

출처 : <https://www.youtube.com/watch?v=1I3hMwQU6GU>

0. git이란

버전관리시스템

2. 깃설치

2-1. Git bash 포함

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

Git Bash - 리눅스/맥(유닉스)에서 사용되는 CLI 명령어들을 윈도우에서 사용 가능

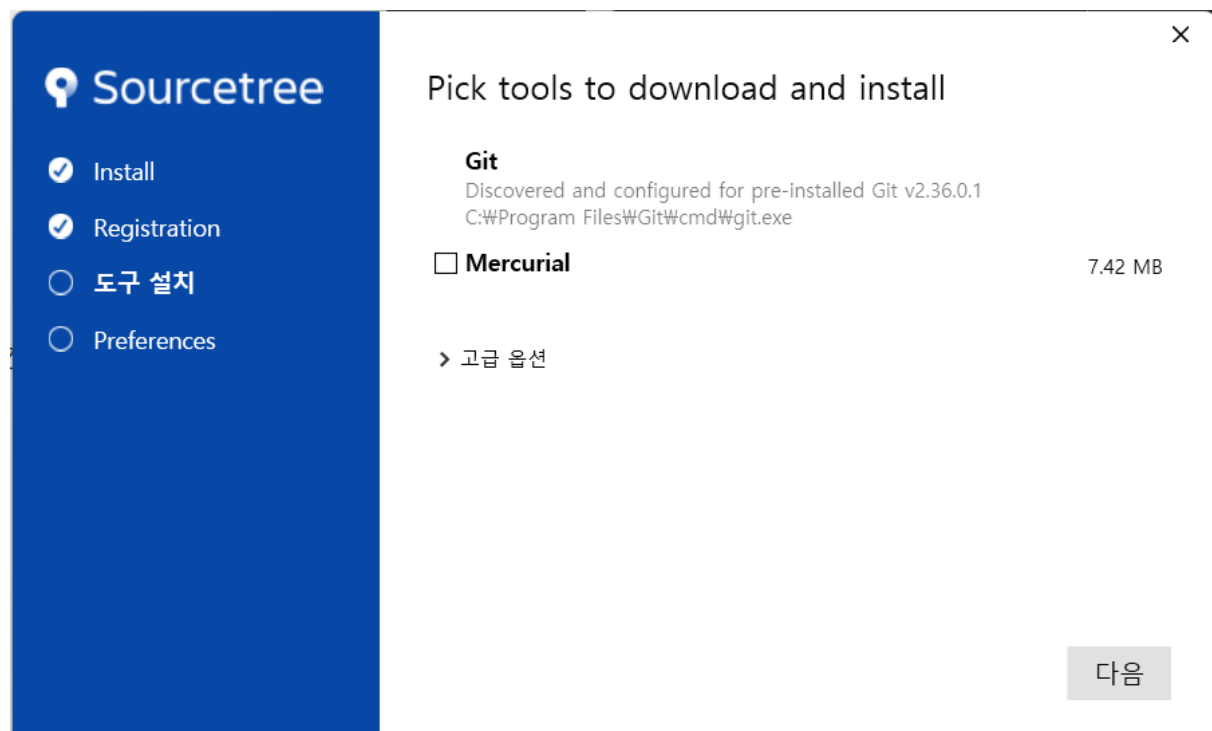
2-2. Git bash 설치후 테스트

```
git --version
```

```
git config --global core.autocrlf true
```

2-3 소스트리 설치 - gui

<https://www.sourcetreeapp.com/>



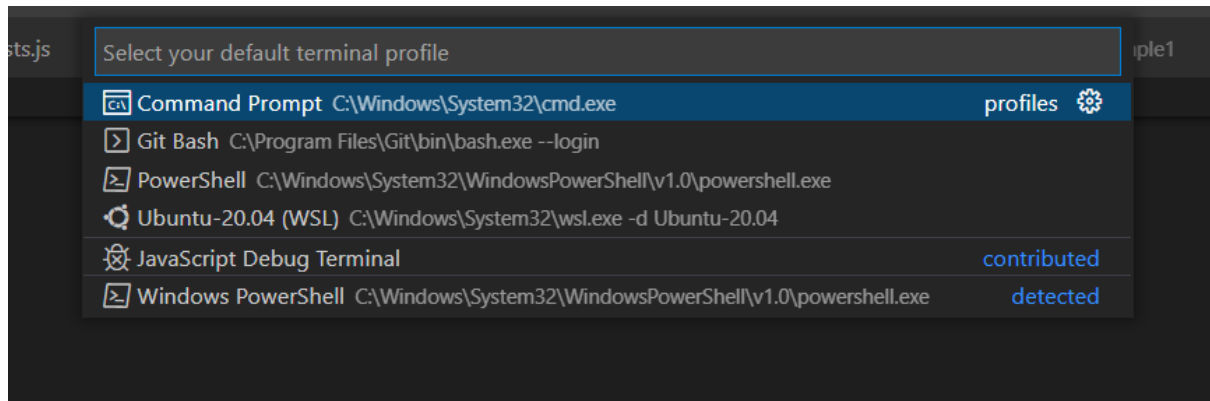
2-4 vscode 설치 – code editor

<https://code.visualstudio.com/>

Ctrl + `로 터미널 열기

Ctrl + Shift + P

Select Default Profile 검색 – git bash 선택



Sts 터미널 설치

Window – show view – terminal – git bash

<https://pncode.tistory.com/112>

4. CLI vs GUI

CLI - 터미널 – put, push같이 바로 명령어 사용할 때

GUI - 소스트리 – 프로젝트 같은 전반적인 상황을 파악할 때

5. 깃 설정

5-1. 사용자 이름, 메일 설정 – 깃헙 계정과 별개

설정

```
git config --global user.name "(본인 이름)"
```

```
git config --global user.email "(본인 이메일)"
```

확인

```
git config --global user.name
```

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

브랜치명 변경

```
git config --global init.defaultBranch main
```

5-2. 프로젝트 생성 & 깃 관리

폴더 생성후 vscode로 열기

```
git init
```

파일생성

tigers.yaml

```
team: Tigers
```

```
manager: John
```

```
members:
```

```
- Linda
```

```
- William
```

```
- David
```

lions.yaml

```
team: Lions
```

```
manager: Mary
```

```
members:
```

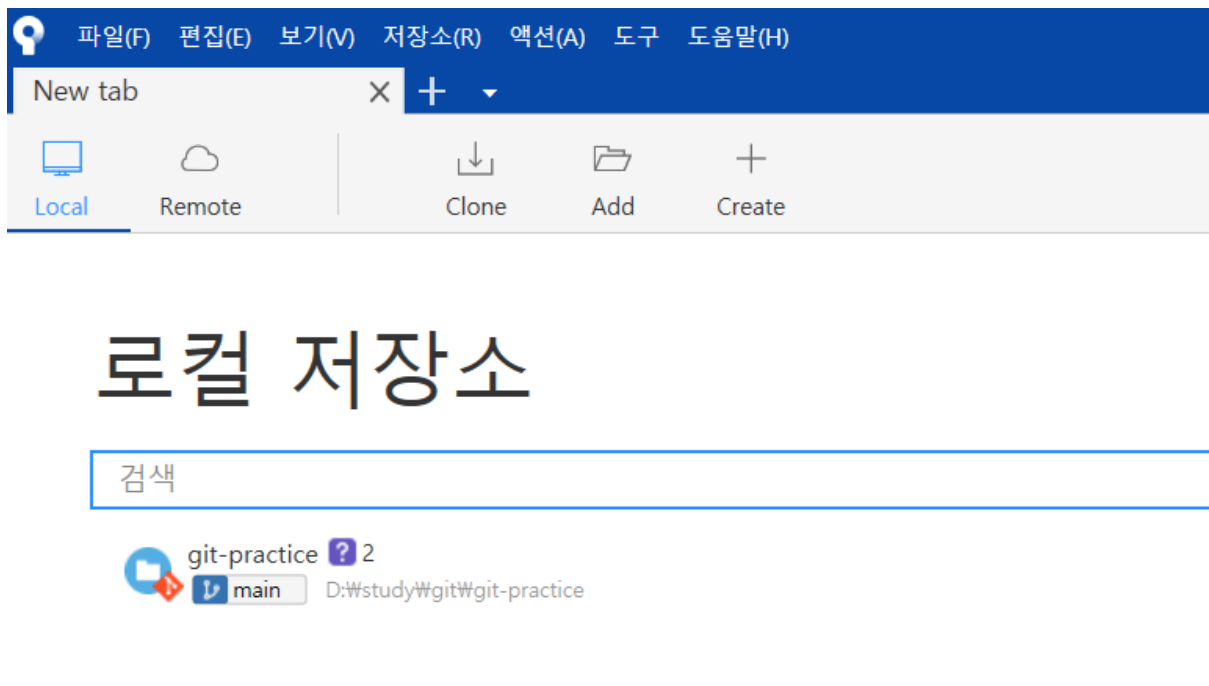
```
- Thomas
```

```
- Karen
```

```
- Margaret
```

```
git status
```

