Git Primer

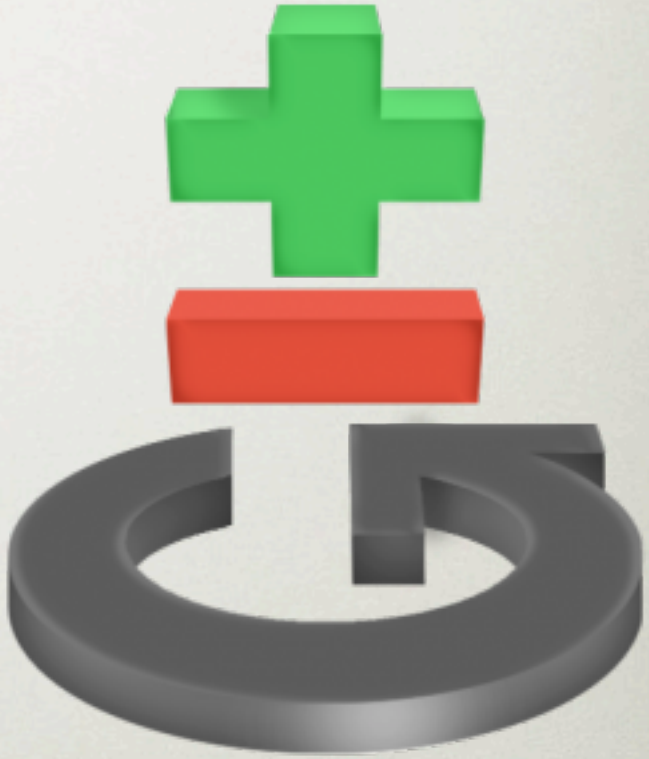
Justin Obara,
Inclusive Design Research Centre
OCADU

Outline

- What is git
- git workflow
- github
- UI's and other tools
- Resources

What is git?

- distributed version control system
- every clone is a full-fledged repository
- branching and merging are designed to be fast and easy



git Workflow: Ready, Get Set Go!

git Workflow: Ready

- Install git (it doesn't come by default)
- Setup your git config file

git Workflow: Ready

Config:
configuration for git

Command:
git config --global user.name “First Last”
git config --global user.email “email”

For line endings see:
http://help.github.com/dealing-with-lineendings/

git Workflow: Ready

Command:

```
git config --global color.diff auto
git config --global color.status auto
git config --global color.branch auto
git config --global color.interactive auto
git config --global alias.st status
git config --global alias.ci commit
git config --global alias.co checkout
git config --global alias.br branch
git config --global core.excludesfile ~/.gitignore
echo .DS_Store >> ~/.gitignore
```

git Workflow: Get

Clone:
import/fork a repository

Command:
git clone “repo url”

git Workflow: Get

Fetch:
retrieve updates
from the parent repo

Command:
git fetch

Pull:
retrieve and merge
updates from parent
repo

Command:
git pull

git Workflow: Set

Branch:
a virtual or mini fork

Command:
git branch <-- list branches
git branch “branch name” <-- create
git checkout “branch name” <-- switch to
git merge “branch name” <-- merge
git branch -d “branch name” <-- delete

git Workflow: Set

Tag:
an alias to a specific commit

Command:
git tag <-- list tags
git tag -a “tag name” <-- create tag at HEAD
git tag -a “tag name” “commit”<-- create tag at commit

git Workflow: Go

Log:
displays the logs of the commits

Command:
git log <-- displays logs
git log -p <-- displays logs with patch
git log “br1” ^”br2” <-- logs changes that are in br1 but not br2

git Workflow: Go

Diff:
displays the logs of the commits

Command:
git diff <-- of unstaged changes
git diff --cached <-- of staged changes
git diff HEAD <-- with latest commit
git diff “commit <-- with specified commit
git diff “br1” “br2” <-- diff branches

git Workflow: Go

Status:

Status of files in git repo (e.g. modified, unknown, added)

Command:

git status <-- verbose status of files

git status -s <-- short status

git Workflow: Go

Add:
stages/prepares files for commit

Command:
git add “file name” <-- stages file
git add “directory” <-- stages files
git add -u <-- stages files that are tracked

git Workflow: Go

Commit:
Commits to local repo.

