

Object – Oriented Programming

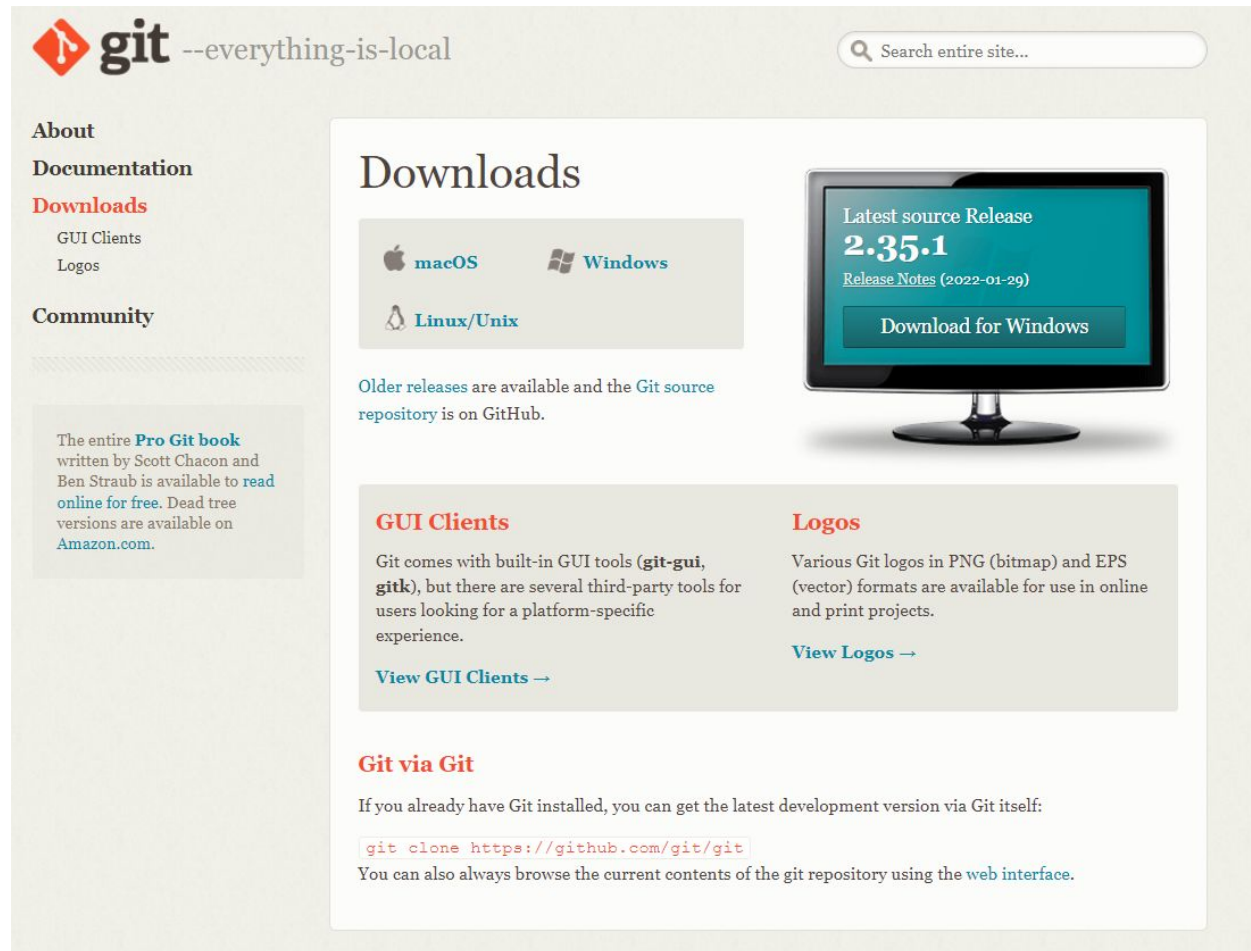
LAB #3-2. GIT

Git

- 소스 코드의 변경 이력을 관리하는 버전 관리 시스템
- 코드는 여러 PC(로컬)와 원격 저장소에 분산하여 저장
- 여러 개발자들과 협업 시 코드 관리에 유리 : 병렬 개발 가능
- 저장소 종류 : GitHub, GitLab, BitBucket

