$ git add .

$ git commit –m “ ”

底層命令Plumbing

高階命令Porcelain add commit

Git != SVN + Magic

git add. git add --all 新增 修改 刪除

add.只會在當前目錄 –all專案哪一層目錄

**分散式**

上傳到gitserver上其他人拉下來，也可以用farpage的方式送給其他人，有人刪了一個檔案也可以取得，沒有gitserver大家依舊可透過網路交換資訊

把commit打包 git format-path -2 HEAD –o /tmp : git apply /tmp/\*.patch

git status

git diff

git add –all

git commit –m “update files”

**版本**

當每完成一個任務不管是大或小，或是新增或是修改或是刪除，可能都會造成目錄或檔案上的變化，每一次的變化稱之為一個版本。

Git只在乎檔案的內容，不在乎目錄或檔案名稱，是\*內容追蹤軟體\*，剛好當版控軟體蠻適合。版控軟體主要紀錄每一次的差異，版本1版本2增加兩個方塊，版本2版本3修改兩個圈圈…記錄起來。git是拍快照，把目錄結構和檔案拍下來，版1到版2加了兩個方塊剩下的都沒有變動，只要是沒有異動的就會只到前一個版本，不會再存依次這個檔案。

**SHA1 由40個16進位字元組成 重複!=碰撞**

$git log

commit (HEAD->master)

Blob物件的SHA1計算公式:

**<1>**blob 字樣 **<2>**1個空白 **<3>**輸入內容的長度 **<4>**Null結束符號 **<5>輸入內容**

$ irb

>> require “digest/sha1”

=>true

>>content = “Hello, 5xRuby”

=> “Hello, 5xRuby”

>>input = “blob #{content.length}\0#{content}”

=>”blob 13\u000Hello, 5c=xRuby”

>> Digest::SHA1.hexdigest(input)

=>”4135fc4add33332e…”

$printf “Hello, 5xRuby” | git hash-object --stdin

”4135fc4add33332e…”

“e69de29ebb…”SHA1值 在git裡是空字串的輸出值

$printf”” | git hash-object - -stdin

git物件檔名:SHA1

git物件內容:檔案內容+壓縮

**git commit**

git 物件 blob(檔案) tree(目錄檔名) commit tag

開始使用git把檔案加到暫存區時blob；開始commit tree，指向blob或其他tree；commit完成後，git 長出commit物件指向tree，指向前一個commit物件；tag指向某個commit。

$ echo “hello, 5xruby” > index.html /\*因為檔案還不算加到git裡面，只是路過而已，untracked狀態，還沒被git追蹤的狀態

$ git add index.html /\*把檔案加到暫存區(索引區) /\*長出blob物件

$ mkdir config /\*加到config目錄

$ echo “super-secret-password” > config/ database.yml

/\*在config目錄裡放database.yml的檔案 裡面內容是super-secret-password /\*因為它是實際的檔案，放在config目錄，git認為是untracked狀態

$ git add config/database.yml /\*把檔案加到暫存區 /\*所以會長出另外一個blob物件

$ git commit –m “init commit” /\*進行commit,git根據目錄造出tree物件,commit物件指向tree物件,HEAD指向master指向commit

$ echo “this is a book” >> index.html /\*狀態modified

$ git add index.html /\*檔案已經不一樣了，新的blob物件