Format: 50 char summary, empty new line, body

Command:
git commit <-- commit staged files
git commit -a <-- stages and commits tracked files

git Workflow: Go

Push:

sends commits up to the remote repo

Command:

git push <-- pushes latest commits

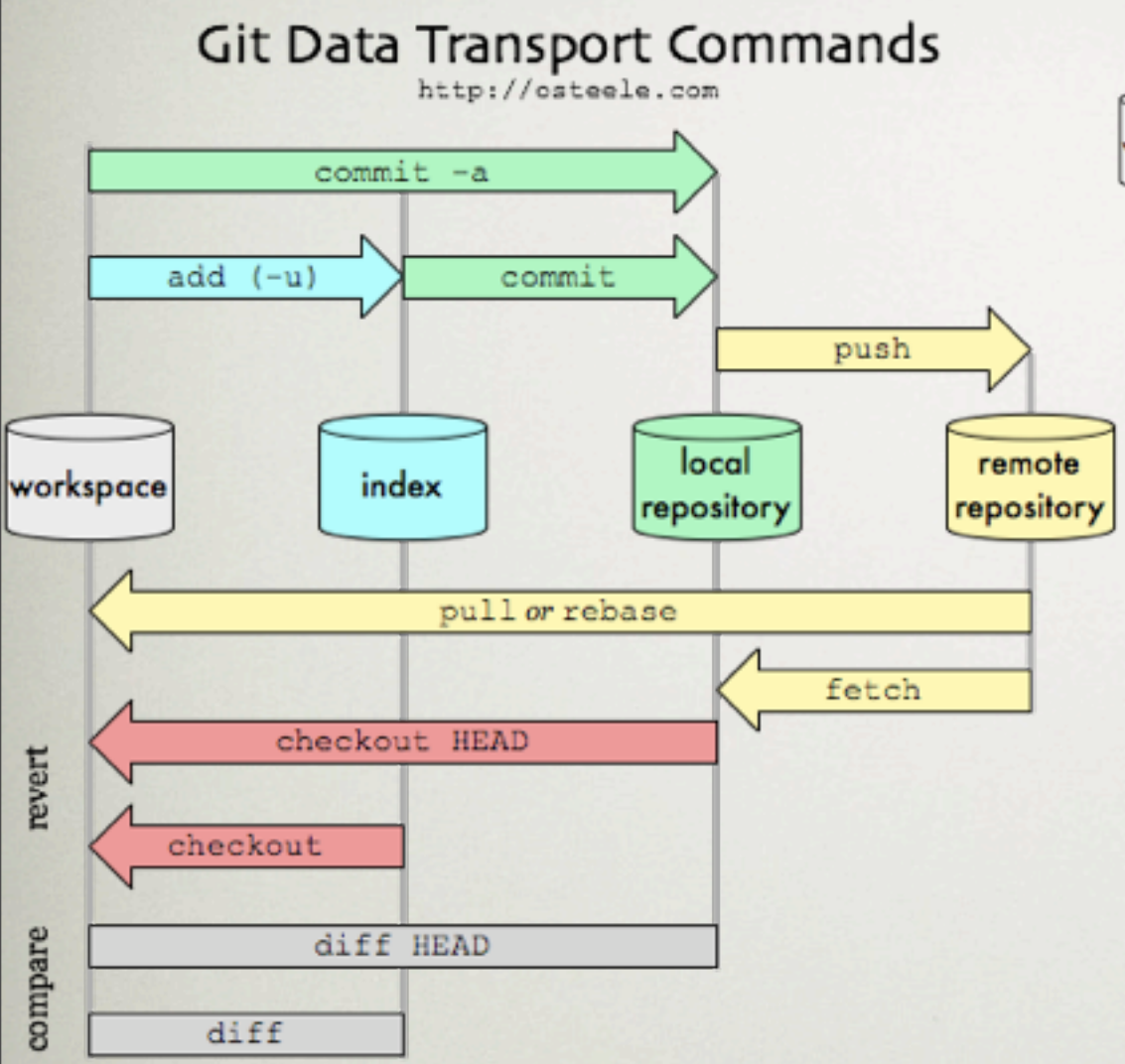
git push “repo” <-- pushes to specified remote repo