Git 설치

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



The screenshot shows the Git Downloads page. At the top left is the Git logo with the tagline "--everything-is-local". To the right is a search bar. On the left sidebar, there are links for "About", "Documentation", "Downloads" (highlighted in red), "GUI Clients", "Logos", and "Community". Below these is a text block about the "Pro Git book". The main content area is titled "Downloads" and features a section for the "Latest source Release 2.35.1" with a "Download for Windows" button. Below this, there are sections for "GUI Clients" and "Logos". At the bottom, there is a section titled "Git via Git" with instructions on how to clone the repository.

git --everything-is-local

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloads

Older releases are available and the Git source repository is on GitHub.

Latest source Release
2.35.1
[Release Notes \(2022-01-29\)](#)
[Download for Windows](#)

GUI Clients
Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.
[View GUI Clients →](#)

Logos
Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.
[View Logos →](#)

Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

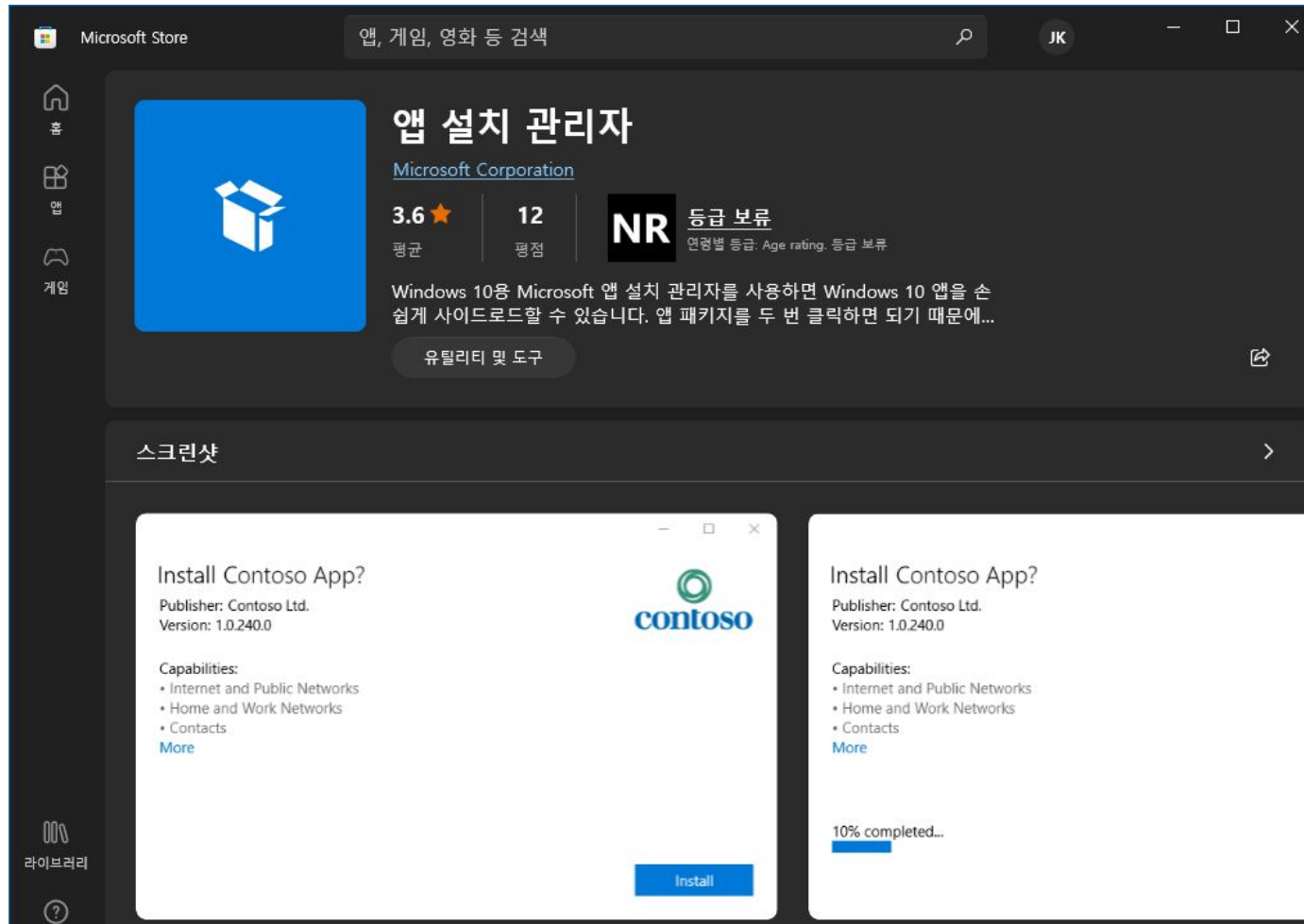
```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the [web interface](#).

Windows

1. 인스톨러 이용 <https://goddaehee.tistory.com/216>
2. winget (Windows Package Manager)
<https://docs.microsoft.com/ko-kr/windows/package-manager/winget/#production-recommended>

Windows



링크 클릭하면
마이크로소프트
스토어로 이동!

<https://www.microsoft.com/en-us/p/app-installer/9nblggh4nns1>

Windows

설치 한 뒤에 Powershell 열고 다음 명령어 입력

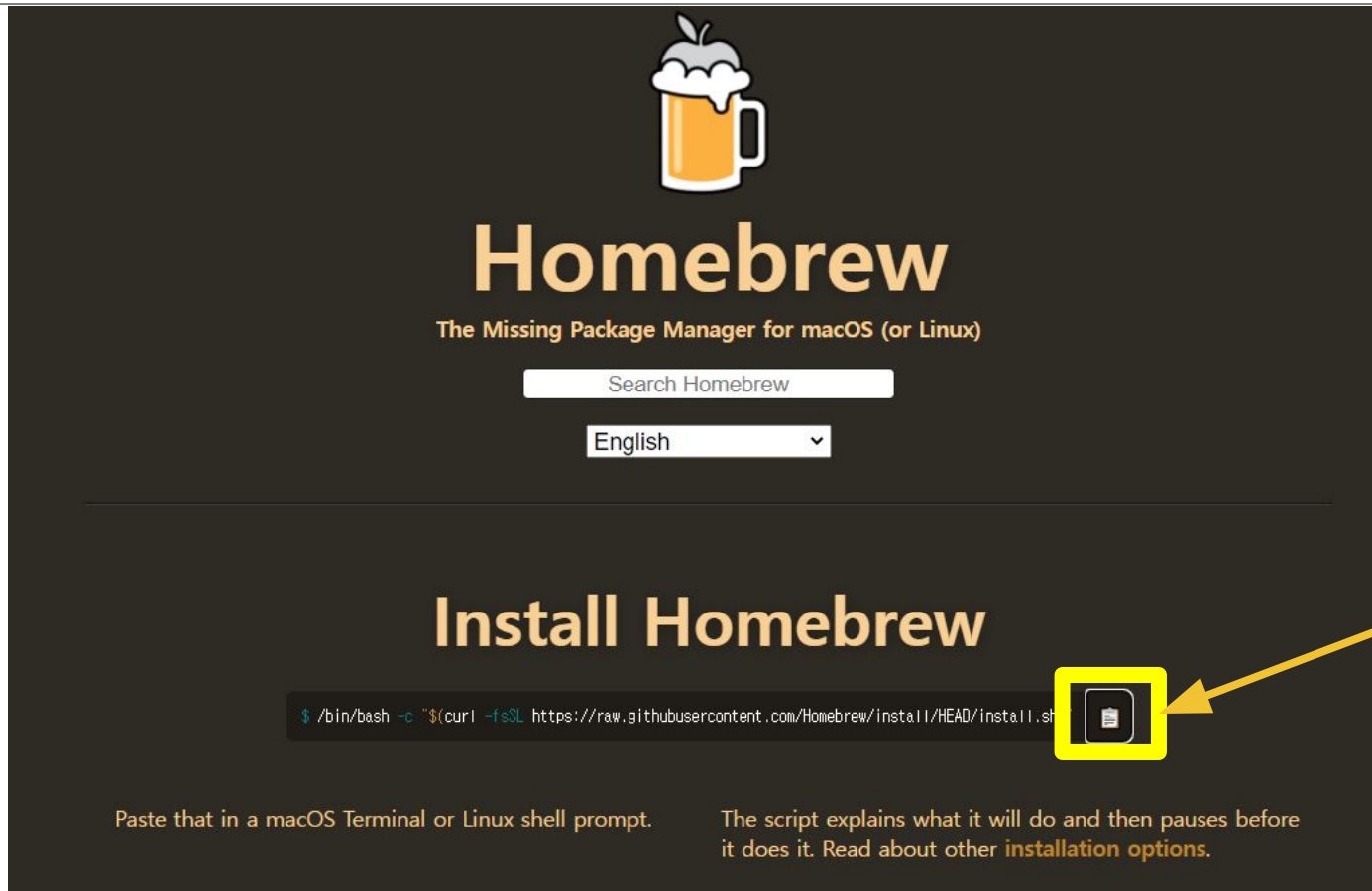
```
winget install --id Git.Git -e --source winget
```

혹시 path 수정해야하면 아래 링크 참조
<https://bongbongreview.tistory.com/55>

macOS

1. Homebrew 설치 (macOS Package Manager)
https://hyeonjiwon.github.io/etc/git_install/
(M1 사용자는 Rosetta설정 주의-링크에 설명되어있음)
2. `brew install git`

macOS



복사 버튼 클릭 하고
터미널에 붙여넣고 엔터!

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```


Linux

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

리눅스는 각자 운영체제에 맞는 스크립트를 입력!

예를 들어 우분투의 경우 아래 스크립트 입력

Debian/Ubuntu

For the latest stable version for your release of Debian/Ubuntu

