

Introduction to Git for Data Science (DataCamp)

陽明生資所 陳卓逸老師實驗室
專任研究助理 陳躍中 製作
2018.07.12

<https://www.datacamp.com/courses/introduction-to-git-for-data-science>

Content

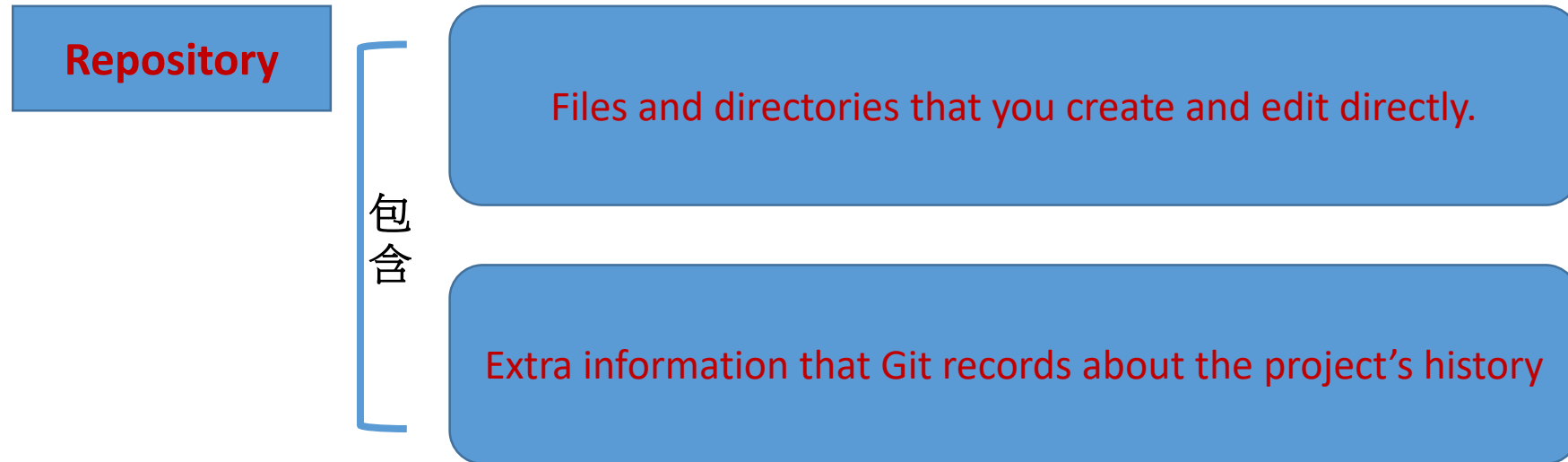
- 1.Basic workflow
- 2.Repositories
- 3.Undo
- 4.Working with Branches
- 5.Collaborating

Introduction to Git for Data Science (DataCamp)

- Version Control System 版本控制系統
- Git 為其中一種版本控制系統
- You can always go back to see which results were generated by which versions of your programs.
- Git will automatically notify you when your work conflicts with someone else's. (prevent overwrite)
- Git can synchronize work done by different people on different machines.

Where does Git store information?

- Git projects (files, directories / extra information):



1. Git stores all of its extra information in directory called `.git` which located in the root directory of the repository.
2. Git expects this information to be laid out in a very precise way, so you should never edit or delete anything in `.git`.

Pop quiz:

Suppose your home directory `/home/repl` contains a repository called `dental`, which has a sub-directory called `data`. Where is information about the history of the files in `/home/repl/dental/data` stored?

由於題目是要問 Git 將 `/home/repl/dental/data` 的資料存在哪裡？
尋找 `data` 資料夾的 `root` 資料夾，
故答案為 `/home/repl/dental/.git`

How can I check the state of a repository?