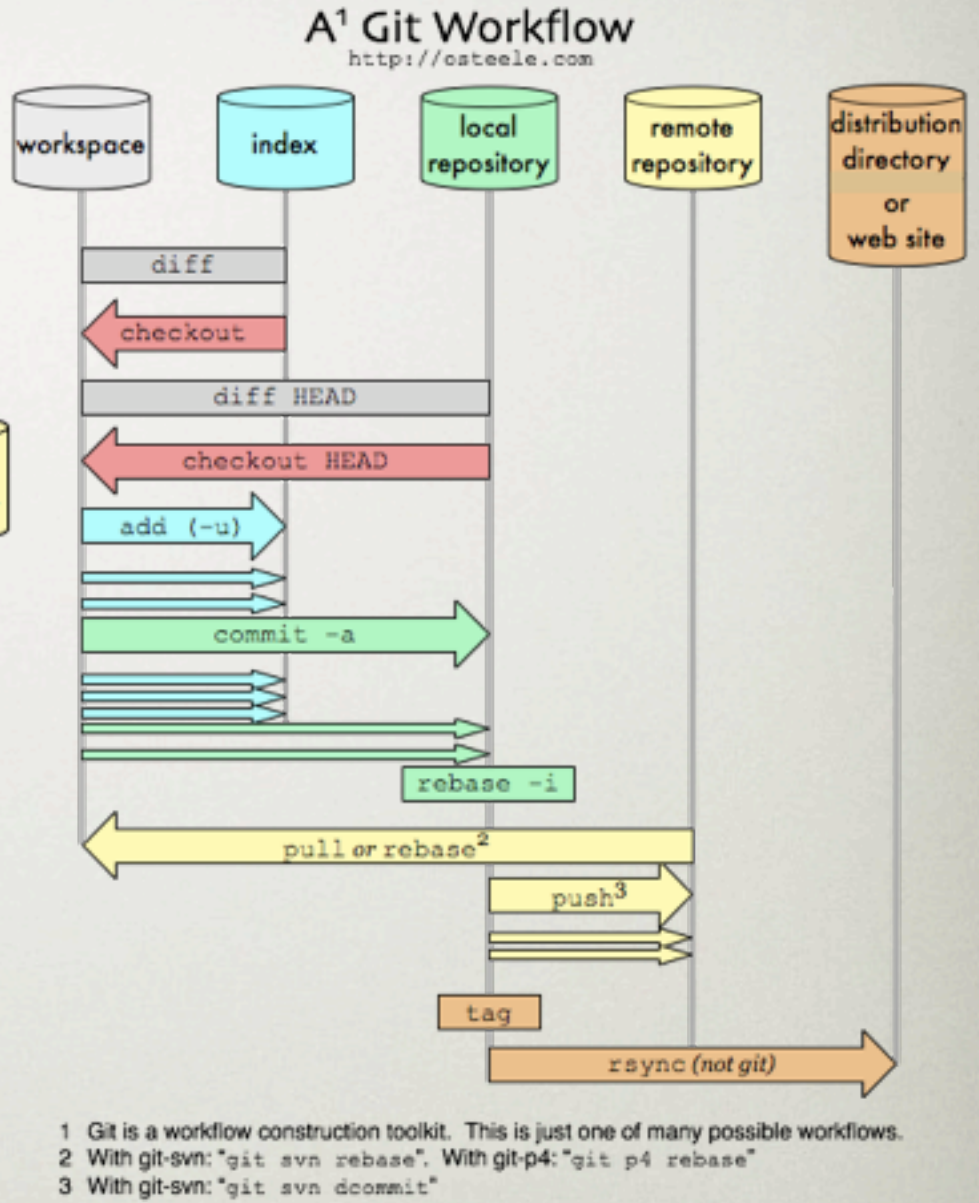
git push “repo” “branch” <-- pushes specified branch