```
apt-get install git
```

Terms of Git

- Repository : 프로젝트가 저장되는 공간(서버)
- Remote : 프로젝트가 저장된 원격 서버
- Branch : 분기점, 새로운 작업 시 현재 상태를 복사하여 branch에 작업 후 합병(Merge)
- Head : 현재 작업 중인 branch
- Master(Main) : 가장 기본이 되는 branch
- Commit : 유의미한 작업이 완료 되었을 때 복구/평가를 할 수 있는 체크포인트
- Merge : 특정 branch에서 작업한 내용을 현재 branch로 가져와 합치는 작업

Git commands

- git init : 버전관리를 할 pc의 폴더에서 초기화
- git clone (git 주소) : 원격 저장소의 코드와 작업 이력들을 내 pc로 가져옴
- git status : git 저장소의 상태를 체크
- git add (파일) : 작업하는 파일을 git이 관리할 수 있게 함
- git commit : 작업한 파일들과 내용을 로컬 저장소에 기록을 남김
- git push : 로컬 저장소에서 수행한 commit들을 원격 저장소에 올림
- git pull : 원격 저장소에 있는 새로운 작업 내용들을 로컬로 가져옴

Git commands

- git branch (새 branch 이름) : 특정 시점에서 새로운 branch를 만듦
- git checkout (branch 이름): 새로운 branch로 이동
- git merge (합병할 branch) : 특정 branch에서 작업한 내용들을 head로 합병
- git reset (돌아갈 commit) : 작업 이력을 지우고 특정 commit 시점으로 돌아 감
- git revert (되돌릴 commit) : 작업 이력을 지우지 않고 특정 commit을 취소함
 - revert는 되돌릴 commit 작업 이력도, revert 수행 기록도 남김

Git 기본 사용법 안내

2. git clone 시 요구하는 Username은 학번으로,
Password는 GitLab webpage에서 설정한 password로 입력

3. Clone받은 폴더로 이동 (처음에는 텅 빈 디렉토리)

```
$ cd 2022_ite2037_{학번}
```

4. Git 사용자 설정하기

```
$ git config user.name "{학번}"  
$ git config user.email "{학번}@hanyang.ac.kr"
```

(user.name은 학번으로,
user.email은 GitLab에 등록해놓은 email로 (기본값: 학번 + @hanyang.ac.kr))

Git 기본 사용법 안내

1. 설치 완료 후, 생성되어 있는 학생의 Git repository clone받기

