Introduction to Git

Version control systems, Git, GitHub

Ian Leifer

Why should I care?

Teams of people can work simultaneously with same files (papers, code, figures etc.)

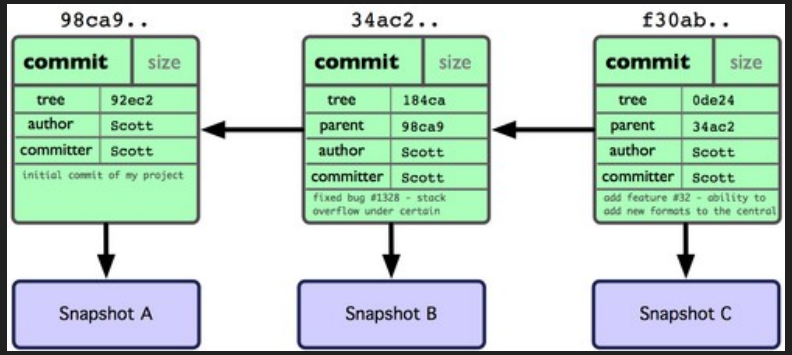
All work is stored and tracked in one place

Access from anywhere with cloud storage

Easy to see past versions for changes and corrections

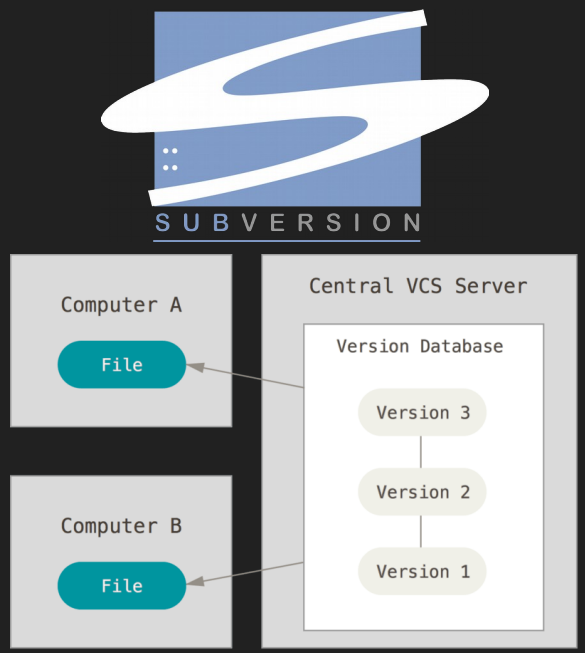
What is VCS

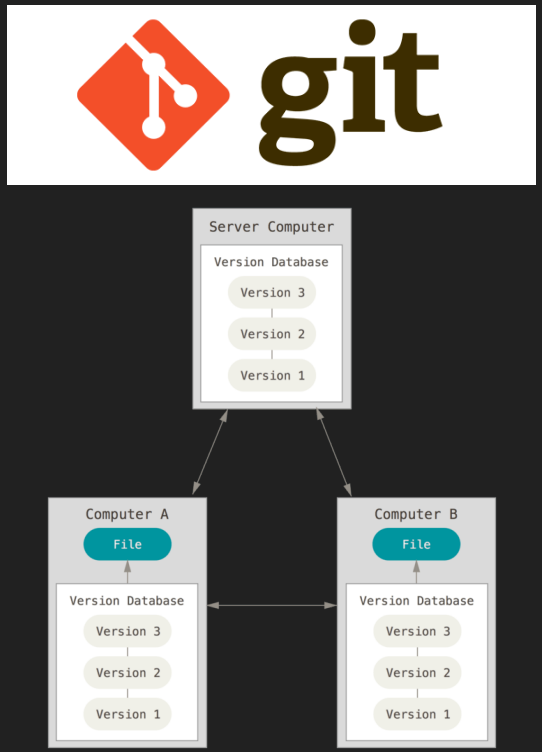
Version control system is a system that records changes to a file or set of files over time so that you can recall specific versions later



VCS: centralized vs distributed

Centralized Version Control Systems (CVCSs)





Distributed Version Control Systems (DVCSs)

