



## 성신여대 오픈소스 팀 프로젝트 실습

**Summary:** Web 페이지를 프로젝트화 하여 서로 협업하면서 작업을 진행, 웹 프로젝트 소스를 서로 수정함으로 발생하는 문제를 강사와 함께 해결한다.



### 강의 노트

- 다운로드 클릭  
(<https://github.com/oss-sungshin/git-lecture/raw/master/pdf/ci.pptx>)

### 목표

- Webpage 프로젝트를 서로 협업하면서 작업을 진행.
- Markdown으로 구성된 페이지 수정 작업을 통해 git/github 사용법을 완벽 숙지한다.

### PDF 문서

- 다운로드 클릭  
(<https://github.com/oss-sungshin/git-lecture/raw/master/pdf/mydoc.pdf>)

## 사용 언어

- 개발 프로젝트가 아닌 관계로 모든 작업은 Markdown을 기반으로 작성.
- Markdown 사용방법은 아래 사이트를 참고하세요.
- <https://guides.github.com/features/mastering-markdown>  
(<https://guides.github.com/features/mastering-markdown>)
- Markdown viewer 활용 <https://jbt.github.io/markdown-editor/>  
(<https://jbt.github.io/markdown-editor/>)

## 실습 진행 방법

- 모든 실습자들은 웹 프로젝트(<https://github.com/oss-sungshin/git-lecture> (<https://github.com/oss-sungshin/git-lecture>))의 소스를 수정하면서 발생하는 문제를 2명의 강사와 함께 해결한다.
- 실습 진행 도중 여러 이벤트(브랜치 작업, 개발 도중 머지) 들의 미션이 발생되며, 강사와 함께 문제점을 같이 해결한다.

## Git 내용 참고 사이트

- <https://git-scm.com/book/ko/v1> (<https://git-scm.com/book/ko/v1>)

## 2월 9일(목)시간표

- 10:00 ~ 11:50 : 실습 진행 방법 설명, 팀 빌딩, 실습 진행
- 11:00 : 브랜치 후 작업 진행 (강사 개별 지도)
- Event #1(11:20) : 1차 머지(팀원 push -> 팀장 pull request -> 서버)
  - 원격 저장소(oss-sungshin)를 등록한다.
  - 개별 브랜치 및 머지 충돌을 실습한다.
  - 팀장은 팀 저장소를 업데이트 한다.
- Event #2(15:10) : 2차 머지(팀원 push -> 팀장 pull request -> 서버)
  - 개별 브랜치 및 머지 충돌을 실습한다.
  - gitk를 통해 커밋 로그를 분석한다.
- Event #3(16:00) : 커밋 가져오기

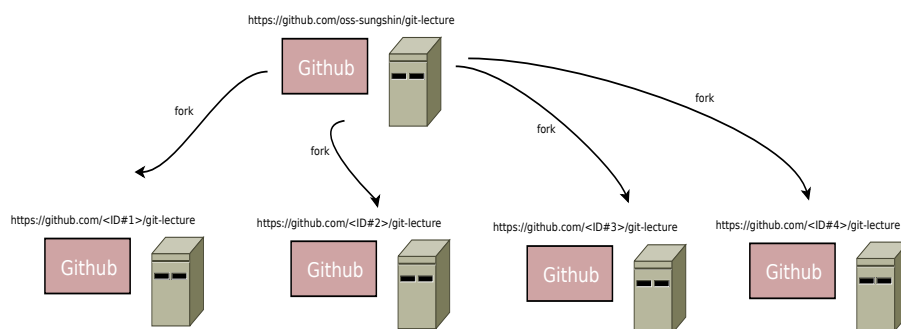
- 최근 oss-sungshin 커밋을 팀장 github 저장소에 적용하기(git cherry-pick)
- Event #4(16:30) : 팀별 커밋 내용 공유 및 설명(강사 진행)
  - 내용 공유

