

5주차 결과보고서

전공: 신문방송학과

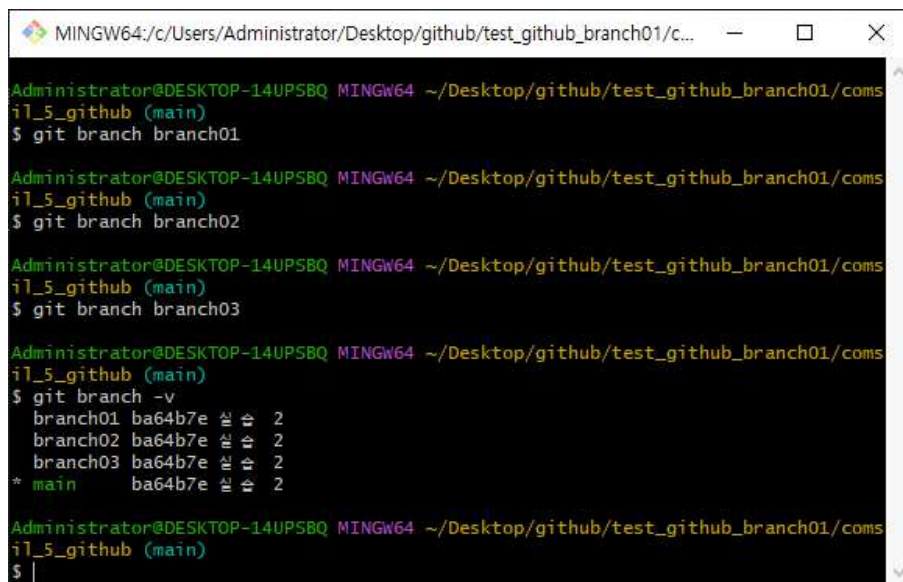
학년: 3학년

학번: 20191150

이름: 전현길

1. 앞의 실습 제출물들을 보고서에 포함할 것.

1. 3개 이상 branch를 작성하고 각 branch에서 내용을 수정해 커밋한다.



```
Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (main)
$ git branch branch01

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (main)
$ git branch branch02

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (main)
$ git branch branch03

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (main)
$ git branch -v
  branch01 ba64b7e 실습 2
  branch02 ba64b7e 실습 2
  branch03 ba64b7e 실습 2
* main     ba64b7e 실습 2

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (main)
$
```

branch 3개를 생성하고, 연결된 branch 3개를 확인한다.

```
MINGW64:/c/Users/Administrator/Desktop/github/test_github_branch01/comsil_5_github
main
Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
(branch01)
$ git add test_branch01.txt

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
(branch01)
$ git commit -m "(branch01) create test_branch01.txt"
[branch01 7a33c9a] (branch01) create test_branch01.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 test_branch01.txt

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
(branch01)
$ git push origin branch01
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 280 bytes | 280.00 KiB/s, done.
Total 3 (delta 1), reused 1 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'branch01' on GitHub by visiting:
remote:   https://github.com/alogeclock/comsil_5_github/pull/new/branch01
remote:
To https://github.com/alogeclock/comsil_5_github.git
 * [new branch]      branch01 -> branch01

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
(branch01)
$
```

branch01에 test_branch01.txt 파일을 생성해 커밋한 뒤 푸쉬한다.

```
MINGW64: c:/Users/Administrator/Desktop/github/test_github_branch01/co...
Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (branch01)
$ git checkout branch02
Switched to branch 'branch02'

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (branch02)
$ git add .

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (branch02)
$ git commit -m "(branch02) create test_branch02.txt"
[branch02 7176649] (branch02) create test_branch02.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 test_branch02.txt

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (branch02)
$ git push origin branch02
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 280 bytes | 280.00 KiB/s, done.
Total 3 (delta 1), reused 1 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'branch02' on GitHub by visiting:
remote:   https://github.com/alogeclock/comsil_5_github/pull/new/branch02
remote:
To https://github.com/alogeclock/comsil_5_github.git
 * [new branch]      branch02 -> branch02

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/coms
il_5_github (branch02)
$
```