Git init 또는 아래의 소스트리 폴더 추가



6, git에게 말하지 않을 것- .gitignore

자동 생성파일 또는 라이브러리

보안 파일

secrets.yaml

id: admin

pw: 1234abcd

git status

.gitignore 파일 생성 후 추가 secrets.yaml

git status

.gitignore 형식

<https://git-scm.com/docs/gitignore> 참조

```
# 이렇게 #를 사용해서 주석
```

```
# 모든 file.c
file.c
```

```
# 최상위 폴더의 file.c
/file.c
```

```
# 모든 .c 확장자 파일
*.c
```

```
# .c 확장자지만 무시하지 않을 파일
!not_ignore_this.c
```

```
# logs 란 이름의 파일 또는 폴더와 그 내용들
logs
```

```
# logs 란 이름의 폴더와 그 내용들
logs/
```

```
# logs 폴더 바로 안의 debug.log 와 .c 파일들
logs/debug.log
logs/*.c
```

```
# logs 폴더 바로 안, 또는 그 안의 다른 폴더(들) 안의 debug.log
logs/**/*.debug.log
```

Section 2. 시간여행

2-1. 변화를 타임캡슐(버전) 에 담기

아직은 버전관리를 한적 없음 . 아래의 명령어로 담기

```
git add tigers.yam
```

```
git add
```

```
git status
```

2-2. 타임캡슐 묻기

```
git commit
```

```
git commit -m "FIRST COMMIT"
```

```
git log
```

```
$ git log
commit ee41de6bb919ebe401eac8ead525f4b975ef8038 (HEAD -> main)
Author: JYH <salvayh@gmail.com>
Date: Tue May 3 15:00:09 2022 +0900

    first commit
```

추가 , 변경, 삭제 후 다시 커밋해보기

```
Git add .
```

```
Git status
```

```
Git diff
```

```
git commit -m "Replace Lions with Leopards"
```

추가된 파일이 없을 때

```
git commit -am "(메시지)"
```

```
git log
```

vi모드에서 열림 j위 k아래 q종료

|

2. 과거로 돌아가는 2가지 방법

Reset vs revert

Reset – 해당 시점으로 돌아가고 이후 히스토리 지움

Revert – 거꾸로 수행하는 캡슐을 하나 추가함. 그전 단계로 가는 것

- 히스토리를 남기기 위해서거나
- 기존 내역들은 남겨두고 딱 어떤 시점의 내용만 되돌리고 싶을 때
- 한번 공유된건 지우면 문제가 되기 때문에 협업시는 반드시 리버트

3. 실습

```
git log
```

```
git reset --hard (돌아갈 커밋 해시)
```

```
git revert (되돌릴 커밋 해시)
```

수정된파일 내역 때문에 충돌 날수 있음. 삭제 또는 수정후 진행

`git revert --continue`

```
$ git revert 0ccd16dfbb344bbeed26946292d7aa1b00721501
CONFLICT (modify/delete): leopards.yaml deleted in parent of 0ccd16d (Replace L
s.yaml left in tree.
error: could not revert 0ccd16d... Replace Lions with Leopards
hint: After resolving the conflicts, mark them with
hint: "git add/rm <pathspec>", then run
hint: "git revert --continue".
hint: You can instead skip this commit with "git revert --skip".
hint: To abort and get back to the state before "git revert",
hint: run "git revert --abort".

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (main|REVERTING)
```

