



Git / GitHub

(1주차)

Contents

**I
Git
Intro**

**II
Hands-on
(Basic)**

**III
Hands-on
(Advanced)**

**IV
Git
Review**

Contents

**I
Git
Intro**

**II
Hands-on
(Basic)**

**III
Hands-on
(Advanced)**

**IV
Git
Review**

What is Git?

- Formal **version control system**
- Developed by Linus Torvalds(developer of Linux)
 - used to manage the source code for Linux
- Tracks any content
 - source code
 - data analysis projects
 - websites
 - presnetations



Git 의 기능

Ctrl+c, v 나 Alzip 압축파일 관리법

Source code management **tool**

그 사이에 뭐가 바뀌었는지
차이 (Diff) 를 알 수 없다 .



과제 1_ 최종 _2016_02_28.zip

과제 1_ 진짜최종 _2016_02_29.zip

과제 1_ 진짜진짜최종 _2016_03_01.zip

Git 의 기능

Ctrl+c, v 나 Alzip 압축파일 관리법

Source code management **tool**

그 사이에 뭐가 바뀌었는지
차이 (Diff) 를 알 수 없다 .

Ctrl + c, v 를 할수록
차지하는 **용량** X 2
X 3 ... + diff



과제 1_ 최종 _2016_02_28.zip

과제 1_ 진짜최종 _2016_02_29.zip

과제 1_ 진짜진짜최종 _2016_03_01.zip

Git 의 기능

Ctrl+c, v 나 Alzip 압축파일 관리법

Source code management **tool**

그 사이에 뭐가 바뀌었는지
차이 (Diff) 를 알 수 없다 .

Ctrl + c, v 를 할수록
차지하는 **용량** X 2
X 3 ... + diff



과제 1_ 최종 _2016_02_28.zip

과제 1_ 진짜최종 _2016_02_29.zip

과제 1_ 진짜진짜최종 _2016_03_01.zip

< History 관리가능 >
차이 (Diff) 가 무엇이고
수정 이유를 Log 를 남길수있다 .

Git 의 기능

Ctrl+c, v 나 Alzip 압축파일 관리법

Source code management **tool**

그 사이에 뭐가 바뀌었는지
차이 (Diff) 를 알 수 없다 .

Ctrl + c, v 를 할수록
차지하는 **용량** X 2
X 3 ... + diff



과제 1_ 최종 _2016_02_28.zip

과제 1_ 진짜최종 _2016_02_29.zip

과제 1_ 진짜진짜최종 _2016_03_01.zip

< History 관리가능 >

차이 (Diff) 가 무엇이고
수정 이유를 Log 를 남길수있다 .

< 타임머신 기능 >

현재파일들은 안전한 상태로
과거상태 그대로 복원가능 (반대도 가능)
(각 버전별 **차이만 저장해서 size 감소**)

Git 의 기능

Ctrl+c, v 나 Alzip 압축파일 관리법

Source code management **tool**

그 사이에 뭐가 바뀌었는지
차이 (Diff) 를 알 수 없다 .

Ctrl + c, v 를 할수록
차지하는 **용량** X 2
X 3 ... + diff



과제 1_ 최종 _2016_02_28.zip

과제 1_ 진짜최종 _2016_02_29.zip

과제 1_ 진짜진짜최종 _2016_03_01.zip

< History 관리가능 >
차이 (Diff) 가 무엇이고
수정 이유를 Log 를 남길수있다 .

< 특정 버전 관리 >
tag 나 release 로
관리가능

< 타임머신 기능 >
현재파일들은 안전한 상태로
과거상태 그대로 복원가능 (반대도 가능)
(각 버전별 **차이만 저장해서 size 감소**)

Git 의 기능

Ctrl+c, v 나 Alzip 압축파일 관리법



Git 배우는데 시간소비하느니
Code 한줄이라도 더 개발 ..)

Source code management **tool**



좋은건 알겠는데 ..
Git 을 쓸 이유가 부족 ..

Wants 는 맞지만 Needs 는 아니야



과제 1_ 최종 _2016_02_28.zip

과제 1_ 진짜최종 _2016_02_29.zip

과제 1_ 진짜진짜최종 _2016_03_01.zip



git 이란?

