

11.(심화)Git의 사용1

-init, status, add,
commit, reset

2018.12

일병 김재형

Git의 설치

구름IDE에는 이미 설치가 되어있다.

Git의 설치-Windows

<https://git-scm.com/download/win>

에 접속하여 자신에게 맞는 버전의 Git을 설치한다.

The screenshot shows the Git website's 'Downloading Git' page. The header features the Git logo and the tagline '--local-branching-on-the-cheap', along with a search bar. The left sidebar contains navigation links: 'About', 'Documentation', 'Downloads' (highlighted in red), 'GUI Clients', 'Logos', and 'Community'. Below these links is a text box mentioning the 'Pro Git book' by Scott Chacon and Ben Straub. The main content area is titled 'Downloading Git' and features a large downward arrow icon. The text states: 'Your download is starting... You are downloading the latest (2.20.1) 32-bit version of Git for Windows. This is the most recent maintained build. It was released 2 days ago, on 2018-12-15.' It also provides a link to 'click here to download manually'. Below this, there is a section for 'Other Git for Windows downloads' listing 'Git for Windows Setup', '32-bit Git for Windows Setup.', '64-bit Git for Windows Setup.', 'Git for Windows Portable ("thumbdrive edition")', '32-bit Git for Windows Portable.', and '64-bit Git for Windows Portable.'. At the bottom, it mentions the current source code release is version 2.20.1 and provides a link to 'the source code'. The footer of the page shows the year '2018-19' and the name '김재형'.


git --local-branching-on-the-cheap

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloading Git

 **Your download is starting...**

You are downloading the latest (**2.20.1**) **32-bit** version of **Git for Windows**. This is the most recent [maintained build](#). It was released **2 days ago**, on 2018-12-15.

If your download hasn't started, [click here to download manually](#).

Other Git for Windows downloads

Git for Windows Setup
[32-bit Git for Windows Setup.](#)
[64-bit Git for Windows Setup.](#)

Git for Windows Portable ("thumbdrive edition")
[32-bit Git for Windows Portable.](#)
[64-bit Git for Windows Portable.](#)

The current source code release is version 2.20.1. If you want the newer version, you can build it from [the source code](#).

2018-19 김재형

Git의 설치-Windows

자세한 설치과정은 싸지방에서 설치할 수 없어 다른 링크를 남긴다.

<https://coding-factory.tistory.com/entry/Git-%EC%9C%88%EB%8F%84%EC%9A%B0%EB%B2%84%EC%A0%84-Git%EC%84%A4%EC%B9%98%ED%95%98%EA%B8%B0-Git-for-Windows>

※ 나중에 휴가나가면 추가하겠습니다.

Git의 설치-Windows

시작메뉴에서 Git-bash를 실행한다.



```
MINGW64: c:/Users/Eunki7
Eunki7@DESKTOP-8MP3KFP MINGW64 ~
$ git
```

Git의 설치-Windows

- Git Bash에서는 linux와 똑같은 명령어를 사용하여 git을 사용할 수 있다.
- Git CMD를 사용하면 윈도우 명령프롬프트를 통해 git을 사용할 수 있다.

구름IDE에서 Git을 시작하기 전에

처음 접속한 위치에서 Git을 사용하려고 하면 Git이 실행되지 않는다.

```
160986414_daggp@goorm:/workspace/PythonSeminar18$ git status
fatal: Not a git repository (or any parent up to mount point /workspace)
Stopping at filesystem boundary (GIT_DISCOVERY_ACROSS_FILESYSTEM not set).
160986414_daggp@goorm:/workspace/PythonSeminar18$ git init
fatal: Could not switch to '.git': Permission denied
160986414_daggp@goorm:/workspace/PythonSeminar18$
```

이는 권한설정 때문임으로, 본인의 과제 제출 directory에서 git을 사용하도록 한다.

