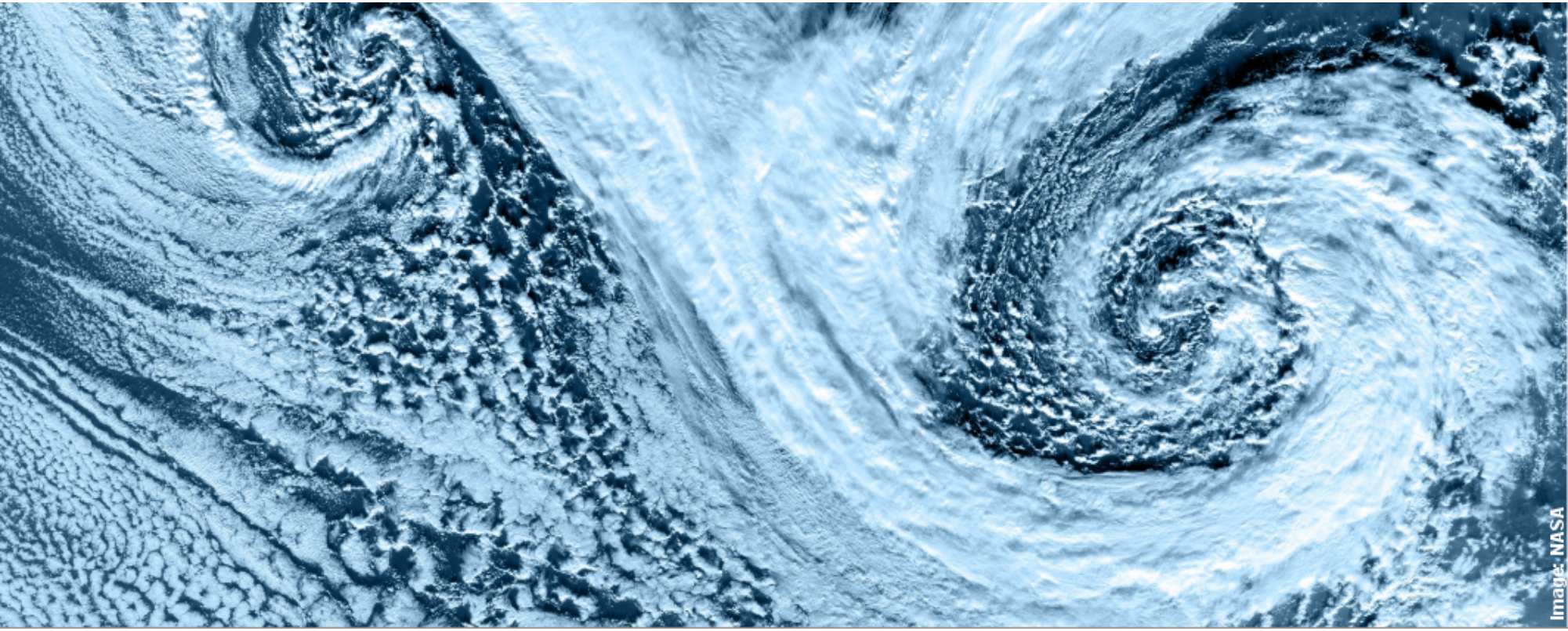
C2SM
Center for Climate
Systems Modeling

ETH zürich

Git and Github

Katie Osterried
C2SM
COSMO General Meeting 2016



Outline

C2SM Center for Climate Systems Modeling

Outline

C2SM
Center for Climate
Systems Modeling

What is git?