개발과정, 소스파일 등을 관리하는 도구

History 관리가 되어 개발 되어온 과정, 역사를 볼 수 있고, 특정시점으로 복구 가능

왜 git 을 써야할까?

개발자로서 평생 간단한 코드만 짤 순 없기 때문에

가면 갈수록 점점 더 복잡하고 관리하기 힘든 코드를 다루는데 도움을 줄 강력한 도구!

혹은 팀 프로젝트에 있어서 협업을 도와 줄 수 있는 유용한 도구!

Contents

I
Git
Intro

II
**Hands-on
(Basic)**

III
Hands-on
(Advanced)

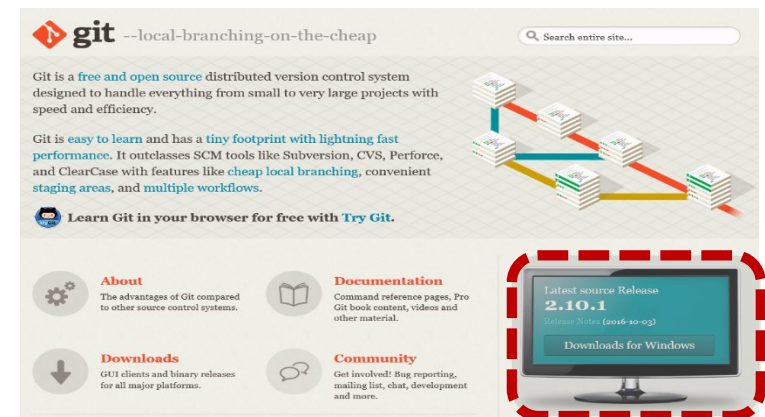
IV
Git
Review

Git/GitHub 실습 본격적인 시작에 앞서서..

- 이 수업(오픈소스소프트웨어실습)은 Git/Github 수업 X 강의 후 부족한 부분을 연습 필요
- 본 강의는 실습 90% 이론 10%
- 내 손으로 명령어를 직접 입력해보자
- 처음 다루는 팀원들을 도와주며 친해지자 (팀 프로젝트 진행 사전 대비)
- 터미널 환경 경험하자
- GUI(Graphic User Interface)대신 CLI(Command Line Interface)경험하자
- Vi에디터를 사용해보자

Git 설치

- 다운로드: <https://git-scm.com/download>
- 리눅스의 경우 배포판에 따라
 - 데비안 계열 : `$ sudo apt-get install git-all`
 - Fedora 배포판 : `$ sudo yum install git-all`
- GUI: git-gui, GitKraKen, Source Tree



Step0 Git 설치 확인

1) Git-bash 혹은 터미널 실행 후에

1-1) CLI 창에 명령어를 입력하여 Git 설치 여부 확인 (아래와 같이 나올 시 설치 O)

```
$ git
```

```
VR01@DESKTOP-4LN2P3L MINGW64 ~
$ git
usage: git [--version] [--help] [-C <path>] [-c name=value]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

These are common Git commands used in various situations:


start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one


work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv         Move or rename a file, a directory, or a symlink
  reset      Reset current HEAD to the specified state
  rm         Remove files from the working tree and from the index


examine the history and state (see also: git help revisions)
  bisect     Use binary search to find the commit that introduced a bug
  grep       Print lines matching a pattern
```

※ Git 과 관련 된 주요 명령어 설명이니 자주 활용

Step1 저장소 만들기 및 초기화

1) Git bash를 실행 (명령어칠 준비), 폴더 생성하기

```
$ mkdir git-test
```

2) 경로 이동 (pwd 명령어로 현재경로 확인하기)

```
$ cd git-test
```

3) 해당 폴더를 git 초기화 (전 후로 ls -al 명령어로 생성된 .git폴더 확인하기)

```
$ git init
```


Step2 add(파일 등록)

