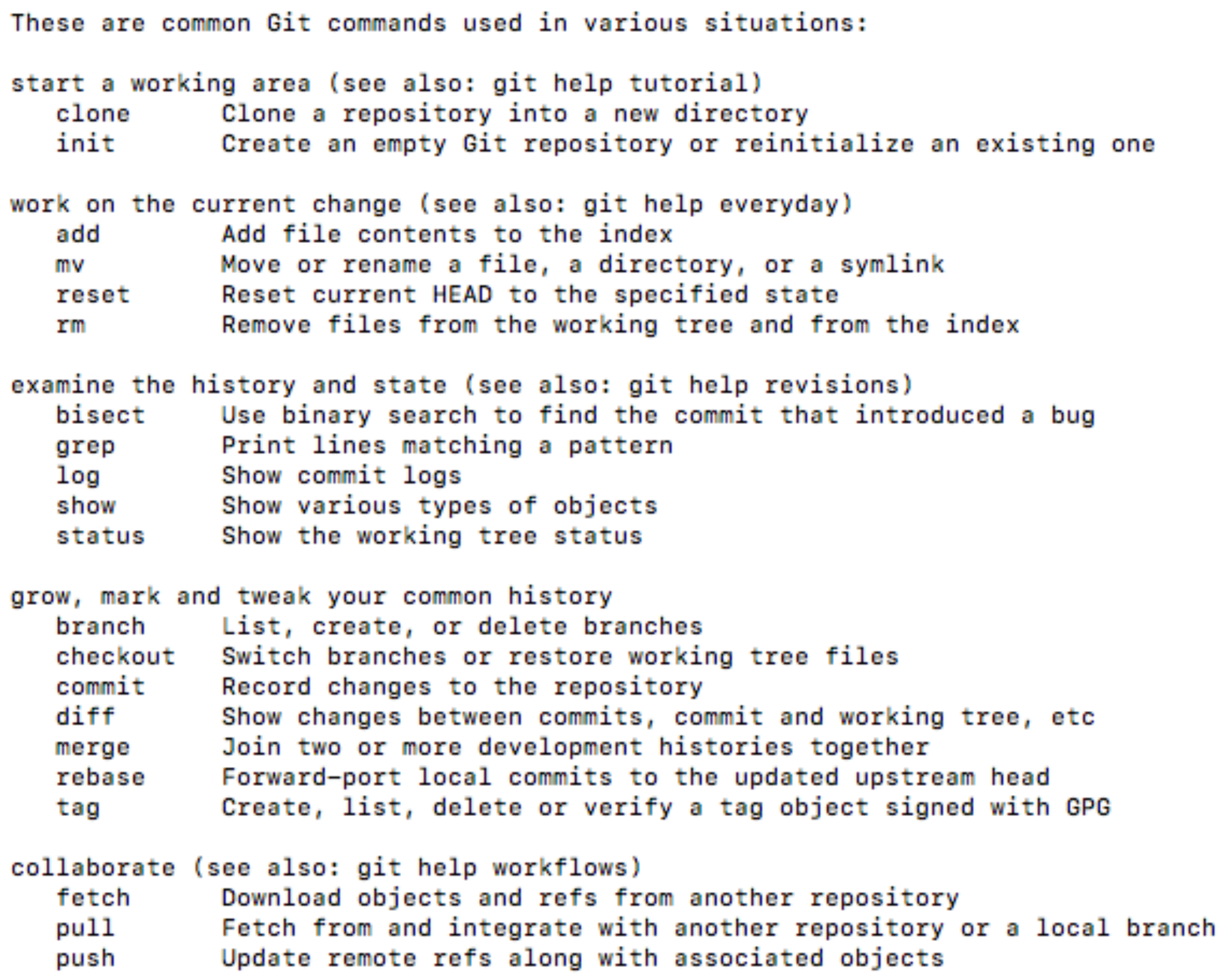
Git

- Source code Management software
- Distributed
- Open Source
- Keep track of version your files is in
- Can merge different lines of development and integrate them into a single baseline
- Used by many companies like: Google, Facebook, Twitter, Microsoft, Netflix, …

Git - workflow



Git base commands(1)



