

분산 버전 관리 도구: Git

Advanced #2

Hyunchan, Park

<http://oslab.jbnu.ac.kr>

Division of Computer Science and Engineering

Jeonbuk National University



강의 일정 (updated)

주차	월 (100분)	수 (50분)
1 (9/1)	Introduction	OSS 역사 1
2	OSS 역사 2 / OSS 개요 1	추석
3	OSS 개요 2 / OSS 라이선스 1	OSS 라이선스 2
4	OSS 활용 방법 / 버전 관리 도구 Git 1	버전 관리 도구 Git 2
5 (9/30)	버전 관리 도구 Git 3	버전 관리 도구 Git 4
6	버전 관리 도구 Git 5	한글날
7	특강 1 (10/14)	코드 분석 1
8 (10/21)	중간고사	개인 프로젝트 1 : 프로젝트 선정 및 계획 마감 (휴강)
9	특강 2 (10/28)	코드 분석 2
10 (11/4)	클라우드를 이용한 개발 환경 구축 1	클라우드를 이용한 개발 환경 구축 2
11	GitHub를 이용한 협업 1	GitHub를 이용한 협업 2
12	개인 프로젝트 2: 대상 프로젝트 코드 분석 결과 발표 및 기능 구현 방향, 커뮤니티 참여 방안 발표 및 피드백	
13 (11/25)	코드 리뷰	CI/CD 구축
14	개별 프로젝트 진행	
15 (12/9)	개인 프로젝트 3: 최종 프로젝트 발표, 기능 구현 소개 및 시연, OSS 커뮤니티 활동 소개	

공지사항: 특강

- [한국공개SW협회] 2019 대학생 공개SW 체험교육 특강
 - 1. Zeppelin (빅데이터 분석 도구)
 - 일정 : 10/14(월) 16:00~18:00
 - 강사명 : 류아영
 - ZEPL Software Development Engineer
 - Apache Zeppelin 커뮤니티의 성장을 봐온, Apache Zeppelin PMC (Project Management Committee)
 - 2. Chromium (크롬 브라우저 엔진)
 - 일정: 10/28(월) 16:00~18:00
 - 강사명: 방진호
 - Samsung Internet Developer
 - Chromium Owner and Committer
 - W3C Spec Editor, W3C Web Platform Tests Reviewer

공지사항: 중간고사

- 10/21 (월) 16:00~17:00
 - 10/23 수요일 수업은 휴강, but 프로젝트 1차 마감
- 장소:
 - 7호관 534호
- 범위: 시험 전까지 배운 내용 모두
 - 이론, Git 포함
- 부정 행위 적발 시, 교칙에 의거 처분
 - 학생증 지참
 - 필수적인 필기구 제외한 모든 물건은 책상 위에 두지 말 것

공지사항: 프로젝트 1차

- OSS 프로젝트 선정 및 계획
- 기한: 10/23 (수) 23:59 까지
- 제출: IEILMS “프로젝트 1차”
- 양식: A4 2장 이내
 - 학번, 이름
 - 프로젝트 명, GitHub 등 프로젝트 주소, 간략한 설명
 - 진행할 내용 및 계획, 일정 (1장 내외)
 - (이미 진행된 내용이 있는 경우, 추가 1장 내에 자세히 설명)

학습 내용

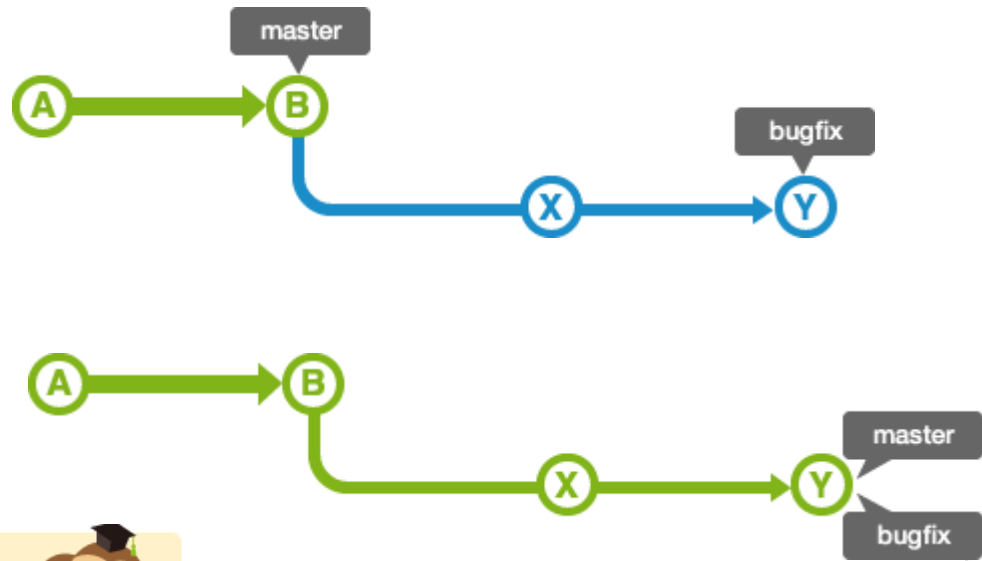
- Git: Branch 관리
- Rebase 를 이용한 commit 정리
 - Git 개인실습 #4
- Branch 관리 전략

Git: Branch 관리

(앞의 slide 내용에서 이어서 진행)

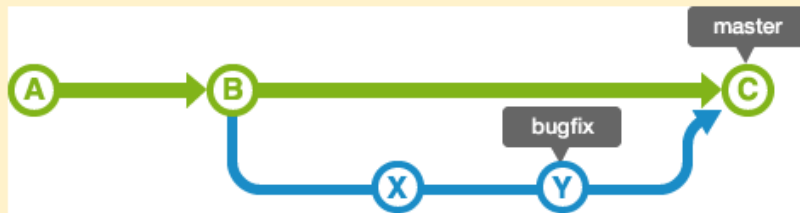
Branch: Merge

- `$ git checkout master`
- `$ git merge bugfix`
- Fast-forward merge



Note

병합 실행 시에 'fast-forward 병합'이 가능한 경우라도 'non fast-forward 병합' 옵션을 지정하여 아래 그림과 같이 만들어 낼 수도 있습니다.