1) 임의 파일 생성(vim 에디터 저장하고 나가기 :wq)

```
$ vim test1.txt
```

2) 현재 파일의 상태 확인

```
$ git status
```

3) 파일을 저장소에 추가(추가 후 다시 git status로 확인/파일 내용 : a)

```
$ git add test1.txt
```

Step3 Commit(버전 만들기)

1) step2와 이어서 진행

2) 나의 이메일과 이름 등록

```
$ git config --global user.email "email"
```

```
$ git config --global user.name "username"
```

3) commit 생성 후 메시지 작성(저장하고 나가기 :wq)

```
$ git commit
```

4) log를 통해 commit 등록 확인

```
$ git log
```

5) test.txt 파일 내용 수정 후 다시 add, commit 연습
(파일 내용 : b / commit message : 2)로 수정

```
$ vim test1.txt
```

```
$ git add test.txt
```

```
$ git commit
```

Step4 변경사항 확인

- 1) step3와 이어서 진행
- 2) text.txt 파일 내용 수정 후 다시 add, commit 연습
(파일 내용 : c / commit message : 3)로 수정

```
$ vim test1.txt
```

```
$ git diff
```

```
$ git add test.txt
```

```
$ git commit
```

- 3) 변경사항을 확인 하는 다양한 방법
- 3-1) 순차적으로 변경사항을 확인

```
$ git log -p
```

- 3-2) 현재 커밋내용과 지정 커밋 변경사항을 확인

```
$ git diff 비교할 커밋 ID
```

Step5 commit 간소화

- 1) step4와 이어서 진행
- 2) text.txt 파일 내용 수정 후 다시 add, commit 연습
(파일 내용 : d) 로 수정

```
$ vim test1.txt
```

```
$ git add test.txt
```

- 3) Commit을 간단하게 방법

```
$ git commit -m “커밋 메시지”
```

GitHub 회원가입

Join GitHub

Create your account

Username *

Email address *



?




Choose a plan

Individual

Pick the plan that's right for you, personally.

Team

Choose a plan to help your team grow and collaborate.




Free

\$0 USD

The basics of GitHub for every developer

Choose Free

- ✓ Unlimited public repositories
- ✓ Unlimited private repositories
- ✓ Limited to 3 collaborators for private repositories
- ✓ 2,000 total Action minutes/month
See pricing details
- ✓ 500MB of GitHub Packages storage
See pricing details
- ✓ Advanced vulnerability scanning for public repositories
- ✓ Automated security updates
- ✓ GitHub Security Advisories
- ✓ Issues and bug tracking
- ✓ Project management



Pro

\$7 USD

Per month

Pro tools for developers with advanced requirements










Choose Pro

- ← Includes everything in Free
- ✓ Unlimited collaborators
- ↑ 3,000 total Action minutes/month
See pricing details
- ↑ 1GB of GitHub Packages storage
See pricing details
- ✓ Private GitHub Pages and Wikis
- ✓ Private protected branches
- ✓ Code owners
- ✓ Repository insights

GitHub 회원가입

What do you plan to use GitHub for?

(Select up to 3)

 Learn to code	 Learn Git and GitHub	 Host a project (repository)
 Create a website with GitHub Pages	 Collaborating with my team	 Find and contribute to open source
 School work and student projects	 Use the GitHub API	 Other

I am interested in:

languages, frameworks, industries

We'll connect you with communities and projects that fit your interests.

For example: `kotlin` `xml` `publishing`

Complete setup

Skip this step



Please verify your email address

Before you can contribute on GitHub, we need you to verify your email address.

An email containing verification instructions was sent to `skkusosc@gmail.com`.

Resend verification email

Change your email settings

[GitHub] Please verify your email address.

받은편지함

GitHub <noreply@github.com>
나에게



Almost done, @soscskku! To complete your GitHub sign up, we just need to verify your email address:

skkusosc@gmail.com.

Verify email address

Once verified, you can start using all of GitHub's features to explore, build, and share projects.

Button not working? Paste the following link into your browser: https://github.com/users/soscskku/emails/99554249/confirm_verification/26568d4ca59f0bfb73f9bb873b46257070341b94

You're receiving this email because you recently created a new GitHub account or added a new email address. If this wasn't you, please ignore this email.