C2SM Center for Climate Systems Modeling

- Version control system (like SVN)
- Tool for tracking of changes in files in order to:
  - Record reasons for changes
  - Compare with and incorporate versions from other sources
  - Have multiple people developing the same code
  - Maintain several parallel versions of the same code in a systematic way
- Designed for collaborative, open source workflows

Source: https://git-scm.com/downloads/logos

Status of the C2SM git migration

C2SM
Center for Climate
Systems Modeling

- C2SM hosted COSMO and related codes were successfully migrated from SVN to git in December 2015
- All the code development history was retained during the migration
- Code is now hosted on github.com in 42 separate code repositories
- Fieldextra and Extpar official versions now hosted on Github

Git Terminology

C2SM Center for Climate Systems Modeling

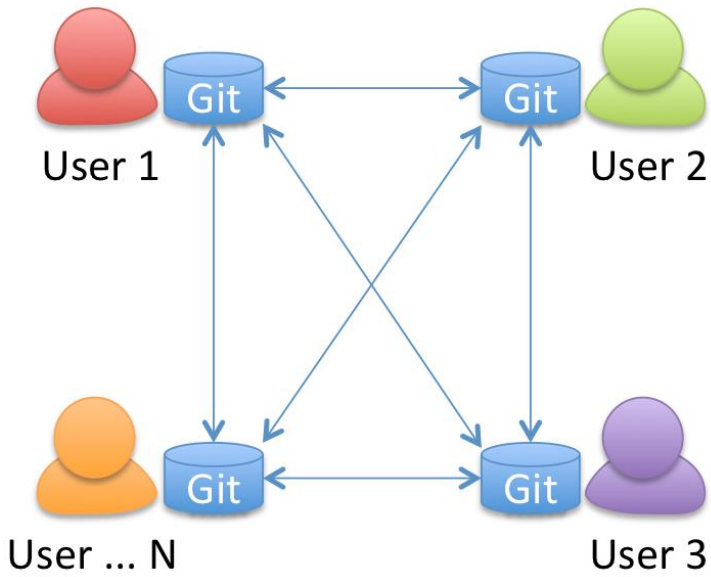
1. **repository** : the location of the saved code and its history
2. **branch** : an independent line of development
3. **master** : the default branch
4. **commit** : a snapshot of your project at a certain time
5. **tag** : a frozen reference to a particular commit
6. **HEAD** : the currently checked out commit
7. **index/staging area** : area between working directory and repository
8. **remote** : a repository linked to the local repository

Important differences from SVN

C2SM Center for Climate Systems Modeling

Git is a distributed version control system

- Every user has the whole repository
- Users can save changes to the local repository without a network
- Repositories can be located anywhere and linked together easily
- Workflow for a group must be clearly defined



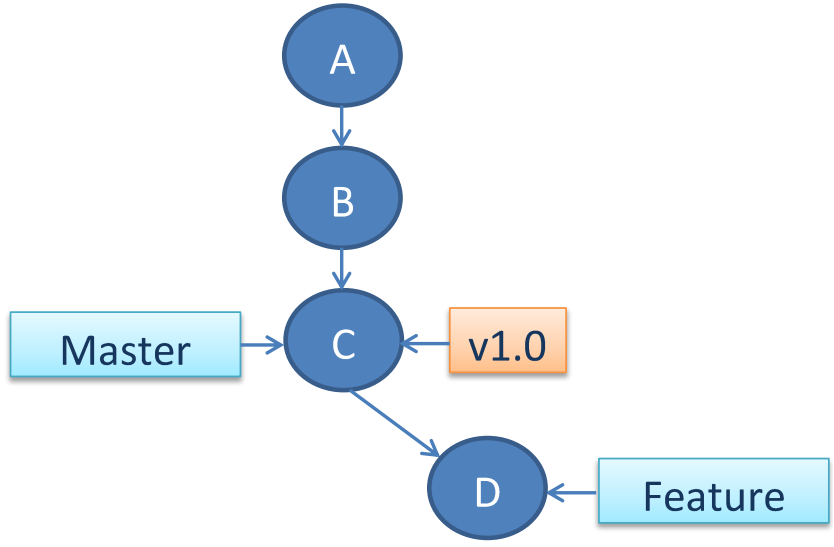
Source: http://www.codemag.com/Article/1105101