- 檢測目前 git 狀態的指令：`< git status >`

```
$ cd dental
$ 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)

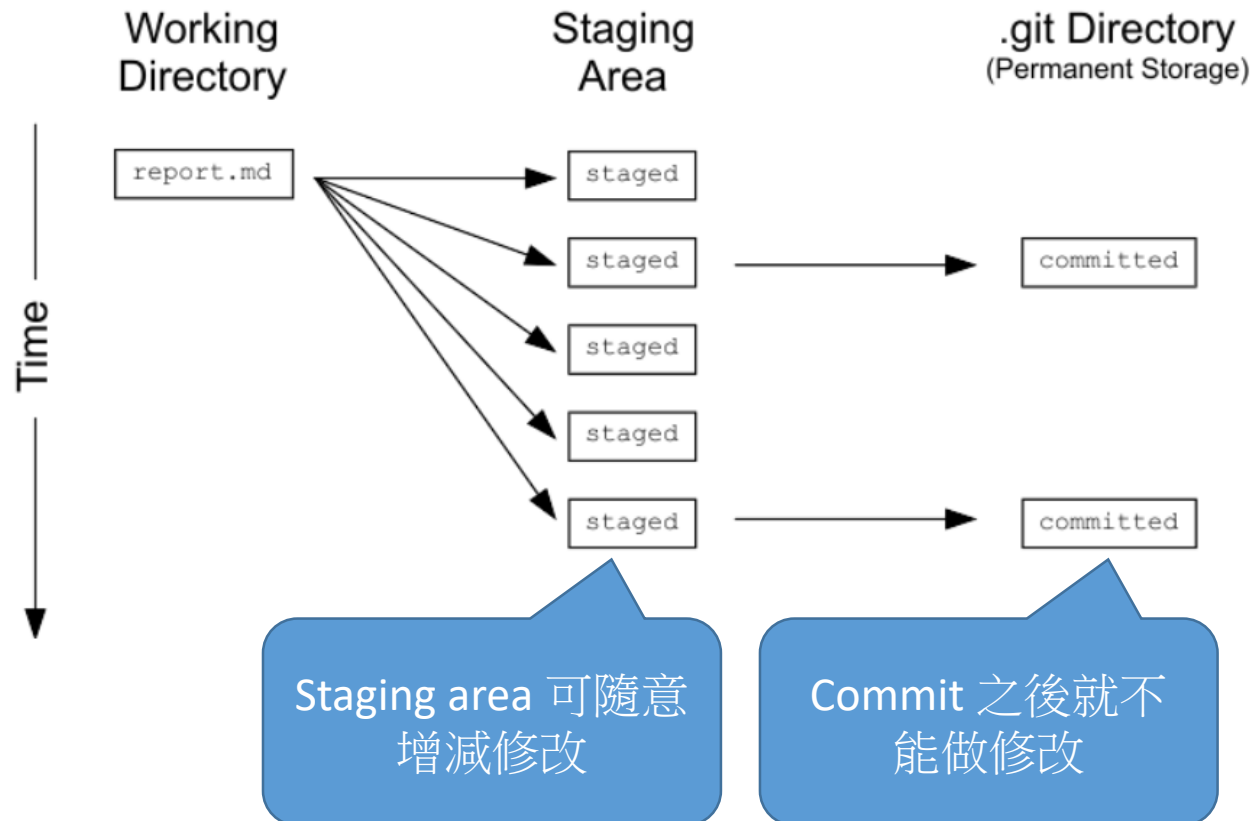
        modified:   report.txt

no changes added to commit (use "git add" and/or "git commit -a")
$ █
```

從上圖終端機結果可以知道輸入 `git status` 後可以看到在 `dental repository` 中，距離上一次儲存後，有被更改過的檔案為 `report.txt`

How can I tell what I have changed?

- Git has a **staging area** in which it stores files with changes you want to save that haven't been saved yet.



```
$ git diff
diff --git a/data/northern.csv b/data/northern.csv
index 5eb7a96..5a2a259 100644
--- a/data/northern.csv
+++ b/data/northern.csv
@@ -22,3 +22,4 @@ Date,Tooth
 2017-08-13,incisor
 2017-08-13,wisdom
 2017-09-07,molar
+2017-11-01,bicuspid
```

<git diff>

可以查詢你曾經在此 repository 中修改過什麼樣的檔案。(接下頁)

What is in a diff?

```
$ git diff
1 diff --git a/data/northern.csv b/data/northern.csv
2 index 5eb7a96..5a2a259 100644
3 --- a/data/northern.csv
4 +++ b/data/northern.csv
5 @@ -22,3 +22,4 @@ Date,Tooth
6 2017-08-13,incisor
7 2017-08-13,wisdom
8 2017-09-07,molar
9 +2017-11-01,bicuspid
```

1. a, b 為 placeholder, 代表第一個版本(a), 第二個版本(b)
2. Index keys into Git's internal database of changes (往後再著墨)
3. 3&4 – 為刪除的 lines, + 為增加的 lines
5. @@ 指出哪裡被修改過, 此處為 22-3, 被刪除, 被 22-4 取代。(被修改的 line 數量為 one line)

What's the first step in saving changes?

- Commit changes to a Git repository in two steps:

1. Add one or more files to the staging area.
2. Commit everything in the staging area.

指令：< git add filename>

```
$ cd dental
$ git add report.txt
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   report.txt
```

輸入 git add report.txt 之後
用 git status 看狀態
可以發現 report.txt 已經變成綠色字體

How can I tell what's going to be committed?

