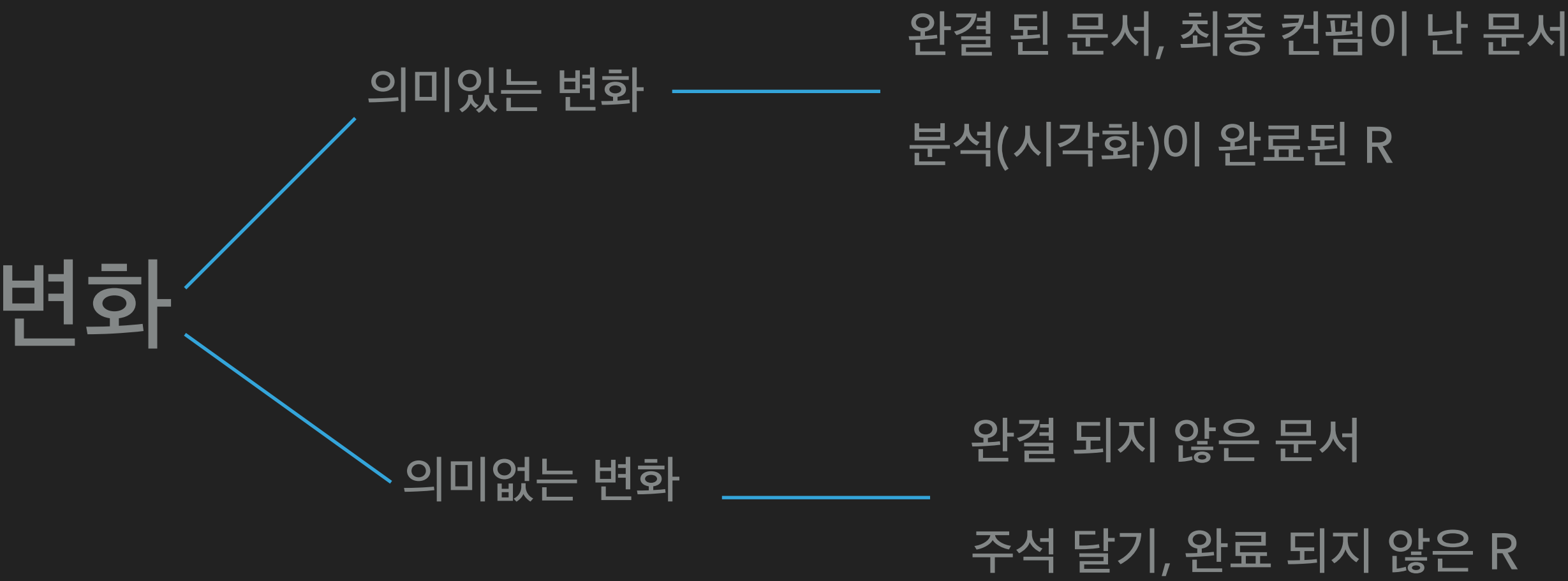


GIT 그리고 GITHUB 

버전 = 변화가 일어나는 모든 과정



버전 관리 시스템 = 변화들을 관리 하는 시스템

광고 카피.txt

발자크는 잉크 대신 커피로 원고지를 채웠다 -

원고지 넘어가는 수만큼 커피를 마셨던 세계적인 문호 발자크

(동서식품)

버전 관리 시스템 ?

광고 카피.txt

발자크는 잉크 대신 커피로 원고지를 채웠다 -

(동서식품)

버전 관리 시스템 ?

돌려놔

보통 어떻게 하시나요?

광고 카피1.txt

발자크는 잉크 대신 커피로 원고지를 채웠다 -

원고지 넘어가는 수만큼 커피를 마셨던 세계적인 문호 발자크

(동서식품)