Git. File status lifecycle

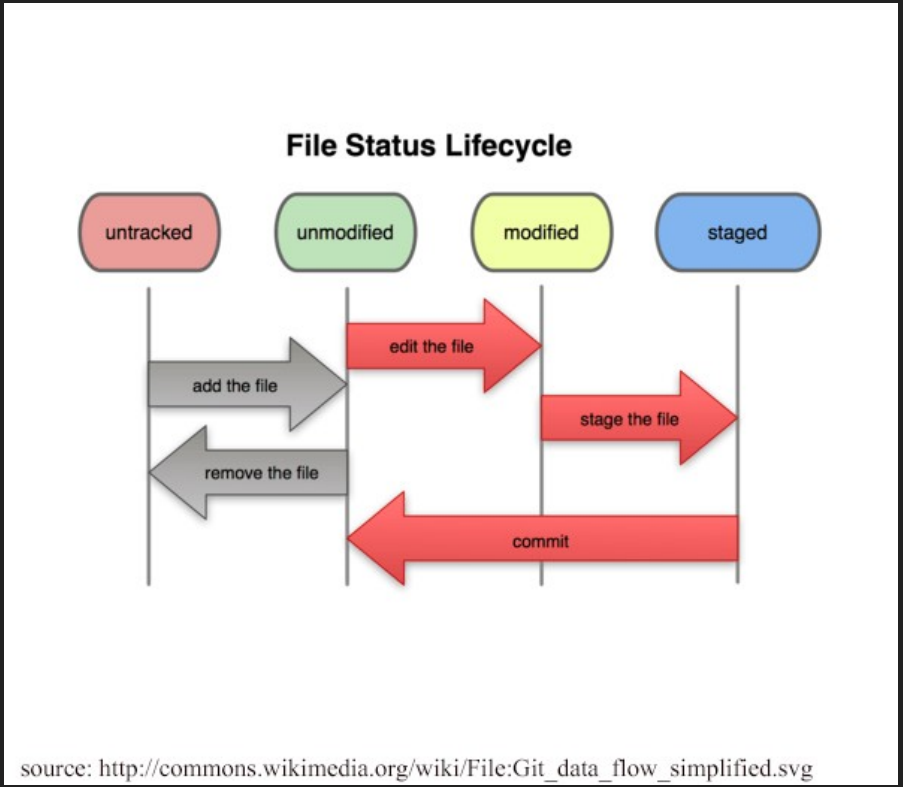
We create new file

We start tracking it

We change it

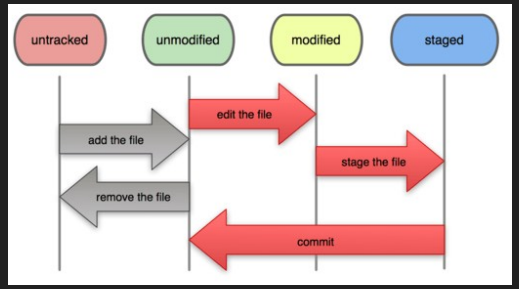
We stage it

We commit it



source: http://commons.wikimedia.org/wiki/File:Git_data_flow_simplified.svg

Example



Initialize →

```
ian@ian-desktop:~/Desktop/GitTalk$ git init
Initialized empty Git repository in /home/ian/Desktop/GitTalk/.git/
ian@ian-desktop:~/Desktop/GitTalk$ git status
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
```

Create new file →

untracked

```
ian@ian-desktop:~/Desktop/GitTalk$ echo "Hello, world" > file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        file.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Start tracking new file →

staged

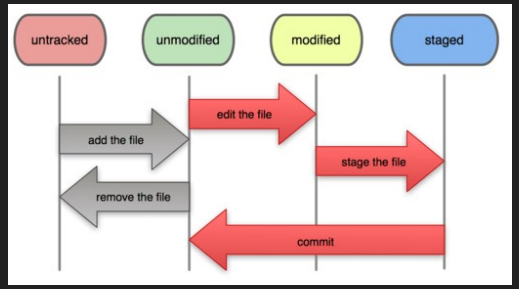
```
ian@ian-desktop:~/Desktop/GitTalk$ git add file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   file.txt
```

Commit your changes →

unmodified

```
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "Initial commit"
[master (root-commit) 326fee0] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git status
On branch master
nothing to commit, working directory clean
```

Example



Initialize

Create new file and commit

unmodified

modified

Stage file and commit

Create new file

untracked

unmodified

Change old file

untracked

modified

staged

modified

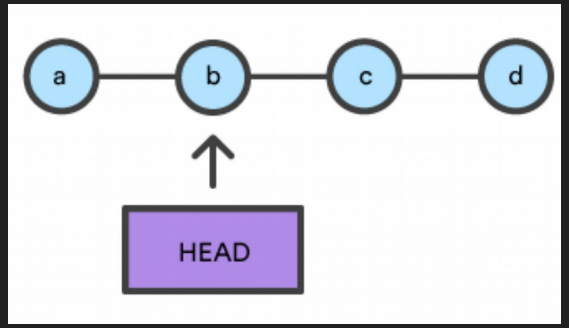
Stage new file and commit

unmodified

modified

