git™ Quick Reference

COMMAND OVERVIEW

Concepts and Definitions

repository a project tracked by Git, consisting of commits & branches, usually stored with project files and directories in a working directory

working directory aka. working tree or workspace, the directory containing a working copy of project files and directories

index aka. cache or stage, staging area for building a commit of changes in the working directory

commit history a database storing past commits

commit a snapshot or record of changes to files in the working directory at some point in time

branch a reference to a commit at the end of a chain of commits

HEAD a reference to the commit that is currently checked out

merge a commit joining divergent development paths or branches

merge conflict a condition that arises from a failed automatic merge; requires manual editing to resolve the conflict

Ref Notation

HEAD	Reference to the commit currently checked out
ref	placeholder for branch, tag, or commit SHA-1 hash
ref^*n*	the *n*th parent of *ref*, where *n*=1 when omitted (only merge commits have multiple parents)
ref~*n*	the *n*th ancestor of *ref*, where *n*=1 when omitted
ref@{*n*}	the *n*th reflog entry of *ref*

Examples:

HEAD^	denotes parent of HEAD
master~3	great grandparent of the latest commit on master
HEAD~5^2	HEAD's great-great-great grandparent's 2nd parent
HEAD@{1}	previous value of HEAD
0c708f	Refers to a commit by its SHA-1 hash (unique ID)

Initial Setup

```
git config --global user.name "Foo Bar"
git config --global user.email
            "foo.bar@example.com"
ssh-keygen -t rsa
cat ~/.ssh/id_rsa.pub
```

Then copy and paste the output to your SSH keys on the remote server.

Creating a New Repository

```
mkdir myrepo
cd myrepo
git init
# create or add files
echo "hello" > foo.txt
git add .
git commit -m "initial commit"
```

Push Existing Repo to Remote

```
git remote add origin remote-repo
git push --all –u origin
```

Downloading a Repository

git clone *remote-repo*

where *remote-repo* is a path of the form **user@server:/path/to/repo**

Viewing Changes

git status	View list of changed files
git diff	View changes to files in the working directory
git diff --cached	View changes in index from HEAD commit

Committing Changes

git add *file*	Add changes in *file* to index
git commit	Commit staged changes in the index to the local repo

git commit *file*

Same as above two commands, except *file* must already be tracked

To commit all changes to tracked files and new or removed files:

```
git add --all
git commit -m "commit message"
```

Commit all changes (to tracked files only):

```
git commit -a -m "commit message"
```


HOFSTRA UNIVERSITY

git™ Quick Reference

COMMAND OVERVIEW

Branches

git branch *branch*
Create new branch named *branch* at the HEAD (current commit)

git checkout *branch*
Check out (i.e. switch to) *branch*

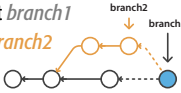
git checkout -b *branch*
Same as above two commands, i.e. create new branch named *branch* at current commit and check it out

git branch -d *branch*
Delete branch named *branch*

Merging

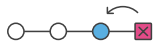
To merge *branch2* into *branch1*:

git checkout *branch1*
git merge *branch2*

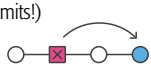


Undoing Commits

git reset *commit*
Rewind current branch to *commit*, e.g. HEAD^ (never do this on published commits!)



git revert *commit*
Does not do what you would think it does – creates a new commit to undo changes of a previous commit



Viewing History

git log	List commit history of the current branch
git log --oneline	Show one per line
git log --follow *file*	Show history of *file*
git show *ref*	View changes in commit
git blame *file*	See who changed what (and when) in given file
git diff *A…B*	Compare two branches

Rebase

Doing a rebase sequentially regenerates a series of commits onto another branch.

git checkout *B*
git rebase *A* Rebase branch *B* onto *A*

git rebase --onto *A C* [*B*]
Rebase branch *B* starting at commit *C* onto branch *A*. If *B* isn't specified, rebase up to and including HEAD.

Pushing and Pulling

git push	Upload commits to default upstream remote repository (To set default upstream: **git push -u** *remote branch*)
git push *remote branch*	Push new commits on *branch* to *remote*, e.g. **git push origin master**
git pull	Pull latest changes from origin (does fetch & merge)
git pull *remote branch*	Pull latest commits on *branch* from *remote*

Restoring Files

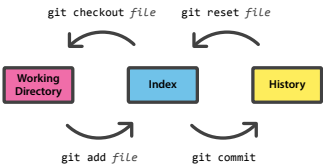
git checkout *commit* **--** *file*
Restore *file* from the given commit

git checkout HEAD -- *file*
Discard uncommitted changes to *file*

git reset --hard HEAD
Discard all uncommitted changes

Staging Files

git add *file*	Add changes in *file* to index
git reset *file*	Unstage *file*, i.e. remove *file* from index, e.g. to keep it from being committed when you do **git commit**



Resolving Merge Conflicts

git status	List the files with conflicts
vim *file*	Edit files to fix conflicts…

problematic areas are marked as follows:

```
<<<<<<< HEAD
text changed in current branch
=======
text changed in other-branch
>>>>>>> refs/heads/other-branch
```

…or use a dedicated merge tool:
git mergetool

Then, **git add** *file* to mark each file resolved and finally **git commit** to conclude the merge. Alternatively, run **git merge --abort** to cancel the merge.