## 2월 10일(금) 시간표

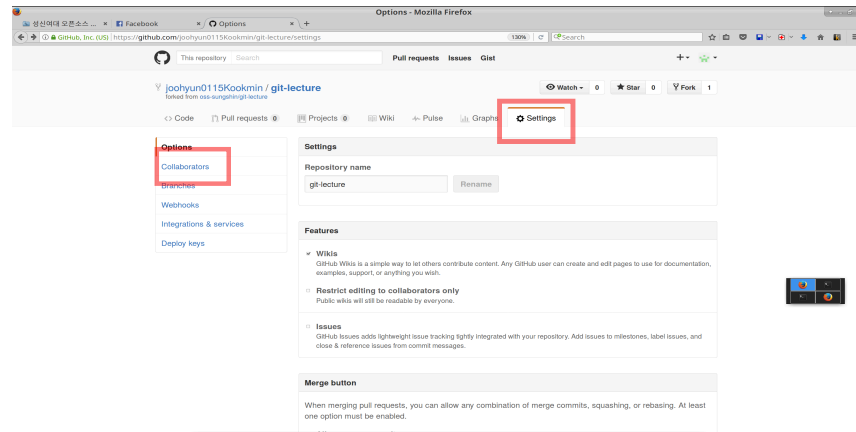
- 13:00 ~ 15:50 : 실습 진행 및 시상
- Event #5(13:20) : 4차 머지(팀원 push -> 팀장 pull request -> 서버)
  - 머지 문제 해결
- Event #5(14:00) : reset 실습
  1. 현재 커밋 백업 (git format-patch -2)
  2. 과거 커밋으로 이동 (git reset COMMIT\_ID)
  3. 백업 커밋 적용 (git am 이용)
- Event #5(15:00) : 최종 머지 및 작업 내용 발표
  - 작업 한 내용 서로 공유
- Event #5(15:30) : 종합 및 시상식

## 팀장이 해야 할 일

- 자신의 github에 oss-sungshin 프로젝트 fork 하기



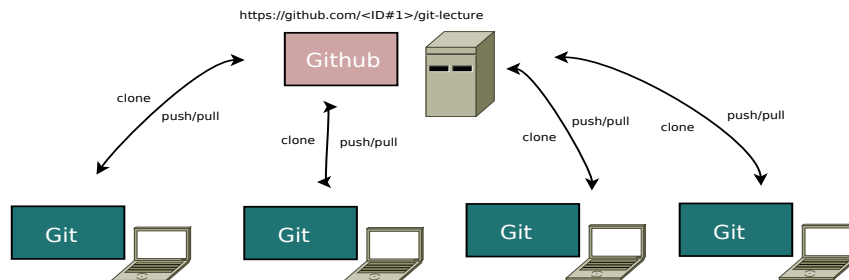
- 팀원 github 계정을 프로젝트에 등록하기



- 통합은 pull request를 사용하여 병합 할 예정

## 팀원이 해야 할 일

- 팀장의 github 프로젝트를 local PC에 clone 하기



## 팀별 결과 공유 및 시상

최종 결과물은 아래 페이지에 영구 보존되며, 우수 팀은 팀별 경품이 있습니다.

## 소스 트리 구조

**Summary:** 문서 소스 트리 구조에 대한 설명

## 소스 트리 구조

- `_data` 폴더는 side바 메뉴 설정 시 사용 & `pages` 폴더는 페이지 생성 및 수정

Branch: master	New pull request	Create new file	Upload files	Find file	Clone or download
joohyun0115Kookmin committed on GitHub Merge pull request #9 from joohyun0115Kookmin/master Latest commit f27725c 10 hours ago					
<code>_bundle</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>_data</code>	메인 페이지 설명 내용 추가	a day ago			
<code>_includes</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>_layouts</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>_posts</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>_tooltips</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>css</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>fonts</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>images</code>	메인 페이지 설명 내용 추가	a day ago			
<code>js</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>licenses</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			
<code>pages</code>	메인 페이지 설명 내용 추가	a day ago			
<code>pdf</code>	add pote	11 hours ago			
<code>pdfconfigs</code>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago			