```
ian@lan-desktop:~/Desktop/GitTalk$ git init
Initialized empty Git repository in /home/ian/Desktop/GitTalk/.git/
ian@lan-desktop:~/Desktop/GitTalk$ echo "Hello, world!" > file.txt
ian@lan-desktop:~/Desktop/GitTalk$ git add file.txt
ian@lan-desktop:~/Desktop/GitTalk$ git commit -m "Initial commit"
[master (root-commit) aaa72d0] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
ian@lan-desktop:~/Desktop/GitTalk$ echo "Hello, world!!!" > file.txt
echo "Hello, worldgit commit -m "Initial commit"!" > file.txt
ian@lan-desktop:~/Desktop/GitTalk$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   file.txt

no changes added to commit (use "git add" and/or "git commit -a")
ian@lan-desktop:~/Desktop/GitTalk$ git add file.txt
ian@lan-desktop:~/Desktop/GitTalk$ git commit -m "Second commit"
[master a9a988c] Second commit
 1 file changed, 1 insertion(+), 1 deletion(-)
ian@lan-desktop:~/Desktop/GitTalk$ echo "Hello, world!" > file1.txt
ian@lan-desktop:~/Desktop/GitTalk$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        file1.txt

nothing added to commit but untracked files present (use "git add" to track)
ian@lan-desktop:~/Desktop/GitTalk$ echo "Hello, world!" > file.txt
ian@lan-desktop:~/Desktop/GitTalk$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   file.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        file1.txt

no changes added to commit (use "git add" and/or "git commit -a")
ian@lan-desktop:~/Desktop/GitTalk$ git add file1.txt
ian@lan-desktop:~/Desktop/GitTalk$ git commit -m "Second commit"
[master 3630cea] Second commit
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
ian@lan-desktop:~/Desktop/GitTalk$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   file.txt

no changes added to commit (use "git add" and/or "git commit -a")
ian@lan-desktop:~/Desktop/GitTalk$
```

HEAD and git checkout



HEAD is the pointer to the commit we work with right now

Git changes this pointer

Get list of commits

See files in the folder in the current commit

Checkout old commit

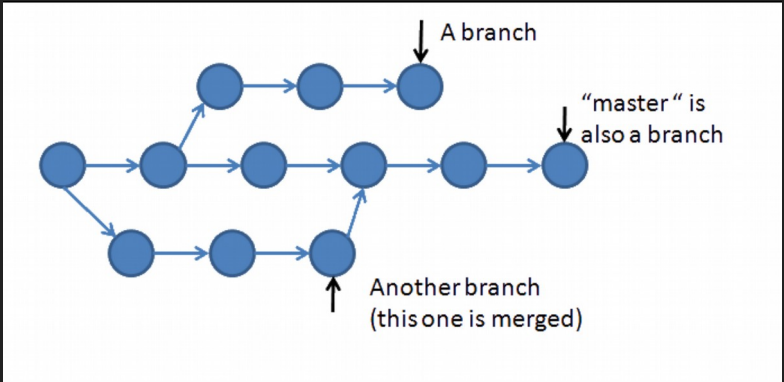
Now folder looks different

```
ian@ian-desktop:~/Desktop/GitTalk$ git log
commit 3630ceab4253a602ba575bd3d2506c2ef3623ef1
Author: Ian Leifer <ianleifer93@gmail.com>
Date:   Fri Jun 14 18:33:11 2019 -0400

    Second commit

commit a9a988cfa3901d8569e6a2a3bff7fc9afc16152e
Author: Ian Leifer <ianleifer93@gmail.com>
Date:   Fri Jun 14 18:32:01 2019 -0400

    Second commit

commit aaa72d042259e68c86219a12fec7e2d67b9d63be
Author: Ian Leifer <ianleifer93@gmail.com>
Date:   Fri Jun 14 18:31:06 2019 -0400