커밋하지 않고 revert하기 – 커밋하기전 상태로 돌려놓고 추가 수정후 다시 커밋할 때

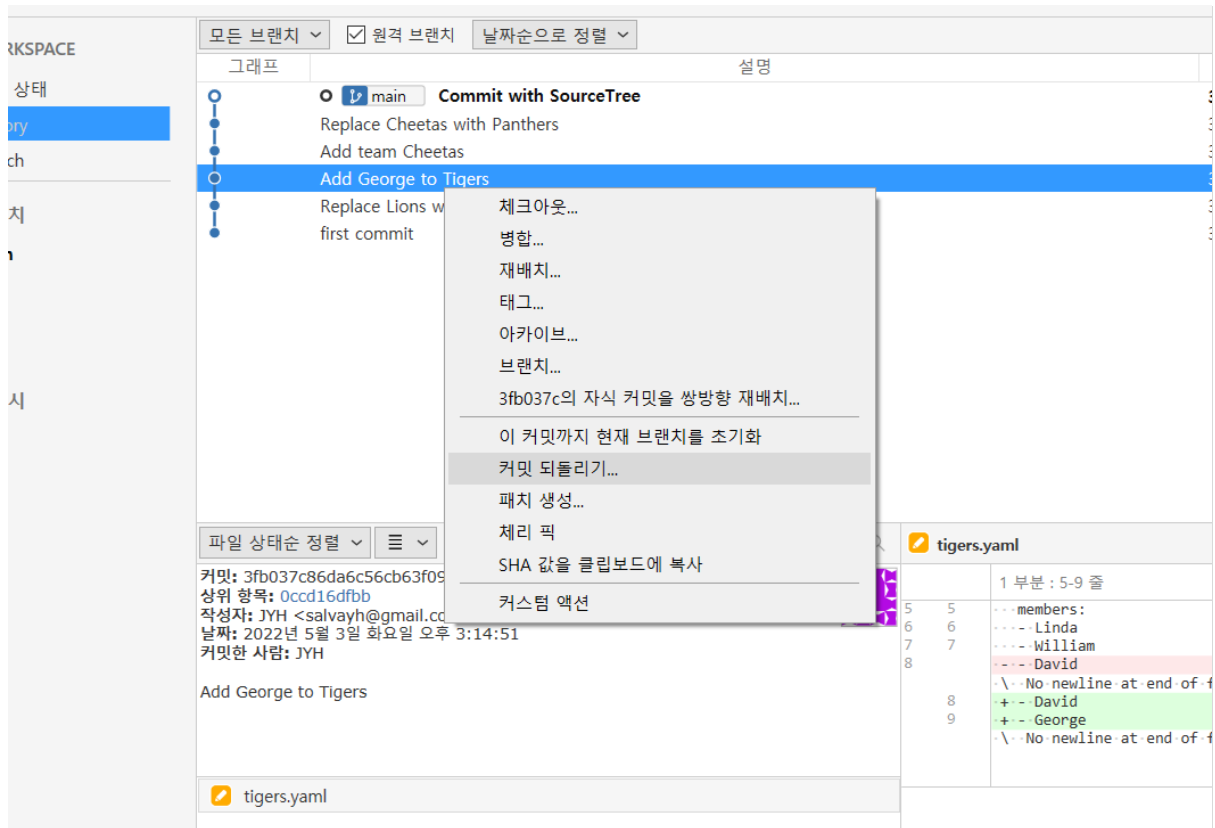
`git revert -continue`

4. 소스트리로 진행하기

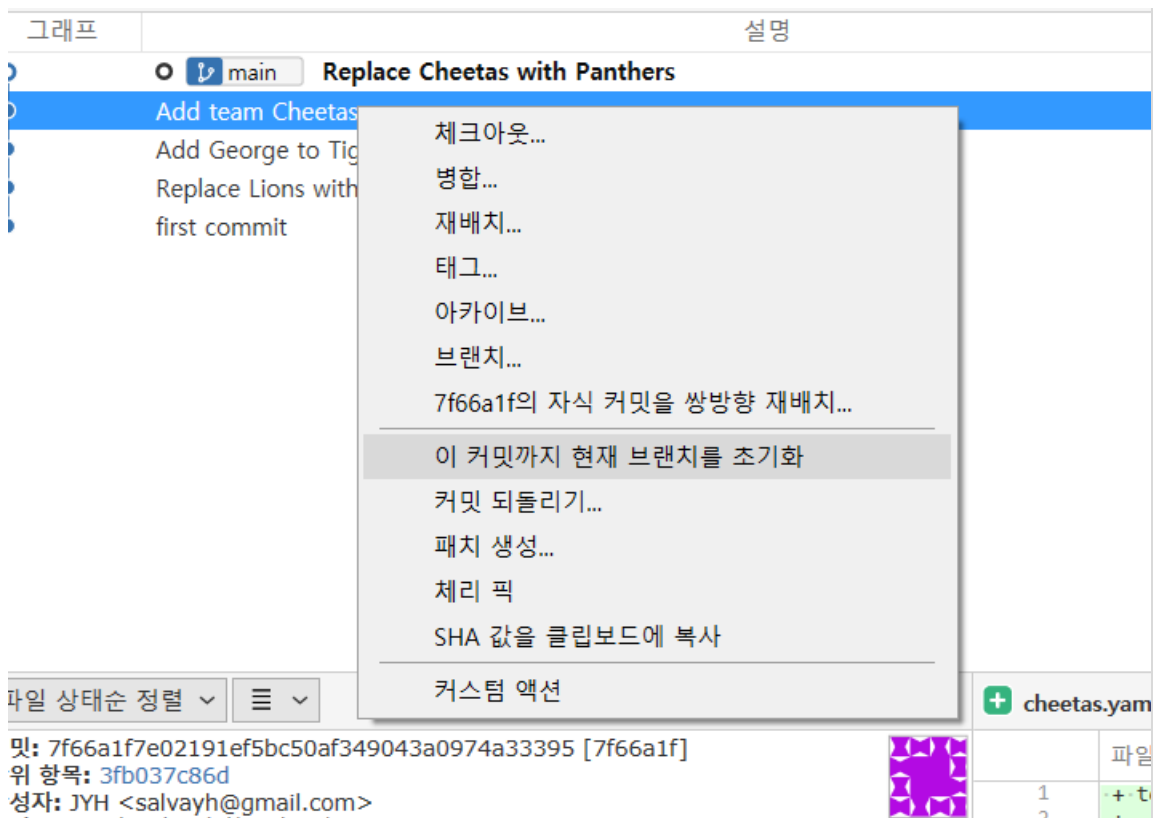
1. git add

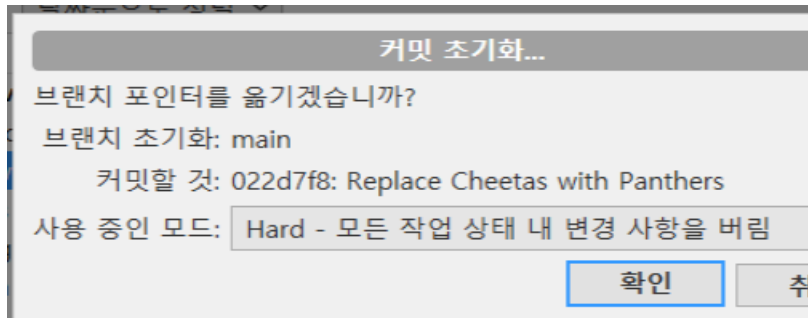
The screenshot shows the Git GUI application interface. The top menu bar includes options like '파일(F)', '편집(E)', '보기(V)', '저장소(R)', '액션(A)', '도구', and '도움말(H)'. The toolbar contains icons for commit, pull, push, patch, branch, merge, stash, reset, and tag. The left sidebar shows the 'WORKSPACE' section with '파일 상태', 'History', and 'Search' options. The main area displays a commit history table with columns for '설명' (Description), '날짜' (Date), '작성자' (Author), and '커밋' (Commit). The table lists several commits, including 'Replace Cheetas with Panthers' and 'Add team Cheetas'. Below the table, there are sections for '대기 중인 파일, 파일 상태순 정렬' and '스테이지에 올라간 파일'. The '스테이지에 올라간 파일' section shows a list of files: '.gitempty', 'leopards.yaml', and 'hello.txt'. The bottom right area displays the message 'Select a file to view the diff'.

2. revert

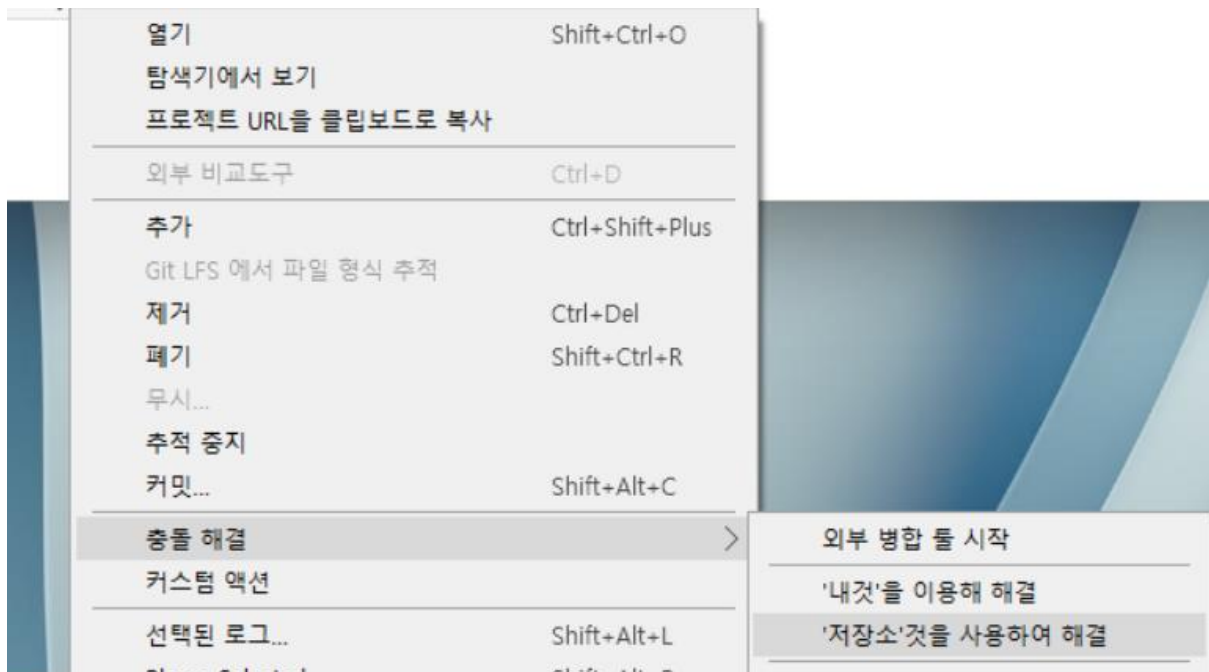
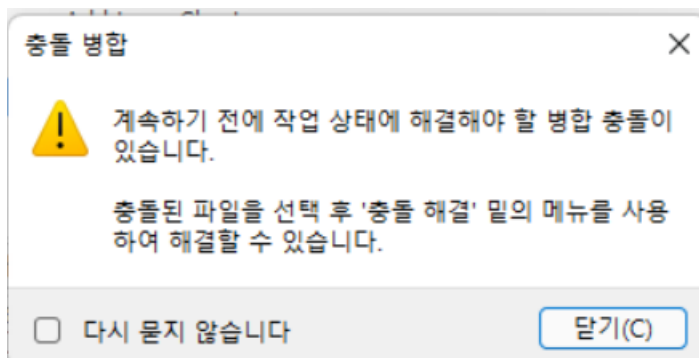


3. reset





revert충돌시



3. 차원 넘나들기

1. 여러 branch 만들기

프로젝트를 하나 이상의 모습으로 관리해야 할 때 - 테스트, 새로운 시도

여러작업이 각각 독립되어 진행될 때 – 각각의 기능을 따로 따로 기능개발후 메인버전에 병합

브랜치 생성

```
git branch add-coach
```

브랜치 목록

```
git branch
```

브랜치 이동

```
git switch add-coach
```

브랜치생성 이동

```
git switch -c new-teams
```

브랜치삭제

```
git branch -d (삭제할 브랜치명)
```

```
add-coach
* main

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (main)
$ git switch add-coach
Switched to branch 'add-coach'

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (add-coach)
$ git branch
* add-coach
  main

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (add-coach)
```