페이지 Layout 파일

Markdown 페이지 디렉토리

- 메인 웹페이지 Markdown 페이지 `index.md`

<a href="#">.gitignore</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">.travis.yml</a>	Merge branch 'master' into master	2 days ago
<a href="#">404.md</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">Dockerfile</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">Gemfile</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">Gemfile.lock</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">_config.yml</a>	add note	11 hours ago
<a href="#">feed.xml</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">index.md</a>	add note	11 hours ago
<a href="#">pdf-all.sh</a>	layout design : add team building page	2 days ago
<a href="#">pdf-mydoc.sh</a>	layout design : add team building page	2 days ago
<a href="#">run.sh</a>	insert logo and pdf link	2 days ago
<a href="#">search.json</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">sitemap.xml</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">tooltips.html</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">tooltips.json</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago
<a href="#">update.sh</a>	import http://idratherbewriting.com/documentation-theme-jekyll/	2 days ago

## 메인 웹 페이지 Markdown 파일

- 사이드바 메뉴는 `_data/sidebars/mydoc_sidebar.yml`에서 설정

Branch: master [git-lecture](#) / [\\_data](#) / [sidebars](#) / [mydoc\\_sidebar.yml](#) Find file Copy path

[joohyun0115](#) 메인 페이지 설명 내용 추가 1c3df4b a day ago

3 contributors

98 lines (74 sloc) | 2.1 KB Raw Blame History

```

1 # This is your sidebar TOC. The sidebar code loops through sections here and provides the appropriate formatting.
2
3 entries:
4 - title: sidebar
5   product: git lecture
6   version: 1.0
7   folders:
8
9 - title: 강의 노트
10  output: web, pdf
11  folderitems:
12
13 - title: 실습 방법
14   url: /index.html
15   output: web, pdf
16   type: homepage
17
18 - title: 소스 트리 구조
19   url: /mydoc_source.html
20   output: web, pdf

```

Branch: master [git-lecture](#) / [pages](#) / [mydoc](#) / Create new file Upload files Fin

[joohyun0115](#) 메인 페이지 설명 내용 추가 Latest commit 1c3

- [mydoc\\_git\\_resolve\\_conflict.md](#) added usage part, team building, document structure
- [mydoc\\_git\\_revert.md](#) added usage part, team building, document structure
- [mydoc\\_git\\_track\\_files.md](#) added usage part, team building, document structure
- [mydoc\\_git\\_update.md](#) added usage part, team building, document structure
- [mydoc\\_git\\_useful.md](#) 메인 페이지 설명 내용 추가
- [mydoc\\_github.md](#) fixed release note url
- [mydoc\\_introduction.md](#) modified links to image with relative path
- [mydoc\\_release\\_notes\\_v1.md](#) 메인 페이지 설명 내용 추가
- [mydoc\\_source.md](#) Insert logo and pdf link
- [mydoc\\_teambuilding.md](#) 메인 페이지 설명 내용 추가
- [mydoc\\_travis.md](#) fixed release note url

- 메뉴와 markdown 파일은 1:1 매핑 관계임

## git lecture 1.0

강의 노트	▼
git 사용	▲
1. Create(A)	
2. Browse(B)	
3. Branch(C)	
4. Track files:(D)	
5. Record(D)	
6. Revert(C)	
7. Publish(B)	
8. Update(A)	
9. Resolve Conflicts(A, B, C, D)	
10. Useful Commands(A, B, C, D)	
github 사용법	▼
Travis CI 사용법	▼
Release notes	▼

Branch: master git-lecture / pages / mydoc /	
joohyun0115 메인 페이지 설명 내용 추가	
..	
mydoc_git_branch.md	added usage part, team building,
mydoc_git_browse.md	added usage part, team building,
mydoc_git_create.md	added usage part, team building,
mydoc_git_publish.md	added usage part, team building,
mydoc_git_record.md	added usage part, team building,
mydoc_git_resolve_conflict.md	added usage part, team building,
mydoc_git_revert.md	added usage part, team building,
mydoc_git_track_files.md	added usage part, team building,
mydoc_git_update.md	added usage part, team building,
mydoc_git_useful.md	메인 페이지 설명 내용 추가

## 팀 빌딩

### Overview

실습 수업 진행을 위한 팀 빌딩: 한 조당 5명, 5분동안 정하세요.

team 빌딩 결과

팀명	조원이름
A	김경민, name, name, name and name
B	이정희, 이한나, 유조영, name and name



# 문서 구조

## 참고 자료

### Git Cheat Sheet

Back – Command Quick Reference

#### Create

From existing files

```
git init
git add .
git commit
```

From remote repository

```
git clone -->old.../new
git clone git://...
git clone ssh://...
```

#### Branch

```
git checkout branch
git merge branch
git branch branch
git checkout -b new other
git checkout -b new other
git checkout -b new other
```

#### Configuration

Change options using `git config [--global] varname value`. The following variable names are useful:

`core.bare` True for repositories without a working tree (usually public repositories).