'non fast-forward 병합'을 실행하면, 브랜치가 그대로 남기 때문에 그 브랜치로 실행한 작업 확인 및 브랜치 관리 면에서 더 유용할 수 있습니다.

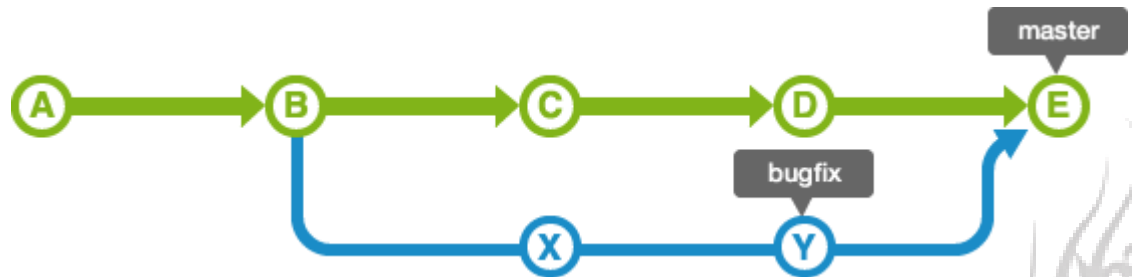
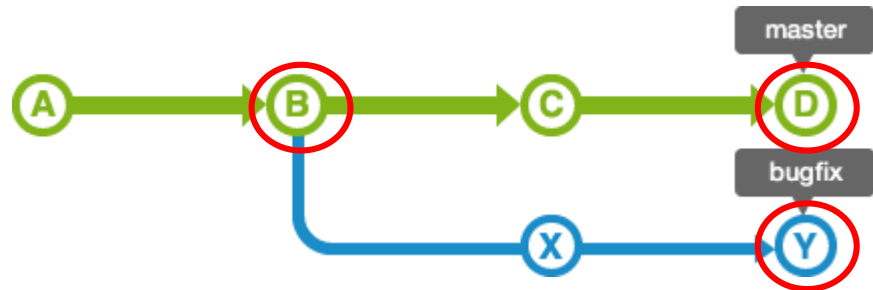
Branch: Merge

- `$ git checkout master`

- `$ git merge bugfix`

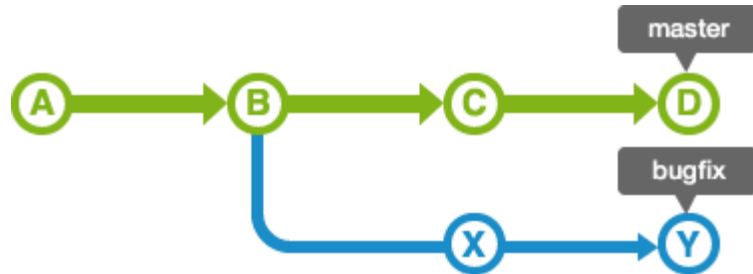
- Three-way merge

- 3개 commit에 대해 서로 다른 부분을 검사해야 함

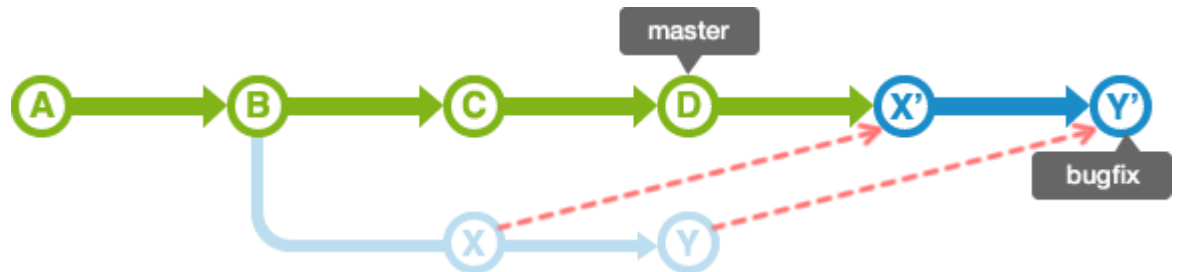


Branch: rebase

- `$ git checkout master`
- `$ git rebase bugfix`



- 커밋 이력까지 동일한 브랜치에 통합
- 충돌은?