지워질 브랜치에만 있는 내용의 커밋이 있을경우

```
git branch -D (강제삭제할 브랜치명)
```

브랜치 이름 바꾸기

```
git branch -m (기존 브랜치명) (새 브랜치명)
```

현재 브랜치 내역

```
git log
```

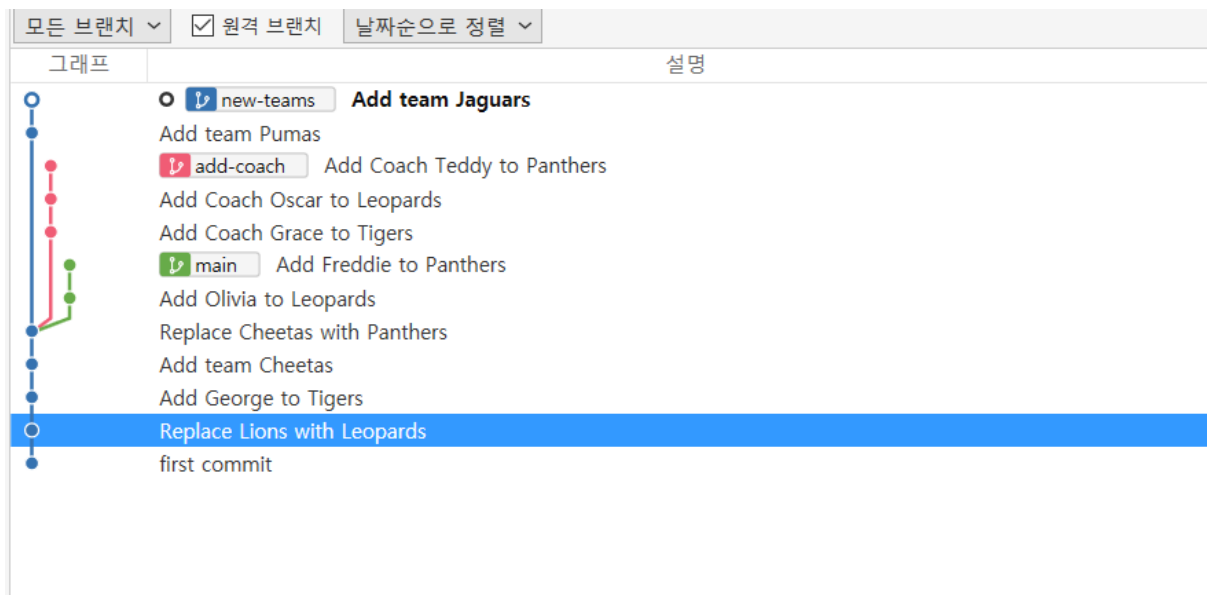
전체 브랜치 내역

```
git log --all --decorate --oneline --graph
```

```

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (new-teams)
$ git log --all --decorate --oneline --graph
* f0c5071 (HEAD -> new-teams) Add team Jaguars
* 5f58eb6 Add team Pumas
| * 6407c5e (add-coach) Add Coach Teddy to Panthers
| * fe02b7a Add Coach Oscar to Leopards
| * dc5ee44 Add Coach Grace to Tigers
| /
| * 556bbc2 (main) Add Freddie to Panthers
| * 6449b8a Add Olivia to Leopards
| /
* 022d7f8 Replace Cheetas with Panthers
* 7f66a1f Add team Cheetas
* 3fb037c Add George to Tigers
* 0ccd16d Replace Lions with Leopards
* ee41de6 first commit

```



3-2. 각각의 브랜치에서 작업한 내용을 합치기

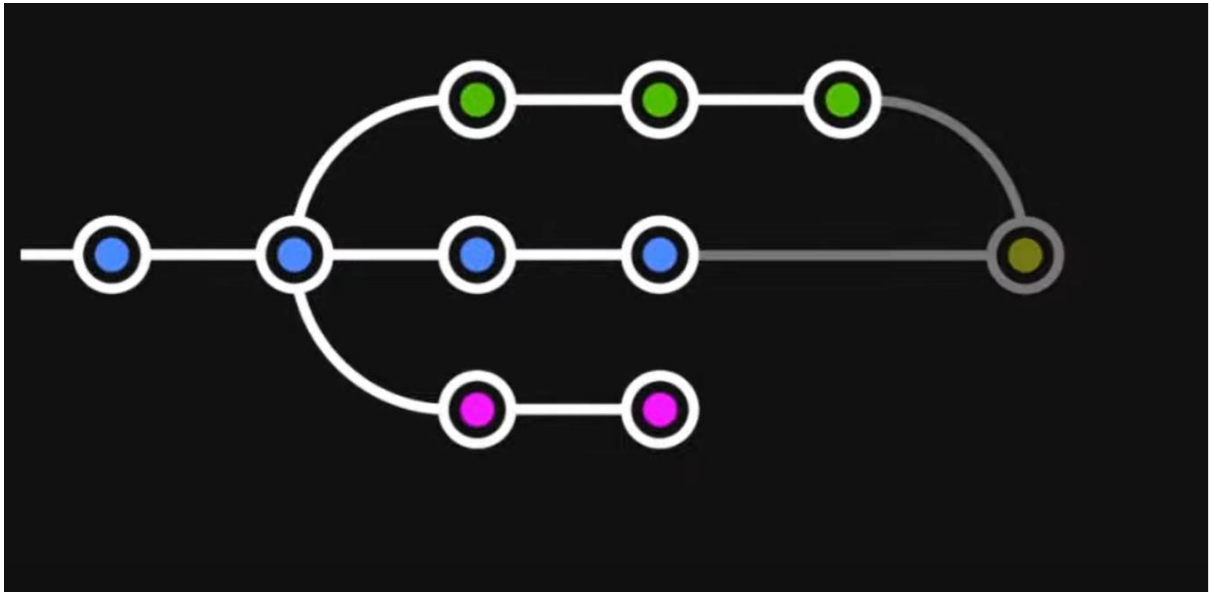
Merge – 두 브랜치를 한 커밋에 이어붙임

-브랜치 사용내역을 남길필요가 있을 때

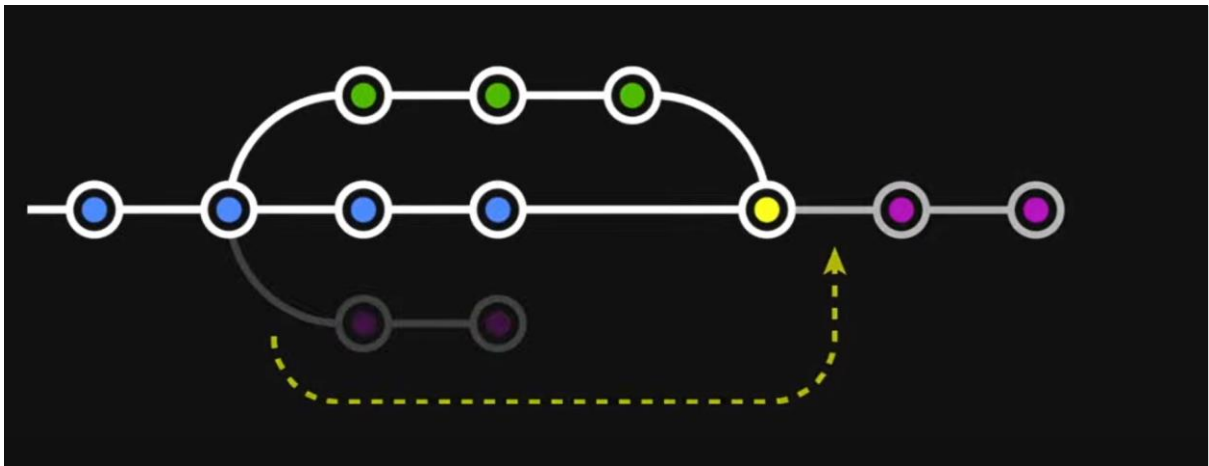
Rebase – 브랜치를 다른 브랜치에 이어 붙임

- 한줄로 깔끔히 정리된 내역을 유지하고 싶을 때
- 이미 팀원과 공유된 커밋들에 대해서는 사용하지 않는 것이 좋음.

Merge – 브랜치 사용내역 필요할 때



Rebase – 깔끔한 히스토리를 남길 때 (브랜치를 제거하기 때문에 협업에는 사용하지 말기)