Git 저장소 만들기

- 본인의 과제 디렉터리로 이동한다.

```
160986414_daggp@goorm:/workspace/PythonSeminar18$ cd Exercise/Jaehyeong/  
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$
```

- 본인의 디렉터리에서 git init를 입력한다.

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git init  
Initialized empty Git repository in /workspace/PythonSeminar18/Exercise/Jaehyeong/.git/
```

- Git 저장소가 만들어졌다.

Git 환경설정

기초 환경설정

- Commit에서 사용할 이메일과 이름을 입력한다.
- git config --global user.email "이메일주소"
- git config --global user.name "본인의 이름"
- git config --list를 통해 입력값을 확인한다.

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git config --list
user.email=imn00133@gmail.com
user.name=Jaehyeong Kim
core.repositoryformatversion=0
core.filemode=true
core.bare=false
```

Git 환경설정

잘못 입력했을 경우 다음과 같이 삭제한다.

—git config --global --unset 잘못 입력한 변수명

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git config --list
user.email=imn00133@gmail.com
user.name=Jaehyeong Kim
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git config --global --unset user.email
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git config --list
user.name=Jaehyeong Kim
core.repositoryformatversion=0
```

Git저장소 상태 확인하기

Git 저장소 상태 확인(git status)

- git status를 입력한다.
- 폴더 안에 있는 자신의 파일 상태가 보인다.

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git status
On branch master

Initial commit