Branch: rebase

- merge
 - 변경 내용의 이력이 모두 그대로 남아 있기 때문에 이력이 복잡해짐.
- rebase
 - 이력은 단순해지지만, 원래의 커밋 이력이 변경됨.
정확한 이력을 남겨야 할 필요가 있을 경우에는 사용하면 안됨.
 - Local repository 에서 브랜치를 만들어 작업하다 push 해야 하는 경우,
굳이 브랜치의 흔적을 남길 필요가 없는 경우가 많음



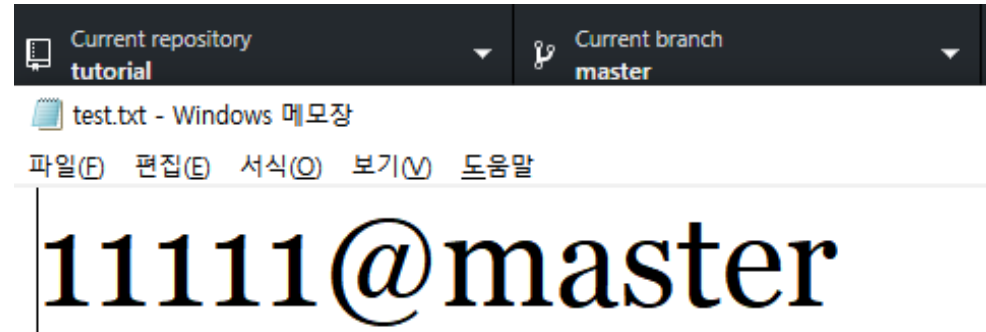
Branch: 충돌 관리

- “11111”
 - Mater: “11111@master”
 - Testing: “11111@testing”

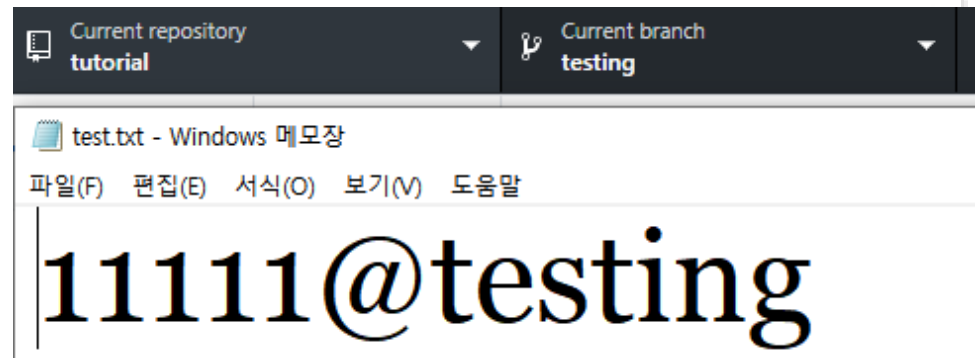
“11111”
commit



Master



Testing



Git bash 에서 기존 commit checkout 후 확인

```
MINGW64:/c/Users/phcph/Documents/GitHub/tutorial

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master|REVERTING)
$ pwd
/c/Users/phcph/Documents/GitHub/tutorial

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master|REVERTING)
$ ls
LICENSE  README.md  test.txt

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master|REVERTING)
$ cat test.txt
11111@master

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master|REVERTING)
$ git log
commit 9dae91ff431f5348d357acd2e75ecd5f7279e2f0 (HEAD -> master, origin/master)
Author: hcpark <phcphc@gmail.com>
Date:   Sun Sep 29 22:12:08 2019 +0900

    made other changes

    At master branch
    commit cc3e11ac6a54782e8edb96db91fb70e72d035310
    Author: hcpark <phcphc@gmail.com>
    Date:   Wed Sep 25 23:04:35 2019 +0900

        Create test.txt

commit 4b46c1ab5a37936b4a350edb706507ca9965bad4
Author: hcpark <phcphc@gmail.com>
Date:   Wed Sep 25 22:46:40 2019 +0900

    Initial commit
```

Git bash 에서 기존 commit checkout 후 확인

```
phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master|REVERTING)
$ git checkout cc3e11
Note: switching to 'cc3e11'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

    git switch -c <new-branch-name>

Or undo this operation with:

    git switch -

Turn off this advice by setting config variable advice.detachedHead to false

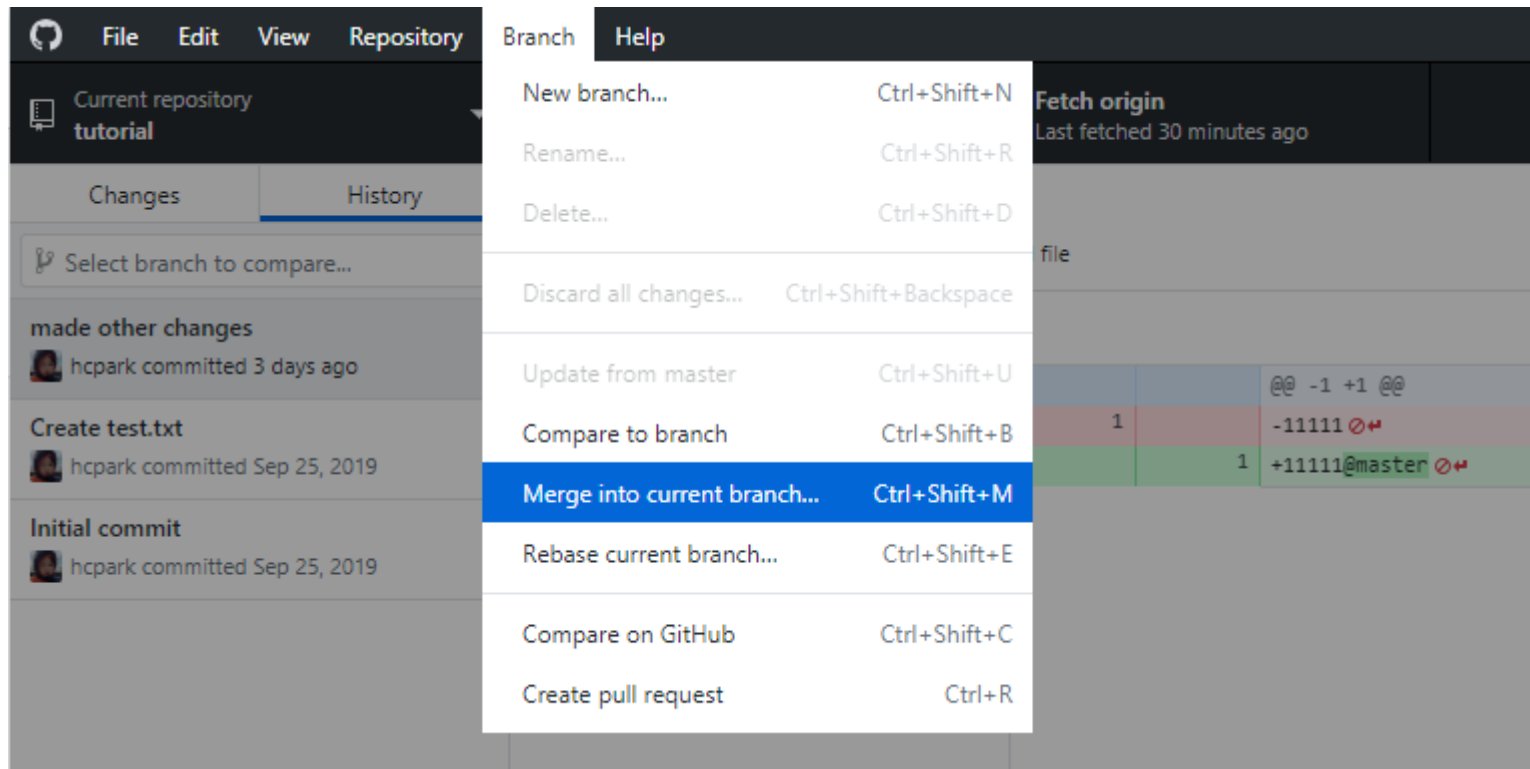
HEAD is now at cc3e11a Create test.txt
warning: cancelling a revert in progress

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial ((cc3e11a...))
$ cat test.txt
11111
phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial ((cc3e11a...))
$ git checkout master
Previous HEAD position was cc3e11a Create test.txt
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master)
$ cat test.txt
11111@master
phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master)
$
```