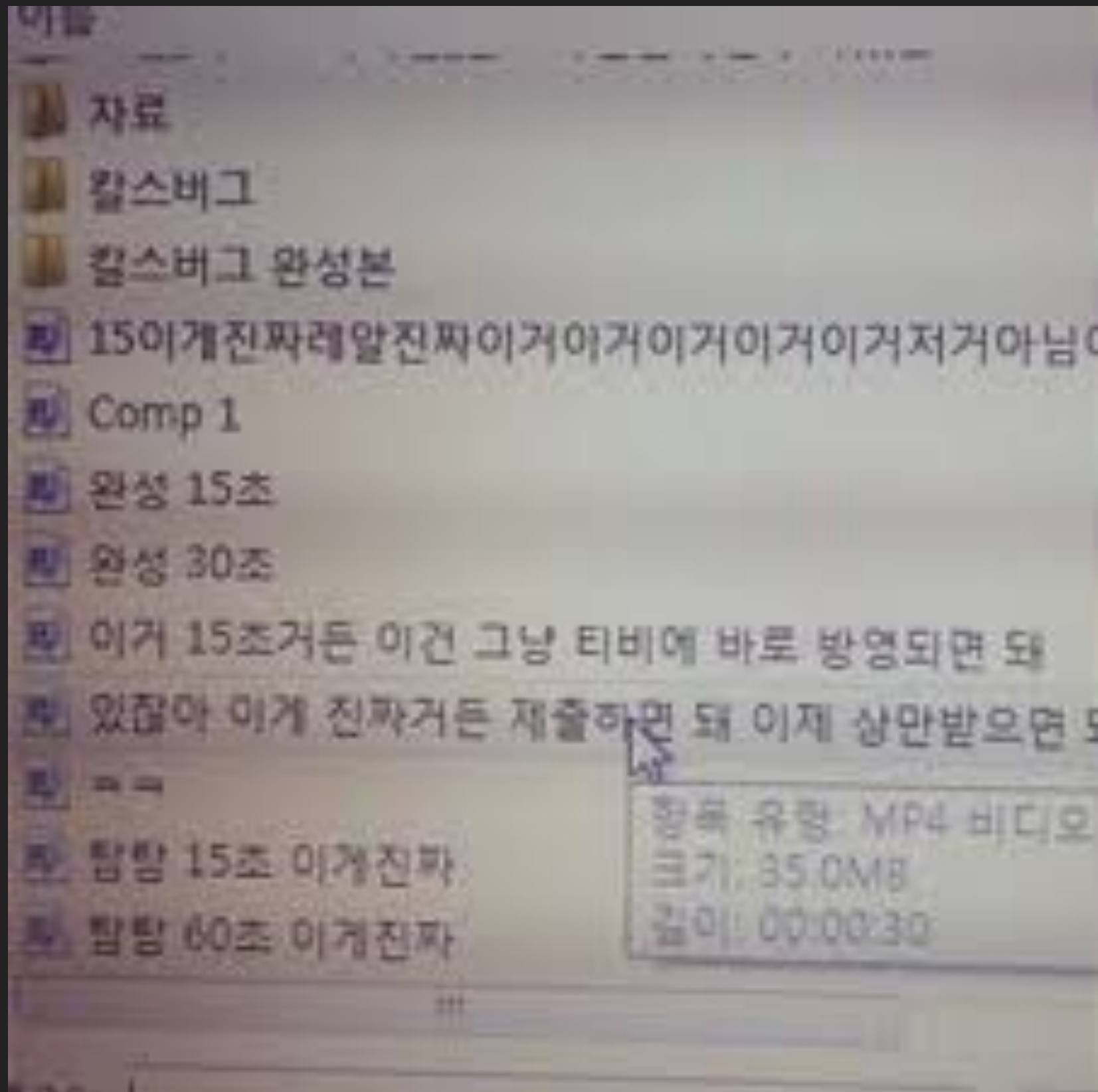
광고 카피2.txt

발자크는 잉크 대신 커피로 원고지를 채웠다 -

원고지 넘어가는 수만큼 커피를 마셨던 세계적인 문호 발자크

(동서식품)

버전 관리 시스템 ?



이게 진짜 마지막.txt

final.txt


final-1.txt

final!!!!!!!.txt

등등 ...

버전 관리 시스템 ?

실제 웹서비스에 적용이 되고 있는 사례



위키백과
우리 모두의 백과사전

대문

사용자 모임

요즘 화제

최근 바뀜

모든 문서 보기

임의 문서로

도움말

기부

도구

여기를 가리키는 문서

가리키는 글의 바뀜

파일 올리기

특수 문서 목록

고유 링크

문서 정보

위키데이터 항목

이 문서 인용하기

인쇄/내보내기

책 만들기

PDF로 다운로드

인쇄용 판

문서 토론

읽기 편집 역사 보기

검색

통계학

위키백과, 우리 모두의 백과사전.

통계학(統計學, 영어: statistics)은 수량적 비교를 기초로 하여, 많은 사실을 통계적으로 관찰하고 처리하는 방법을 연구하는 학문이다. 근대 과학으로서의 통계학은 19세기 중반 벨기에의 케틀레가 독일의 "국상학(國狀學, Staatenkunde, 넓은 의미의 국가학)"과 영국의 "정치 산술(Political Arithmetic, 정치 사회에 대한 수량적 연구 방법)"을 자연과학의 "확률 이론"과 결합하여, 수립한 학문에서 발전되었다.^[1]^[2]

목차 [숨기기]

1 개요

1.1 수리 통계학

1.2 어원

1.3 역할

2 용어

3 통계적 방법

3.1 실험 계획

3.2 설문지 작성

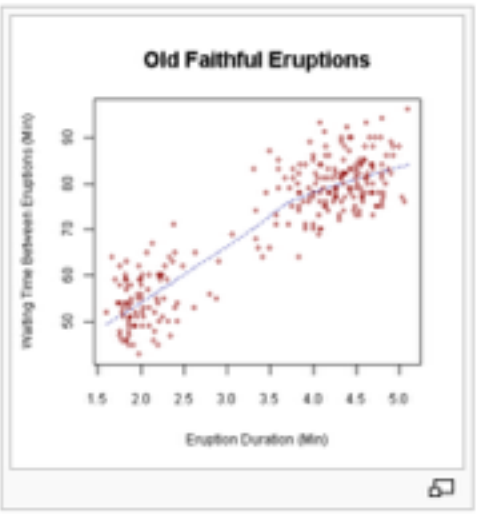
3.3 추론 통계

3.4 기술 통계

4 통계분석 소프트웨어

5 통계학 관련 학문

6 통계학의 변화



Old Faithful Eruptions

Waiting Time Between Eruptions (Min)

Eruption Duration (Min)



"통계학"의 두 판 사이의 차이

위키백과, 우리 모두의 백과사전.

2015년 3월 13일 (금) 15:28 판 (편집)

Garden of the zion (토론 | 기여)

(→용어)

(태그: 모바일 편집, 모바일 사이트를 이용한 편집)

← 이전 편집

2016년 3월 29일 (화) 07:00 기준 최신판 (편집) (편집 취소)

TedBot (토론 | 기여)

잔글 (봇: 틀 이름 및 스타일 정리)

(사용자 6명의 중간 판 9개는 보이지 않습니다)

1번째 줄:

[[파일:Oldfaithful3.png|thumb|right|200px]]

— '''통계학'''(統計學, {{lang|en|statistics}})은 수량적 비교를 기초로 하여, 많은 사실을 통계적으로 관찰하고 처리하는 방법을 연구하는 학문이다. 근대 과학으로서의 통계학은 19세기 중반 벨기에의 케틀레가 독일의 "국상학(國狀學, Staatenkunde, 넓은 의미의 국가학)"과 영국의 "정치 산술(Political Arithmetic, 정치 사회에 대한 수량적 연구 방법)"을 자연과학의 "확률 이론"과 결합하여, 수립한 학문에서 발전되었다.<ref>{{뉴스 인용|url=http://news.khan.co.kr/kh_news/khan_art_view.html?artid=201301182111155&code=900308|제목=명저 새로 읽기, 이연 해킹 "우연을 길들이다"|성=**이름**=윤해동, 한양대 비교역사문화연구소 교수|작성일자=2013-01-18|출판사=경향신문|확인일자=2013-03-05}}</ref><ref name="통계학">{{서적 인용|저자=정상윤, 오경환|제목=알기 쉬운 기초통계학|꺾쇠표= 예연도=2012|출판사=형설출판사|ISBN=9788947271820}}</ref>

1번째 줄:

