CSE306 Software Quality in Practice

Dr. Carl Alphoncse
alphonce@buffalo.edu
343 Davis Hall
Exercise from last time
```c
#include <stdio.h>
#include <stdlib.h>

int main() {
    int x = 0;
    while (x < 10) {
        printf("x has value %d\n", x);
        x = x + 1;
    }
    exit(EXIT_SUCCESS);
}
```
```c
#include <stdio.h>
#include <stdlib.h>

int x = 0;
int main() {
    while (x < 10) {
        printf("x has value %d\n",x);
        x = x + 1;
    }
    exit(EXIT_SUCCESS);
}
```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int[] x = {0};
    while (x < 10) {
        printf("x has value %d\n", x[0]);
        x[0] = x[0] + 1;
    }
    exit(EXIT_SUCCESS);
}
#include <stdio.h>
#include <stdlib.h>

int main() {
    int *x = (int*)malloc(sizeof(int));
    *x = 0;
    while (*x < 10) {
        printf("x has value %d\n",*x);
        *x = *x + 1;
    }
    exit(EXIT_SUCCESS);
}
git

- distributed version control system
Local Machine
(e.g. your laptop, or timberlake if you've ssh'ed in)

stash
workspace
index staging
local repository

Remote
(e.g. bitbucket, github, CSE servers)
remote repository
What you see when working:

- stash
- workspace
- index staging
- local repository
- remote repository
Cloning a remote
Makes a copy of remote repo in local repo and checks out branch into workspace

git clone
Add a file to the staging area (add it to the index)

- stash
- workspace
- index
- staging
- local
- repository
- remote
- repository

`git add`
Create a new commit object with the staged items from the index.

```
git commit
```
Push files from local repo to remote repo

- stash
- workspace
- index staging
- local repository
- remote repository

`git push`
Pulling files

"git pull is shorthand for git fetch followed by git merge FETCH_HEAD"
[https://git-scm.com/docs/git-pull]
Grab files from remote

- stash
- workspace
- index staging
- local repository
- remote repository

`git fetch`
Create a commit combining the contents of two branches

- stash
- workspace
- index staging
- local repository
- remote repository

/git_merge/
Let's start by cloning an existing repository

git clone
**GitIntro**

A first repo to show students how to interact with repo on GitHub
Clone the repo

```bash
% git clone git@github.com:UB-CSE306/GitIntro.git
Cloning into 'GitIntro'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.
```

```bash
% ls -la
total 0
drwxr-xr-x  3 alphonce  staff    96 Feb  8 13:20 .
drwxr-xr-x  48 alphonce  staff  1536 Feb  8 13:24 ..
drwxr-xr-x  4 alphonce  staff   128 Feb  8 13:20 GitIntro
```

These slides were made years ago, and the repo may look different now.
% cd GitIntro/
% ls -la

```
total 8
drwxr-xr-x  4 alphonce  staff  128 Feb  9 13:20 
. 
drwxr-xr-x  3 alphonce  staff   96 Feb  9 13:20 ..
.. 
-drwxr-xr-x 12 alphonce  staff  384 Feb  9 13:20 .git
-rw-r--r--  1 alphonce  staff   85 Feb  9 13:20 README.md
```
cd .git
ls -la

total 40
-rw-r--r--  1 alphonse  staff   23 Feb  9 13:20  HEAD
-rw-r--r--  1 alphonse  staff  320 Feb  9 13:20  config
-rw-r--r--  1 alphonse  staff   73 Feb  9 13:20  description
-rw-r--r--  1 alphonse  staff  137 Feb  9 13:20  index
-rw-r--r--  3 alphonse  staff   96 Feb  9 13:20  info
-rw-r--r--  4 alphonse  staff  128 Feb  9 13:20  logs
-rw-r--r--  4 alphonse  staff  128 Feb  9 13:20  objects
-rw-r--r--  1 alphonse  staff  114 Feb  9 13:20  packed-refs
-rw-r--r--  5 alphonse  staff  160 Feb  9 13:20  refs
pointer to the current branch
% more HEAD
ref: refs/heads/main
staging area

```
-rw-r--r--  1 alphonce  staff  23 Apr 17 13:26 HEAD
drwxr-xr-x  2 alphonce  staff  68 Apr 17 13:26 branches
-rw-r--r--  1 alphonce  staff  328 Apr 17 13:26 config
-rw-r--r--  1 alphonce  staff  73 Apr 17 13:26 description
drwxr-xr-x 12 alphonce  staff  408 Apr 17 13:26 hooks
-rw-r--r--  1 alphonce  staff  137 Apr 17 13:26 index
drwxr-xr-x  3 alphonce  staff  102 Apr 17 13:26 info
drwxr-xr-x  4 alphonce  staff  136 Apr 17 13:26 logs
drwxr-xr-x  7 alphonce  staff  238 Apr 17 13:26 objects
-rw-r--r--  1 alphonce  staff  107 Apr 17 13:26 packed-refs
drwxr-xr-x  5 alphonce  staff  170 Apr 17 13:26 refs
```
% git ls-files
README.md
The *git man page* seems to be surprisingly bereft of an official definition, other than this (emphasis mine):