Branch: 충돌 관리

- \$ git checkout master
- \$ git merge testing
- @GitHub Desktop: Branch Menu -> Merge into ... (master branch 에서 수행)



Merge: Conflict 경고, Click Merge!

Merge into **master**


Filter

Default branch

✓ master 3 days ago

Recent branches


testing 3 days ago


There will be **1 conflicted file** when merging **testing** into **master**

Merge **testing** into **master**

Resolve conflicts before merging **testing** into **master**

1 conflicted file

 test.txt 1 conflict

Open in Visual Studio Code

[Open in command line](#), your tool of choice, or close to resolve manually.

Commit merge

Abort merge

충돌 발생한 내용이 자동 기재됨

수동으로 내용 수정: 실제로는 다른 개발자와 협의하고, 테스트하며 신중히 merge



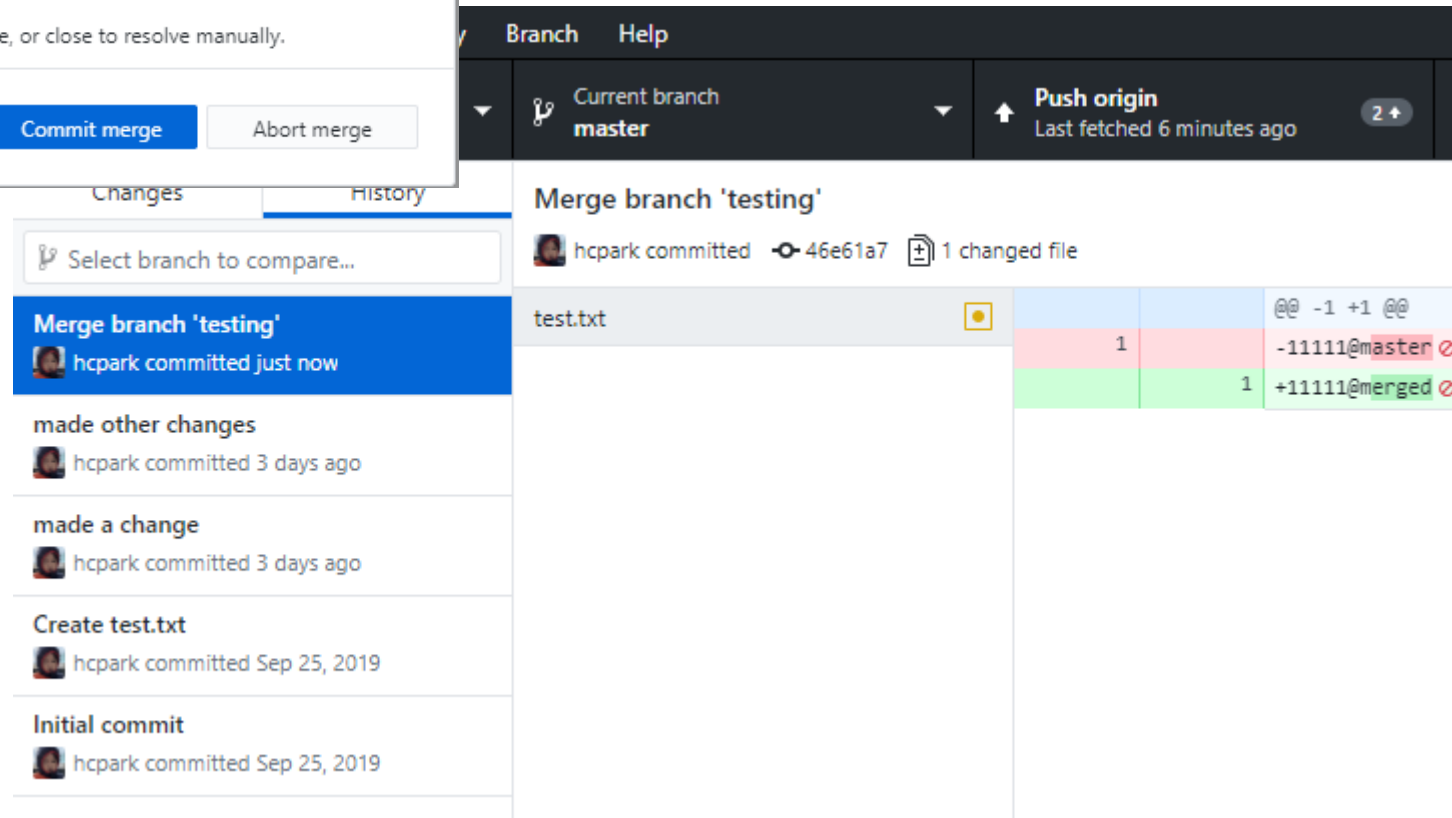
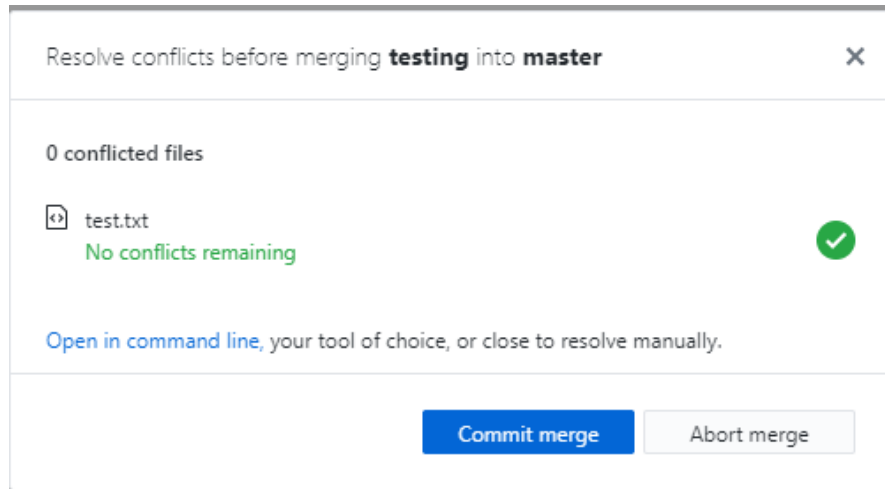
The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe" with the following text:

```
Microsoft Windows [Version 10.0.18362.356]  
(c) 2019 Microsoft Corporation. All rights reserved.  
C:\Users\phcph\Documents\GitHub\tutorial>notepad test.txt  
C:\Users\phcph\Documents\GitHub\tutorial>
```

Overlaid on the command prompt is a Notepad window titled "test.txt - Windows 메모장". The Notepad window contains the following text:

```
<<<<<<< HEAD  
11111@master  
=====|  
11111@testing  
>>>>>>> testing
```

충돌 해결 후 Commit Merge



Push to GitHub

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.



Push 2 commits to the origin remote

You have local commits waiting to be pushed to GitHub.

Always available in the toolbar when there are local commits waiting to be pushed or

Ctrl P

Push origin

Open the repository in your external editor

Select your editor in [Options](#)

Repository menu or Ctrl Shift A

Open in Visual Studio Code

View the files of your repository in Explorer

Repository menu or Ctrl Shift F