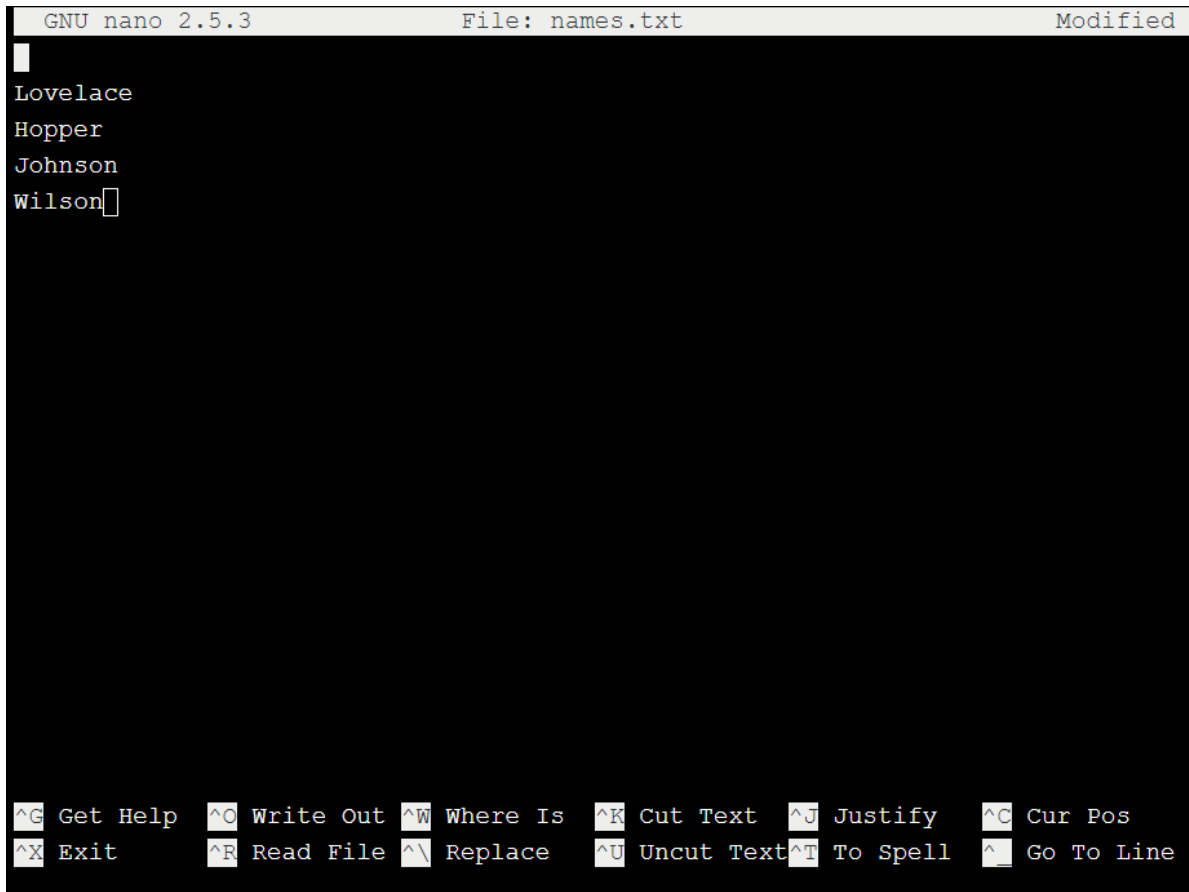
```
$ git diff -r HEAD data/northern.csv
diff --git a/data/northern.csv b/data/northern.csv
index 5eb7a96..5a2a259 100644
--- a/data/northern.csv
+++ b/data/northern.csv
@@ -22,3 +22,4 @@ Date,Tooth
 2017-08-13,incisor
 2017-08-13,wisdom
 2017-09-07,molar
+2017-11-01,bicuspid
```

<git diff -r HEAD path/to/file>

-r flag means compare to a particular revision
HEAD is a shortcut meaning “the most recent commit”

How can I edit a file? (nano as example)

- `<nano filename>` 輸入此指定，若原本有檔案，則會用 nano 開啟。
若原本沒有該檔案，則會創造一個出來並用 nano 開啟。

A screenshot of the GNU nano 2.5.3 text editor. The title bar at the top shows "GNU nano 2.5.3", "File: names.txt", and "Modified". The main editing area has a black background with white text. It contains four lines of text: "Lovelace", "Hopper", "Johnson", and "Wilson". A white cursor is positioned at the end of the word "Wilson" on the fourth line. At the bottom of the editor, there is a status bar with two rows of keyboard shortcuts: the first row includes ^G Get Help, ^O Write Out, ^W Where Is, ^K Cut Text, ^J Justify, and ^C Cur Pos; the second row includes ^X Exit, ^R Read File, ^\ Replace, ^U Uncut Text, ^T To Spell, and ^_ Go To Line.