`core.sharedRepository` Set to group or all to make the repository contents writable for the file group or everybody.

`core.compression` A zlib compression level for objects (0-9, 9 = best compression) or -1 to use zlib's default.

`color.branch` Color-code list of branches (true = always, auto = only when outputting to a terminal)

`color.diff` Color-code diffs (true, auto)

`color.status` Color-code output of `git status` (true, auto).

`user.email` Your e-mail address (used in commits).

`user.name` Your name (used in commits).

#### Browse

```
git status
git diff oldref newref
git log [-p] [file|dir]
git blame file
git show ref[:file]
git branch (shows list, * = current)
git tag -l (shows list)
```

#### Record

In Git, commit only respects changes that have been marked explicitly with add.

```
git commit [-a]
git push (remote)
git tag foo
git tag foo
git tag foo
```

#### Object refs

```
master default devel branch
origin default upstream branch
HEAD current branch
HEAD~ parent of HEAD
HEAD~4 great-great grandp. of HEAD
foo..bar from ref foo to ref bar
```

#### Other Useful Commands

```
git archive Create release tarball
git bisect Binary search for defects
git cherry-pick Take single commit from elsewhere
git fsck Check tree
git gc Compress metadata (performance)
git rebase Forward-port local changes to remote branch
git remote add URL Register a new remote repository for this tree
git stash Temporarily set aside changes
git tag (there's more to it)
```

#### Commit Messages

Some of Git's viewing tools need commit messages in the following format:

```
A brief one-line summary
<blank line>
Details about the commit
```

#### Change

Track Files

```
git add files
git mv old new
git rm files
git rm --cached files
git rm --cached files
git rm --cached files
```

Revert

```
git reset --hard (NO UNDO)
git reset --hard (NO UNDO)
git revert ref
git commit -a -amend
git checkout ref file
```

Update

```
git fetch (from default upstream)
git fetch ref
git pull (= fetch + merge)
git am -3 patch mbox
git apply patch.diff
```

#### Resolve Conflicts

Use add to mark files as resolved.

```
git diff --base
git diff --ours
git diff --theirs
git log --merge
gitk --merge
```

#### Explanation of Syntax

```
[foo] foo is optional
... You can get creative here
foo foo is a placeholder for something you need to fill in
ref An object hash or name (see "Object Refs" for standard names)
```

There's a little bit of room for your own notes here. This is your chance to customize this cheat sheet!

This is version 2.0 of Jan Krüger's Git cheat sheet. You can contact the author by e-mail at: <jk@jk-gs>. Partially based on work by Zack Rusin, <http://rusin.blogspot.com/>

사진이 잘 보이지 않는 경우 마우스 오른쪽 버튼을 이용해서 사진을 저장해서 윈도우 사진 뷰어를 통해 열어서 보시면 됩니다.

## 문서구조

NOTE!: 기본 명령어 및 옵션에 대한 사용법을 예제와 함께 작성해주시면 됩니다. 필요에 따라 검색을 하셔도 좋습니다.

1. Create (page 11): 실제 수정해야할 파일: \$(GIT\_LECTURE)/pages/mydoc/mydoc\_git\_create.md
2. Browse (page 12): 실제 수정해야할 파일: \$(GIT\_LECTURE)/pages/mydoc/mydoc\_git\_browse.md
3. Branch (page 13): 실제 수정해야할 파일: \$(GIT\_LECTURE)/pages/mydoc/mydoc\_git\_branch.md