Show in Explorer

Open the repository page on GitHub in your browser

Repository menu or Ctrl Shift G

View on GitHub



Push to GitHub

[hyunchan-park / tutorial](#)

Watch 0Star 0Fork 0

[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

Branch: master ▾

Commits on Oct 2, 2019

Merge branch 'testing'
hyunchan-park committed 4 minutes ago

[46e61a7](#) [Code](#)

Commits on Sep 29, 2019

made other changes ...
hyunchan-park committed 3 days ago

[9dae91f](#) [Code](#)

made a change ...
hyunchan-park committed 3 days ago

[f7368c0](#) [Code](#)

Commits on Sep 25, 2019

Create test.txt
hyunchan-park committed 7 days ago

[cc3e11a](#) [Code](#)

Initial commit
hyunchan-park committed 7 days ago

[4b46c1a](#) [Code](#)

Branches at GitBash

```
$ git log --graph --pretty=oneline --abbrev-commit
*   46e61a7 (HEAD -> master, origin/master) Merge branch 'testing'
| \
|  * f7368c0 (origin/testing, testing) made a change
* | 9dae91f made other changes
|/
* cc3e11a Create test.txt
* 4b46c1a Initial commit

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master)
$ |
```

Branch: 기타

- \$ git branch --d "branch name"
 - Delete: 작업 중인 branch는 삭제 불가.
먼저 다른 branch로 HEAD를 옮기고 (checkout), 수행
- \$ git branch
 - 현재 브랜치 확인
- \$ git show-branch
 - 브랜치 확인
- \$ git log --graph --pretty=oneline --abbrev-commit

```
hcpark@hcpark-PC MINGW64 ~/Documents
$ git branch
devel
* master

hcpark@hcpark-PC MINGW64 ~/Documents
$ git show-branch
! [devel] devel 2nd
* [master] Merge branch 'devel'
--
- [master] Merge branch 'devel'
+* [devel] devel 2nd

hcpark@hcpark-PC MINGW64 ~/Documents
$
```

```
hcpark@hcpark-PC MINGW64 ~/Documents/temp (master)
$ git log --graph --pretty=oneline --abbrev-commit
* 23b8215 Merge branch 'devel'
|
| \
|  * 8d140c2 devel 2nd
|  * | ed1ba0d master 2nd
|  /
| /
* e2594cc 1st
```