[[파일:Oldfaithful3.png|thumb|right|200px]]

+ '''통계학'''(統計學, {{lang|en|statistics}})은 수량적 비교를 기초로 하여, 많은 사실을 통계적으로 관찰하고 처리하는 방법을 연구하는 학문이다. 근대 과학으로서의 통계학은 19세기 중반 벨기에의 케틀레가 독일의 "국상학(國狀學, Staatenkunde, 넓은 의미의 국가학)"과 영국의 "정치 산술(Political Arithmetic, 정치 사회에 대한 수량적 연구 방법)"을 자연과학의 "확률 이론"과 결합하여, 수립한 학문에서 발전되었다.<ref>{{뉴스 인용|url=http://news.khan.co.kr/kh_news/khan_art_view.html?artid=201301182111155&code=900308|제목=명저 새로 읽기, 이연 해킹 "우연을 길들이다"|성명=윤해동, 한양대 비교역사문화연구소 교수|날짜=2013-01-18|출판사=경향신문|확인일자=2013-03-05}}</ref><ref name="통계학">{{서적 인용|저자=정상윤, 오경환|제목=알기 쉬운 기초통계학|꺾쇠표= 예연도=2012|출판사=형설출판사|ISBN=9788947271820}}</ref>

GIT이란 ?

GIT = 프로젝트 관리도구

GIT = 프로젝트 관리도구

- 어떤 부분도 겹쳐쓰지 않게 프로젝트의 변경을 관리하는 버전관리 소프트웨어
- 리눅스 토발즈가 개발
- 이전에 만들어진 모든 변경사항의 “스냅샷”을 저장하기 때문에 이전 시점의 어떤 버전으로 되돌릴 수도 있다.
- 스냅샷에 변경한 사람, 변경된 내용, 변경한 파일(수정,삭제,추가), 변경한 시간 등을 저장한다.

(이 때 여기서 ‘변경’은 GIT에서 커밋이란 용어로 사용함)

GIT 기본

- 여러 명의 사용자가 한 개의 파일을 수정하였을 때 자신이 수정한 부분을 다른 사람이 지울 수 있는 상황이 발생할 수도 있다.

GIT을 사용하면 다른 사람이 수정한 부분을 합쳐서 한 개의 파일로 만들어 준다.

예) 광고카피.txt를 열어 A,B,C 가 작업을 한다.

A가 첫번째 줄에 작성한 문구 : 발자크는 잉크 대신 커피로 원고지를 채웠다

B가 엔터를 한 번 치고 두번째 줄에 작성한 문구 : 원고지 넘어가는 수만큼 커피를 마셨던 세계적인 문호 발자크

C가 엔터를 두 번 세번째줄에 작성한 문구 : 발자크(동서식품)

**GIT에서 첫번째, 두번째, 세번째 줄을 자동으로 병합(MERGE)하여
'발자크는 잉크 대신 커피로 원고지를 채웠다
원고지를 넘어가는 수만큼 커피를 마셨던 세계적인 문호 발자크
발자크(동서식품)
이렇게 하나의 문서로 만들어준다.**

GIT 기본

예) 광고카피.txt을 열어 A,B,C 가 작업을 한다.

A가 첫번째 줄에 작성한 문구 : 발자크는 잉크 대신 커피로 원고지를 채웠다

B가 첫번째 줄에 작성한 문구 : 원고지 넘어가는 수만큼 커피를 마셨던 세계적인 문호 발자크

C가 첫번째 줄에 작성한 문구 : 발자크(동서식품)

**GIT에서 GITHUB(REMOTE, 원격저장소, 서버)로 올릴 때(COMMIT) 때
첫번째 줄에 충돌(CONFLICT)이 났다는 에러 메시지가 뜨고 파일을 확인하여
수정 후에 하나로 취합(병합, MERGE)할 수도 있다.**

과정을 단순화하여 설명을 한 것이므로

충돌이 났을 때 충돌난 파일을 확인하여 수정 하여 하나로 취합할 수 있다는 것만 여기서 우선 기억 !

GIT 기본

예) 광고카피.txt을 열어 A,B,C 가 작업을 한다.

A가 첫번째 줄에 작성한 문구 : 발자크는 잉크 대신 커피로 원고지를 채웠다

B가 첫번째 줄에 작성한 문구 : 원고지 넘어가는 수만큼 커피를 마셨던 세계적인 문호 발자크

C가 첫번째 줄에 작성한 문구 : 발자크(동서식품)

그리고 A,B,C 순서대로 문서 저장을 했다. B가 작업한 카피가 최종 컨펌이 되었다.

GIT을 이용하여 B가 작업한 시점으로 되돌려 놓는다(REVERT).

저장소(Git Repository) - 파일이나 폴더를 저장해 두는 곳

파일이 변경 이력 별로 구분되어 저장, 비슷한 파일이라도 실제 내용 일부 문구가 서로 다르면 다른 파일로 인식하기 때문에 파일을 변경 사항 별로 구분해 저장

로컬 저장소(Local Repository) - 내 PC에 파일이 저장되는 개인 전용 저장소

원격 저장소(Remote Repository) - 파일이 원격 저장소 전용 서버에서 관리되며

여러 사람이 함께 공유하기 위한 저장소(ex , GITHUB)

로컬 저장소(Local Repository)



철수가



광고카피.txt 파일을



자신의 컴퓨터



저장소(project 폴더)에

추가한다 / 수정한다 / 삭제한다

커밋할 파일들 목록(STAGE)에 추가한다.

모든 변경이력을 저장한다.(COMMIT)

쉽게 말하면 커밋할 파일의 목록

원격 저장소(Remote Repository)로 수정한 파일을 업로드한다.(PUSH)

원격 저장소에 있는 파일을 다운 받는다.(PULL)

다른 사람이 광고카피.txt 파일에서 같은 줄에 있는 텍스트를 수정하여 충돌된 파일을 열어 수정한다. / 이전 버전으로 되돌린다.

원격 저장소(Remote Repository)

영희가 로컬 저장소에서 커밋한 내용을



저장소(myProject폴더)에서



광고카피.txt 파일을 수정

광고시안.txt 파일을 STAGE에
추가한 후 커밋한다.