    Initial commit
ian@ian-desktop:~/Desktop/GitTalk$ ls
file1.txt  file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git checkout aaa72d042259e68c86219a12fec7e2d67b9d63be
Note: checking out 'aaa72d042259e68c86219a12fec7e2d67b9d63be'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b <new-branch-name>

HEAD is now at aaa72d0... Initial commit
ian@ian-desktop:~/Desktop/GitTalk$ ls
file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git checkout 3630ceab4253a602ba575bd3d2506c2ef3623ef1
Previous HEAD position was aaa72d0... Initial commit
HEAD is now at 3630cea... Second commit
ian@ian-desktop:~/Desktop/GitTalk$ ls
file1.txt  file.txt
ian@ian-desktop:~/Desktop/GitTalk$
```

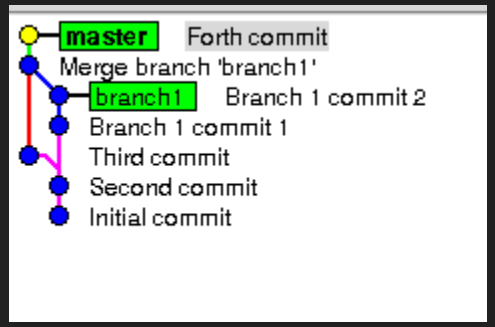
Git. Branches and merging

When a team or a person is working on multiple features in parallel appears need for branches

After work on a feature is done it is usually merged back to main branch (master)



Example



Initialize →

Create couple of commits on master →

Create new branch →

Create couple of commits in new branch →

Work on master →

Merge branch to master →

Keep working on master →