Rebase 를 이용한 commit 정리



많이 발생하는 상황

- 작업 중 여러 Commit 이 발생하였으나,
remote repo 에는 하나의 commit으로 통합하여 push 하려 함
- Rebase: 여러 commit 을 하나로 merge
- 참고
 - [Git-book rebase](#)
 - [초보 몽키 rebase](#)



3개의 commit 작성

Current repository
tutorial

Current branch
master

Push origin
Last fetched 14 minutes ago 3 +

Changes

History

Select branch to compare...

update #3
hcpark committed just now

update #2
hcpark committed just now

update #1
hcpark committed just now

Merge branch 'testing'
hcpark committed 17 minutes ago

update #3
hcpark committed 29817fe 1 changed file

3333

test.txt

		@@ -1,2 +1,2 @@
1	1	11111@merged
2		-2222
	2	+3333

* 어찌됐든 최종 파일은 ->

test.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말

11111@merged

3333

3개의 commit 작성

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.



Push 3 commits to the origin remote

You have local commits waiting to be pushed to GitHub.

Always available in the toolbar when there are local commits waiting to be pushed or

Ctrl P

Push origin

Open the repository in your external editor

Select your editor in [Options](#)

Repository menu or Ctrl Shift A

Open in Visual Studio Code

View the files of your repository in Explorer

Repository menu or Ctrl Shift F

Show in Explorer

Open the repository page on GitHub in your browser

Repository menu or Ctrl Shift G

View on GitHub

- \$ git rebase -i HEAD~3

- HEAD에서 부터 최근 3개의 커밋 표시하며,
vi 를 기반으로 interactive 하게 commit을 수정, 통합함
- squash를 입력하면 Git은 해당 커밋과 바로 이전 커밋을 합치고
커밋 메시지도 Merge 한다.
- 저장하고 나서 편집기를 종료하면 Git은 3개의 커밋 메시지를
Merge 할 수 있도록 에디터를 바로 실행해준다.



```
pick 44b8fff update #1
pick f93269d update #2
pick 29817fe update #3

# Rebase 46e61a7..29817fe onto 46e61a7 (3 commands)
#
# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup <commit> = like "squash", but discard this commit's log message
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .      create a merge commit using the original merge commit's
# .      message (or the oneline, if no original merge commit was
# .      specified). Use -c <commit> to reword the commit message.
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out
```

Interactive Rebase

첫번째 commit 은 pick (base로 사용), 나머지는 squash (기존 commit에 합침)

```
MINGW64:/c/Users/phcph/Documents/GitHub/tutorial  
pick 81d4a35 update #1  
s 3128107 update #2  
s a8c6aaf update #3  
  
# Rebase 46e61a7..a8c6aaf onto 46e61a7 (3 commands)  
..
```

“:wq” 를 입력하여 vi 편집기에서 빠져나오면, 자동으로 커밋 메시지 수정하는 창 나옴

```
MINGW64:/c/Users/phcph/Documents/GitHub/tutorial  
# This is a combination of 3 commits.  
# This is the 1st commit message:  
  
update #1  
  
1111  
  
# This is the commit message #2:  
  
update #2  
  
2222  
  
# This is the commit message #3:  
  
update #3  
  
3333
```



```
MINGW64:/c/Users/phcph/Documents/GitHub/tutorial  
# This is a combination of 3 commits.  
# This is the 1st commit message:  
  
update #1,2,3  
  
1111 -> 2222 -> 3333  
"
```

결과 및 Log 확인

```
phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master)
$ git rebase -i HEAD~3
[detached HEAD b8c1f8f] update #1,2,3
Date: Wed Oct 2 23:36:51 2019 +0900
1 file changed, 2 insertions(+), 1 deletion(-)
Successfully rebased and updated refs/heads/master.

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master)
$ cat test.txt
11111@merged
3333

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master)
$ git log -2
commit b8c1f8fc7c2c7ab8e6093bb1a9784b5c84af0035 (HEAD -> master)
Author: hcpark <phcphc@gmail.com>
Date: Wed Oct 2 23:36:51 2019 +0900

    update #1,2,3

    1111 -> 2222 -> 3333

commit 46e61a7b04b8e3b1836d9d783da1c37d4ef08e90 (origin/master)
Merge: 9dae91f f7368c0
Author: hcpark <phcphc@gmail.com>
Date: Wed Oct 2 23:20:23 2019 +0900

    Merge branch 'testing'

phcph@LAPTOP-1PMORJFD MINGW64 ~/Documents/GitHub/tutorial (master)
$
```

만약 rebase 하다가 실수하면?

- \$ git rebase --edit-todo
 - 계속해서 중단된 작업 수행
- \$ git rebase --abort
 - Rebase 중단 및 원복



GitHub Desktop 확인 및 push (1개 commit)

The screenshot shows the GitHub Desktop interface. At the top, the 'Current repository' is 'tutorial', the 'Current branch' is 'master', and the 'Push origin' button is highlighted with a red box. Below this, the 'History' tab is selected, showing a list of commits. The most recent commit is 'update #1,2,3' by hcpark, committed 32 minutes ago. The commit message is '1111 -> 2222 -> 3333'. The file 'test.txt' is shown with a diff view. The diff shows a change from '-11111@merged' to '+11111@merged' and '+3333'.