[Email preferences](#) · [Terms](#) · [Privacy](#) · [Sign into GitHub](#)

원격 저장소 만들기

The screenshot shows the GitHub homepage. At the top, there's a navigation bar with the GitHub logo, a search bar, and links for Pull requests, Issues, Marketplace, and Explore. Below the navigation bar, a message states "skkusosc@gmail.com is already verified." On the left sidebar, under "Create your first project", the "Create repository" button is highlighted with a red box. Below this, there's a section for "Working with a team?" with a "Create an organization" button. The main content area features a large green box with the text "Learn Git and GitHub without any code!" and two buttons: "Read the guide" and "Start a project". Below this, there's a section for "Discover interesting projects and people to populate your personal news feed." with an "Explore GitHub" button. On the right sidebar, there's a section for "Explore repositories" listing several repositories like "Oteemo/charts", "Helm chart repository", "Smarty", "Chuckerteam/chucker", and "babel/babel".

Search or jump to...

Pull requests Issues Marketplace Explore

skkusosc@gmail.com is already verified.

Create your first project

Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it.

Create repository Import repository

Working with a team?

GitHub is built for collaboration. Set up an organization to improve the way your team works together, and get access to more features.

Create an organization

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Read the guide Start a project

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.

Explore GitHub

ProTip! The feed shows you events from people you follow and repositories you watch.

Subscribe to your news feed

Explore repositories

Oteemo/charts
Helm chart repository
Smarty ★ 22

Chuckerteam/chucker
An HTTP inspector for Android & OkHTTP (like Charles but on device) - More Chucker than Chuck
Kotlin ★ 1k

babel/babel
Babel is a compiler for writing next generation JavaScript.
JavaScript ★ 36.2k

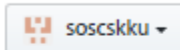
Explore more →

원격 저장소 만들기

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner



Repository name *

git-test



Great repository names are short and memorable. Need inspiration? How about [redesigned-fortnight?](#)

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ Initialize this repository with a README

This will let you immediately clone the repository to your computer.

Add .gitignore: None

Add a license: None



Create repository

Step6 Push(원격 저장소에 저장하기)

- 1) step5와 이어서 진행
- 2) 지금까지 작업한 commit 들을 원격 저장소에 저장

```
$ git remote add origin 원격저장소 주소
```

```
$ git push -u origin master
```

- 3) 원격 저장소의 아이디와 비밀번호 입력

Contents

I
Git/GitHub
Intro

II
Hands-on
(Basic)

III
**Hands-on
(Advanced)**

IV
Git/GitHub
Review

실습

- 과목 합계 및 평균을 구하는 코딩을 Git을 사용하여 버전 관리 해보기

문 제

다음 출력 예와 같이 출력되는 프로그램을 작성하시오.
합계와 평균은 수식을 이용하세요.

출력 예 [Copy]

```
국어 90
수학 80
영어 100
합계 270
평균 90
```

실습 결과 제출

- 29~40 page 실습 수행 후 결과 제출
 - Github username(ID)
 - Github email
 - 실습 저장소 url 혹은 캡처 화면을 넣으셔서 제출

Step1 초기화 및 첫 commit하기

1) 나의 이메일과 이름을 적자 (이미 등록했다면 건너뛰기)

```
$ git config --global user.email "본인메일"
```

```
$ git config --global user.name "본인이름,닉네임"
```

2) Git bash를 실행 (명령어칠 준비), 폴더 생성하기

```
$ mkdir report-card
```

3) 경로 이동 (pwd 명령어로 현재경로 확인하기)

```
$ cd report-card
```

4) 해당 폴더를 git 초기화 (ls-A 명령어로 생성된 .git폴더 확인하기)

```
$ git init
```

5) 프로그램 문제 PDF 파일 추가 (커밋할 목록에 추가 add) (commit1 폴더내 파일 활용)

```
$ git add report_card.pdf
```

6) 첫 commit 하기 (역사 한단위 만들기)

```
$ git commit -m "report card: Add question PDF"
```