- 4. [Track\\_files \(page 14\)](#): 실제 수정해야할 파일: `$(GIT_Lecture)/pages/mydoc/mydoc_git_track_files.md`
- 5. [Record \(page 15\)](#): 실제 수정해야할 파일: `$(GIT_Lecture)/pages/mydoc/mydoc_git_record.md`
- 6. [Revert \(page 16\)](#): 실제 수정해야할 파일: `$(GIT_Lecture)/pages/mydoc/mydoc_git_revert.md`
- 7. [Publish \(page 17\)](#): 실제 수정해야할 파일: `$(GIT_Lecture)/pages/mydoc/mydoc_git_publish.md`
- 8. [Update \(page 18\)](#): 실제 수정해야할 파일: `$(GIT_Lecture)/pages/mydoc/mydoc_git_update.md`
- 9. [Resolve\\_conflict \(page 19\)](#): 실제 수정해야할 파일:  
`$(GIT_Lecture)/pages/mydoc/mydoc_git_resolve_conflict.md`
- 10. [Useful\\_command \(page 20\)](#): 실제 수정해야할 파일:  
`$(GIT_Lecture)/pages/mydoc/mydoc_git_useful.md`

# Create

**Summary:** Writing git manual

## Create

```
From existing files
git init
git add .
git commit
From remote repository
git clone ../old ../new
git clone git://...
git clone ssh://...
```

## Writing git manual!

Writing git manual!

## git browse-related command

**Summary:** Writing git manual

### Writing git manual!

Writing git manual!

## git browse 내용 쓰기

- 리스트1
  - 리스트1-1
- 리스트2
- 리스트3

## git branch-related command

**Summary:** Writing git manual

Writing git manual!

Writing git manual! #test

## git track-related command

**Summary:** Writing git manual

### Writing git manual!

Writing git manual!

#### track-related 명령어

- git add 파일명 : 작업폴더의 파일을 깃이 추적하게 하거나 커밋을 위한 준비상태로 만들. \*\* git add \* \*\* git add . \*\* git add -i // git 대화모드로 파일 staging
- git mv old nes
- git rm files
- git rm -cached files

## Record

**Summary:** Writing git manual

### git record-related command

- In Git, commit only respects changes that have been marked explicitly with add.

```
git commit [-a]  
git push [remote]  
git tag foo
```

## git revert-related command

**Summary:** Writing git manual

Writing git manual!

Writing git manual!



# Publish

**Summary:** Writing git manual

## GIT Publish-related command

Publish는 변경 내용을 발행하는 것입니다. 현재의 변경내용은 아직 로컬 저장소의 head 안에 있습니다. 이제 이 변경 내용을 원격 서버로 올려봅시다.

publish의 명령어에는 이런 것들이 있습니다.

- [git push \(page 17\)](#)
- [git push remote \(page 17\)](#)
- [git format-patch origin \(page 17\)](#)

1. `git push $ git push`
2. `git push remote $ git push remote`
3. `git format-patch origin $ git format-patch origin`

## git update-related command

**Summary:** Writing git manual

### Writing git manual!

Writing git manual!

update-related

### Update

![Update] (C:\Users\IT333-8-PC\git-lecture\images\update.jpeg)

## git resolve-related command

### **Summary:** Writing git manual

«««< HEAD ## Writing git manual! Writing git manual!

```
git diff [--base]
git diff --ours
git diff --theirs
git log --merge
gitk --merge
```

===== ## Writing git manual! Writing git manual!

abcd

- foo
- bar
- baz »»»> 5e3a805d5fa95a10828ab5bd6983cac4e1702a0a

# git useful command

## Summary: Writing git manual

## git tags

## 태그 조회하기

git tag 명령으로 이미 만들어진 태그가 있는지 확인할 수 있다.

```
$ git tag
v0.1
v1.3
```

# Writing git manual!

## Writing git manual!

안녕하세요

- 리스트 입니다.

소스는 아래와 같이 출력하세요

```
#include <stdio.h>
```

이미지 링크는 아래와 같이 하세요

2L 0 2L 2L 0 2L 0 2

LOZL

## Sidebar Navigation

**Summary:** github 사용법에 대한 문서

github 사용법

# Travis CI 사용법

**Summary:** Travis CI 사용법에 대한 문서

## Travis CI 사용법

## Release notes 0.1

**Summary:** Version 0.1 of the Documentation

### Relative links

test