```
ian@ian-desktop:~/Desktop/GitTalk$ git init
Initialized empty Git repository in /home/ian/Desktop/GitTalk/.git/
ian@ian-desktop:~/Desktop/GitTalk$ echo "V" > file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git add file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "Initial commit"
[master (root-commit) 9d45832] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
ian@ian-desktop:~/Desktop/GitTalk$ gitk --all
ian@ian-desktop:~/Desktop/GitTalk$ echo "V!" > file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git add file.txt
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "Second commit"
[master c645cee] Second commit
 1 file changed, 1 insertion(+), 1 deletion(-)
ian@ian-desktop:~/Desktop/GitTalk$ git branch branch1
ian@ian-desktop:~/Desktop/GitTalk$ git checkout branch1
Switched to branch 'branch1'
ian@ian-desktop:~/Desktop/GitTalk$ echo "V!" > file1.txt
ian@ian-desktop:~/Desktop/GitTalk$ git add file1.txt
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "Branch 1 commit 1"
[branch1 ad54848] Branch 1 commit 1
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
ian@ian-desktop:~/Desktop/GitTalk$ echo "V!" > file2.txt
ian@ian-desktop:~/Desktop/GitTalk$ git add file2.txt
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "Branch 1 commit 2"
[branch1 ca7d0fa] Branch 1 commit 2
 1 file changed, 1 insertion(+)
 create mode 100644 file2.txt
ian@ian-desktop:~/Desktop/GitTalk$ git checkout master
Switched to branch 'master'
ian@ian-desktop:~/Desktop/GitTalk$ echo "V" > file0.txt
ian@ian-desktop:~/Desktop/GitTalk$ git add file0.txt
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "Third commit"
[master 16ad416] Third commit
 1 file changed, 1 insertion(+)
 create mode 100644 file0.txt
ian@ian-desktop:~/Desktop/GitTalk$ gitk --all
ian@ian-desktop:~/Desktop/GitTalk$ git merge branch1
Merge made by the 'recursive' strategy.
 file1.txt | 1 +
 file2.txt | 1 +
 2 files changed, 2 insertions(+)
 create mode 100644 file1.txt
 create mode 100644 file2.txt
ian@ian-desktop:~/Desktop/GitTalk$ gitk --all
ian@ian-desktop:~/Desktop/GitTalk$ echo "VC
> ^C
ian@ian-desktop:~/Desktop/GitTalk$ echo "VC" > file0.txt
ian@ian-desktop:~/Desktop/GitTalk$ git add file0.txt
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "Forth commit"
[master e83e800] Forth commit
 1 file changed, 1 insertion(+), 1 deletion(-)
```

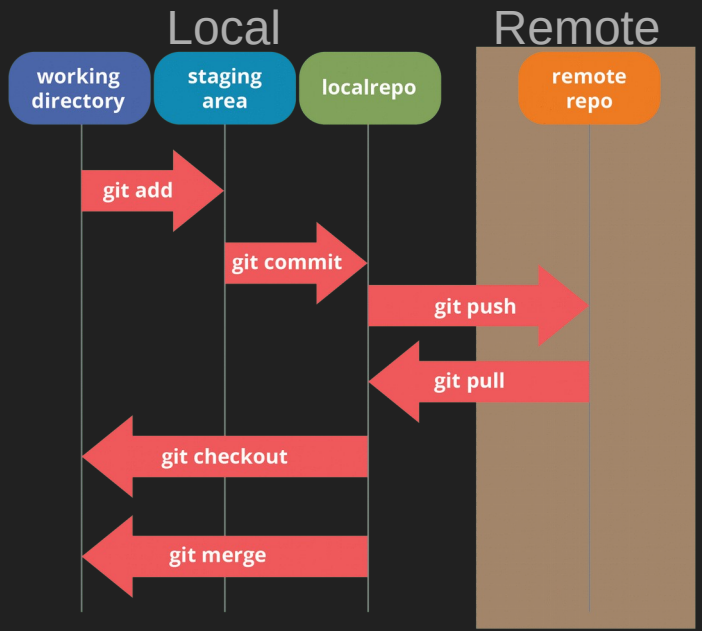
Working with remote repositories

We want to work with others, so we need cloud repositories

Each local Git repository can have few remote repositories

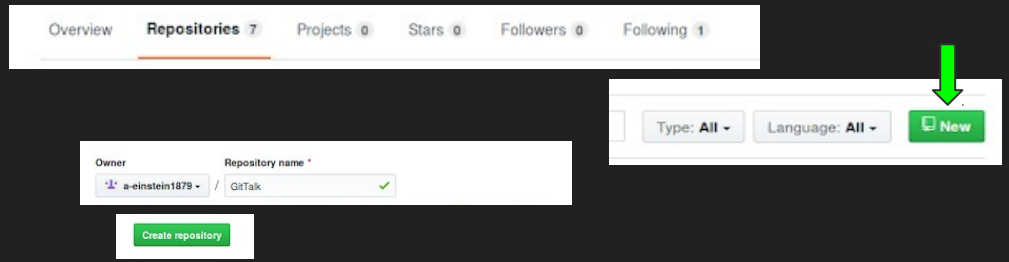
We can push our work and others can fetch changes

Github



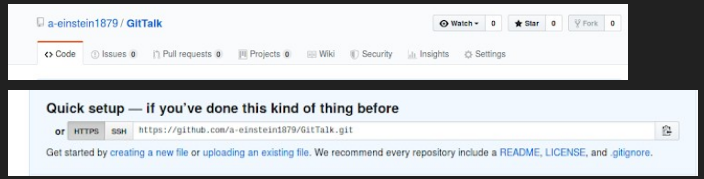
Example. Uploading code to GitHub repository

1. Create repository
   a. Go into repositories in your account
   b. Push “New”
   c. Add name
   d. Push “Create repository”



1. Go to your the repository

2. Copy address from Quick setup line

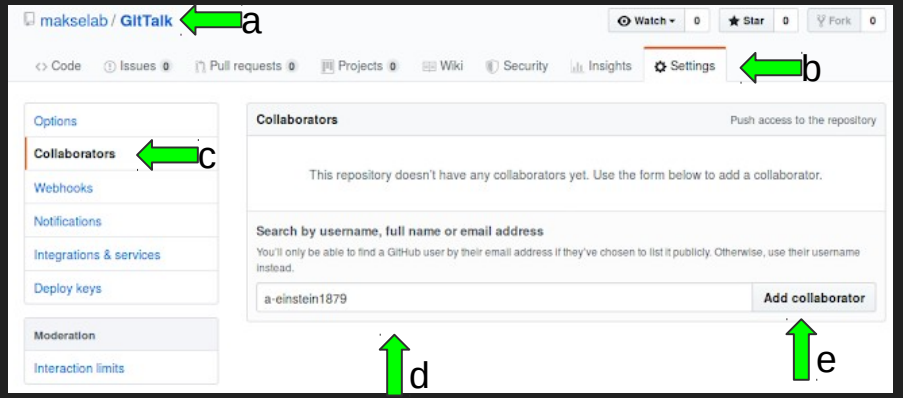


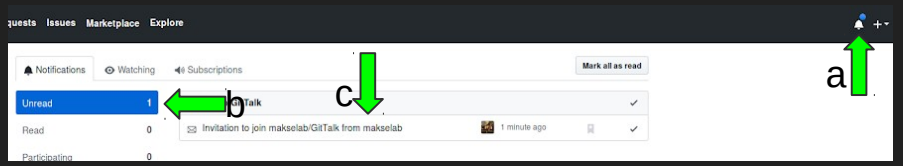
3. Add new remote repository named “origin” with address above
4. Push master branch of your local repository to remote repository named origin

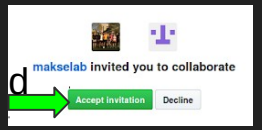
```
ian@ian-desktop:~/Desktop/GitTalk$ git remote add origin https://github.com/a-einstein1879/GitTalk.git
ian@ian-desktop:~/Desktop/GitTalk$ git push -u origin master
Username for 'https://github.com': a-einstein1879
Password for 'https://a-einstein1879@github.com':
Counting objects: 17, done.
Delta compression using up to 12 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (17/17), 1.33 KiB | 0 bytes/s, done.
Total 17 (delta 5), reused 0 (delta 0)
remote: Resolving deltas: 100% (5/5), done.
To https://github.com/a-einstein1879/GitTalk.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
ian@ian-desktop:~/Desktop/GitTalk$
```

Example. Uploading files to someone's Github

1. Owner creates repository (see point 1 on previous slide for details)
2. Owner gives rights to work with repository
   a. Go into repository
   b. Settings
   c. Collaborators
   d. Put collaborator's GitHub nickname and choose from dropdown
   e. Push "Add collaborator"
3. Collaborator accepts invitation
   a. Go into notifications
   b. Unread
   c. Invitation to join …
   d. Accept invitation
4. Collaborator copies address of remote repository from owner repository (following steps 2-3 on previous slide)
5. Collaborator works with new repository (following steps 4-5 on previous slide)