(左圖) 用 nano 開啟一個名叫 `names.txt` 的檔案。
編輯完成之後 `Ctrl-O` 儲存檔案 (按 `Enter` 確認)，`Ctrl-X` 退出編輯器。

- `Ctrl-K`: delete a line.
- `Ctrl-U`: un-delete a line.
- `Ctrl-O`: save the file ('O' stands for 'output').
- `Ctrl-X`: exit the editor.

How do I commit changes?

- `<git commit -m 'Program appears to have become self-aware'>`
- 編輯 commit 的 log message，為了讓下次使用者知道有那些被修改過。

```
$ cd dental
$ git add report.txt
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

       modified:   report.txt

$ git commit -m 'Adding a reference'
```

```
[master f6ca510] Adding a reference
1 file changed, 3 insertions(+)
```

How can I view a repository's history?

- The command `<git log>` is used to view the log of the project's history.

```
$ cd dental
$ git log
commit 7bc27cac9374930aea9f146fe8828ed8d925d465
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Added year to report title.

commit 90675ac9c39e3cc48fd65d08b13569d531b7963e
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Adding fresh data for western region.

commit d3a9c4d7bd9cc2cd9f1db0533723155c9fef4311
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Adding fresh data for southern and western regions.

commit 16a0b5c2146f3b40913e495d5075903d9a2bdd06
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Fixed bug and regenerated results.
:█
```

Commit 那行稱為 hash (稍後補充)
下方則是作者和log message內容與時間。

按空白鍵可以往下看，越上方的資料越接近現今。

按 'q' 可退出 git log 模式

How can I view specific file's history?

- `<git log path>` where path is the path to a specific file or directory

```
$ cd dental
$ git log data/southern.csv
commit d3a9c4d7bd9cc2cd9f1db0533723155c9fef4311
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Adding fresh data for southern and western regions.

commit 7578b729316199a459f944d7c16cd1a88fb1cef1
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Added seasonal CSV data files
```

How do I write a better log message?

- <git commit> : Git launches a text editor with a template.

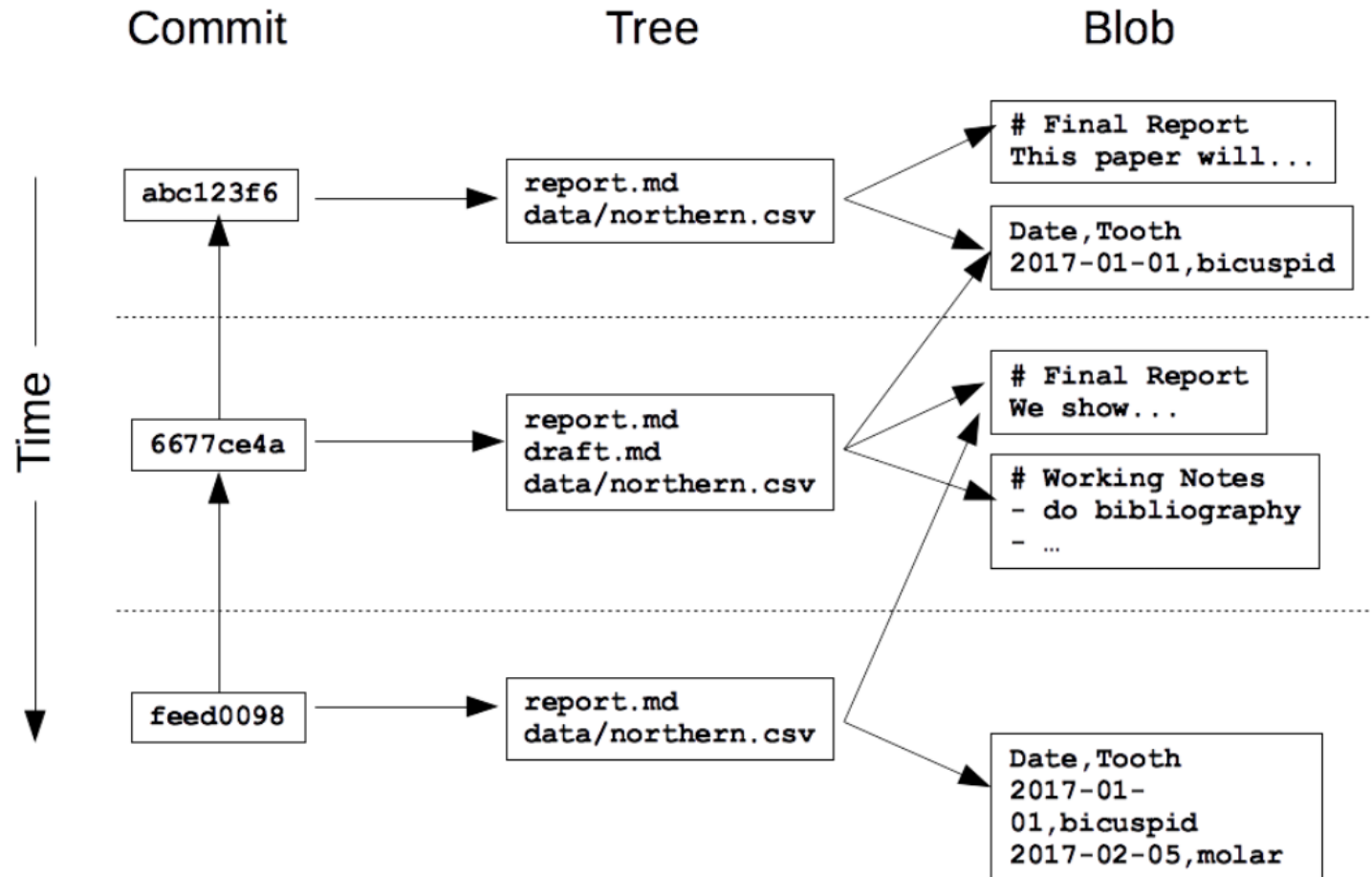
```
GNU nano 2.5.3      File: /home/repl/dental/.git/COMMIT_EDITMSG
#
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Changes to be committed:
#       modified:   report.txt
#
[ Read 7 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line
```