Current repository: tutorial

Current branch: master

Push origin (Last fetched 13 minutes ago)

Changes: update #1,2,3

History: update #1,2,3

1111 -> 2222 -> 3333

test.txt

		@@ -1 +1,2 @@
1		-11111@merged
1		+11111@merged
2		+3333

Happy!

hyunchan-park / tutorial

Watch 0 Star 0 Fork 0

<> Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

Branch: master

Commits on Oct 2, 2019

update #1,2,3
hyunchan-park committed 34 minutes ago

b8c1f8f <>

Merge branch 'testing'
hyunchan-park committed 1 hour ago

46e61a7 <>

Commits on Sep 29, 2019

made other changes
hyunchan-park committed 3 days ago

9dae91f <>

made a change
hyunchan-park committed 3 days ago

f7368c0 <>

Commits on Sep 25, 2019

Create test.txt
hyunchan-park committed 7 days ago

cc3e11a <>

Initial commit
hyunchan-park committed 7 days ago

4b46c1a <>

Git 개인실습 #4

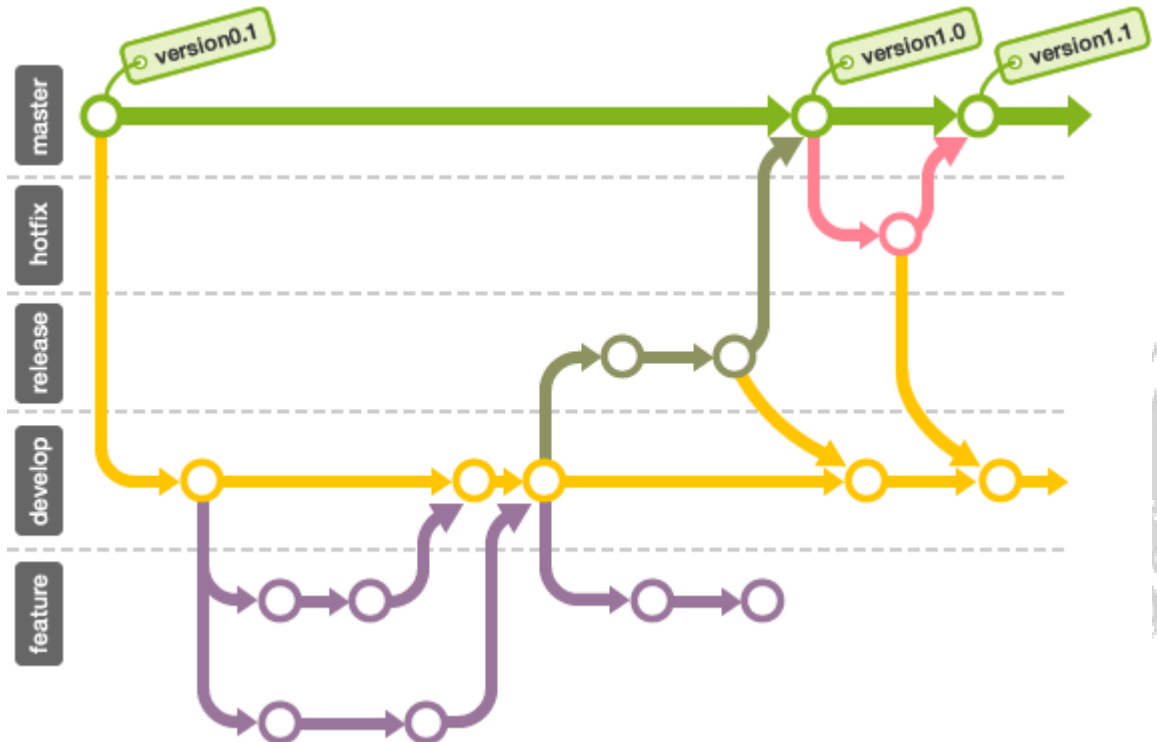
- 충돌 발생
 - Slide #16 -> conflict.jpg
- 충돌 해결 후 merge + GitHub Desktop 화면까지
 - Slide #18 -> resolved.jpg
- GitHub Push 완료 후, Git Bash 에서 브랜치 화면 확인
 - Slide #21 -> branches.jpg
- Rebase: 3개 commit 추가 생성 후, rebase 작업 수행
 - Slide #29 -> rebase.jpg (rebase 및 commit message 수정 화면 각각 포함)
- 제출 기한:
 - 10/27 (일) 23:59
 - 지각 감점: 5%p / day (3주 내 제출해야 함)

Branch 관리 전략



성공적인 (일반적인) Branch model

- 메인 브랜치(Main branch): Master, develop
- 피쳐 브랜치(Feature branch) 또는 토픽 브랜치(Topic branch)
- 릴리스 브랜치(Release branch)
- 핫픽스 브랜치(Hotfix branch)



Main branch: Master and develop

- 'master' 브랜치와 'develop' 브랜치, 이 두 종류의 브랜치를 보통 메인 브랜치로 사용합니다.
- **master**
'master' 브랜치에서는, 배포 가능한 상태만을 관리합니다.
커밋할 때에는 태그를 사용하여 배포 번호를 기록합니다.
- **develop**
'develop' 브랜치는 앞서 설명한 통합 브랜치의 역할을 하며, 평소에는 이 브랜치를 기반으로 개발을 진행합니다.