Important differences from SVN

C2SM Center for Climate Systems Modeling

Git uses a strict definition of branches and tags

- Branches and tags are not associated with different directories (like SVN)
- Branches and tags are simply pointers to a certain commit
- The trunk equivalent is called "master" and is no different from any other branch



Important differences from SVN

C2SM Center for Climate Systems Modeling

Git uses different commit IDs

- Linear revision numbers don't work
- Each commit has a unique identifier generated by an algorithm
- Series of 40 characters and numbers
- Commits can be generally referred to by the first 6-8 characters of the ID

Example: commit f3abe64fc121b75f3f0566c73f2f1a4e8fffd68e

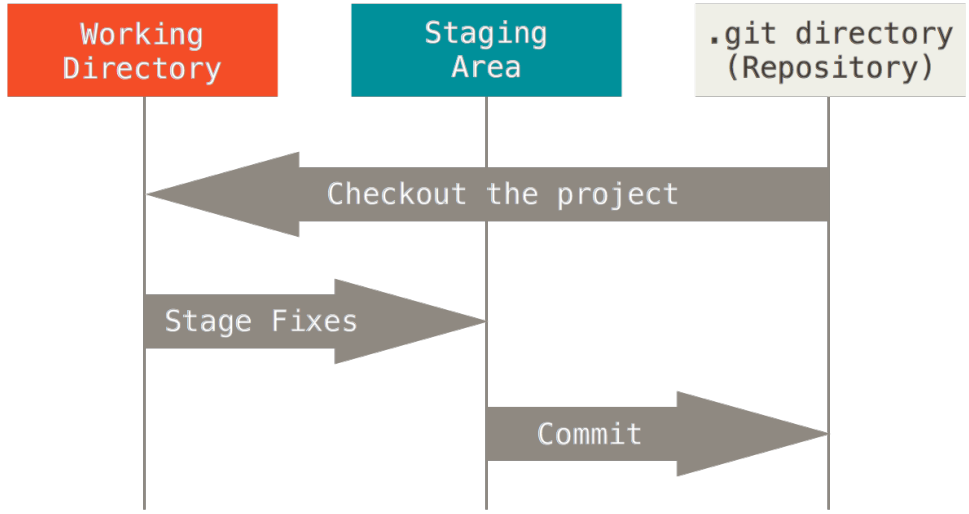
Can be referred to as: f3abe64

Important differences from SVN

C2SM Center for Climate Systems Modeling

Git uses a staging area

- Additional layer between working directory and repository
- Stores information about what will go in the next commit
- Allows you to group commits logically



Source: https://git-scm.com/book/en/v1/Getting-Started-Git-Basics

Outline

Local git workflow

C2SM Center for Climate Systems Modeling

1. Start or copy a git repository locally *(git init, git clone)*
2. Make a feature branch for developing *(git branch, git checkout)*
3. Make some code changes
4. Save the code changes to the staging area *(git add)*
5. Save the code changes to the repository *(git commit)*
6. Merge the changes from the feature branch to the master *(git merge)*

How to start a repository

C2SM Center for Climate Systems Modeling

Commands to start working with git:

git config

- Set configuration variables for git

Usage: *git config user.name "Your Name"*

Usage: *git config --global user.email "youremail@email.com"*

git init

- Creates an empty git repository
- Creates by default the master branch
- Creates the .git folder and contents

Usage: *git init*

git clone

- Copies an existing git repository
- Creates and navigates to the current branch of the copied repository
- Links the original repository as a remote

Usage: *git clone /path_to_original /path_to_copy*

How to make commits

C2SM Center for Climate Systems Modeling

Commands for saving code changes:

- **git add**
  - Saves code changes to the staging area
  - Can add all or some of the current code changes
  - Can be performed multiple times before a commit