```
$ git clone https://hconnect.hanyang.ac.kr/2022_ITE2037_12388/2022_ite2037 {학번}.git
```

Git clone 주소는 GitLab webpage의 해당 프로젝트 메인 화면으로 이동하여 확인 가능

2022_ITE2037_12388 > 2022_ITE2037_2021186018 > Details

The screenshot shows the GitLab interface for a project named '2022_ITE2037_2021186018'. The project ID is 3689. It has 5 commits, 2 branches, 0 tags, and 225 KB of files. A description states: 'This project is for Assistant taking "OBJECT-ORIENTED SYSTEMS DESIGN" class in 2022-1.' The interface includes a 'main' branch selector, a '2022_ite2037_2021186018' path, and a '+ v' button. A 'Clone' button is highlighted with a red box. Below it, a dropdown menu shows 'Clone with SSH' and 'Clone with HTTPS' options. The 'Clone with HTTPS' option is also highlighted with a red box, showing the URL 'https://hconnect.hanyang.ac.kr/' and a lock icon.

2022_ITE2037_2021186018

2022_ITE2037_2021186018

Project ID: 3689

5 Commits 2 Branches 0 Tags 225 KB Files

This project is for Assistant taking "OBJECT-ORIENTED SYSTEMS DESIGN" class in 2022-1.

main 2022_ite2037_2021186018 / + v

History Find file Web IDE Clone v

Update README.md 김재하 authored 1 week ago

README Auto DevOps enabled Add LICENSE Add CHANGELOG

Clone with SSH

git@hconnect.hanyang.ac.kr:2022

Clone with HTTPS

https://hconnect.hanyang.ac.kr/

Git 기본 사용법 안내

5. 작업 파일 생성

```
$ vi test.txt
```

*Windows의 경우

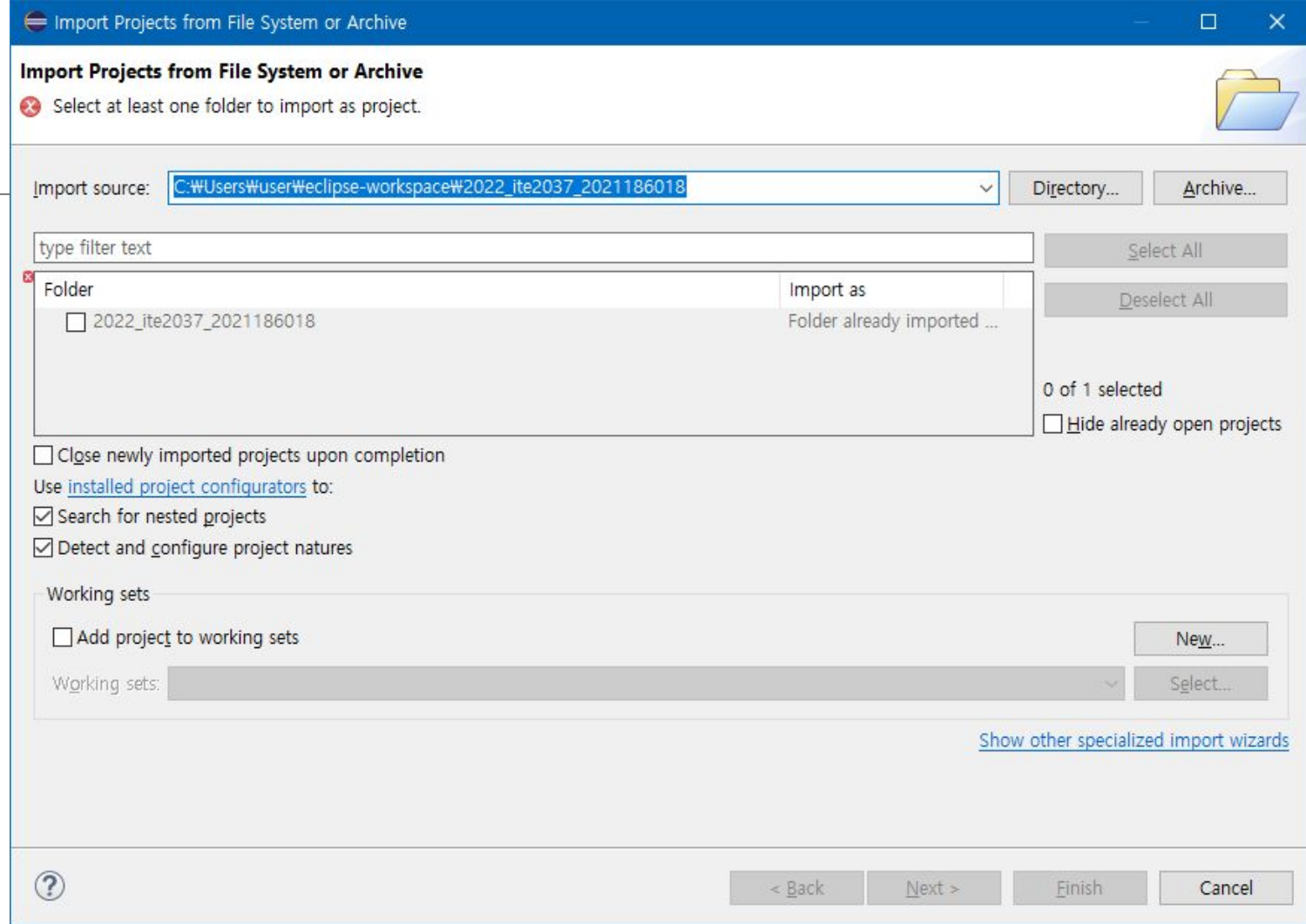
```
$ notepad test.txt (vscode 사용자는 code test.txt)
```

6. 파일 작성

Eclipse 에서 폴더 열기

1. Eclipse 실행
2. 환경 변수 추가 (계정의 환경 변수 추가) -> path 편집
path 에 추가 -> (C:\Users\user\eclipse\java-2021-12\eclipse)
3. 아까 git clone 했던 폴더에서