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

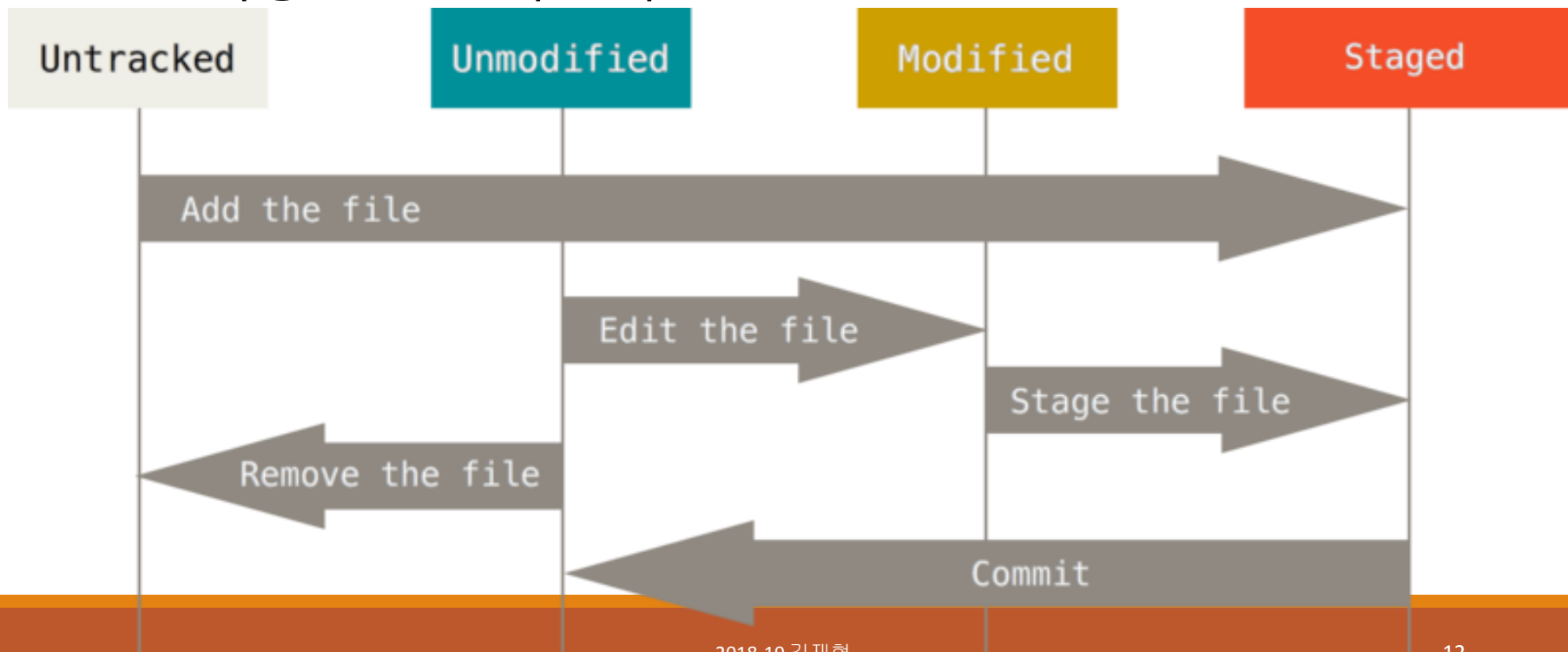
    vending_machine.py

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

Git저장소 상태 확인하기

파일의 상태

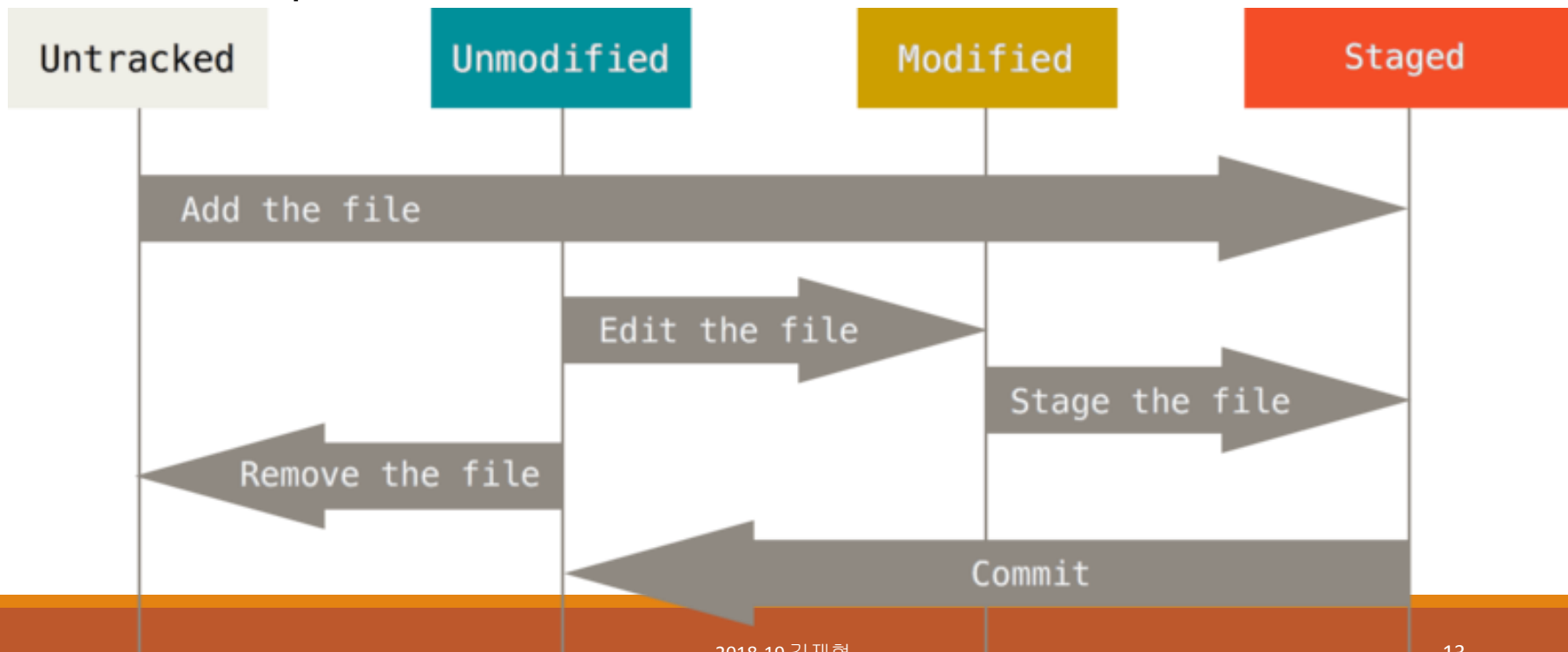
- 크게 Untracked(관리대상이 아님)와 Tracked(관리 대상임)으로 나뉜다.



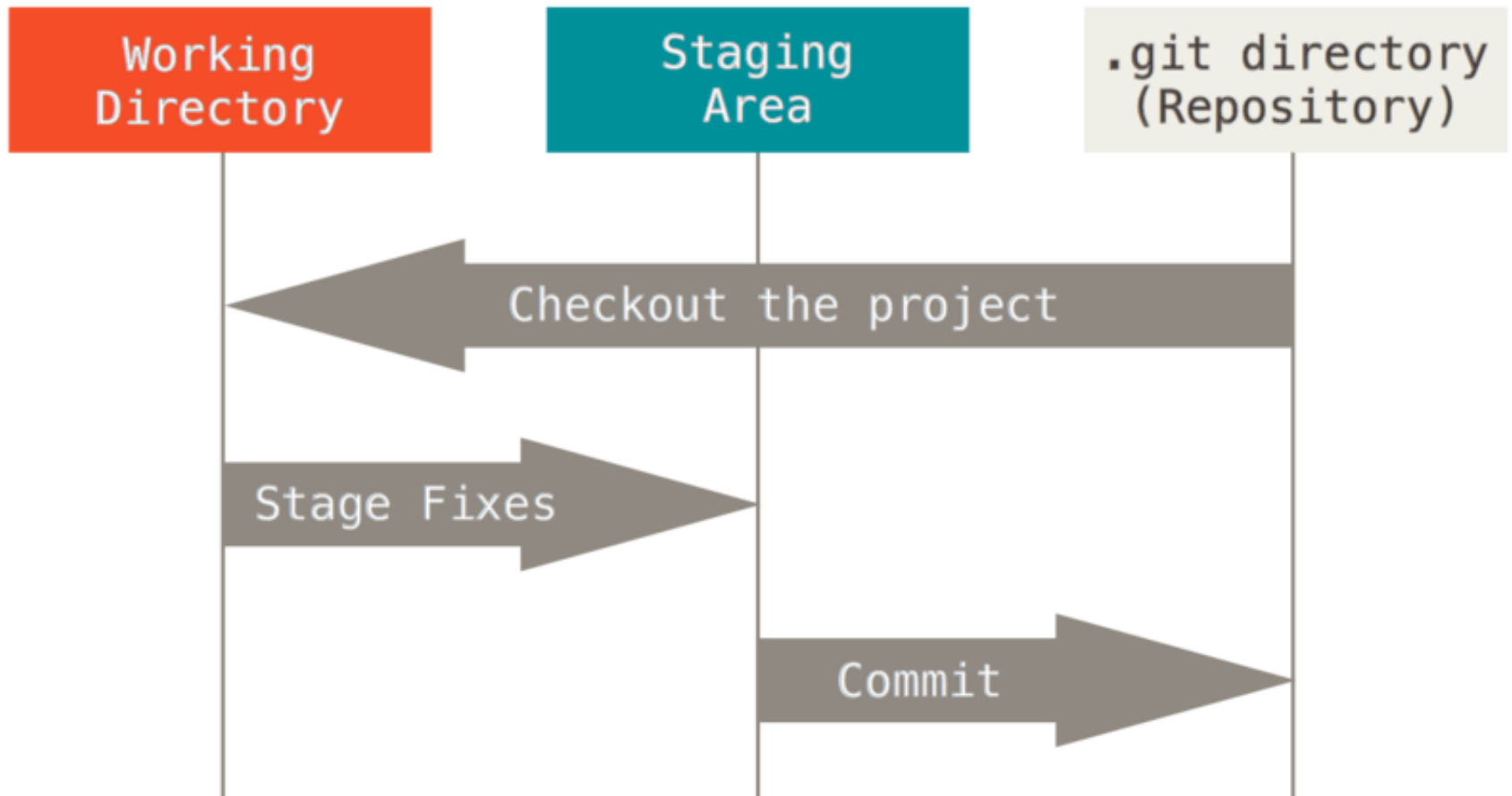
Git저장소 상태 확인하기

파일의 상태

- Tracked 파일은 Unmodified, Modified, Staged로 나뉜다.



Git 기초



Git저장소 상태 확인하기

Untracked파일: Working directory에 있는 파일 중 스냅샷에도 Staging Area에도 포함되지 않음

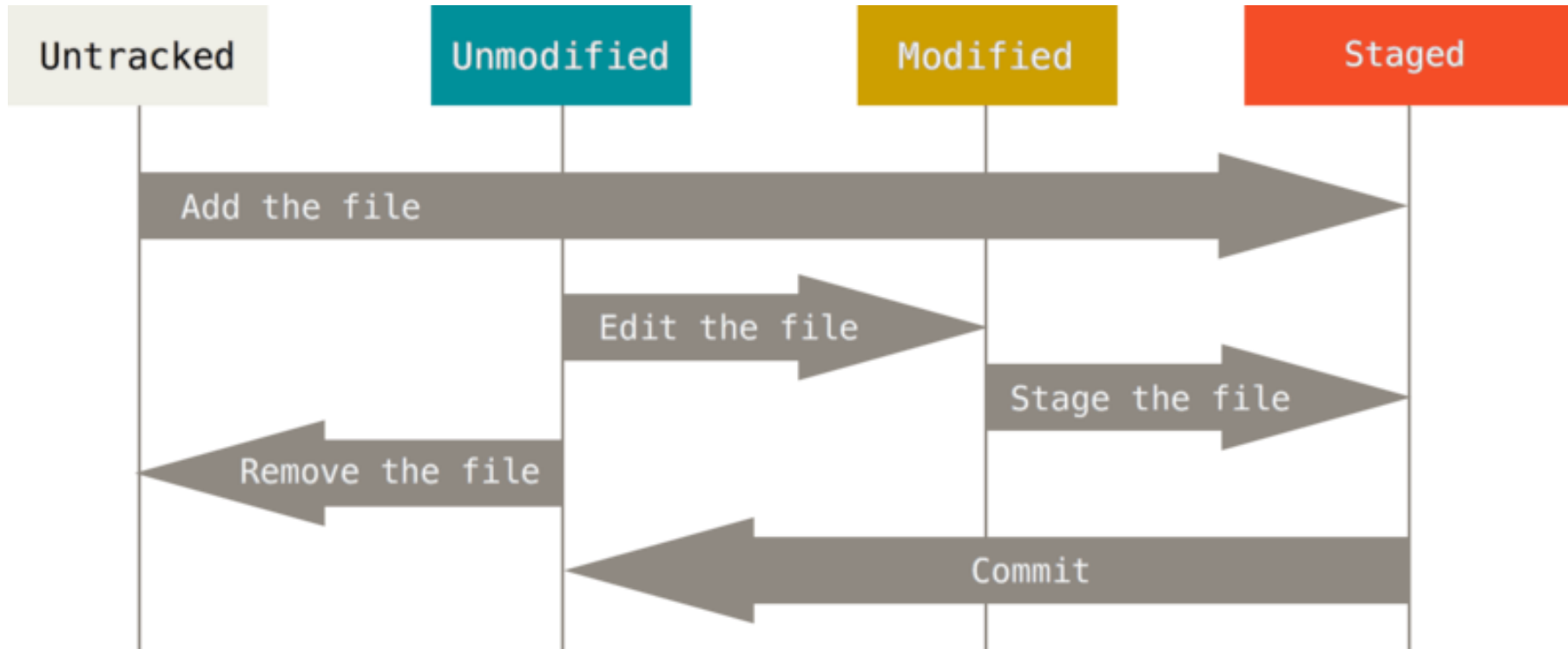
Git저장소 상태 확인하기

Tracked파일의 분류

- Unmodified(수정하지 않음)
- Modified(수정함)
- Staged(커밋으로 저장소에 저장하려고 준비)

Git저장소 상태 확인하기

파일의 상태 라이프 사이클



Git저장소 상태 확인하기

Git 저장소 상태 확인(git status)

- git status를 입력한다.
- 폴더 안에 있는 자신의 파일 상태가 보인다.

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git status
On branch master

Initial commit

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

    vending_machine.py

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

파일을 새로 추적하기

파일을 Untracked에서 Tracked상태로 만든다.

- git add (집어넣을 파일)
- git add .을 클릭하면 현재 디렉터리 내의 모든 파일이 Staged상태가 된다.
- 각각의 파일을 따로 commit하기 위해서 디렉터리나 각 파일을 따로 add해도 된다.

※ 보통 각 파일이나 관리 단위에 따라 add로 추가한 후 commit하는 것을 추천한다.

파일을 새로 추적하기

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git add .
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git status
On branch master
```