git push “repo” “tag” <-- pushes tag

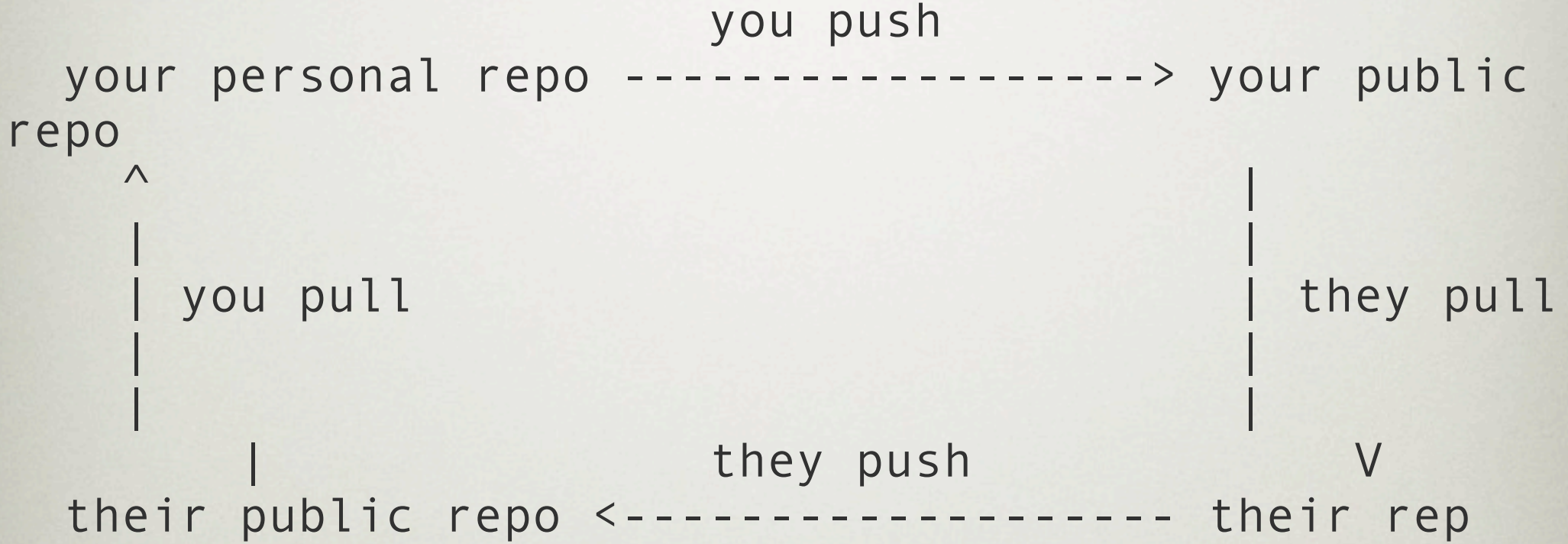
git push “repo” --tags <-- pushes all tags