可以編輯比較詳細的內容
開頭的不會被儲存，只是提示你可以寫一些什麼內容

可以注意到這個文字編輯器是 nano
所以操作方式如同前面教學。

How does Git store information?

- Git uses a multilevel structure to store data. (fast, minimize storage space).
- 1. Every unique version of every file. (Git calls these **blobs** because they can contain data of any kind)
- 2. **Tree** that tracks the names and locations of a set of files.
- 3. A **commit** that records the author, log message, and other properties of a particular commit.



What is hash?

- 40-character hexadecimal string
- 由 hash function 製作出的 Pseudo-random number (假隨機)
- Most of the time, you only have to give Git the first 6 or 8 characters in order to identify the commit you mean.
- Git can therefore tell what information needs to be saved where by comparing hashes rather than comparing entire files.

```
$ git show 7bc27
commit 7bc27cac9374930aea9f146fe8828ed8d925d465
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Added year to report title.

diff --git a/report.txt b/report.txt
index e713b17..4c0742a 100644
--- a/report.txt
+++ b/report.txt
@@ -1,4 +1,4 @@
-# Seasonal Dental Surgeries 2017-18
+# Seasonal Dental Surgeries (2017) 2017-18

TODO: write executive summary.
```

<git show hash>

可以找尋特定檔案的歷史，對比 git log 是整個 repository 的歷史。
Hash 的部分就是輸入最前面幾碼通常就可以

What is Git's equivalent of a relative path?

- The special label HEAD, always refers to the most recent commit. HEAD~1 then refers to the commit before it. HEAD~2..... (tilde 前後不能有空格)。

How can I see who changed what in a file?

- `<git annotate file>` shows who made the last change to each line of a file and when.

```
$ cd dental
$ git annotate report.txt
7bc27cac      ( Rep Loop    2018-07-02 12:30:59 +0000    1)# Seasonal De
ntal Surgeries (2017) 2017-18
a233c117      ( Rep Loop    2018-07-02 12:30:59 +0000    2)
a233c117      ( Rep Loop    2018-07-02 12:30:59 +0000    3)TODO: write e
xecutive summary.
a233c117      ( Rep Loop    2018-07-02 12:30:59 +0000    4)
a233c117      ( Rep Loop    2018-07-02 12:30:59 +0000    5)TODO: include
link to raw data.
a4ebc0f6      ( Rep Loop    2018-07-02 12:30:59 +0000    6)
a4ebc0f6      ( Rep Loop    2018-07-02 12:30:59 +0000    7)TODO: remembe
r to cite funding sources!
```

基本上一個 hash 號碼，代表一次修改次數。
以左圖為例：
總共有 3 次修改：7bc27cac, a233c117, a4ebc0f6

How can I see what changed two commits?

- `<git diff ID1..ID2>`

```
$ git diff 7bc27ca..90675ac
diff --git a/report.txt b/report.txt
index 4c0742a..e713b17 100644
--- a/report.txt
+++ b/report.txt
@@ -1,4 +1,4 @@
-# Seasonal Dental Surgeries (2017) 2017-18
+# Seasonal Dental Surgeries 2017-18

TODO: write executive summary.
```

How do I add new files?

```
$ cd dental
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    sources.txt

nothing added to commit but untracked files present (use "git add" to track)
$ git add sources.txt
$ git commit -m "Starting to track data sources."
[master 4043c24] Starting to track data sources.
 1 file changed, 1 insertion(+)
 create mode 100644 sources.txt
```

How do I tell Git to ignore certain files?