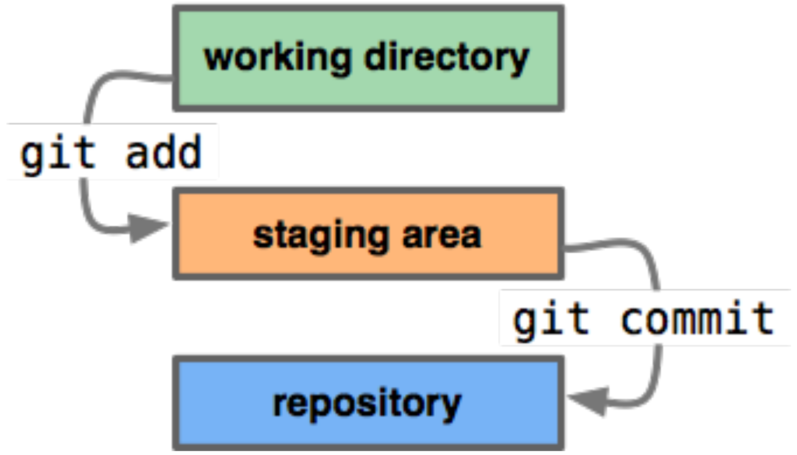
```
These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone      Clone a repository into a new directory
   init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add        Add file contents to the index
   mv         Move or rename a file, a directory, or a symlink
   reset      Reset current HEAD to the specified state
   rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect     Use binary search to find the commit that introduced a bug
   grep       Print lines matching a pattern
   log        Show commit logs
   show       Show various types of objects
   status     Show the working tree status

grow, mark and tweak your common history
   branch     List, create, or delete branches
   checkout   Switch branches or restore working tree files
   commit     Record changes to the repository
   diff       Show changes between commits, commit and working tree, etc
   merge      Join two or more development histories together
   rebase     Forward-port local commits to the updated upstream head
   tag        Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch      Download objects and refs from another repository
   pull       Fetch from and integrate with another repository or a local branch
   push       Update remote refs along with associated objects
```

Git base commands (2)

- **Init:** Initialize empty Git repository in the working directory
  - Usage: ***git init <workingDirectory>*** Initialize a repo with .git
- **Clone:** Clone a repository in the working directory
  - Usage: ***git clone <repositoryUrl>***
- **Add**: add an untracked/modified file to staging area
  - Usage: ***git add <fileName>***
  - These files are not commited yet!

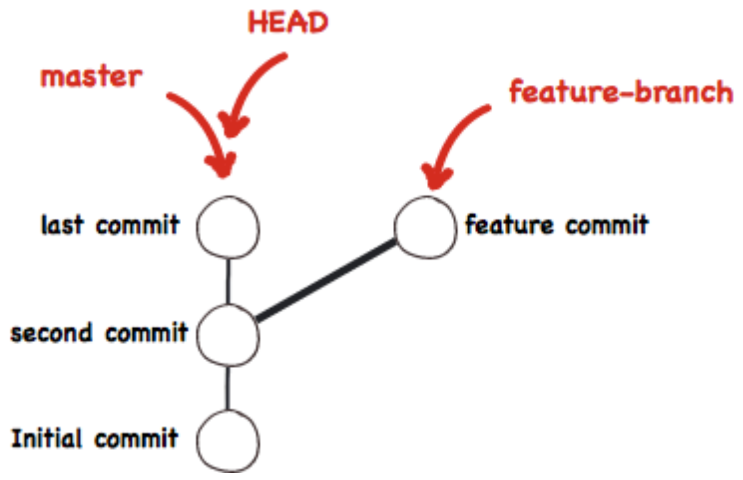
Git base commands (3)

- Commit: commit staged files to the local repository
  - Usage: ***git commit –m "commit message"***



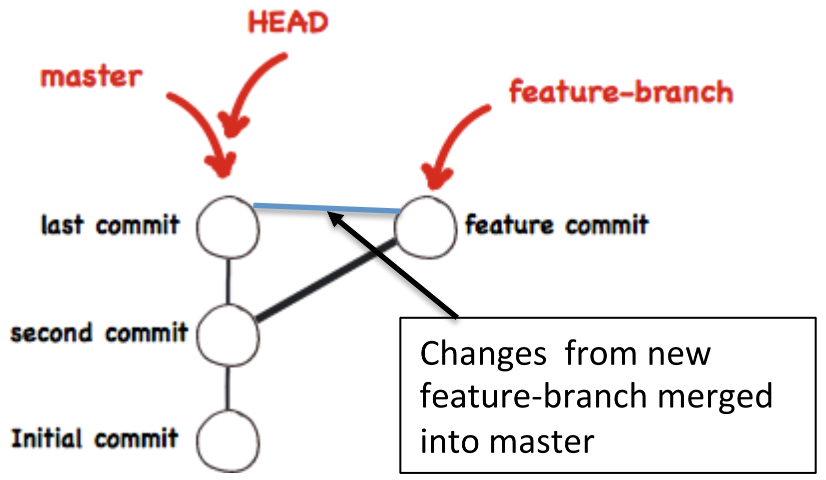
Git base commands (4)