git push “repo” “branch” --tags <-- pushes all tags for a branch





Diagrams from: http://osteele.com/archives/2008/05/commit-policies



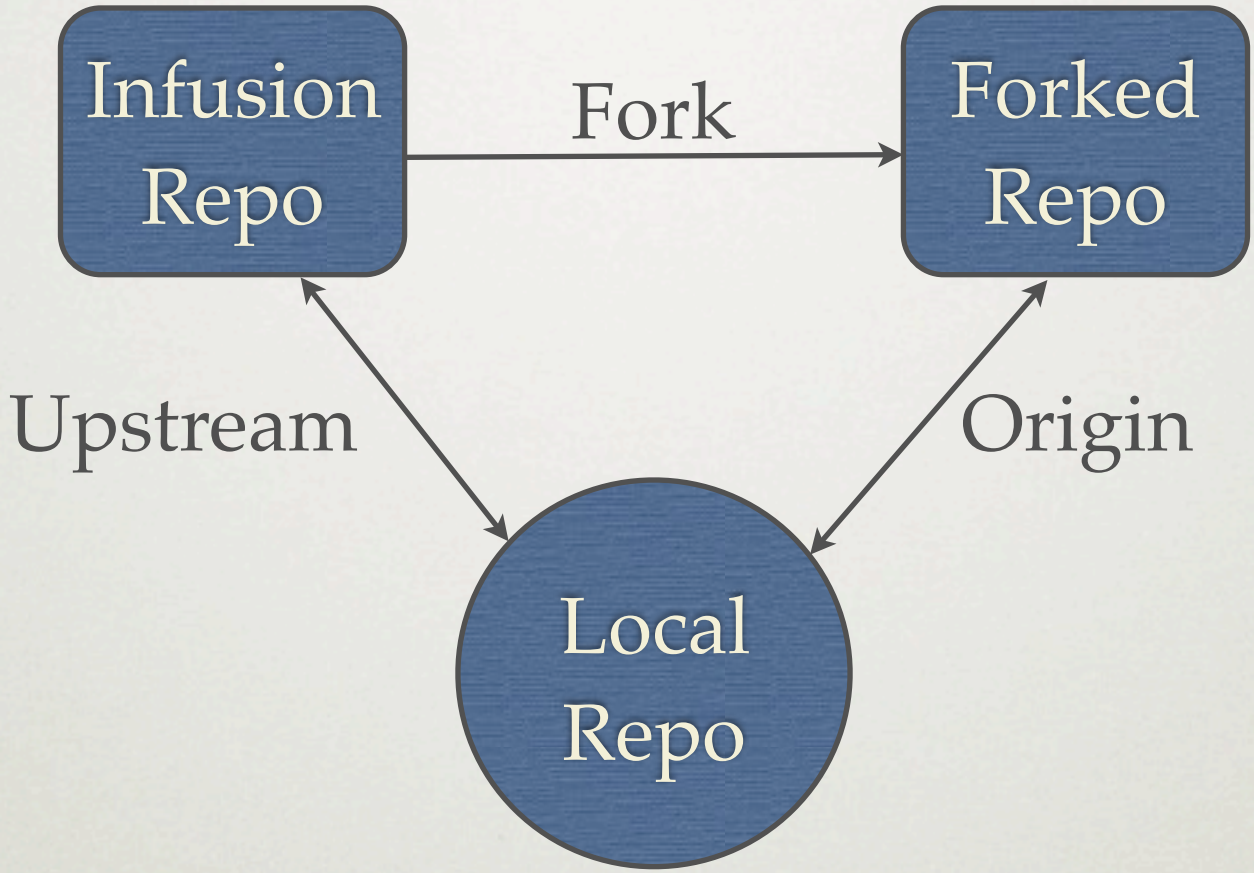
Diagrams from: http://book.git-scm.com/3_distributed_workflows.html

git Workflow: Remotes

Remote:
Remote repos, “origin” created automatically during clone

Command:
git remote add “repo alias” “repo url” <-- verbose status of files

git Workflow: Remotes



See: http://help.github.com/forking/

git Workflow: Gotchas

- stage changes (**git add**), even already tracked files
- **git rm** removes files from versioning and deletes them
- use **git add -u** to stage deleted files that are still under versioning
- push (**git push**) after committing
- branches are garbage collected. If you delete it, it will be gone **forever**
- make sure to **always** use **git tag -a** when creating tags, or else the proper meta data won't be added.

git Workflow: Gotchas

- tags and branches aren't automatically pushed to the remote repository.
- you delete remote branches and tags by pushing nothing to them
  - **git push "repo" :"branch"**

GITHUB

UI's and other tools

- egit: http://www.eclipse.org/egit/?gclid=CKuM64aBsKYCFYa7Kgod2zVanQ
- smartGit: http://www.syntevo.com/smartgit/index.html
- mac autocomplete: http://repo.or.cz/w/git.git/blob/HEAD:/contrib/completion/git-completion.bash
- review of mac clients: http://shiningthrough.co.uk/Mac-OS-X-Git-Clients-Roundup
- list of all git tools: https://git.wiki.kernel.org/index.php/Interfaces,_frontends,_and_tools

Resources

- The Git Community book: http://book.git-scm.com/
- Pro Git (book): http://progit.org/
- Git Reference: http://gitref.org/index.html
- git Docs: http://www.kernel.org/pub/software/scm/git/docs/
- github help: http://help.github.com/
- cheat sheets: https://github.com/AlexZeitler/gitcheatsheet/raw/master/gitcheatsheet.pdf
- Others: http://old.nabble.com/some-git-resources-to30829999.html