branch02에 test_branch02.txt 파일을 생성해 커밋한 뒤 푸쉬한다.

```
Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (branch02)
$ git checkout branch03
Switched to branch 'branch03'

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (branch03)
$ git add .

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (branch03)
$ git commit -m "(branch03) create test_branch03.txt"
[branch03 1af2121] (branch03) create test_branch03.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 test_branch03.txt

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (branch03)
$ git push origin branch03
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 279 bytes | 279.00 KiB/s, done.
Total 3 (delta 1), reused 1 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'branch03' on GitHub by visiting:
remote:   https://github.com/alogeclock/comsil_5_github/pull/new/branch03
remote:
To https://github.com/alogeclock/comsil_5_github.git
 * [new branch]      branch03 -> branch03

Administrator@DESKTOP-14UP58Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (branch03)
$ |
```

branch03에 test_branch03.txt 파일을 생성해 커밋한 뒤 푸쉬한다.

2. 모든 branch를 main branch에 merge한 뒤 push한다.

```
Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
l_5_github (branch03)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
l_5_github (main)
$ git branch -v
  branch01 7a33c9a (branch01) create test_branch01.txt
  branch02 7176649 (branch02) create test_branch02.txt
  branch03 1af2121 (branch03) create test_branch03.txt
* main      ba64b7e 2 commits ahead

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
l_5_github (main)
$ git merge branch01
Updating ba64b7e..7a33c9a
Fast-forward
 test_branch01.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test_branch01.txt

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github
l_5_github (main)
$ git merge branch02
Merge made by the 'ort' strategy.
 test_branch02.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test_branch02.txt

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (main)
$ git merge branch03
Merge made by the 'ort' strategy.
 test_branch03.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test_branch03.txt

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 5 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (main)
$ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 476 bytes | 476.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/alogeclock/comsil_5_github.git
   ba64b7e..be4cc9f  main -> main

Administrator@DESKTOP-14UPS8Q MINGW64 ~/Desktop/github/test_github_branch01/comsil_5_github (main)
$
```

현재 main branch와 연결되어 있는지 git checkout main 명령어를 통해 확인한 뒤, branch를 하나하나 merge했다. 이후 git status로 커밋이 잘 반영되어 있는지 확인하고 push했다.

Public
Pin
Unwatch 1

main
4 Branches
0 Tags

t
+
Code

Merge branch 'branch03'
 be4cc9f · 3 minutes ago
12 Commits

1주차	실습 1	4 days ago
2주차	실습 1	4 days ago
3주차	실습 1	4 days ago
4주차	실습 1	4 days ago
5주차	5주차 실습 제출물	4 days ago
test_branch01.txt	(branch01) create test_branch01.txt	16 minutes ago
test_branch02.txt	(branch02) create test_branch02.txt	11 minutes ago
test_branch03.txt	(branch03) create test_branch03.txt	9 minutes ago
test_github.txt	실습 2	4 days ago
보고서양식.hwp	실습 1	4 days ago
소프트웨어개발도구및환경실습_...	실습 1	4 days ago

실제 repository의 main branch를 확인하면 각 branch의 파일들이 합병되어 있다.

아래는 branch01, branch02, branch03 브랜치의 파일 상태이다. 중간의 test_branch01.txt, test_branch02.txt, test_branch03.txt 파일이 각각 반영되어 있는 모습을 확인할 수 있다.

 alogeclock (branch01) create test_branch01.txt	7a33c9a · 19 minutes ago	 8 Commits
 1주차	실습 1	4 days ago
 2주차	실습 1	4 days ago
 3주차	실습 1	4 days ago
 4주차	실습 1	4 days ago
 5주차	5주차 실습 제출물	4 days ago
 test_branch01.txt	(branch01) create test_branch01.txt	19 minutes ago
 test_github.txt	실습 2	4 days ago
 보고서양식.hwp	실습 1	4 days ago
 소프트웨어개발도구및환경실습_...	실습 1	4 days ago