  Usage: *git add /path_to_file*

- **git commit**
  - Saves the changes in the staging area to the repository
  - Creates a unique commit ID
  - Saves a log message from the user

  Usage: *git commit*

How to examine a repository

C2SM Center for Climate Systems Modeling

Commands for getting information about a repository:

- **git log**
  - Displays the log of all the commits
  - Can be customized through command line options

  Usage: *git log*

- **git status**
  - Shows the status of the working copy
  - States which files have been placed in the staging area
  - Shows which files have been modified but not placed in the staging area

  Usage: *git status*

- **git diff**
  - Shows the changes between two versions of the code
  - Many options for customization

  Usage: *git diff*

How to examine a repository

C2SM Center for Climate Systems Modeling

Commands for looking at previous commits:

- **git checkout (old commit)**
  - Displays the working copy as it was when the commit was made
  - Should be used for looking at old commits, not development
  - Anything committed will NOT be saved to a current branch

  Usage: *git checkout commitID*

- **git checkout (single file)**
  - Updates the file in the current working copy
  - Used for recovering old versions of files
  - Anything committed WILL be saved to the current branch

  Usage: *git checkout commitID /path_to_file*

How to create and switch to a branch

C2SM Center for Climate Systems Modeling

Commands for working with branches:

- **git branch**
  - Lists current branches or creates a new one
  - Creates branch from current HEAD
  - Does not automatically switch to new branch

  Usage: *git branch branch_name*

- **git checkout (branch)**
  - Changes the files in the working copy to the branch
  - Local changes are preserved

  Usage: *git checkout branch_name*

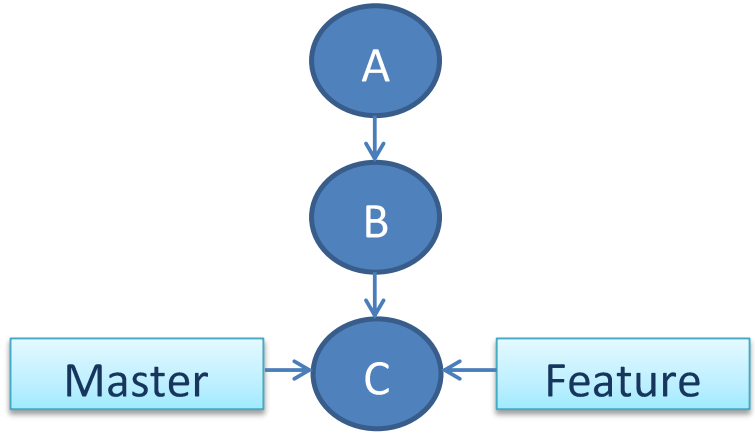
Merging without commits

C2SM Center for Climate Systems Modeling

Fast-forward merge is the default behavior

- **git merge**
  - Combines the target branch with the current branch
  - Does not create a commit unless it has to (or you tell it to)
  - Called from the branch you want to merge into