The **object database** contains objects of three main types: **blobs**, which hold file data; trees, which point to blobs and other trees to build up directory hierarchies; and commits, which each reference a single tree and some number of parent commits.

The repeated use of the term "object database" across git documentation suggests a borrowing of "blob" specifically from DBMSs.

In its article on [Binary large objects](https://en.wikipedia.org/wiki/Binary_large_objects) Wikipedia defines the term as "a collection of binary data stored as a single entity in a database management system", further offering the following:

Blobs were originally just amorphous chunks of data invented by Jim Starkey at DEC, who describes them as "the thing that ate Cincinnati, Cleveland, or whatever" from "the 1958 Steve McQueen movie", referring to The Blob. Later, Terry McKiever, a marketing person for Apollo, felt that it needed to be an acronym and invented the backronym Basic Large Object. Then Informix invented an alternative backronym, Binary Large Object.

So, though it's not a definitive answer, the term "blob" has a conventional and well-defined usage across computer science as an opaque string of binary data, and git adheres to that definition without further specifying it.

*answered Jul 24 '15 at 17:12*

**Jeff Bowman**

33.9k65685
% ls -l objects

% git cat-file -t 25b4
commit
% git cat-file -t 9ce9
tree
% git cat-file -t 39af
blob

% ls -l

% ls -l objects

% git cat-file -t
commit
% git cat-file -t
tree
% git cat-file -t
blob

% ls -l

% git cat-file -p 25b4

Tree 9ce959348ab4c2ebe61549393b4b1acc0504a649
Author Carl Alphonce <alphonce@buffalo.edu> 1492449992 +0000
Committer Carl Alphonce <alphonce@buffalo.edu> 1492449992 +0000

README.md created online with Bitbucket
% git cat-file -p 9ce9
100644 blob 39af52c077c0d4c3bc7730b362592e0bf7f635db README.md
% git cat-file -p 39af
# README #

This README would normally document whatever steps are necessary to get your application up and running.

### What is this repository for? ###
* Quick summary
* Version
* [Learn Markdown](https://bitbucket.org/tutorials/markdowndemo)

### How do I get set up? ###
* Summary of set up
* Configuration
* Dependencies
* Database configuration
* How to run tests
* Deployment instructions

### Contribution guidelines ###
* Writing tests
* Code review
* Other guidelines

### Who do I talk to? ###
* Repo owner or admin
* Other community or team contact
pointers to commits

-rw-r--r-- 1 alphonce  staff  23 Apr 17 13:26 HEAD
-dwxr-xr-x 2 alphonce  staff  68 Apr 17 13:26 branches
-rw-r--r-- 1 alphonce  staff  328 Apr 17 13:26 config
-rw-r--r-- 1 alphonce  staff  73 Apr 17 13:26 description
-dwxr-xr-x 12 alphonce  staff  408 Apr 17 13:26 hooks
-rw-r--r-- 1 alphonce  staff  137 Apr 17 13:26 index
-dwxr-xr-x  3 alphonce  staff  102 Apr 17 13:26 info
-dwxr-xr-x  4 alphonce  staff  136 Apr 17 13:26 logs
-dwxr-xr-x  7 alphonce  staff  238 Apr 17 13:26 objects
-rw-r--r-- 1 alphonce  staff  107 Apr 17 13:26 packed-refs
-dwxr-xr-x  5 alphonce  staff  170 Apr 17 13:26 refs
Possible states of a file

- **unmodified**
  - edit
  - commit

- **modified**
  - add

- **untracked**
  - add

- **staged**
  - add
commit preserves contents
(accidental removals can be recovered from)
create a file

Suppose we create a file in the workspace.

How do we get it into the local repository?
add to index (staging area)

```
git add <filename>
```
git add
commit to local repo

```bash
  git commit -m "message"
```