merge

main에서

```
git merge add-coach
```

커밋메시지 작성후 마무리 – 커밋은 하나이기 때문에 reset가능

병합된 브랜치 삭제

```
git branch -d add-coach
```

rebase

new-teams 에서

```
git rebase main
```

main이동후

```
git merge new-teams ????
```

new-teams 브랜치 삭제

3-4. 충돌 해결하기

머지 중단

```
git merge --abort
```

해결했으면

git add ., git commit으로 병합

리베이스 중단

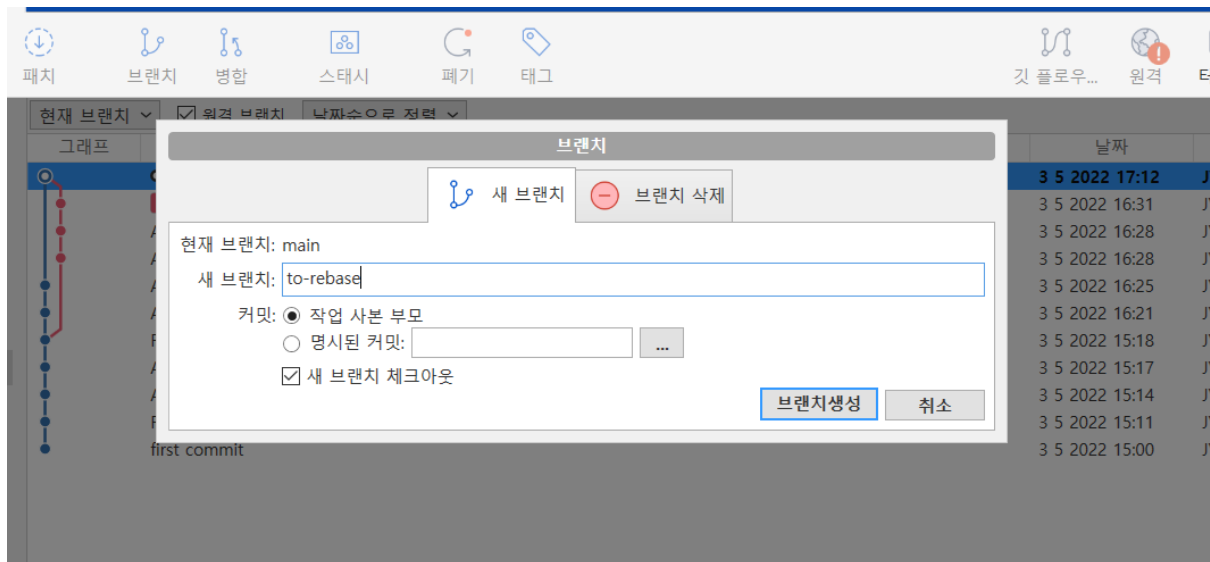
```
git rebase --abort
```

해결했으면

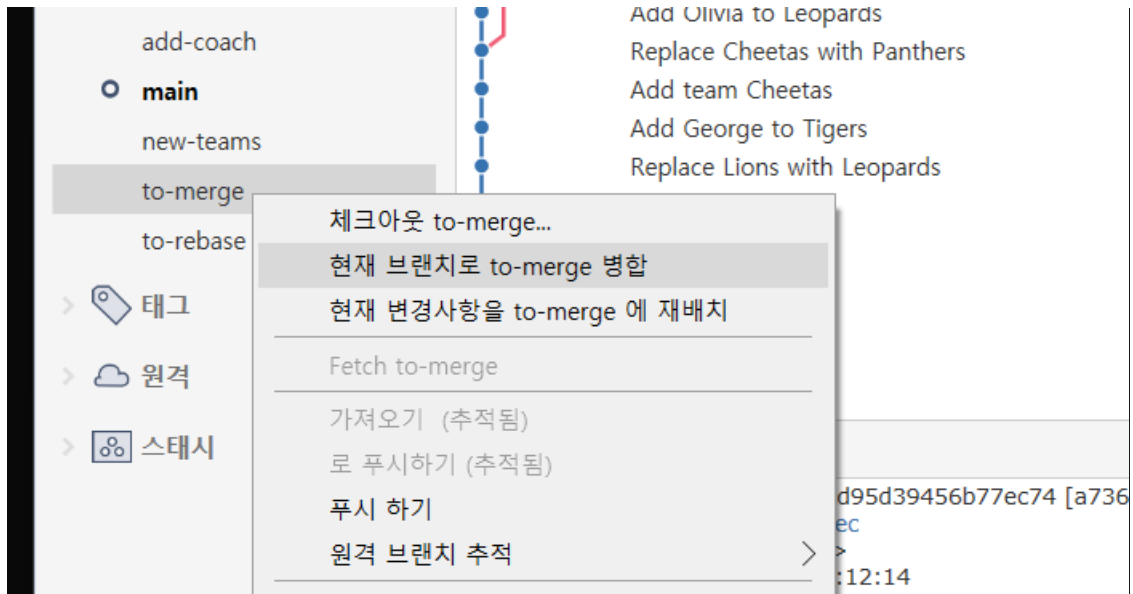
```
Git add
```

```
git rebase --continue
```

3-5소스트리로 진행하기



머지

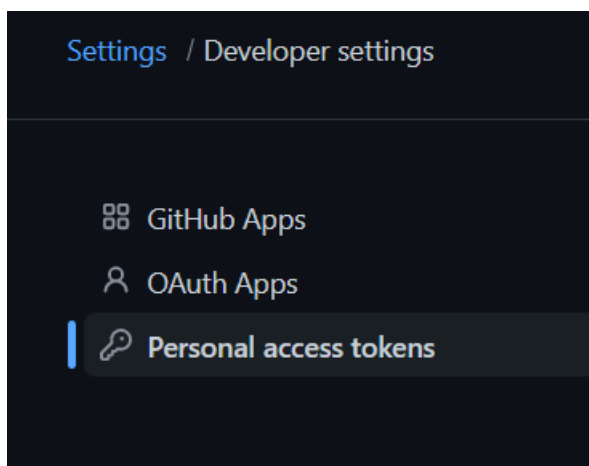


Rebase

충돌시 머지는 소스트리 해도 되나 rebase는 cli에서 하기를 권장함

4. 깃헙 사용하기 – 원격 저장소

가입하고 personal access token만들기



토큰관리는 일일이 하지 말고 컴퓨터에 셋팅



자격 증명 관리자

→ < 사용자 계정 > 자격 증명 관리자

제어판 홈

자격 증명 관리

웹 사이트, 연결된 응용 프로그램 및 네트워크에 대해 저장된 로그인 정보를 보고 삭제합니다.

 웹 자격 증명
  Windows 자격 증명

자격 증명 백업(B) 자격 증명 복원(R)

웹 사이트 주소 및 자격 증명 정보를 입력하십시오.

입력한 사용자 이름과 암호를 사용하여 해당 위치에 액세스할 수 있는지 확인하십시오.

인터넷 또는 네트워크 주소:

사용자 이름:

암호:

소스트리 추가


Git GUI for Windows

옵션

일반 Updates 비교 Git Mercurial 커스텀 액션 인증 네트워크

계정 추가

Git 저장된 비밀번호

 github.com

부분 : 2-9 중

깃헙에 저장소 만들기

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

[Set up in Desktop](#) or [HTTPS](#) [SSH](#) <https://github.com/junsun0708/git-practice.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#).

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

```
echo "# git-practice" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/junsun0708/git-practice.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/junsun0708/git-practice.git
git branch -M main
git push -u origin main
```

사용자 추가

junsun0708 / **git-practice** (Public) [Pin](#) [Unpin](#)

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

Who has access

General

Access

- Collaborators**
- Moderation options

Code and automation

- Actions

PUBLIC REPOSITORY

This repository is public and visible to anyone.

[Manage](#)

DIRECT ACCESS

0 collaborators have access to this repository. Only you can contribute to this repository.

로컬 깃저장소에 원격저장소 연결

git remote add origin <https://github.com/junsun0708/git-practice.git>

깃헙 기본 브랜치명

git branch -M main

로컬저장소의 커밋내역을 원격으로 push(업로드)

`git push -u origin main`

원격저장소 명

`git remote`

협업시 다운받기

폴더이동 – 오른쪽 클릭 – `git bash` – `git clone` “깃주소”

4-4. push pull

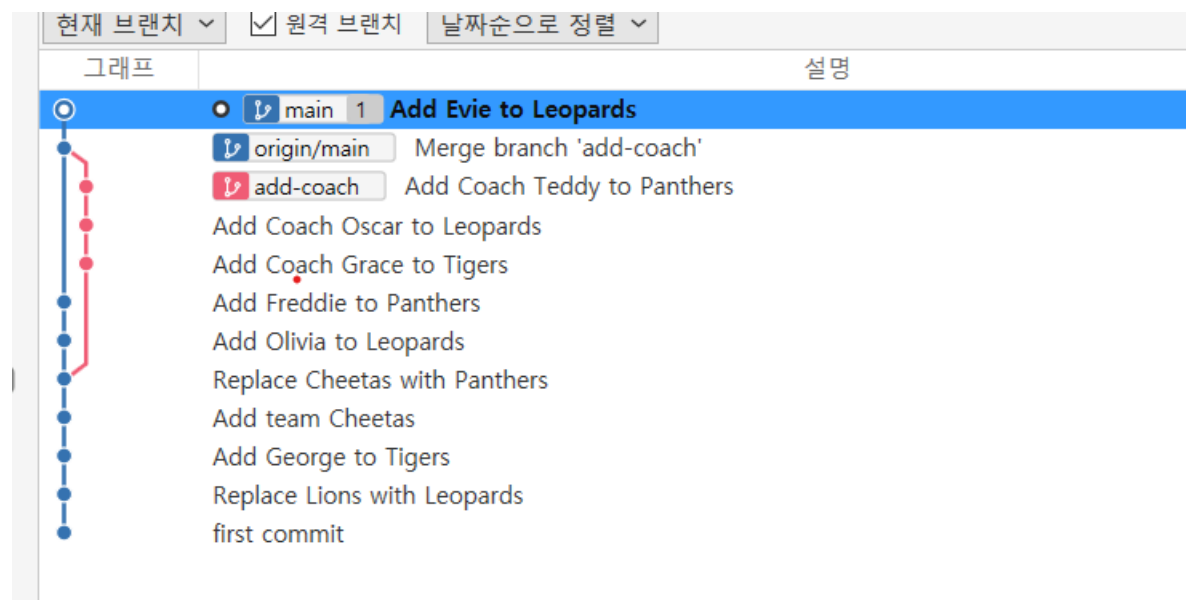
수정후 `git commit -am 'Add Evie to Leopards'`

로컬은 한단계 더, 원격은 한단계 아래임. 원격저장소로 저장이 필요

Git push – 이미 `git push -u origin main` 으로 대상 원격 브랜치가 지정되었기 때문에 가능

2 원격에서 커밋 당겨오기

Git pull



Pull 할것이 있을 때 push 하면 -> 원격에 먼저 저장되었기 있으므로 적용불가

먼저 원격의 버전을 받아오고 push해야함

Pull은 두가지 방식이 있음.