Step1 초기화 및 첫 commit하기

7) 소스코드 추가하기 (커밋할 목록에 추가 add) (commit2 폴더내 파일 활용)

```
$ git add report_card.c
```

8) commit 하기 (역사 한단위 만들기)

```
$ git commit -m "report card: Add base code"
```

Git 상태확인 명령어
(중간중간 치면서 수시로 확인하자)

```
$ git log
```

```
$ git shortlog
```

```
$ git diff
```

```
$ git status
```

Step2 diff 사용과 추가 commit하기

1) 상태를 확인한다

```
$ git status
```

2) **commit3** 폴더에 있는 report_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
$ git diff
```

3) diff를 통해서 변화분을 확인했다면 add 진행

```
$ git add report_card.c
```

4) 준비된 소스파일을 commit 한다

```
$ git commit -m "report card: Print a message of introduction"
```

5) 지금까지한 3개의 commit들을 확인해보자

```
$ git log
```

Step2 diff 사용과 추가 commit하기

6) **commit4 폴더**에 있는 report_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
$ git diff
```

7) diff 를 통해서 변화분을 확인했다면 add 진행

```
$ git add report_card.c
```

8) 준비된 소스파일을 commit 한다.

```
$ git commit -m "report card: Print grades of each subject"
```

9) 지금까지한 4개의 commit들을 확인해보자

```
$ git log
```


Step3 commit하기,반복

1) **commit5 폴더**에 있는 report_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
$ git diff
```

2) diff 를 통해서 변화분을 확인했다면 add 진행

```
$ git add report_card.c
```

3) 준비된 소스파일을 commit 한다.

```
$ git commit -m "report card: Show the sum of each grade"
```

Step3 commit하기,반복

4) **commit6 폴더**에 있는 report_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
$ git diff
```

5) diff 를 통해서 변화분을 확인했다면 add 진행

```
$ git add report_card.c
```

6) 준비된 소스파일을 commit 한다

```
$ git commit -m "report card: Get a average of grades"
```

Step4 지금까지의 commit을 push 하자

1) 상태를 확인하고 현재 브랜치명 master 를 확인하자

```
$ git status
```

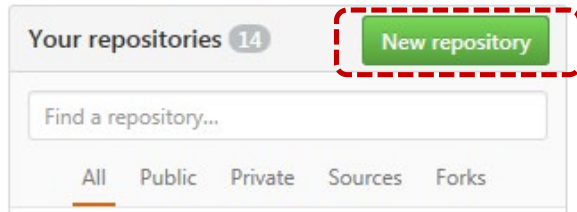
2) 지금까지한 commit들을 확인하자 (6개가 아니면 다시 확인하자)

```
$ git shortlog
```

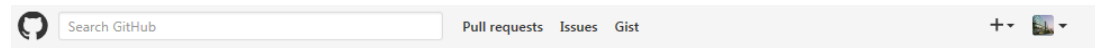
3) Github 원격저장소 URL을 등록하자

(잠깐 멈추고 <http://github.com> 켜고 repository 새로 생성하자)

잠깐, GitHub에서 원격저장소 만들기



새로운 **원격 저장소**를 생성하자



Create a new repository

A repository contains all the files for your project, including the revision history.

프로젝트명 자유(ex. test)

Owner: Repository name:

Great repository names are short and memorable. Need inspiration? How about **cuddly-tribble**.

Description (optional):