- `<.gitignore>`
- 如果 `.gitignore` 包含 `build, *.mpl`，則被包含的檔案即會被 Git 忽略。
- 如果有權不包含關鍵字的檔案或是資料夾名稱，則需要加入 wildcard `'*'`。不然就只能忽略跟 `line` 一模一樣。
- 比如 `.gitignore` 有 `line` 為 `pdf`，但因為 `pdf` 沒有 wildcard 符號，所以 Git 只會忽略檔名或是資料夾叫做 `pdf` 的檔案，`apple.pdf` 則不會被忽略。

How can I remove unwanted files?

- `<git clean -n>` will show you a list of files that are in the repository, but whose history Git is not currently tracking.
- `<git clean -f>` will delete those files.

```
$ cd dental
$ ls
backup.log  bin  data  report.txt  results
$ git clean -f
Removing backup.log
$ ls
bin  data  report.txt  results
```

How can I see how Git is configured?

- Git allows you to change its default settings.
- `<git config --list + --system/--global/--local>`
- `--system` : settings for every user on this computer.
- `--global` : settings for every one of your projects.
- `--local` : settings for one specific project.

```
$ git config --list --local
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
```


How can I change my Git configuration?

- `<git config --global setting.name setting.value>`
- With the setting's name and value in the appropriate places. The keys that identify your name and email address `user.name` and `user.email` respectively.

```
$ git config --global user.email rep.loop@datacamp.com
```

How can I commit changes selectively?

```
$ cd dental
$ 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:   data/eastern.csv
        modified:   data/northern.csv

no changes added to commit (use "git add" and/or "git commit -a")
```

```
$ $ git add data/northern.csv
$ git commit -m "Adding data from northern region"
[master 9cdb918] Adding data from northern region
1 file changed, 1 insertion(+)
```

How can I undo changes to unstaged files?

- `<git checkout -- filename>` discard the changes that have not yet been staged. Checkout 要小心使用，一旦丟棄的檔案將會永久消失。

```
$ cd dental
$ git add data/*.csv
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   data/eastern.csv
        modified:   data/northern.csv

$ git checkout -- data/northern.csv
```

How can I unstage a file that I have staged?

- `<git reset HEAD filename>` undo changes that have been staged. However, it doesn't restore the file to the state it was in before you started making changes. Instead, it resets the file to the state you last staged.

```
$ cd dental
$ git add data/*.csv
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   data/eastern.csv
        modified:   data/northern.csv

$ git reset HEAD data/northern.csv
Unstaged changes after reset:
M       data/northern.csv
$
```

How do I restore an old version of a file?

- `<git checkout hash file_name>`

```
$ cd dental
$ git log report.txt
commit 7bc27cac9374930aea9f146fe8828ed8d925d465
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Added year to report title.

commit 93daacc499a095fd7e1ab6b120269aef52e39b1e
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Changed title because purpose of report has changed.

commit a4ebc0f6f4a4450d6a222377b2c61610dee53b91
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Added reminder to cite funding sources.

commit 796cdce3896f1f046847d2d3a599635f2bae1614
Author: Rep Loop <repl@datacamp.com>
Date:   Mon Jul 2 12:30:59 2018 +0000

    Renamed report as plain text file rather than Markdown.
$ git checkout 7bc27cac report.txt
$ git commit
On branch master
nothing to commit, working directory clean
```

How can I undo all of the changes I have made?

- `<git reset HEAD directory_name>` will unstage any files from the `directory_name` directory that you have staged.
- `<git checkout -- data>` restore those files to their previous state

```
$ git reset HEAD .  
Unstaged changes after reset:  
M      data/eastern.csv  
M      data/northern.csv  
M      report.txt  
$ git checkout -- .
```

‘.’ dot 表示目前所在的資料夾，
或是在此資料夾的所有檔案。

Branch: a parallel universe

- Changes you make in one branch do not affect other branches until you merge them back together.