```
$ eclipse .
```
4. Finish



(추가 자료) Windows Vim 설치

1. winget install -e --id vim.vim

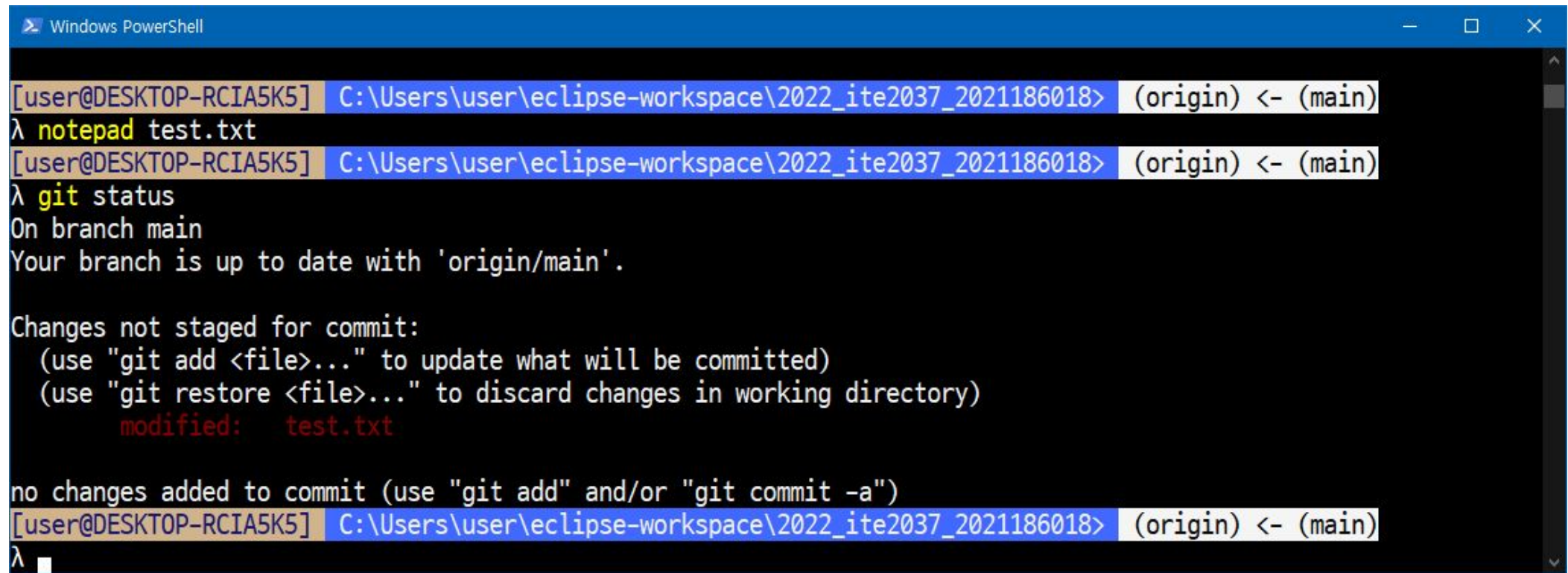
2. 환경변수 편집

(path 에 C:\Program Files\Vim\vim82 추가)

Git 기본 사용법 안내

7. 현재 git 관리 상태를 확인하면 test.txt가 관리되지 않는 상태로 표시된다.