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

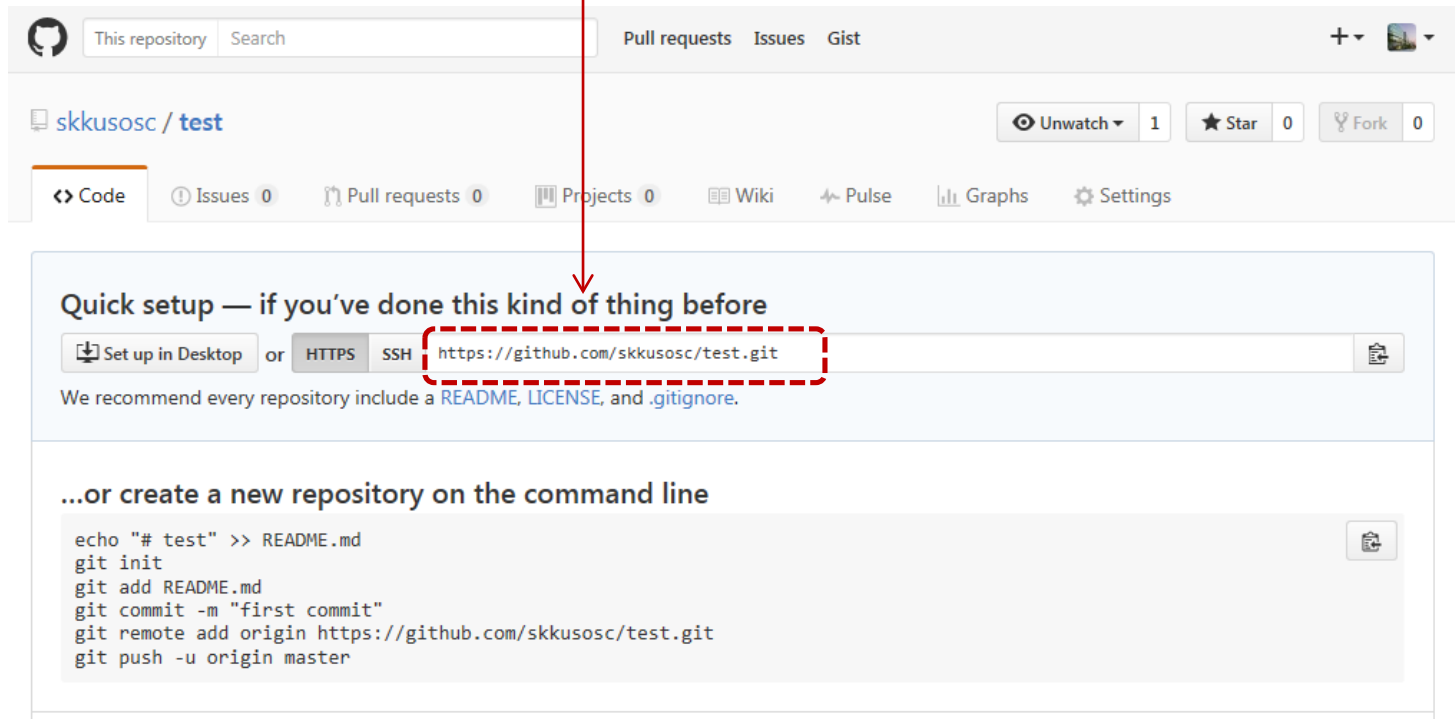
☐ **Private**
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.



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

잠깐, GitHub에서 원격저장소 만들기

해당 URL 복사해서 Step4의 4)으로 이어서 진행



Quick setup — if you've done this kind of thing before

 Set up in Desktop or **HTTPS** **SSH** `https://github.com/skkusosc/test.git` 

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# test" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/skkusosc/test.git
git push -u origin master
```

Step4 지금까지의 commit을 push 하자

4) 방금 복사한 URL로 GitHub 원격저장소 등록하자