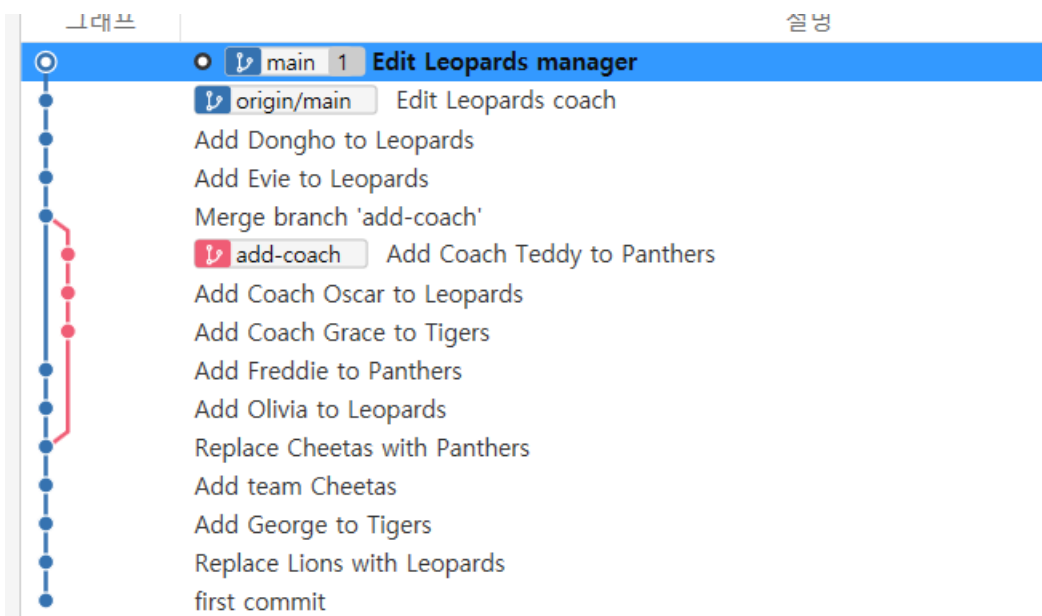
```
salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (main)
$ git push
To https://github.com/junsun0708/git-practice.git
! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/junsun0708/git-practice.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Git pull --no-rebase (merge방식)

로컬과 원격의 분기된 히스토리를 하나로 합침 (원격수정사항과 내 수정사항을 하나로)

Git pull --rebase (rebase방식)

시간선을 깃헙꺼로 붙이고 내 로컬을 붙임



Pull을 받은후 push하기

올릴때는 rebase하지 말고

내려받을 때 pull할때는 rebase사용 ok

로컬내역 강제푸쉬 - 원격에 있는게 잘못됐을 때 - 합의후

Git push --force

5. 원격 브랜치 다루기

git branch from-local

git switch from-local

git push --set-upstream origin from-local = git push -u origin from-local

git branch -a

```
salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (from-local)
$ git branch -a
  add-coach
* from-local
  main
  new-teams
  remotes/origin/from-local
  remotes/origin/main
```

git push origin --delete from-remote

<https://blog.namhj.com/m/52?category=846084>

<https://hucet.tistory.com/88>

<https://velog.io/@juho00ng/Git-workflow>

<https://kibua20.tistory.com/16>

<https://milooy.wordpress.com/2017/06/21/working-together-with-github-tutorial/>

<https://tre2man.tistory.com/256>

<https://jackinstitute.tistory.com/236>

<https://inpa.tistory.com/entry/GIT-%E2%9A%A1%EF%B8%8F-%EA%B9%83%ED%97%99-%EC%9B%90%EA%B2%A9-%EC%A0%80%EC%9E%A5%EC%86%8C-%EA%B4%80%EB%A6%AC-git-remote>

svn이점, 어디서나 커밋가능 . 인터넷 연결 안되어있어도 로컬커밋내용을 나중에 원격에 올리면 되기 때문에

자동 머지, 병합기능

고의 충돌

```
12 lines (9 sloc) | 101 Bytes

1  team: Tigers
2
3  manager: Donald
4
5  coach: Grrrrrrrr
6
7  members:
8    - Linda
9    - William
10   - David
11   - George
12   - abcde

tigers.yaml
1  team: Tigers
2
3  manager: mac
4
5  coach: Grace
6
7  members:
8    - Linda
9    - William
10   - George
11   - 123456
```

```

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (main)
$ git commit -am '123456'
[main f334c7f] 123456
1 file changed, 3 insertions(+), 3 deletions(-)

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (main)
$ git push
To https://github.com/junsun0708/git-practice.git
! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/junsun0708/git-practice.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