Usage: *git merge branch_name*



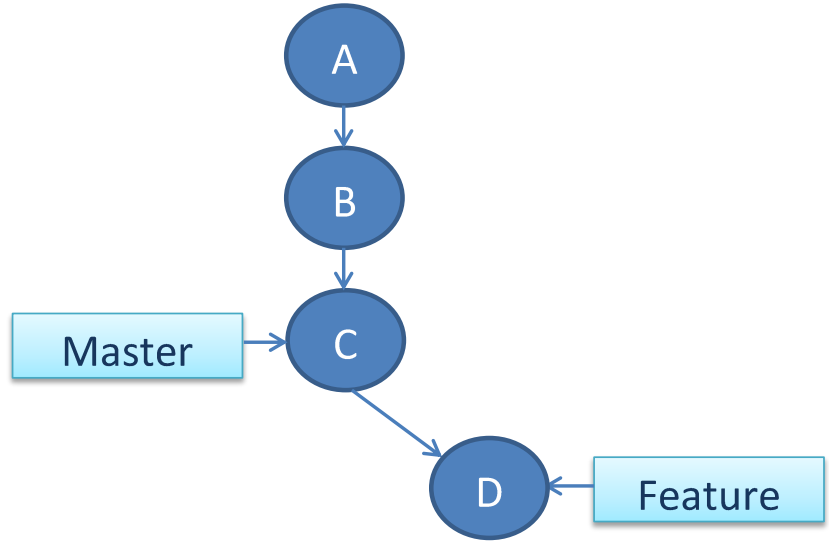
Merging without commits

C2SM Center for Climate Systems Modeling

Fast-forward merge is the default behavior

- **git merge**
  - Combines the target branch with the current branch
  - Does not create a commit unless it has to (or you tell it to)
  - Called from the branch you want to merge into

Usage: *git merge branch_name*



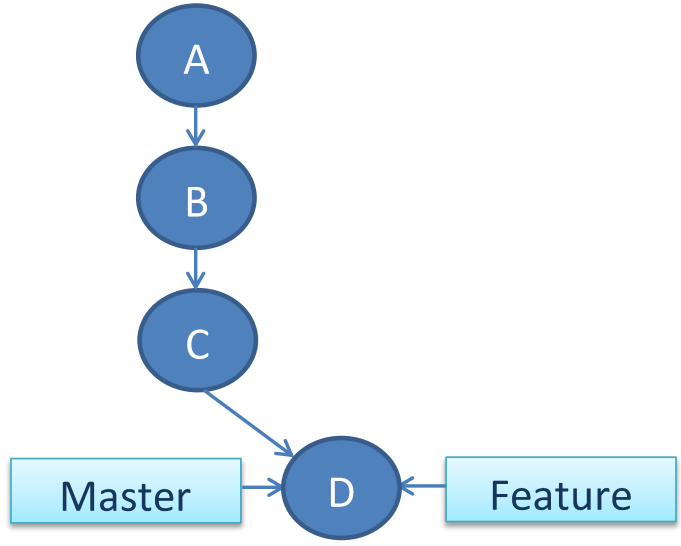
Merging without commits

C2SM Center for Climate Systems Modeling

Fast-forward merge is the default behavior

- **git merge**
  - Combines the target branch with the current branch
  - Does not create a commit unless it has to (or you tell it to)
  - Called from the branch you want to merge into