```
$ git remote add origin 방금복사한 URL
```

5) GitHub 원격저장소(origin)에다가 밀어 넣자

```
$ git push origin master
```

6) GitHub 웹페이지 열고 확인해보자



Step4 지금까지의 commit을 push 하자

Github 들어가서 **본인** 프로젝트의 commit 기록 눌러보자

soscskku / report-card

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. [Manage topics](#)

6 commits 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

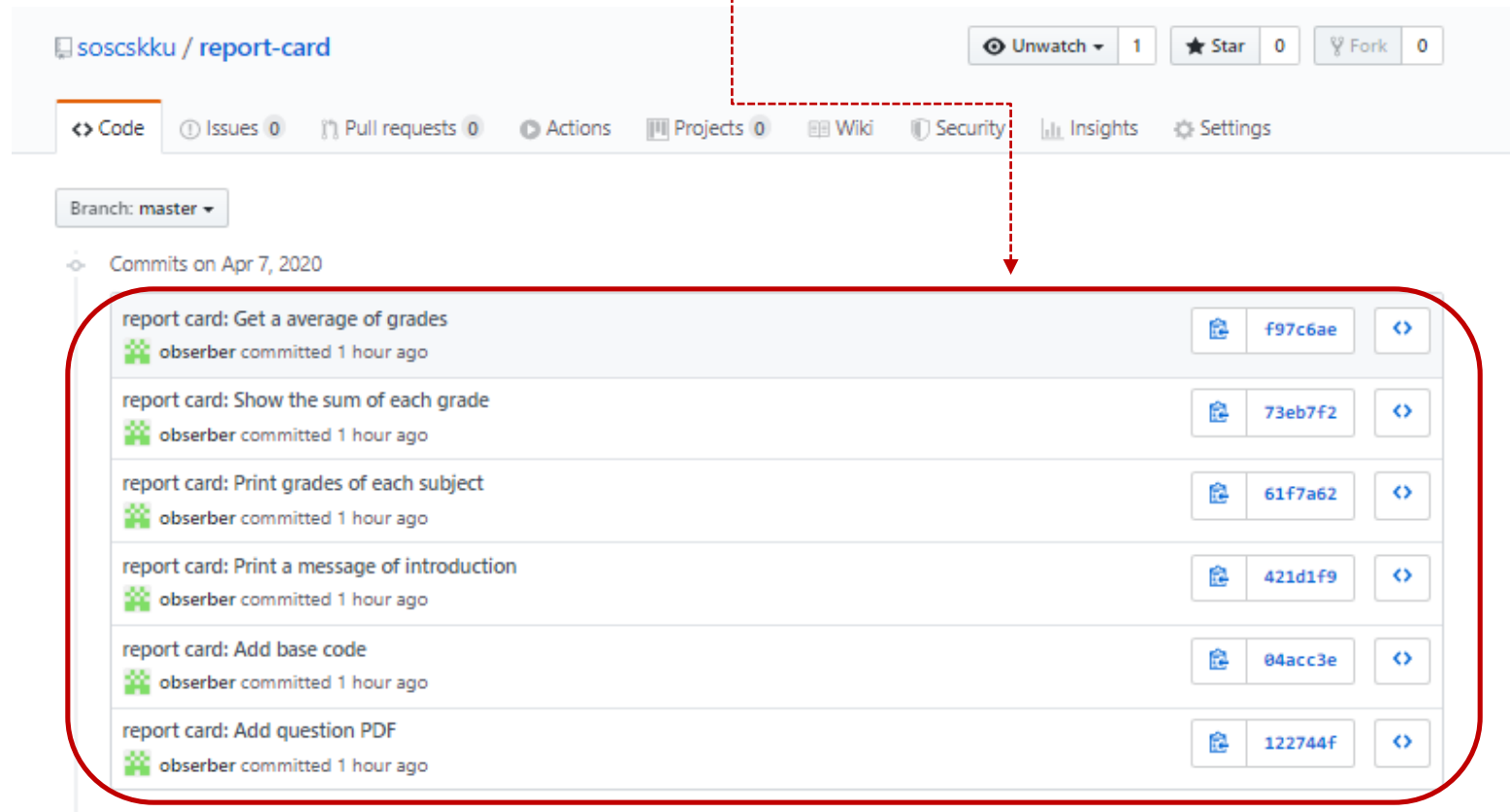
observer report card: Get a average of grades Latest commit f97c6ae 1 hour ago

File	Commit Message	Time
report_card.c	report card: Get a average of grades	1 hour ago