Initial commit

```
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   vending_machine.py
```

Modified 상태의 파일을 Stage하기

Modified 상태에서 Stage로 만들어야 한다.

```
root@goorm:/workspace/PythonSeminar18(master)# git status
On branch master
Your branch is up-to-date with 'origin/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:   README.md
```

Modified 상태의 파일을 Stage하기

```
root@goorm:/workspace/PythonSeminar18(master)# git add README.md
root@goorm:/workspace/PythonSeminar18(master)# git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   README.md
```

Staged 파일 commit전 수정

git add를 한 시점을 저장

```
root@goorm:/workspace/PythonSeminar18(master)# git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   README.md

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:   README.md
```

Staged 파일 되돌리기

git reset을 입력하면 staged하기 이전으로 되돌아간다.

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   vending_machine.py

160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git reset
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git status
On branch master

Initial commit

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

        vending_machine.py
```


git commit

git commit -m "작업 내용 입력"

```
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git commit -m "init commit"
[master (root-commit) 139f732] init commit
1 file changed, 2 insertions(+)
create mode 100755 vending_machine.py
160986414_daggp@goorm:/workspace/PythonSeminar18/Exercise/Jaehyeong$ git status
On branch master
nothing to commit, working directory clean
```

- ※ 쌍따옴표를 닫지 않고 여러 줄을 입력해도 된다.
- ※ 보통 첫 줄은 요약, 그 다음 줄부터 자세한 내용을 입력하도록 알려져있다.

git commit-vi 편집기

git commit만 쳐서 vi 편집기로 들어갔을 때
—esc를 누른 뒤 :q를 입력하여 나온다.

```
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   new file:   vending_machine.py
#
~
~
~
~
~
~
~
~
~
~
~
```

git commit-vi편집기

이후 다시 commit을 한다.

만일 add를 따로 하고 싶지 않다면,
git commit -a -m "작업내용"으로도 할 수 있다.

Git commit

Git 커뮤니티에서는 버전 관리를 위해 commit 을 자주하는 것을 권고한다.

git과제1

본인의 과제 폴더에서 진행한다.

- 본인의 과제 디렉터리에서 git init로 git 저장소를 생성한다.
- Git에 대한 global 환경설정을 한다.
- add를 통해 파일을 tracking상태로 만든다.
- commit을 통해 처음으로 파일을 git 저장소에 저장한다.