How can I see what branches my repository has?

```
TERMINAL
$ cd dental
$ git branch
  alter-report-title
* master
  summary-statistics
$
```

How can I view the differences between branches?

```
$ git diff summary-statistics..master
diff --git a/bin/summary b/bin/summary
deleted file mode 100755
index eeec501..0000000
--- a/bin/summary
+++ /dev/null
@@ -1,4 +0,0 @@
-#!/usr/bin/env bash
-
-echo $(wc -l < /home/repl/dental/results/dates.csv) 'unique dates'
-echo $(wc -l < /home/repl/dental/results/teeth.csv) 'unique teeth'
diff --git a/report.txt b/report.txt
index e713b17..4c0742a 100644
--- a/report.txt
+++ b/report.txt
@@ -1,4 +1,4 @@
-# Seasonal Dental Surgeries 2017-18
+# Seasonal Dental Surgeries (2017) 2017-18

TODO: write executive summary.

diff --git a/results/summary.txt b/results/summary.txt
deleted file mode 100644
index 2e880b9..0000000
```

\$ git diff summary-statistics..master
顯示這兩個 branch 的差別，記得..前後不要有空格。

How can I switch from one branch to another?

```
TERMINAL
$ cd dental
$ git branch
  alter-report-title
* master
  summary-statistics
$ git checkout summary-statistics
Switched to branch 'summary-statistics'
$ git rm report.txt
rm 'report.txt'
$ git commit -m "Removing report"
[summary-statistics 60f3eb3] Removing report
 1 file changed, 7 deletions(-)
 delete mode 100644 report.txt
$ ls
bin  data  results
$ git checkout master
Switched to branch 'master'
$ ls
bin  data  report.txt  results
```

- \$ git checkout branch-name
 1. Switch to summary-statistics branch
 2. use git rm to delete report.txt
 3. commit your change
 4. use ls to check that it is gone
 5. switch back to master and use ls to ensure report.txt is still there

How can I create a branch

- The easiest way to create a new branch is to run
\$ git checkout -b branch-name
which creates the branch and switches you to it.
The contents of the new branch is identical to the contents of the original. Once you start making changes, they only affect the new branch.

TERMINAL

```
$ cd dental
$ git branch
alter-report-title
* master
summary-statistics
$ git checkout -b deleting-report
Switched to a new branch 'deleting-report'
```

How can I merge two branches?

- `$ git merge source_branch_name destination_branch_name`

TERMINAL

```
$ cd dental
$ git merge summary-statistics master
Merge made by the 'recursive' strategy.
 bin/summary          | 4 ++++
 results/summary.txt | 2 ++
2 files changed, 6 insertions(+)
create mode 100755 bin/summary
create mode 100644 results/summary.txt
```

How can I merge two branches with conflicts?

```
TERMINAL
$ cd dental
git branch
$ git branch
  alter-report-title
* master
  summary-statistics
$ git merge alter-report-title master
Auto-merging report.txt
CONFLICT (content): Merge conflict in report.txt
Automatic merge failed; fix conflicts and then commit the result.
$ git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")

Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both modified:   report.txt

no changes added to commit (use "git add" and/or "git commit -a")
$ nano report.txt
$ git add report.txt
$ git commit -m "message"
[master 0e11d9e] message
```

- 1. Merge and find conflict
- 2. \$git status (to see which file has conflicts.)
- 3. use text editor like nano to remove the conflict markers.
- 4. add and commit your changes.

How can I create a brand new repository?

- Create a new repository called optical.

TERMINAL

```
$ git init optical  
Initialized empty Git repository in /home/repl/optical/.git/
```

How can I turn a existing project into a Git repository?

TERMINAL

```
pwd
$ pwd
/home/repl/dental
$ git init /home/repl/dental
Initialized empty Git repository in /home/repl/dental/.git/
$ git status
On branch master

Initial commit

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

    bin/
    data/
    report.txt
    results/

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

- 將本地資料夾變成可以給 Git 控管：
- `$ git init /Path/to/File`
- 結束後可以用 `git status` 看狀態

How can I create a copy of an existing repository?

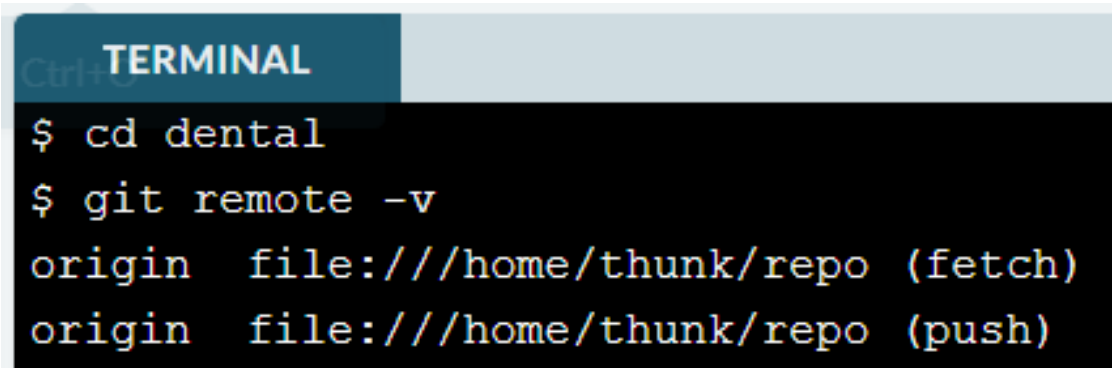
- \$ git clone URL destination/file_name
- (URL=identity of the repository you want to clone)

TERMINAL