Usage: *git merge branch_name*



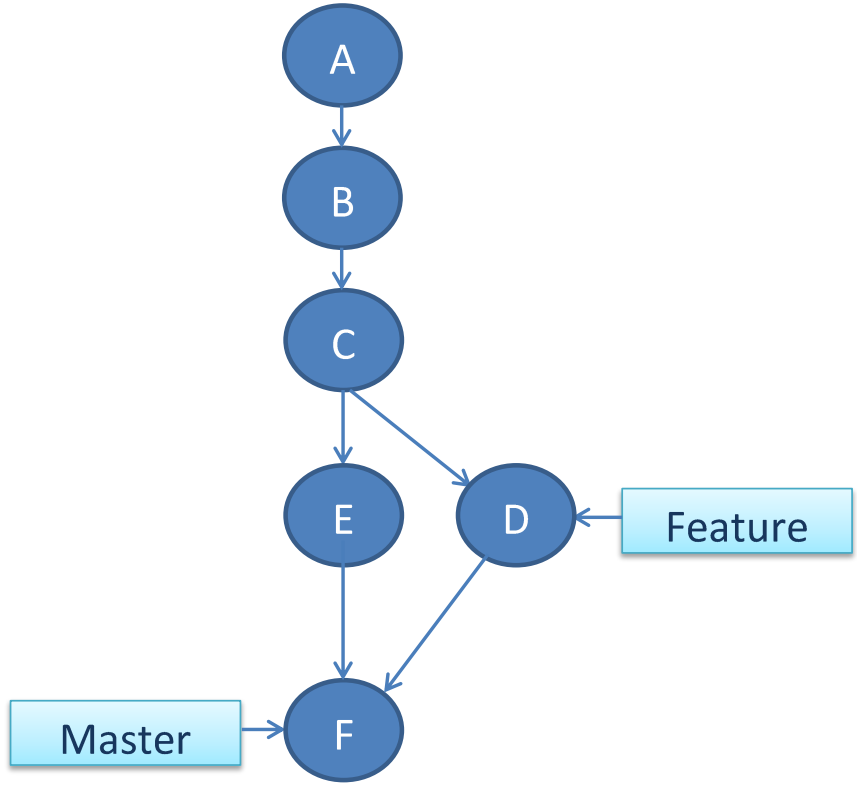
Merging with commits

C2SM Center for Climate Systems Modeling

Conflicting merges also occur

- **git merge**
  - A commit is made for a conflicting merge
  - Conflicts must be resolved before merge is completed
    - Remove conflict markers from conflicted files
    - *git add* conflicted file
    - *git commit* conflicted file

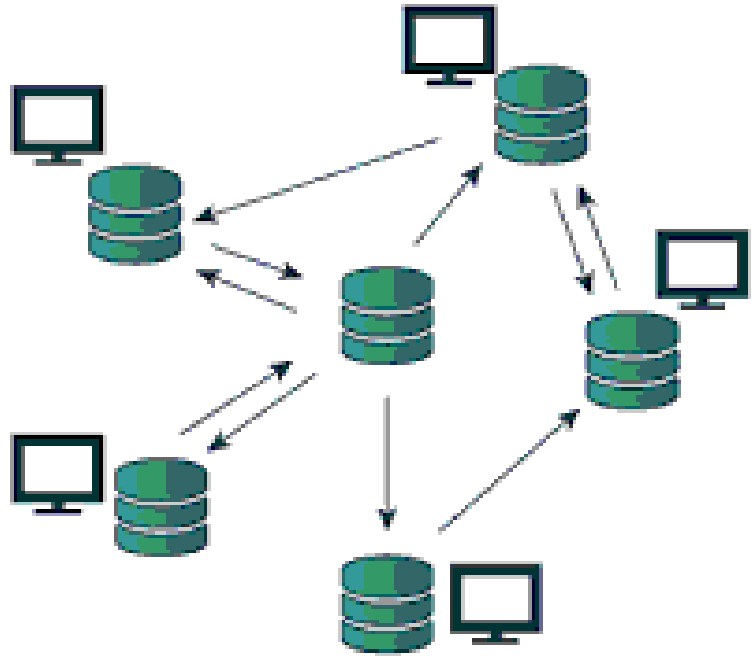
Usage: *git merge branch_name*



Outline

git uses remotes

- Each local repository can connect to multiple remote repositories
- Remotes can be local or across a network
- Remotes can be read-only or read-write access
- Workflow must be clearly defined from the beginning



Source: http://thepilcrow.net/explaining-basic-concepts-git-and-github

How to link to a remote

C2SM Center for Climate Systems Modeling

Commands for connecting to and examining remotes:

- **git remote**
  - Lists all of the remote repositories
  - Using -v option lists all the remote repositories and their paths

  Usage: *git remote (-v)*

- **git remote add**
  - Connects an existing repository with a remote one

  Usage: *git remote add remote_name /path_to_remote*

- **git remote show**
  - Displays detailed information about the selected remote
  - Lists branches in remote repository and how they are linked to the local repository

  Usage: *git remote show remote_name*

How to get code from a remote

C2SM Center for Climate Systems Modeling

Commands for exchanging code with remotes:

- **git fetch**
  - Updates data in remote branches of local repository
  - Can then inspect and/or merge this data into local branches

  Usage: *git fetch remote_name*

- **git pull**
  - Updates data in remote branches of local repository
  - Automatically merges remote data into local branches
  - git pull = git fetch + git merge

