

# Git: Remotes, Branching, and Rebasing

Seth House <seth@eseth.com>

Cars.com

2012-04-27

# Remotes

## Viewing remotes

```
% git remote -v  
origin  git+ssh://git@git.myremote.com/myrepo
```

## Adding remotes

On a remote machine:

```
git init --bare ~/myrepo.git
```

Then add the new remote:

```
git remote add myremote git+ssh://myhost/myrepo
```

## Mirroring a repo

```
git push --mirror myremote
```

or:

```
git push myremote \
  refs/remotes/myremote/*:refs/heads/*
```

# Remote branches

```
git branch -r
```

## Remote-tracking branches

Create a new branch that tracks the remote:

```
git checkout --track origin/somebranch
```

Or for existing branches:

```
git branch --set-upstream somebranch\  
origin/somebranch
```

Or for new branches:

```
git push -u origin mynewbranch
```

## Remote-tracking shorthand

```
git rev-parse --short @{upstream}  
git rev-parse --short @{u}
```



# Viewing differences with upstream

## Overview:

`git status`

## Incoming changesets:

`git log --oneline --decorate @{u}`

## Outgoing changesets:

`git log --oneline --decorate @{u}..`

## All local branches:

`git branch -v`

## Viewing differences in your prompt

```
wget https://raw.githubusercontent.com/git/git/master/\
contrib/completion/git-completion.bash
```

Add this to your `~/ .bashrc`:

```
source ~/path/to/git-completion.bash
GIT_PS1_SHOWUPSTREAM="verbose" # or "auto"
```

Add `$(__git_ps1 " (%s) ")` to your `PS1`:

```
PS1='\u@\h:\W$(__git_ps1 " (%s) ") \ $ '
```

## git pull considered harmful

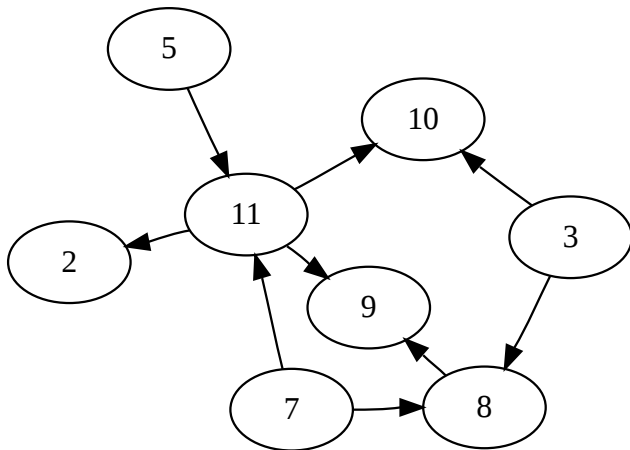
```
git fetch  
git merge @{u}
```

Or:

```
git fetch  
git rebase @{u}
```

# Branching

# The DAG



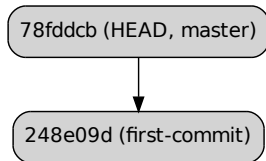
## refs/heads

```
% cat .git/refs/heads/master  
6bf4e7278d0cd3301ac40874d6aca6636c21975d
```

# Rebasing

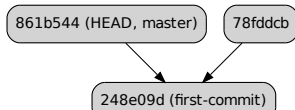
# commit --amend

```
touch A && git add A
git commit -m "Added A"
git tag first-commit
touch B && git add B
git commit -m "Added B"
```



---

```
git commit --amend \
-m "Added B and stuff"
```





# Interactive

```
% git status -s -b  
## mybranch...origin/develop [ahead 7]  
% git rebase -i @{u} # or git rebase -i HEAD~7
```

# Upstream

```
git fetch  
git rebase @{u}
```

## rebase --onto

```
git rebase --onto newbranch oldbranch branchname
```

## reset --soft

- 1 Moves the ref HEAD points to.

## reset [--mixed]

- 1 Moves the ref HEAD points to.

## reset [--mixed]

- 1 Moves the ref HEAD points to.
- 2 Updates the index to match HEAD.

## reset --hard

- 1 Moves the ref HEAD points to.

# reset --hard

- 1 Moves the ref HEAD points to.
- 2 Updates the index to match HEAD.



## reset --hard

- 1 Moves the ref HEAD points to.
- 2 Updates the index to match HEAD.
- 3 Updates the working directory to match HEAD.

# The reflog

```
git reflog --date=relative
```

## Garbage collection

- Git garbage collects objects with no references that are older than 30 days.

# Garbage collection

- Git garbage collects objects with no references that are older than 30 days.
- The reflog counts as a reference.

# Garbage collection

- Git garbage collects objects with no references that are older than 30 days.
- The reflog counts as a reference.
  - **reflog entries are pruned after 90 days.**