salva@Thingspire-jyh MINGW64 /d/study/git/git-practice (main)
$

```

깃

5 tigers.yaml		
...	@@ -2,10 +2,11 @@	team: Tigers
2	2	
3	3	manager: Donald
4	4	
5	-	coach: Grace
5	+	coach: Grrrrrrrr
6	6	
7	7	members:
8	8	- Linda
9	9	- William
10	10	- David
11	-	- George
11	+	- George
12	+	- abcde

The screenshot shows a version control interface with a commit history table and a context menu.

Commit Hash	Author	Date	Message
d861724	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 9:54:04	Edit Leopards
4011a06	Jung YoungHoo <98569881+@github.com>	2022 5월 4일 수요일 오전 9:43:12	Edit Leopards
c5eba4f	Jung YoungHoo <98569881+@github.com>	2022 5월 4일 수요일 오전 9:39:00	Add Dongho
c0a71d4	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 9:31:00	Add Evie to L
a736ac6	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 9:17:00	Merge branch
6407c5e	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:31:00	add-coach
fe02b7a	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:28:00	Add Coach O
dc5ee44	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:28:00	Add Coach G
556bbc2	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:25:00	Add Freddie t
6449b8a	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:21:00	Add Olivia to
022d7f8	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:18:00	Replace Chee
7f66a1f	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:17:00	Add team Ch
3fb037c	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:14:00	Add George t
0ccd16d	JYH <salvayh@gmail.com>	2022 5월 4일 수요일 오전 8:11:00	Replace Lions

Commit details for d861724:

- Commit: d861724ba8d65a0cc52bd070f86a62d5bd375295 [d861724]
- 상위 항목: 4011a06028
- 작성자: JYH <salvayh@gmail.com>
- 날짜: 2022년 5월 4일 수요일 오전 9:43:12
- 커밋한 사람: JYH
- 커밋 날짜: 2022년 5월 4일 수요일 오전 9:54:04

Context menu options:

- 선택된 로그... (Shift+Alt+L)
- Blame Selected... (Shift+Alt+B)
- 커밋 초기화...
- 현재 버전 열기 (Shift+Ctrl+O)
- 선택한 버전 열기
- 프로젝트 URL을 클립보드로 복사

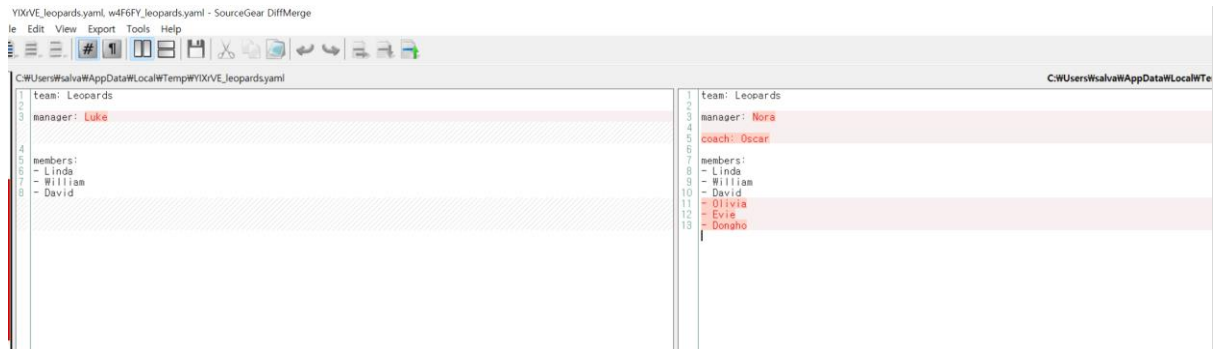
디프머지 설치 – 꼭 이거 설치 할 필요없음.

<https://sourcegear.com/diffmerge/>

<https://copynull.tistory.com/283>

도구- 옵션 -비교

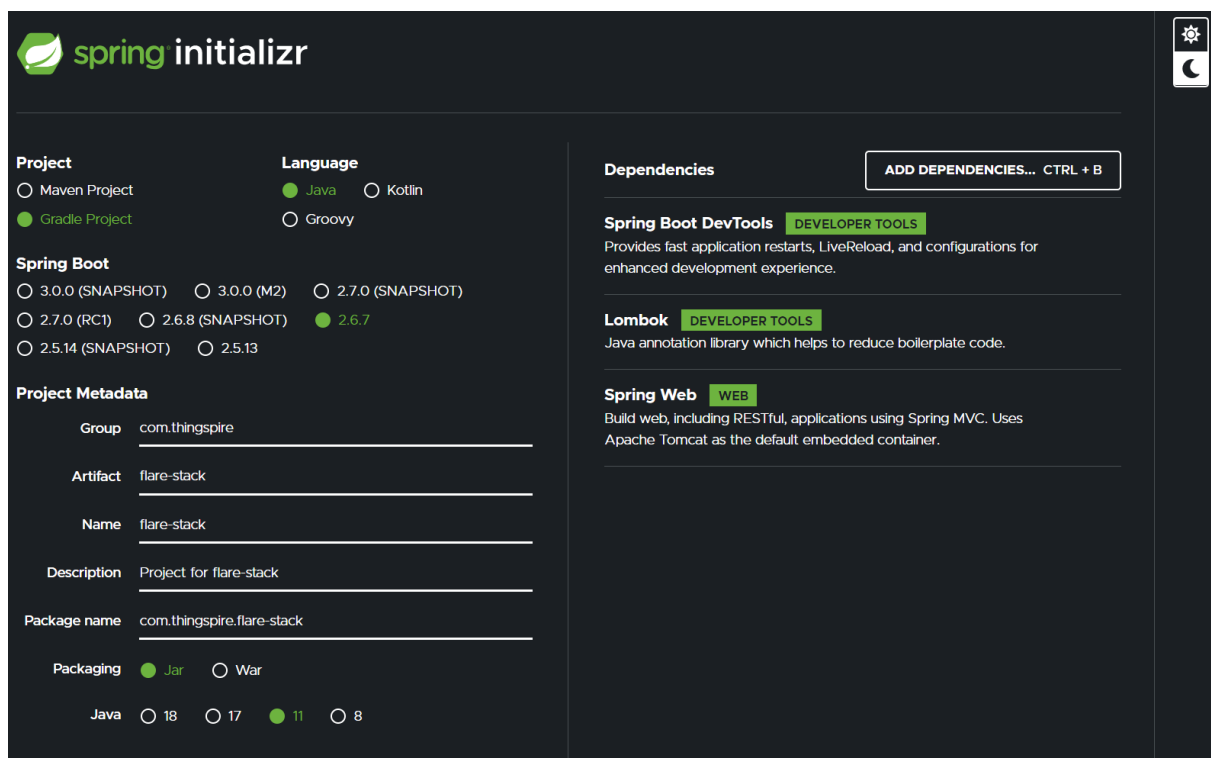
외부도구, 병합도구 선택



깃허브 220504 플레어스택 저장소 생성

소스트리 연결

소스트리 - 설정 - 원격저장소



그레이들 임포트

```

salva@Thingspire-jyh MINGW64 /d/workSpace/flare-stack
$ git init
Initialized empty Git repository in D:/workSpace/flare-stack/

salva@Thingspire-jyh MINGW64 /d/workSpace/flare-stack (main)
$ git status
On branch main

No commits yet

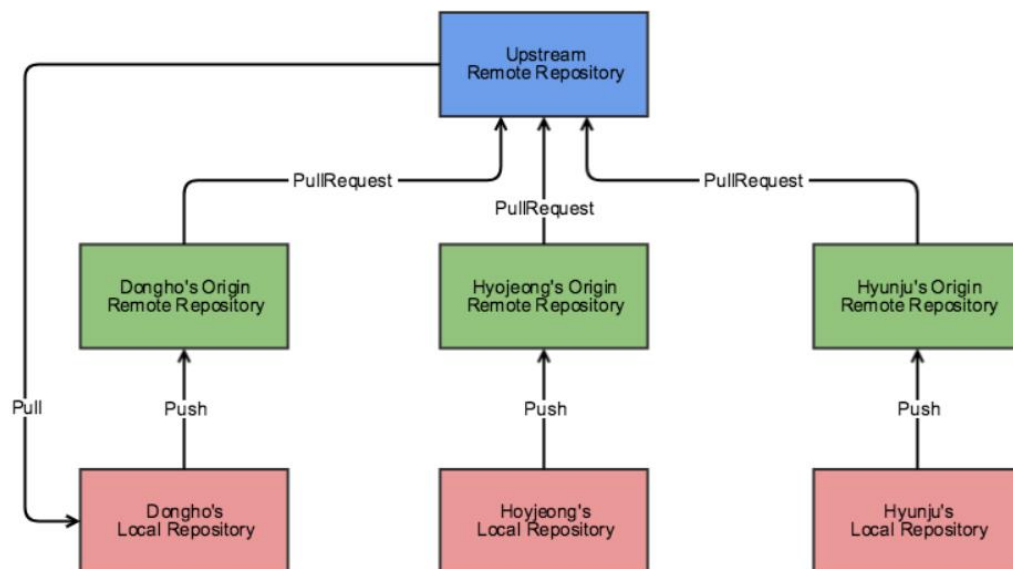
```

깃 초기

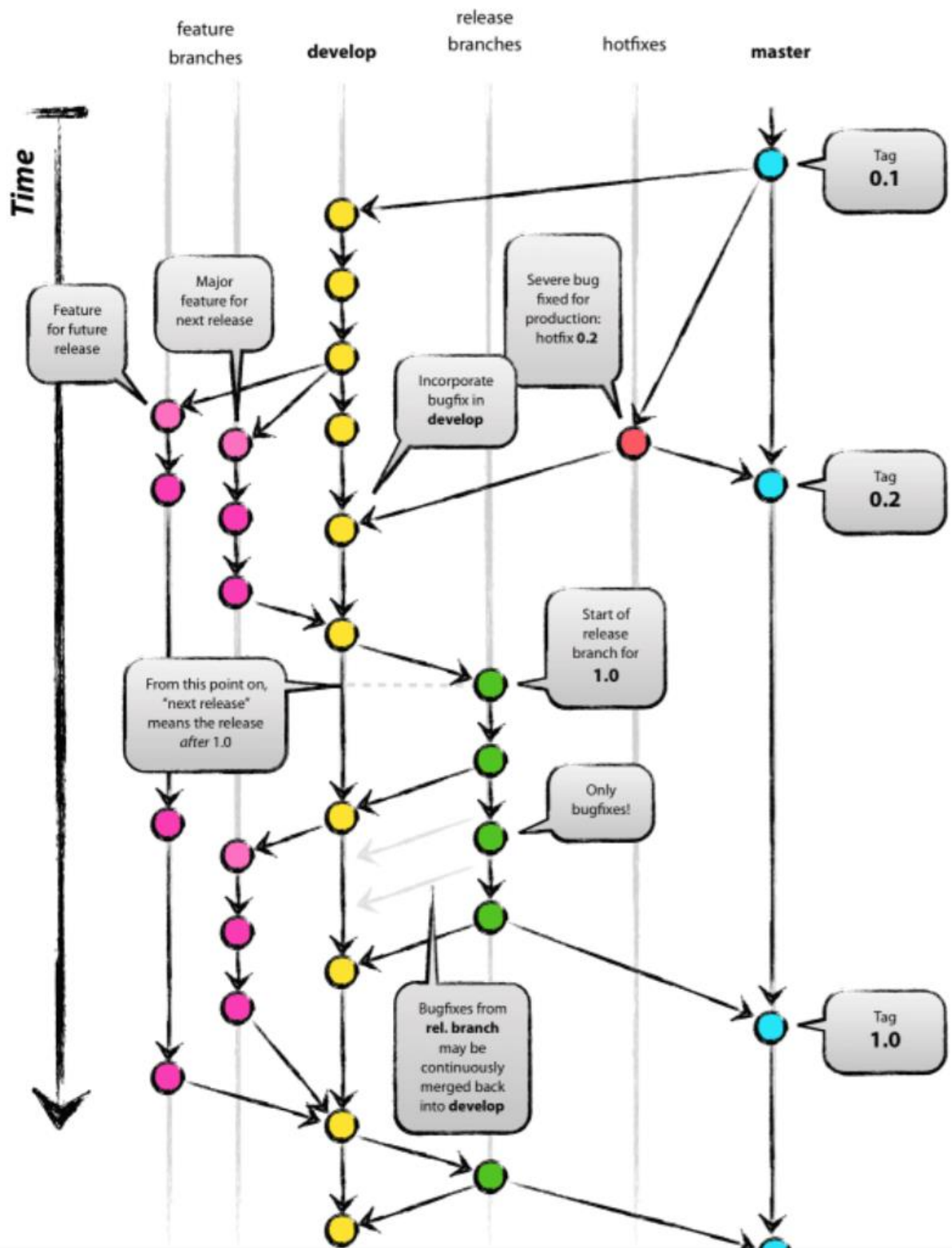
<https://devbirdfeet.tistory.com/63>

‘기획-디자인-개발-QA-출시’

Git-flow



<https://techblog.woowahan.com/2553/>



<https://overcome-the-limits.tistory.com/7>

깃설치, 깃허브가입

작업할 공간 터미널 열기