  Usage: *git pull remote_name*

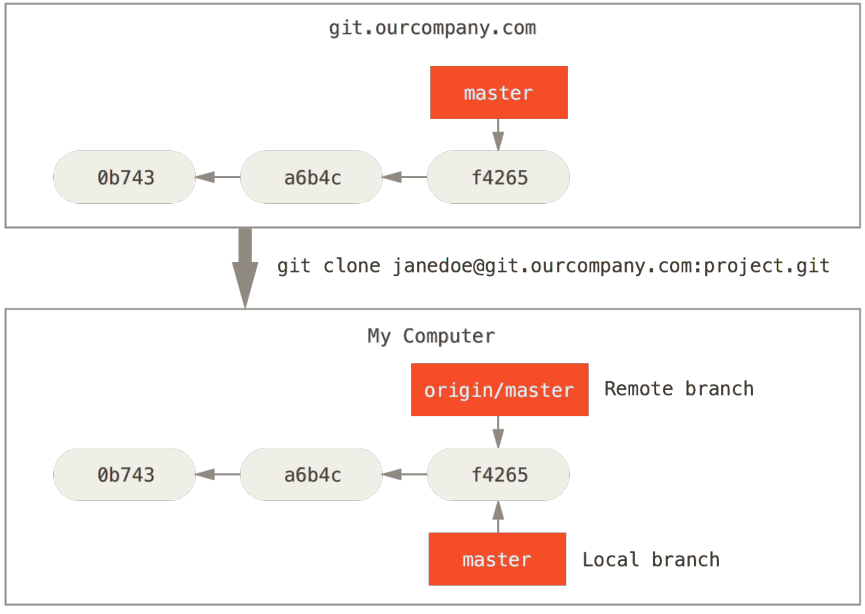
- **git push**
  - Sends changes into remote repository
  - Must do a git fetch and git merge first, to make sure that the local branch is up to date with the remote

  Usage: *git push remote_name branch_name*

Remote branches

Git uses remote branches to track changes to remote repositories

- Branches in the local repository containing data from remotes
- Can be displayed using *git branch -a*
- Created during git clone automatically
- Take the form *remote_name/branch_name*



Source: https://git-scm.com/book/en/v2/Git-Branching-Remote-Branches

Demonstration

C2SM
Center for Climate
Systems Modeling

Outline

C2SM
Center for Climate
Systems Modeling

Github Web interface for hosting remotes

C2SM Center for Climate Systems Modeling

- Web services host remote repositories (can be public or private)
- Provide interfaces for visualizing repositories
- Support collaboration and good coding practices
- Can also edit files and make commits there

github
SOCIAL CODING

Source: https://github.com/logos

Permissions

C2SM Center for Climate Systems Modeling

There are three different levels of permissions for accessing the code repositories on Github:

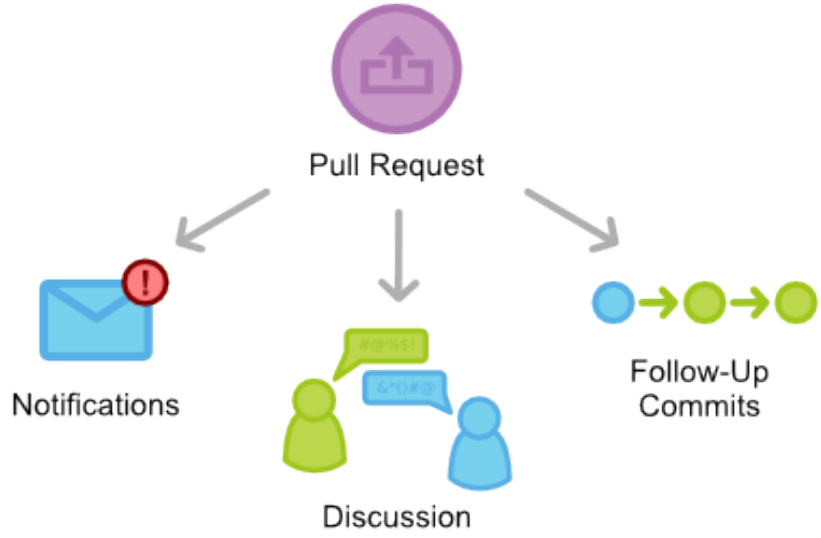
1. Owners
   - Two or three people only
   - Have complete control over the code repositories
   - Can create or delete repositories, add users, and write to every repository
2. Admins
   - One or two people for each repository (admin-*codename*)
   - Have write and read access to their assigned repository
   - Add new versions of code and incorporate new features and bug fixes
3. Users
   - Everyone who is not an owner or admin
   - Have read access to all of the central repositories