올린다(PUSH)

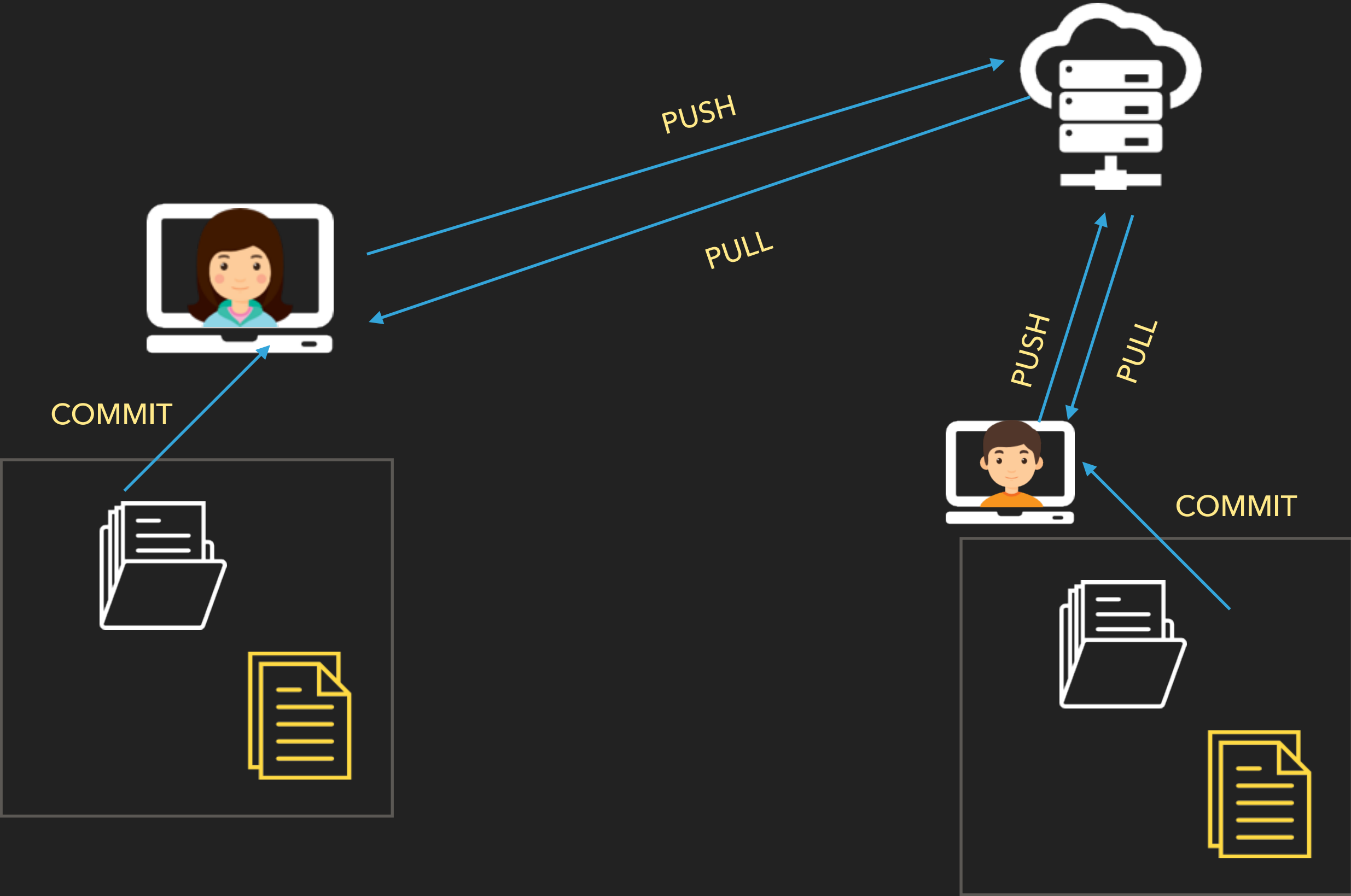
철수가 커밋한 내용을 내려받는다(PULL)
=원격 저장소에 있는 소스를 내려 받는다



PUSH

PULL





GIT 명령어

git init: 깃 저장소를 초기화한다. 깃의 저장소로 생성한다.

git config: "configure"의 준말, 처음에 깃을 설정할 때 가장 유용하다.

git help: 깃 명령어 도움말, "git help init"처럼 "git help 명령어"으로 사용하여 도움말 검색도 가능하다.

git status: 저장소 상태를 체크. 어떤 파일이 저장소 안에 있는지, 커밋이 필요한 변경사항이 있는지, 현재 저장소의 어떤 브랜치에서 작업하고 있는지 등을 볼 수 있다.

git add:저장소의 스테이지에 변경된 파일을 추가한다. 파일을 추가하면, 깃의 저장소 "스냅샷"에 포함된다.

git commit: 깃의 가장 중요한 명령어. 어떤 변경사항이라도 만든 후, 저장소의 "스냅샷"을 찍기 위해 이것을 입력한다. 보통 "git commit -m "Message hear." 형식으로 사용한다. -m은 명령어의 그 다음 부분을 메시지로 읽어야 한다는 것을 말한다.

git branch: 여러 협업자와 작업하고 자신만의 변경 내용만 이력에 남기고 싶을 때 사용. 이 명령어는 새로운 브랜치를 만들고, 자신만의 변경사항과 화일 추가 등의 커밋 타임라인을 만든다. 당신의 제목이 명령어 다음에 온다. 새 브랜치를 "cats"로 부르고 싶으면, git branch cats를 타이핑한다.

GIT 명령어

git checkout: 글자 그대로, 현재 위치하고 있지 않은 저장소를 “체크아웃”할 수 있다. 이것은 체크하길 원하는 저장소로 옮겨가게 해주는 탐색 명령이다. master 브랜치를 들여다 보고 싶으면,

git checkout master를 사용할 수 있고, git checkout cats로 또 다른 브랜치를 들여다 볼 수 있다.

git merge: 브랜치에서 작업을 끝내고, 모든 협업자가 볼 수 있는 master 브랜치로 병합할 수 있다.

git merge cats는 “cats” 브랜치에서 만든 모든 변경사항을 master로 추가한다.

git push: 로컬 컴퓨터에서 작업하고 당신의 커밋을 깃허브에서 온라인으로도 볼 수 있기를 원한다면,


이 명령어로 깃허브에 변경사항을 “push”한다.

git pull: 로컬 컴퓨터에서 작업할 때, 작업하고 있는 저장소의 최신 버전을 원하면,

이 명령어로 깃허브로부터 변경사항을 다운로드한다(“pull”).