- branch/merge commands:
  - The branch command has the effect of creating a new branch of the repository
    - Usage: ***git branch <new_branch_name>***



Git base commands (5)

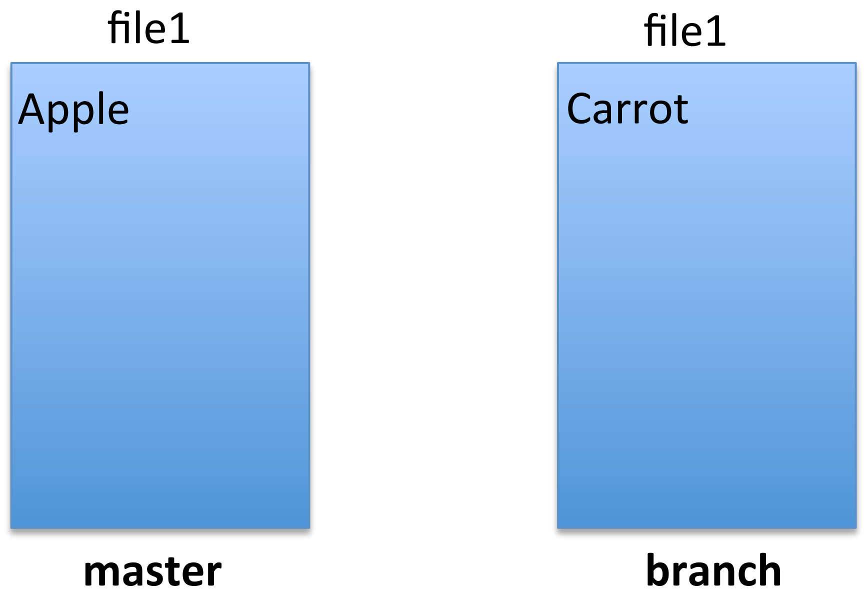
- merge: has the effect of merging a branch into your current master (integrate all the commits of your branch)
  - Usage: ***git merge <branch_to_merge>***



Changes from new feature-branch merged into master

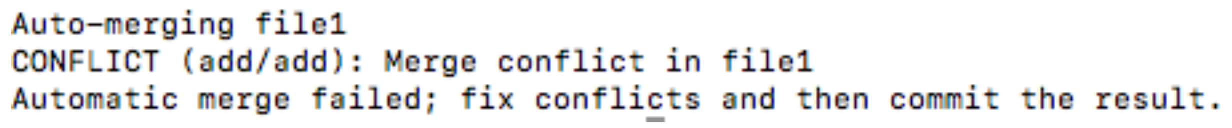
Merge conflicts

- What happens when merging and there are two commits contains different changes to same line?



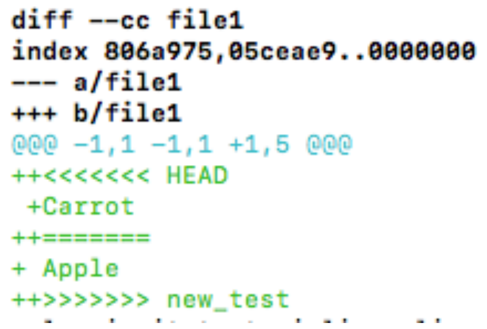
Merge conflicts (1)

- Git will notice conflicts in the files



```
Auto-merging file1
CONFLICT (add/add): Merge conflict in file1
Automatic merge failed; fix conflicts and then commit the result.
```

- Use*: **git diff <fileName>*** to check conflicts



```
diff --cc file1
index 806a975,05ceae9..0000000
--- a/file1
+++ b/file1
@@@ -1,1 -1,1 +1,5 @@@
++<<<<<<< HEAD
 +Carrot
++=======
+ Apple
++>>>>>>> new_test
```

- Solutions:
  - Manually fix the conflict
  - Abort the merge using: ***git merge -abort***

Git base commands (6)

- Remote commands: **push/pull**
  - **git push:** to push commits made on your local branch to a remote repository
  - **git pull:** to fetch from remote repository and merge with the local one
    - Note that conflicts during the merge phase occurs locally!