Pull Requests

C2SM Center for Climate Systems Modeling

Pull requests are used to review code before merging new features into the main codebase

- Request for changes from a feature branch to be put into central repository
- Generated through web interface (not command line)
- Can be merged using web interface if no merge conflicts exist
- Web interface facilitates review of and commenting on code before pull request is granted



Source: https://www.atlassian.com/zh/git/workflows#!pull-request

Issue tracking

C2SM Center for Climate Systems Modeling

We use the issue tracker on Github as an organizational tool

- Issue trackers allow you to keep track of known bugs, desired features, and other to-do items for the code
- Issues can be assigned to a specific person
- Other users can subscribe to be notified when known issues are resolved
- Issues can be color-coded and labeled so they are easily filtered
- Anybody with access to the repository can comment on issues

Cosmo-prerelease repository workflow

C2SM Center for Climate Systems Modeling

1. Copy the repository to your local machine *(git clone)*
2. Make a feature branch for your own development *(git branch)*
3. Make changes to your local repository following the local Git workflow
4. Save the changes to the cosmo-prerelease repository *(git push)*
5. Make a pull request to start the code review process *(Generate pull request using web interface)*
6. Test the branch using the automated testing program Jenkins *('launch jenkins' command in pull request comments)*
7. Once tests have passed and code has been reviewed, the code owner will merge the pull request *(Merge pull request using web interface)*

Demonstration

C2SM
Center for Climate
Systems Modeling

Outline

C2SM
Center for Climate
Systems Modeling

Best Practices

C2SM Center for Climate Systems Modeling

Some best practices when working with Git:

- Choose a workflow at the beginning of a project and stick with it
  - Where will development of new features occur? (branches, forks)
  - What is the naming convention for branches and forks?
  - Who is responsible for the central repository?
  - How will the code review/pull request process work?
- Review code in staging area before committing it
- Commit small logical changes
- Make useful commit messages that can be understood by anyone
  - First line of message should be a one line summary
  - Details of commit follow the summary
- Keep repository clean - remove unused/finished branches

Git Resources

C2SM Center for Climate Systems Modeling

- **git help**
  - Displays the man page for the given command
  - Displays general git information when no command name is given

  Usage: *git help (command_name)*
- **http://git-scm.com/**
  - Comprehensive description of Git commands and concepts
- **http://gitref.org/**
  - Quick reference guide for commands
- **https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf**
  - Cheat sheet with Git commands

Graphical git tools

C2SM Center for Climate Systems Modeling

Built-in git graphical tools

- **gitk**
  - Displays changes in a repository; information about commits
  - Cannot be used to make commits
- **git-gui**
  - Used for making changes to a repository
  - Can commit,branch, merge, and interact with remotes
  - Does not show code history

Third party git graphical tools

- **Source Tree**
  - Can commit, branch, merge, and interact with remotes
  - Can view history and commits
  - Only for OS: Mac and Windows
- **SmartGit**
  - Can commit, branch, merge, and interact with remotes
  - Can view history and commits
  - OS: Mac, Linux, and Windows
  - http://www.syntevo.com/smartgit/