GITHUB 사용


GITHUB 가입 (<https://github.com>)


 [Personal](#) [Open source](#) [Business](#) [Explore](#) [Pricing](#) [Blog](#) [Support](#) [Sign in](#) [Sign up](#)

Join GitHub

The best way to design, build, and ship software.

 **Step 1:**
Set up a personal account

 **Step 2:**
Choose your plan

 **Step 3:**
Tailor your experience

Create your personal account

Username

This will be your username — you can enter your organization's username next.

Email Address

You will occasionally receive account related emails. We promise not to share your email with anyone.

You'll love GitHub

Unlimited collaborators
Unlimited public repositories

✓

 Great communication

✓

 Frictionless development

✓

 Open source community

GITHUB 사용

온라인(GITHUB)에 저장소 만들기

- 'New Repository' 클릭

The screenshot shows the GitHub homepage. At the top, there is a search bar and navigation links for Pull requests, Issues, and Gist. A notification banner at the top states: "You don't have any verified emails. We recommend [verifying](#) at least one email. Email verification helps our support team verify ownership if you lose account access and allows you to receive all the notifications you ask for."

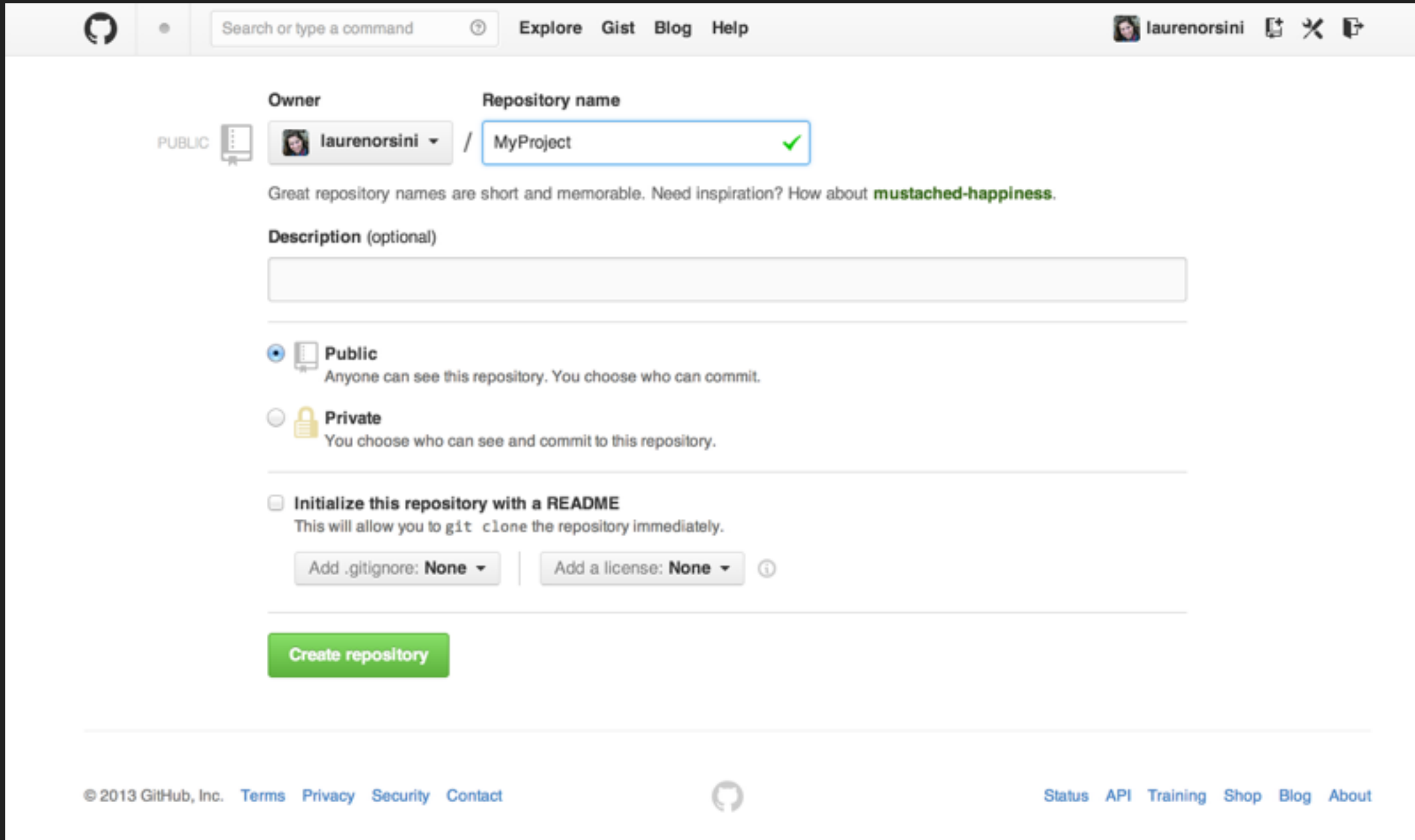
The main content area is divided into two columns. The left column features a dashed box containing a grid of repository icons and the text: "Discover interesting projects and people to populate your personal news feed. Your news feed helps you keep up with recent activity on repositories you [watch](#) and people you [follow](#)." Below this is a button labeled "Explore GitHub".

The right column contains a notification box titled "Reorder issues within a milestone" with a close button (X). Below this is a link "View all broadcasts". The "Your repositories" section shows a list of repositories: "devhaeyeon.github.com" and "test". A red box highlights the "New repository" button in the top right corner of the repository list.

GITHUB 사용

온라인(GITHUB)에 저장소 만들기

- 저장소 이름 입력, Public 선택 후 Create repository




Search or type a command

Explore Gist Blog Help

laurenorsini


Owner


Repository name

PUBLIC  laurenorsini / MyProject ✓

Great repository names are short and memorable. Need inspiration? How about **mustached-happiness**.