Feature branch

- Topic branch
- 새로운 기능 개발 및 버그 수정이 필요할 때
 - 'develop' 브랜치로부터 분기
 - 각 개발자에게 작업 분배 시 활용
 - 일반적으로 공유할 필요가 없기 때문에, 원격으로 관리하지 않음
 - 개발이 완료되면 'develop' 브랜치로 병합하여 다른 사람들과 공유



Release branch

- Master 브랜치로 병합하기 이전에, 병합 및 테스트 수행을 위한 브랜치
 - 해당 릴리즈를 위한 최종적인 버그 수정 등의 개발 수행
 - 버그를 수정하거나 새로운 기능을 포함한 상태로 모든 기능이 정상적으로 동작하는지 확인
 - 모든 준비를 마치고 배포 가능한 상태가 되면 'master' 브랜치로 병합
 - 릴리즈 번호 태그로 구분
 - 기타 사항
 - 관례적으로 브랜치 이름 앞에 ' release- ' 를 붙임
 - 다음 릴리즈를 위한 개발 작업은 'develop' 에서 따로 계속 진행
 - 릴리즈 브랜치에서 기능을 점검하며 발견한 버그 수정 사항은 'develop' 브랜치에도 적용
 - 배포 완료 후 'develop' 브랜치에 대해서도 병합 작업을 수행

Hotfix branch

- 배포한 버전에 긴급하게 수정을 해야 할 필요가 있을 경우, 'master' 브랜치에서 분기하는 브랜치
 - 이미 작업 중인 develop 에서 수정하기는 어려우므로, 기존 배포 버전을 기반으로 hotfix 수행
 - 일반적으로 'hotfix-###' 식으로 명명
 - 이후 develop 에도 병합하여 수정 사항 반영



Tensorflow: branches upon releases

[tensorflow / tensorflow](#)

[Watch](#) 8,595

[Star](#) 135,260

[Fork](#) 77,668

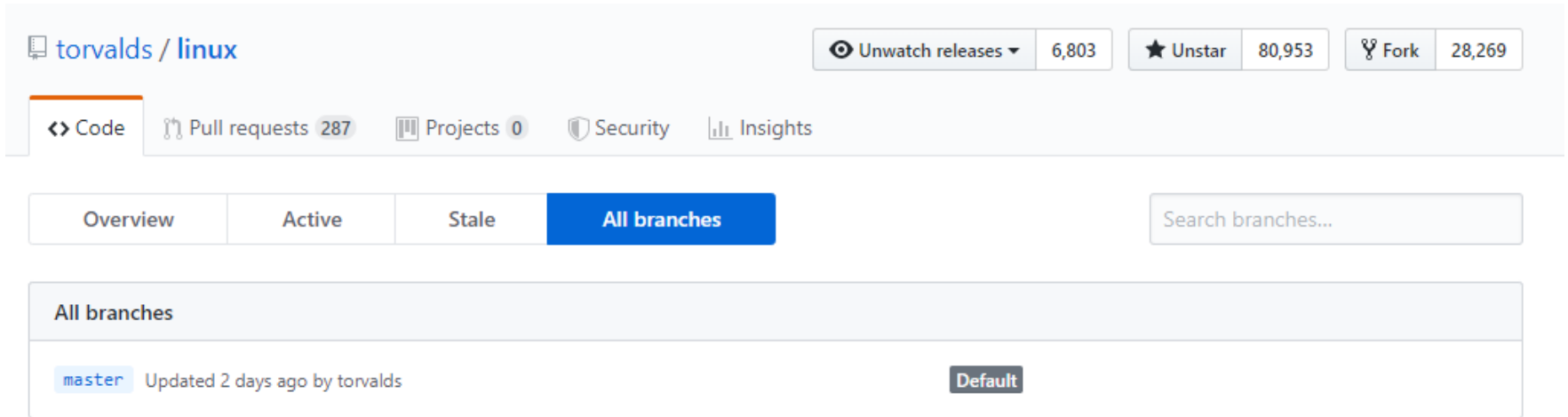
[Code](#) [Issues 2,537](#) [Pull requests 223](#) [Projects 1](#) [Security](#) [Insights](#)

[Overview](#) [Active](#) [Stale](#) [All branches](#)

Active branches

r1.15 Updated 2 days ago by yifeif		<div><div></div><div>5608 94</div></div>	Compare
r2.0 Updated 5 days ago by goldiegadde	✓	<div><div></div><div>6863 451</div></div>	Compare
jvishnuvardhan-patch-9 Updated 13 days ago by jvishnuvardhan	✗	<div><div></div><div>10681 6</div></div>	#29895 Open
1.8.0 Updated last month by gharibian	✗	<div><div></div><div>5223 0</div></div>	Compare
r1.14 Updated last month by mihairuseac	✓	<div><div></div><div>12729 282</div></div>	Compare
master-where-we-want-it Updated 2 months ago by nicolasvasilache	✗	<div><div></div><div>6863 0</div></div>	Compare
r1.13 Updated 3 months ago by mihairuseac	✗	<div><div></div><div>23133 217</div></div>	#31740 Closed

Linux: only 1 branch



The screenshot shows the GitHub interface for the 'torvalds / linux' repository. At the top, the repository name is displayed with icons for watching releases (6,803), starring (80,953), and forking (28,269). Below this is a navigation bar with links for Code, Pull requests (287), Projects (0), Security, and Insights. The 'All branches' tab is selected, showing a list of branches. Only one branch, 'master', is listed, with the note 'Updated 2 days ago by torvalds' and a 'Default' label.

torvalds / linux

Unwatch releases 6,803 Unstar 80,953 Fork 28,269

<> Code Pull requests 287 Projects 0 Security Insights

Overview Active Stale **All branches** Search branches...

All branches

master Updated 2 days ago by torvalds Default