```
git clone https://github.com/junsun0708/thingspire-flare-stack-dev.git
```

```
git remote add origin https://github.com/junsun0708/thingspire-flare-stack-dev.git
```

git remote

git remote -v

에러 : 기존 리모트명 맘에 안들면 삭제

```
git remote remove origin
```

파일 수정후

Git add .

발생에러 : warning: LF will be replaced by CRLF in .gitignore.

```
해결 : git config --global core.autocrlf true
```

Git status

Git commit -m '커밋메시지'

git push origin

발생에러 : error: failed to push some refs to 'https://github.com/junsun0708/thingspire-flare-stack-dev.git'

hint: Updates were rejected because the remote contains work that you do

hint: not have locally. This is usually caused by another repository pushing

hint: to the same ref. You may want to first integrate the remote changes

hint: (e.g., 'git pull ...') before pushing again.

hint: See the 'Note about fast-forwards' in 'git push --help' for details.

해결 : 원격저장소와 로컬저장소 동기화

에러 : \$ git push origin

fatal: The current branch main has no upstream branch.

To push the current branch and set the remote as upstream, use

```
git push --set-upstream origin main
```

해결 : git push --set-upstream origin main

git push -u origin main

Git push

JYH 플레어스택 첫 커밋 - 스프링 프로젝트 구성	
gradle/wrapper	플레어스택 첫 커밋 - 스프링 프로젝트 구성
src	플레어스택 첫 커밋 - 스프링 프로젝트 구성
.gitignore	플레어스택 첫 커밋 - 스프링 프로젝트 구성
README.md	Update README.md
build.gradle	플레어스택 첫 커밋 - 스프링 프로젝트 구성
gradlew	플레어스택 첫 커밋 - 스프링 프로젝트 구성
gradlew.bat	플레어스택 첫 커밋 - 스프링 프로젝트 구성
settings.gradle	플레어스택 첫 커밋 - 스프링 프로젝트 구성


소스트리 설정


New tab


×


+


▼

Local

Remote


Clone


Add

Create

로컬 저장소

검색

git-practice

main

1 1 ↓

D:\study\git\git-practice

Add a repository

Choose a working copy repository folder to add to Sourcetree

D:\workspace\thingspire-flare-stack-dev

탐색

저장소 종류: ? Git 저장소입니다

main

Local Folder:

[루트]

추가

The screenshot shows the Sourcetree application interface. The top menu bar includes File, Edit, View, Repository, Action, Tools, and Help. The toolbar contains buttons for Commit, Pull, Push, Patch, Branch, Merge, Stash, Queue, and Tag. The left sidebar shows the Workspace view with a tree of branches (main, origin/main) and tags. The main area displays the commit history for the 'main' branch, showing a commit by JYH on 6/5/2022 at 15:05. The bottom panel shows the 'build.gradle' file with the following content:

```
11 configurations {
12     compileOnly {
13         extendsFrom annotationProcessor
14     }
15 }
16
17 repositories {
18     mavenCentral()
19 }
20
21 dependencies {
22     implementation 'org.springframework.boot:spring-boot-starter-w
23     compileOnly 'org.projectlombok:lombok'
24     developmentOnly 'org.springframework.boot:spring-boot-devtools
25     annotationProcessor 'org.projectlombok:lombok'
26     testImplementation 'org.springframework.boot:spring-boot-start
27 }
28
29 tasks.named('test') {
30     useJUnitPlatform()
31 }
```

//////////신규개발자//////////

git clone <https://github.com/junsun0708/thingspire-flare-stack-dev.git>

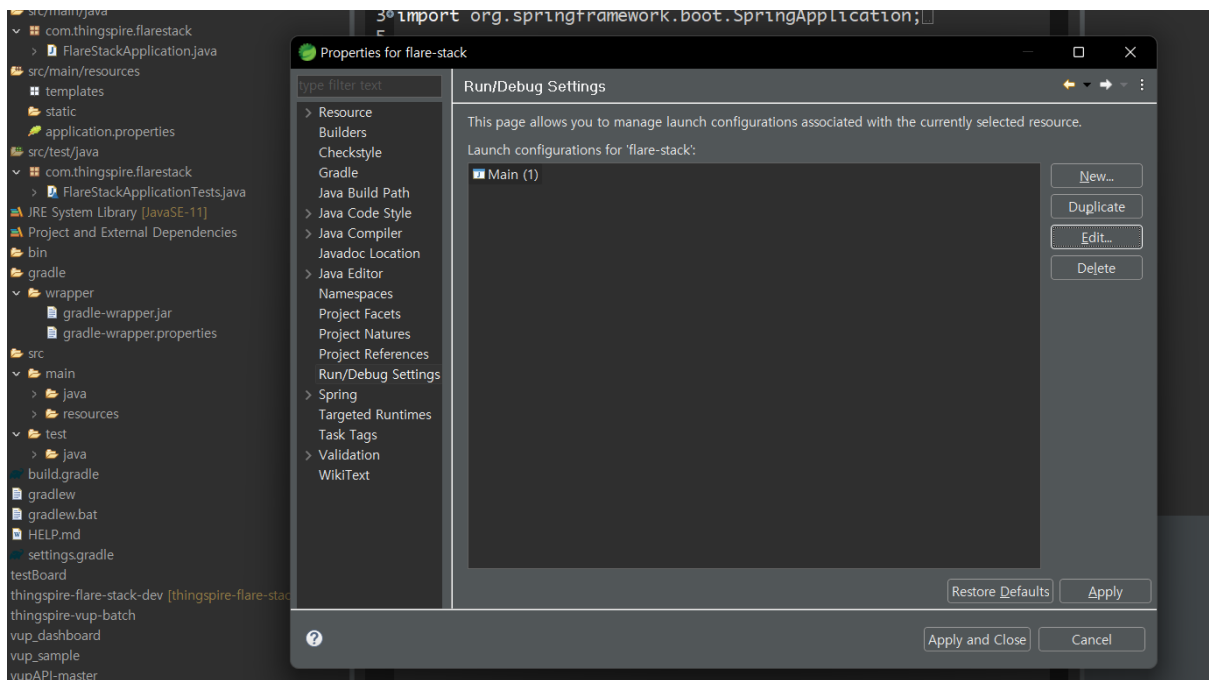
해당 폴더 가서 sts 열고 임포트 그레이들 프로젝트 -

프로젝트 엔진이 안올라갈 때

- 1) Go to Project->properties.
- 2) In properties window's left pane select "Project Facets".
- 3) Then click on "Convert to faceted form"
- 4) Then choose your server and JDK version.

Apply and Close.

Note: Verify the project and the default IDE JRE is same version.



Description	Resource	Path	Location Type
-------------	----------	------	---------------

Project at 'D:\workSpace\thingspire-flare-stack-dev' can't be named 'flare-stack' because it's located directly under the workspace root. If such a project is renamed, Eclipse would move the container directory. To resolve this problem, move the project out of the workspace root or configure it to have the name 'thingspire-flare-stack-dev'.	thingspire-flare-stack-dev	line 0	Gradle Error Marker
--	----------------------------	--------	---------------------

Setting.gradle 수정 `rootProject.name = 'thingspire-flare-stack-dev'`

특수문자로 인해 git commit이 안됨

Git commit 만쓰면 편집기 열리기 때문에 거기에 쓰기