Description (optional)

☒  **Public**
Anyone can see this repository. You choose who can commit.


☐  **Private**
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately.

Add .gitignore: **None** | Add a license: **None** ⓘ

Create repository

© 2013 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Contact](#)



[Status](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)

GITHUB 사용

GIT BASH 실행

- 저장소 폴더를 생성, 저장소 폴더로 이동 후 오른쪽 버튼 GIT BASH클릭 (Window)
- 맥의 경우 터미널 실행 후 저장소 폴더로 이동 (cd 명령어 이용)
- * GIT BASH가 안깔려 있다면 GIT설치를 해야함.

(<https://git-scm.com/downloads>)

The screenshot shows the GitHub homepage. At the top, there is a search bar with the GitHub logo and the text "Search GitHub". To the right of the search bar are links for "Pull requests", "Issues", and "Gist". Further right are icons for notifications, a plus sign, and a user profile icon.

Below the navigation bar, there is a yellow banner with the text: "You don't have any verified emails. We recommend [verifying](#) at least one email. Email verification helps our support team verify ownership if you lose account access and allows you to receive all the notifications you ask for."

The main content area is divided into two columns. The left column features a dashed box containing a grid of project cards, each with a star icon and a repository name. To the right of this grid is the text: "Discover interesting projects and people to populate your personal news feed. Your news feed helps you keep up with recent activity on repositories you [watch](#) and people you [follow](#)." Below this text is a button labeled "Explore GitHub".

The right column contains a blue box with the text: "Reorder issues within a milestone X Reorder Issues within a Milestone to indicate priority using drag-and-drop." Below this box is a link that says "View all broadcasts".

At the bottom right, there is a section titled "Your repositories 6" with a green button labeled "New repository". Below this is a search bar with the text "Find a repository...". Under the search bar are tabs for "All", "Public", "Private", "Sources", and "Forks". Below the tabs is a list of repositories, including "devhaeyeon.github.com" and "test".

GITHUB 사용

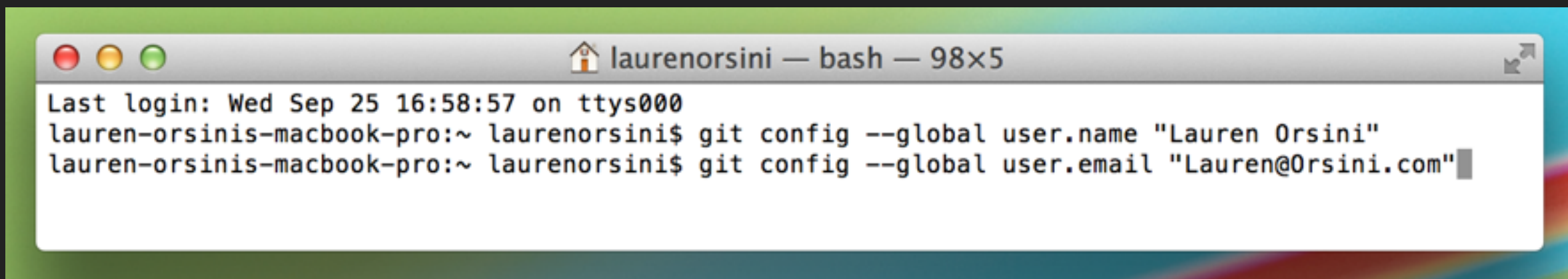
초기 깃 설정

1. `git config --global user.name "Your Name Here"` 입력

"Your Name Here" 부분에 자신의 이름을 쓴다.(그냥 쉽게 github에 가입할 때 쓴 이름을 기입)

2. `git config --global user.email "your_email@youremail.com"` 입력

"your_email@youremail.com" 부분에 자신의 이메일 주소를 쓴다. (그냥 쉽게 github에 가입할 때 쓴 이메일을 기입)

A screenshot of a macOS terminal window. The title bar shows a home icon, the username 'laurenorsini', and the shell 'bash' with window dimensions '98x5'. The terminal text shows the last login time and two successful git configuration commands: setting the global user name to 'Lauren Orsini' and the global user email to 'Lauren@Orsini.com'.

```
laurenorsini — bash — 98x5
Last login: Wed Sep 25 16:58:57 on ttys000
lauren-orsinis-macbook-pro:~ laurenorsini$ git config --global user.name "Lauren Orsini"
lauren-orsinis-macbook-pro:~ laurenorsini$ git config --global user.email "Lauren@Orsini.com"
```

