git-0.01

256个目录 在 init_db 时创建好.

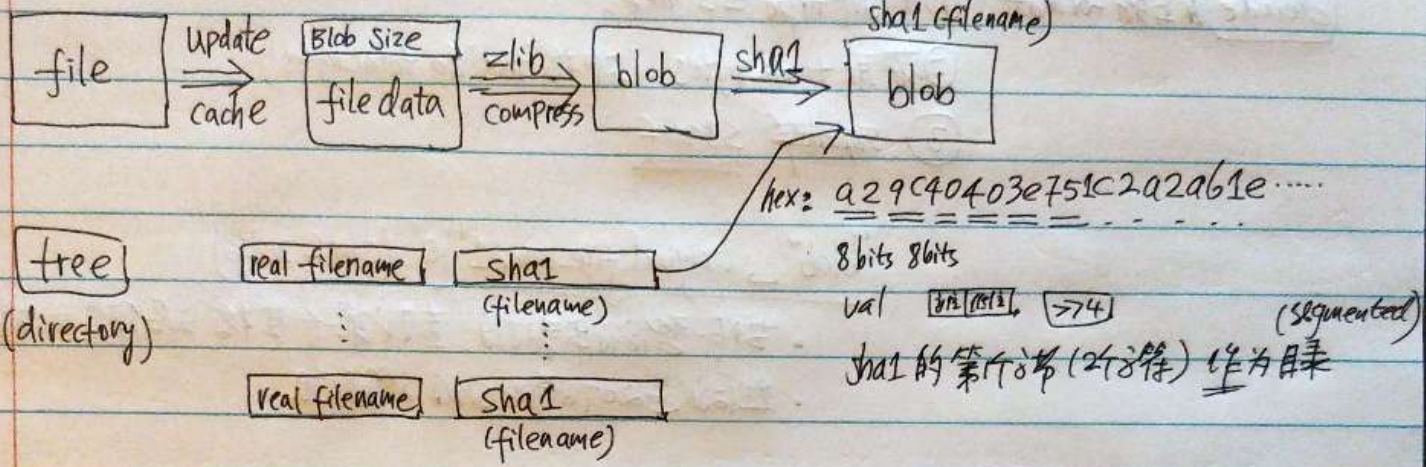
Working directory

objects/ 00 01 02 ... FF

sha1

dircache .dircache/

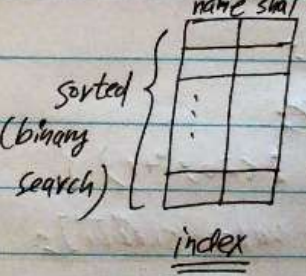
index



.update-cache f1 f2 f3 (f1, f2, f3 添加或修改的文件) 到 index文件中

↳ add_file_to_cache objects/xx/xx... ⇒ add_cache_entry

↳ write_cache (index文件)




```
struct cache_header {
    uint32_t signature;
    uint32_t version;
    uint32_t entries;
    unsigned char sha1[20];
};
```

(cache_header + 所有 cache_entry) 的 sha1.

```
struct cache_entry {
    struct cache_time ctime;
    struct cache_time mtime;
    ... 文件的 stat information
    unsigned char sha1[20];
    unsigned short namelen;
    unsigned char name[0];
};
```

name[0]: 文件真正的名字

index文件: 写到文件: ① header binary data.

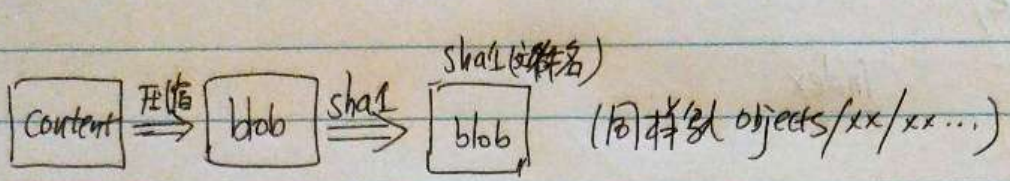
② 所有的 cache_entry binary data

每次 update_cache 运行都要先 read_cache, 将 index db 读入(deserialize)到内存数据结构中. 直接用 mmap 读取映射内存中.

tree index cache中的所有entries，以下是content（type字符串，数据字节数 打头）

"tree" total_bytes "blob" original_file_name1 '\0' sha1 (二进制)
(20字节)

"blob" original_file_name2 '\0' sha1 "blob" .....



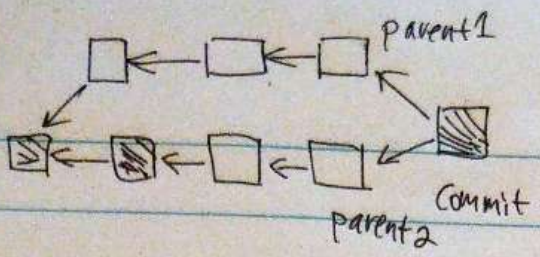
decode 树的format. ①判断type == "tree",
②获取total_bytes (不包括'tree'和total_bytes)
③ ~~sprintf~~ sscanf

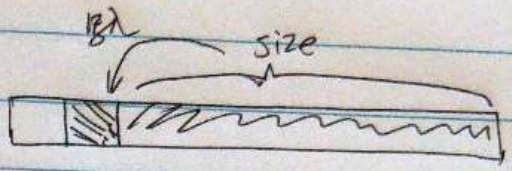
"blob" original_file_name1 '\0' (sha1, 20字节)
- 当作一整个字符串数据，查找第一个空格，后便是文件名.
- 然后就是20个字节的sha1 字节数组.

```
sha1_to_hex (unsigned char* sha1)
{
  static const char hex[] = "0123456789abcdef";
  char buffer[50];
  char* buf = buffer;
  for(int i=0; i<20; i++) {
    unsigned int val = sha1[i];
    *buf++ = hex[val>>4];
    *buf++ = hex[val&0xF];
  }
}
```

① 新文件/更改的文件会被"add"编码成blob对入到 objects/目录，并记录在 index文件.
② index文件是所有文件的最新描述，文件名，sha1值.
③ 每次update都会从disk中读出，然后再进行更新，再写入，组装成
④ commit时，将index文件内容（内存中的cacheEntry）作为tree作为一个数据对入 objects/目录文件.
返回 sha1 (文件名). 一个commit 对应一个 objects 文件.
都有个sha1

commit





"commit" format:

"tree" <sha1> '\n'

["parent" <sha1> '\n']

["parent" <sha1> '\n']

- - - - -

"author" xx <email> date

"committer" xx <email> date (real user)

"[comment]"

"commit" 字节数 '\0'

tag header

Ex: "commit" "123" '\0'

(123字节)

↓

压缩

↓

sha1

↓

写入 objects/ blob 对象

返回 commit 的 sha1 作为引用

---

show-diff

working directory

index

index 中文件有文件名 + sha1

↓ 最新文件在 working directory

↓ 上一次更改后的文件

命令: diff -u - %s

%s ==> 文件名

```
f = popen(cmd, "w");
fwrite(sha1文件内容, sha1文件数大小, 1, f);
pclose(f);
```