report_card.pdf	report card: Add question PDF	1 hour ago

Help people interested in this repository understand your project by adding a README. [Add a README](#)

Step4 지금까지의 commit을 push 하자

본인이 추가한 **commit** 들이 나오는걸 확인하자(본인 Github)



Contents

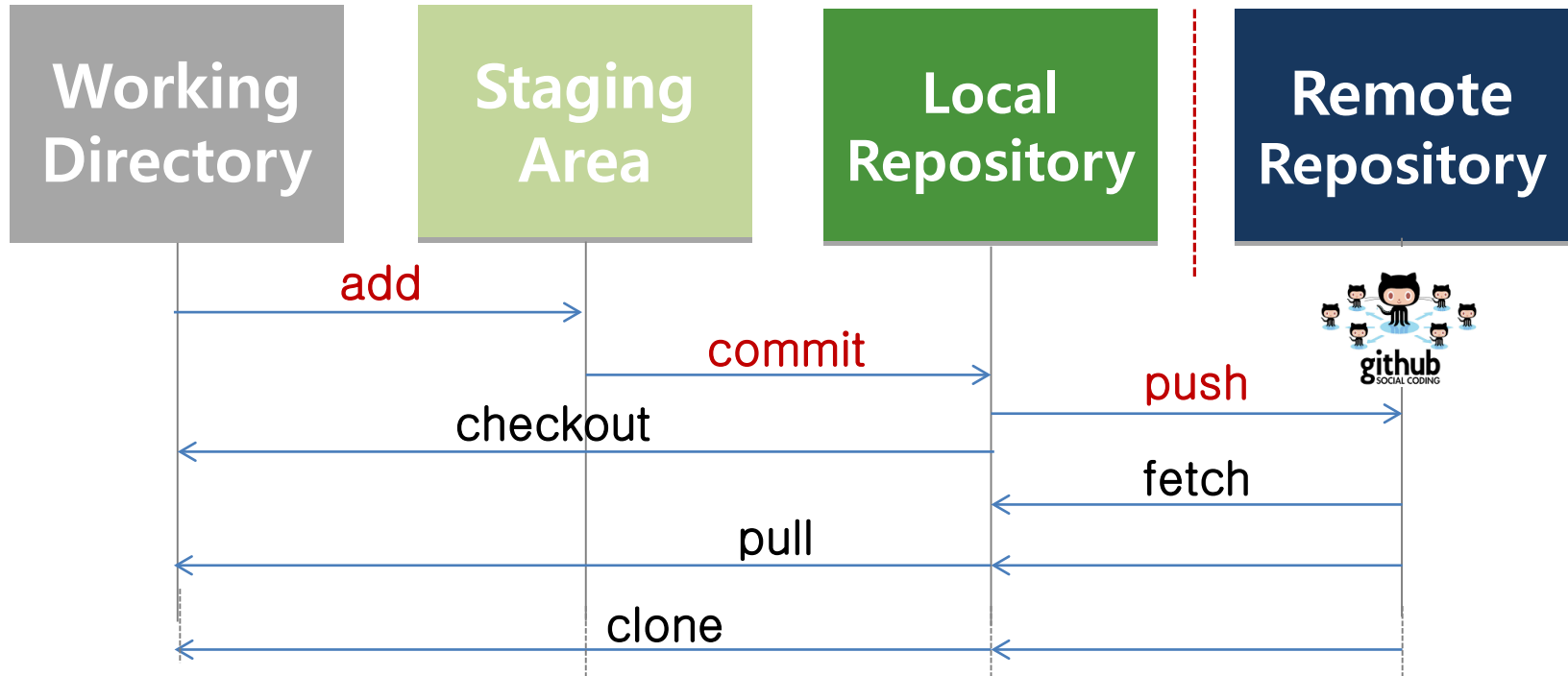
I
Git
Intro

II
Hands-on
(Basic)

III
Hands-on
(Advanced)

IV
Git
Review

Git 필수 개념 – Git의 세 가지 상태



Git을 통한 작업 순서

- 워킹 디렉토리에서 파일을 수정
- 워킹 디렉토리에서 변경된 파일을 **스테이징 영역에 추가**(커밋할 스냅샷 생성)
- 스테이징 영역의 파일을 커밋하여 **Git 디렉토리에 영구적으로 저장**

Q&A

- 오픈소스소프트웨어 실제 실습시간(매주 목요일 오후 6시 ~ 9시)에 아이캠퍼스 토론방을 통해서 질문 및 대답 진행
- 추가로 질문이 있으면 문의게시판 및 조교에게 메시지 보내기 등 활용