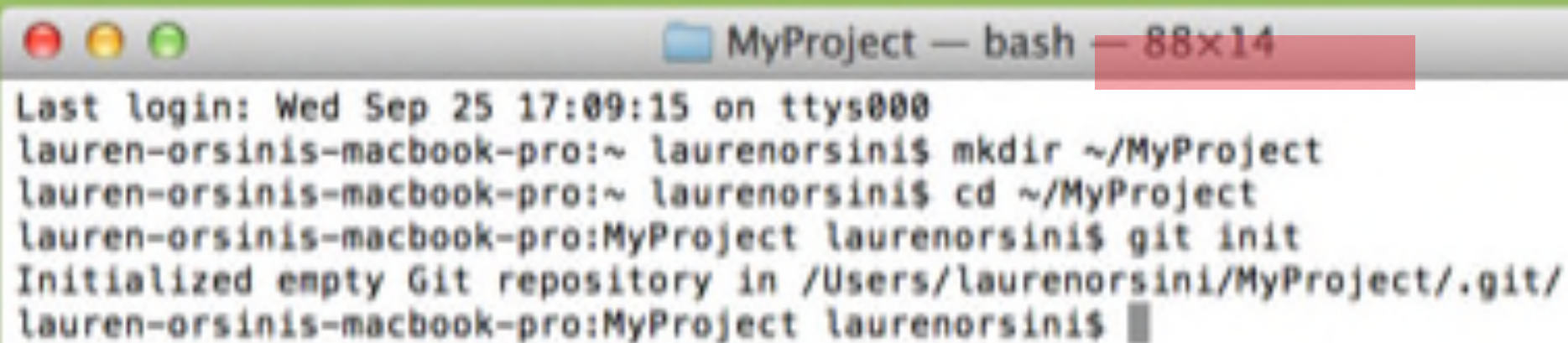
GITHUB 사용

로컬 저장소 만들기

git init

“initialize(초기화)”를 뜻한다. 이 코드를 입력하면 이 디렉토리를 로컬 깃 저장소라고 컴퓨터에게 말해주는 것

이 디렉토리를 Git-ready로 인식하고, 깃 명령어를 입력할 수 있다.



```
MyProject — bash — 88x14
Last login: Wed Sep 25 17:09:15 on ttys000
lauren-orsinis-macbook-pro:~ laurenorsini$ mkdir ~/MyProject
lauren-orsinis-macbook-pro:~ laurenorsini$ cd ~/MyProject
lauren-orsinis-macbook-pro:MyProject laurenorsini$ git init
Initialized empty Git repository in /Users/laurenorsini/MyProject/.git/
lauren-orsinis-macbook-pro:MyProject laurenorsini$
```

GITHUB 사용

로컬 저장소에 Readme.txt 파일 만들어보기

1. touch Readme.txt 입력

touch는 깃명령어가 아닌 CLI(도스, 터미널) 명령어으로써 파일을 생성하는 명령어이다.

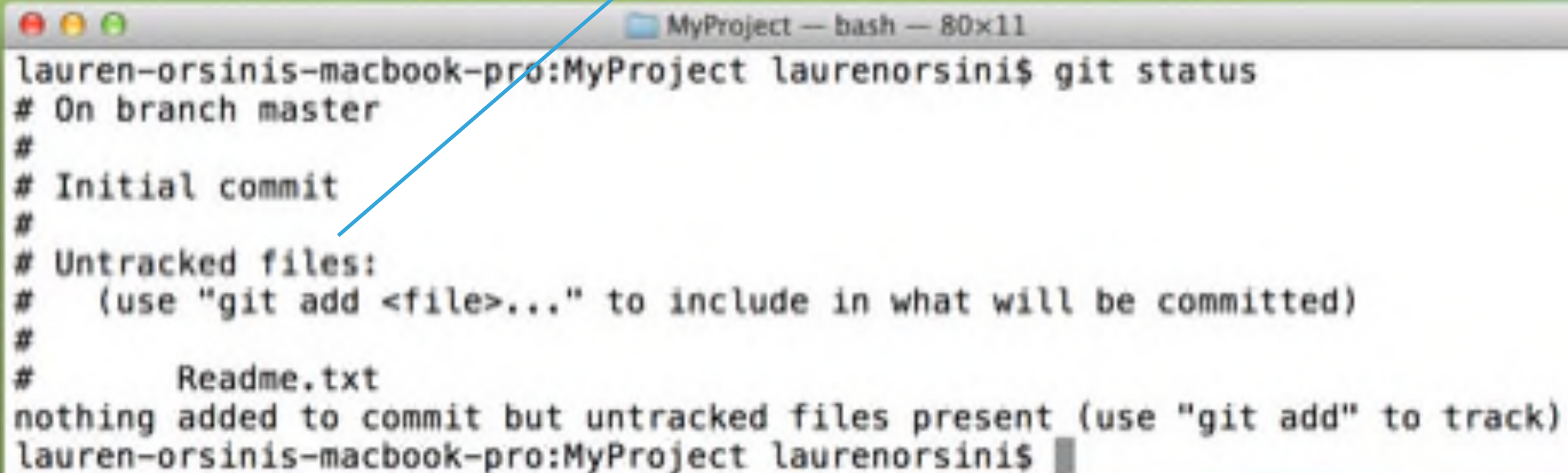
윈도우에서는 해당 로컬 저장소에서 그냥 Readme.txt 파일을 생성해도 된다.

GITHUB 사용

로컬 저장소에 상태 확인 하기

1. git status

커밋 스테이지에 파일을 올려둔 상태가 아니므로...

A screenshot of a macOS terminal window titled "MyProject — bash — 80x11". The terminal shows the output of the "git status" command. The output indicates that the user is on the master branch and has made an initial commit. It lists "Untracked files:" as "Readme.txt" and provides instructions to use "git add" to track the file. A blue arrow points from the Korean text above to the "Untracked files:" section of the terminal output.

```
lauren-orsinis-macbook-pro:MyProject laurenorsini$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       Readme.txt
nothing added to commit but untracked files present (use "git add" to track)
lauren-orsinis-macbook-pro:MyProject laurenorsini$
```