```
$ pwd
/home/repl
$ git clone file:///home/thunk/repo /home/repl/dental
Cloning into '/home/repl/dental'...
remote: Counting objects: 58, done.
remote: Compressing objects: 100% (40/40), done.
remote: Total 58 (delta 13), reused 54 (delta 12)
Receiving objects: 100% (58/58), 6.06 KiB | 0 bytes/s, done.
Resolving deltas: 100% (13/13), done.
Checking connectivity... done.
```

How can I find out where a cloned repository originated?

- \$ git remote -v

A terminal window with a dark background and a light blue title bar. The title bar contains the word "TERMINAL" in white. The terminal shows the following commands and output:

```
$ cd dental
$ git remote -v
origin  file:///home/thunk/repo (fetch)
origin  file:///home/thunk/repo (push)
```

TERMINAL

```
$ cd dental
$ git remote -v
origin  file:///home/thunk/repo (fetch)
origin  file:///home/thunk/repo (push)
```


How can I define remotes?

- \$ git remote add remote-name URL
- \$ git remote rm remote-name

TERMINAL

```
$ cd dental  
$ git remote add thunk file:///home/thunk/repo
```

you are in the dental repository. Add <file:///home/thunk/repo> as a remote called thunk to it.

How can I pull in changes from a remote repository?

- Git keeps track of remote repositories so that you can pull changes from those repositories and push changes to them.
- `$ git pull remote branch` (get everything in branch)
- You are in quarterly-report branch of your local repository, the command: `$git pull thunk latest-analysis` (would get changes from latest-analysis branch in the repository associated with the remote called thunk and merge them into your quarterly-report branch.

```
$ git pull origin master
From file:///home/thunk/repo
 * branch                master      -> FETCH_HEAD
Updating 90675ac..28e7582
Fast-forward
 report.txt | 4 +++-
1 file changed, 3 insertions(+), 1 deletion(-)
```

Origin 為 remote
Master 為 branch

What happens if I try to pull when I have unsaved changes?

```
$ git pull origin master
From file:///home/thunk/repo
 * branch          master      -> FETCH_HEAD
Updating 7bc27ca..28e7582
error: Your local changes to the following files would be overwritten by merge:
    report.txt
Please, commit your changes or stash them before you can merge.
Aborting
$ git checkout --
M       report.txt
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.
  (use "git pull" to update your local branch)
$ git pull origin master
From file:///home/thunk/repo
 * branch          master      -> FETCH_HEAD
Updating 7bc27ca..28e7582
error: Your local changes to the following files would be overwritten by merge:
    report.txt
Please, commit your changes or stash them before you can merge.
Aborting
```

How can I push my changes to a remote repository?

- `$git push remote-name branch-name`

```
TERMINAL
$ cd dental
$ git add data/northern.csv
$ git commit -m "Added more northern data."
[master fde0f79] Added more northern data.
 1 file changed, 1 insertion(+)
$ git push origin master
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 373 bytes | 0 bytes/s, done.
Total 4 (delta 3), reused 0 (delta 0)
To file:///home/thunk/repo
 28e7582..fde0f79  master -> master
```

1. You are in the master branch of the dental repository, which has a remote called origin. You have changed data/northern.csv; add it to the staging area.
2. Commit your changes with the message "Added more northern data."
3. Push your changes to the remote repository's master branch.

What happens if my push conflicts with someone else's work?

```
TERMINAL
$ cd dental
$ git add data/northern.csv
$ git commit -m "Adding a record"
[master 2cb20f0] Adding a record
1 file changed, 1 insertion(+)
$ git push origin master
To file:///home/thunk/repo
 ! [rejected]        master -> master (non-fast-forward)
error: failed to push some refs to 'file:///home/thunk/repo'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
$ git pull origin master
From file:///home/thunk/repo
 * branch            master      -> FETCH_HEAD
Merge made by the 'recursive' strategy.
$ git push origin master
Counting objects: 2, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 332 bytes | 0 bytes/s, done.
Total 2 (delta 1), reused 0 (delta 0)
To file:///home/thunk/repo
 fde0f79..7e1909c  master -> master
```

- 1. you have made changes to the dental repository. Use git push to push those changes to the remote repository
- 2. In order to prevent you overwriting remote work, Git has refused to execute your push. Use git pull to bring your repository up to date with origin
- 3 Now that you have merged the remote repository's state into your local repository, try the push again.