 alogeclock (branch02) create test_branch02.txt	7176649 · 14 minutes ago	 8 Commits
 1주차	실습 1	4 days ago
 2주차	실습 1	4 days ago
 3주차	실습 1	4 days ago
 4주차	실습 1	4 days ago
 5주차	5주차 실습 제출물	4 days ago
 test_branch02.txt	(branch02) create test_branch02.txt	14 minutes ago
 test_github.txt	실습 2	4 days ago
 보고서양식.hwp	실습 1	4 days ago
 소프트웨어개발도구및환경실습_...	실습 1	4 days ago

 alogeclock (branch03) create test_branch03.txt	1af2121 · 12 minutes ago	 8 Commits
 1주차	실습 1	4 days ago
 2주차	실습 1	4 days ago
 3주차	실습 1	4 days ago
 4주차	실습 1	4 days ago
 5주차	5주차 실습 제출물	4 days ago
 test_branch03.txt	(branch03) create test_branch03.txt	12 minutes ago
 test_github.txt	실습 2	4 days ago
 보고서양식.hwp	실습 1	4 days ago
 소프트웨어개발도구및환경실습_...	실습 1	4 days ago

2. 협업을 할 때 github를 사용하는 것이 어떤 장점이 있는지 기술하여라.

git이란 리눅스 토르발스에 의해 개발된 버전 관리 시스템(version control system: VCS)으로, 프로그램의 소스 코드 및 문서의 변경된 점을 관리해주는 프로그램이다. 특히 분산형 버전 관리 시스템이기 때문에 로컬 파일을 다운받으면 중앙 서버와 관계 없이 협업을 진행할 수 있으며, 서버가 다운되거나 심지어 데이터가 손상되더라도 로컬 파일을 통해 백업할 수 있다는 장점이 있다.

github는 이러한 git을 기반으로 한 호스팅 서비스를 지원하여, 온라인상으로 코드를 공유하고 함께 작업할 수 있도록 한다. 사용자는 원격 저장소(repository)를 생성하여 파일을 커밋(commit)한 뒤 푸쉬하거나, 다른 사람이 수정한 파일을 풀(pull)함으로써 파일을 보다 쉽게 관리하고, 보다 안전하게 백업할 수 있다.

git과 github를 사용하지 않는다면 ppt에서 볼 수 있었듯이 프로그램을 직접 버전별로 분류하여 저장하게 되는데, 이 경우 버전, 날짜, 수정한 사람 등을 일일이 구분하여 기록하는 것이 쉽지 않다. 수정한 파일이 겹치거나, 한쪽에서는 삭제된 파일이 다른 쪽에서는 존재하는 등의 문제가 발생할 경우에도 문제가 발생한다. 프로그램이 작을 때에는 큰 문제가 되지 않겠지만, 프로그램의 기능이 추가되고 구현사항이 복잡해질수록 문제는 심각해진다.

버그가 발생했을 때의 디버깅 문제, 파일 안전성 문제를 고려하면 깃허브의 중요성은 더욱 커진다. 깃허브를 사용하지 않는다면 버그가 발생했을 때 디버깅을 위해 단순 주먹구구식으로 이전 버전을 사용해볼 수밖에 없다. 깃허브도 큰 개요는 같지만, commit message를 통해 구현 사항을 정리한 뒤 이와 함께 파일을 갱신한다는 점이 다르다. 커밋 메시지를 확인하면 문제가 발생할 만한 기능이 추가되었던 날을 추론할 수 있다. 뿐만 아니라 branch를 이용해 파일을 갱신했다가 프로그램이 안전한 것을 확인한 뒤 main branch에 merge하는 방식을 통해 안정성을 높일 수 있다.

파일 백업의 문제 역시 github를 사용하는 쪽이 더 효과적이다. 중앙 서버의 DB 또는 로컬 컴퓨터에 프로그램의 버전을 직접 저장하여 관리하는 방식은 항상 파일이 날아가거나 손상될 위험을 안고 있다. 그에 비해 중앙 서버와 실제 작업용 로컬 컴퓨터 양쪽에 데이터를 저장하는 github의 방식은 파일 손상의 문제에서 훨씬 안전하다고 볼 수 있다.