GITHUB 사용

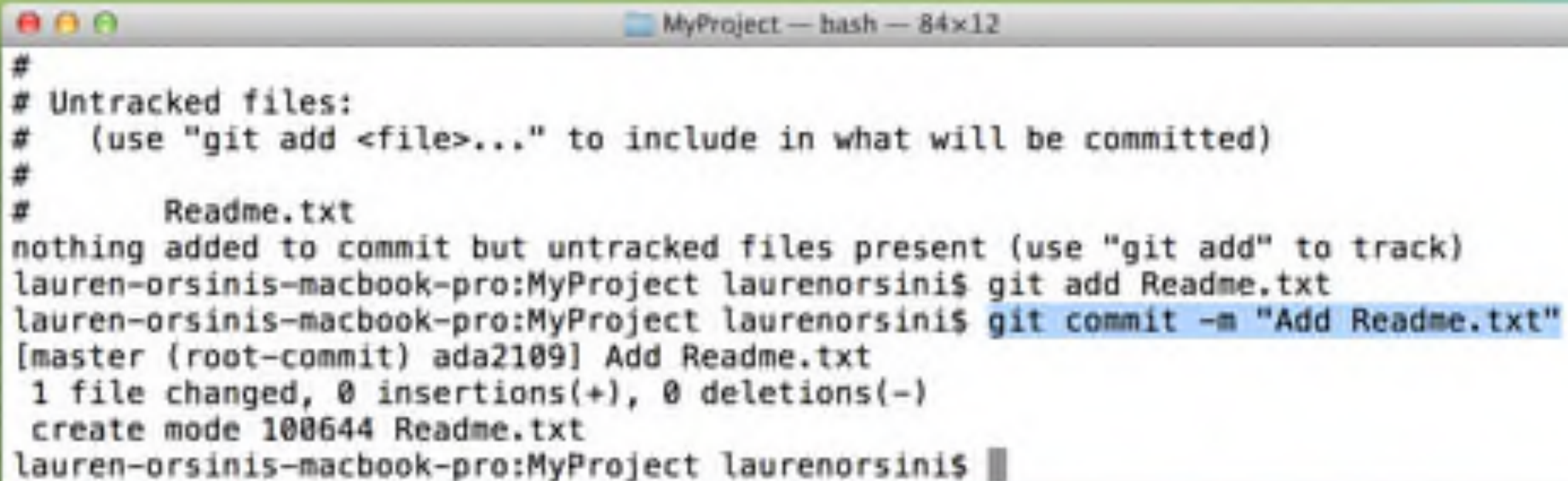
로컬 저장소의 스테이지에 파일 추가하기

1. git add Readme.txt 입력

로컬 저장소에 변경된 사항을 커밋하기

1.git commit -m "Add Readme.txt" 입력

"Add Readme.txt"부분에 변경된 내용을 입력한다.

A screenshot of a macOS terminal window titled "MyProject — hash — 84x12". The terminal shows the output of a git status command, indicating that Readme.txt is an untracked file. It then shows the execution of "git add Readme.txt" and "git commit -m 'Add Readme.txt'", resulting in a new commit on the master branch.

```
#  
# Untracked files:  
#   (use "git add <file>..." to include in what will be committed)  
#  
#       Readme.txt  
nothing added to commit but untracked files present (use "git add" to track)  
lauren-orsinis-macbook-pro:MyProject laurenorsini$ git add Readme.txt  
lauren-orsinis-macbook-pro:MyProject laurenorsini$ git commit -m "Add Readme.txt"  
[master (root-commit) ada2109] Add Readme.txt  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 Readme.txt  
lauren-orsinis-macbook-pro:MyProject laurenorsini$
```

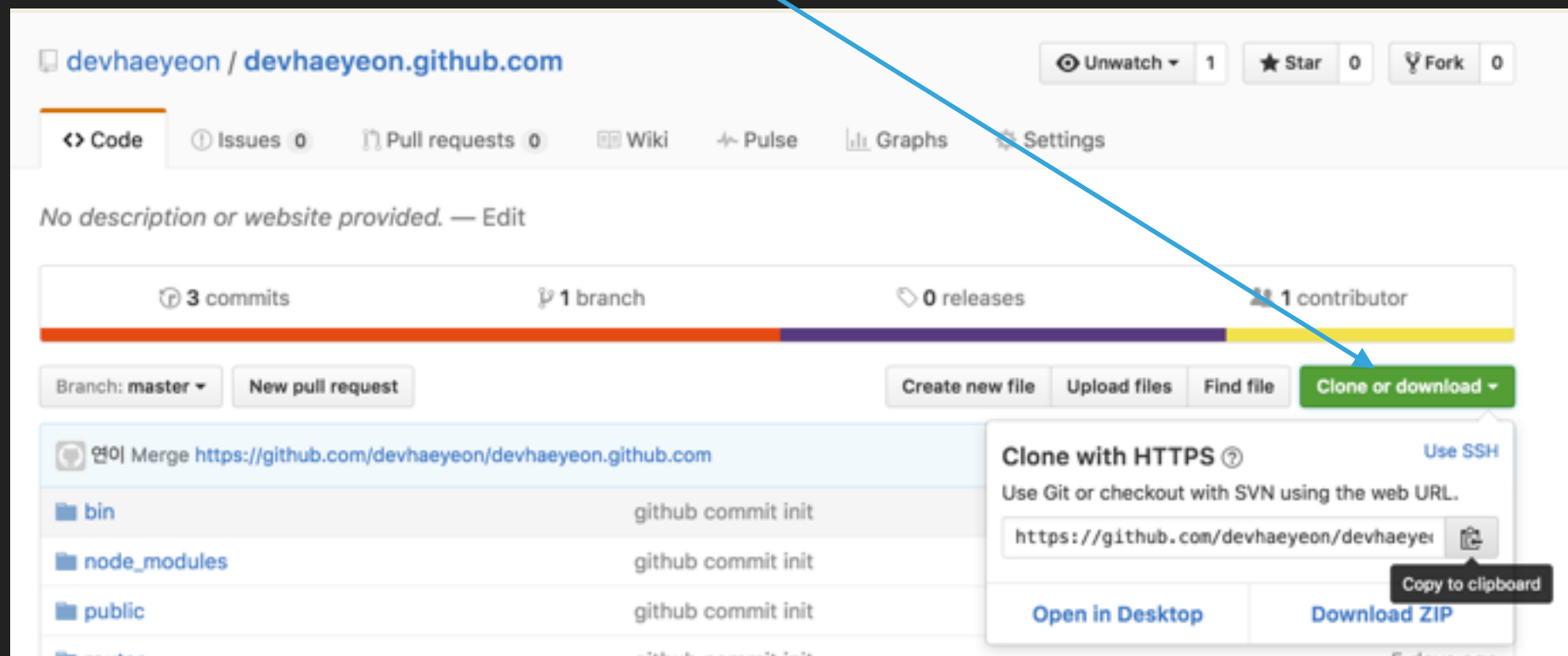
지금까지는 로컬 저장소에 파일을 변경된 사항을 스테이지에 추가하기, 커밋하는 방법

GITHUB 사용

로컬 저장소와 원격저장소(깃허브) 연결하기

1. `git remote add origin` <https://github.com/username/myproject.git> 입력

여기서 origin 뒤에 있는 git 주소는



clone or download를 통해 알 수 있다.

GITHUB 사용

원격저장소(GITHUB)로 커밋내용 및 파일 올리기

1. `git push origin master` 입력

원격저장소(GITHUB)에서 커밋내용 및 파일 내려 받기

1. `git pull origin master` 입력

참고 사이트

<http://nolboo.kim/blog/2013/10/06/github-for-beginner/>

해당 사이트를 참고 하였으며, 이미지를 사용하였습니다.

****추가 R에서 패키지를 제공한다고 함.**

<http://r-pkgs.xwmooc.org/git.html#git-rstudio> 사이트 참고