```
lan@lan-desktop:~/Desktop/GitTalk$ git remote add origin https://github.com/makselab/GitTalk.git
lan@lan-desktop:~/Desktop/GitTalk$ git push -u origin master
Username for 'https://github.com': a-einstein1879
Password for 'https://a-einstein1879@github.com':
Counting objects: 17, done.
Delta compression using up to 12 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (17/17), 1.33 KiB | 0 bytes/s, done.
Total 17 (delta 5), reused 0 (delta 0)
remote: Resolving deltas: 100% (5/5), done.
To https://github.com/makselab/GitTalk.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
```

Example. Sample workflow

1. Clone remote repository

2. Work with files in repository
3. Stage the file and commit changes

4. Push your changes

```
ian@ian-desktop:~/Desktop$ git clone https://github.com/makselab/GitTalk.git
Cloning into 'GitTalk'...
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 20 (delta 6), reused 20 (delta 6), pack-reused 0
Unpacking objects: 100% (20/20), done.
Checking connectivity... done.
ian@ian-desktop:~/Desktop$ cd GitTalk/
ian@ian-desktop:~/Desktop/GitTalk$ echo "Hello, world!" > file5.txt
ian@ian-desktop:~/Desktop/GitTalk$ git add file5.txt
ian@ian-desktop:~/Desktop/GitTalk$ git commit -m "File 5 added to master"
[master 1d0ca70] File 5 added to master
 1 file changed, 1 insertion(+)
 create mode 100644 file5.txt
ian@ian-desktop:~/Desktop/GitTalk$ git push
Username for 'https://github.com': a-einstein1879
Password for 'https://a-einstein1879@github.com':
Counting objects: 2, done.
Delta compression using up to 12 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 248 bytes | 0 bytes/s, done.
Total 2 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/makselab/GitTalk.git
   3e5d62d..1d0ca70  master -> master
ian@ian-desktop:~/Desktop/GitTalk$
```

Basic commands

git init - initialize local Git repository
git add - add new file to list of tracked files or stage modified file
git commit - commit changes (only staged files will be committed)
git remove - stop tracking file
git checkout - restore repository or file to specific version
git branch - create new branch
git merge <branch_name> - merge branch_name to current branch
git push - push changes to remote repository
git fetch - fetches changes from remote repository
git clone - clones remote Git repository
git clean - clean untracked files from git repository
gitk - visual tool for managing commit history
https://git-scm.com/ - useful resource

Thank you for your attention!

Questions?