```
$ git status
```



```
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ notepad test.txt
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ
```

Git 기본 사용법 안내

8. 현재 디렉토리에 있는 모든 추가/수정된 파일들을 Stage 영역으로 이동(test.txt가 git에 의해 관리됨)

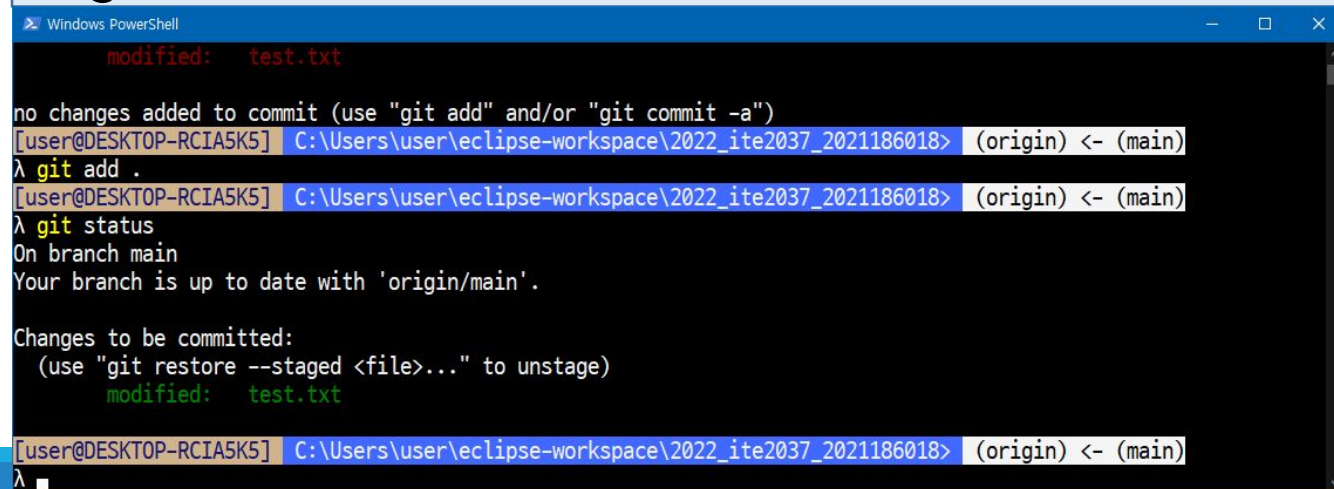
```
$ git add .
```

또는, 원하는 파일만 stage 영역으로 이동할 수 있음

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

9. Git 관리 상태를 다시 확인

```
$ git status
```



```
Windows PowerShell
modified: test.txt

no changes added to commit (use "git add" and/or "git commit -a")
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ git add .
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   test.txt

[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ
```

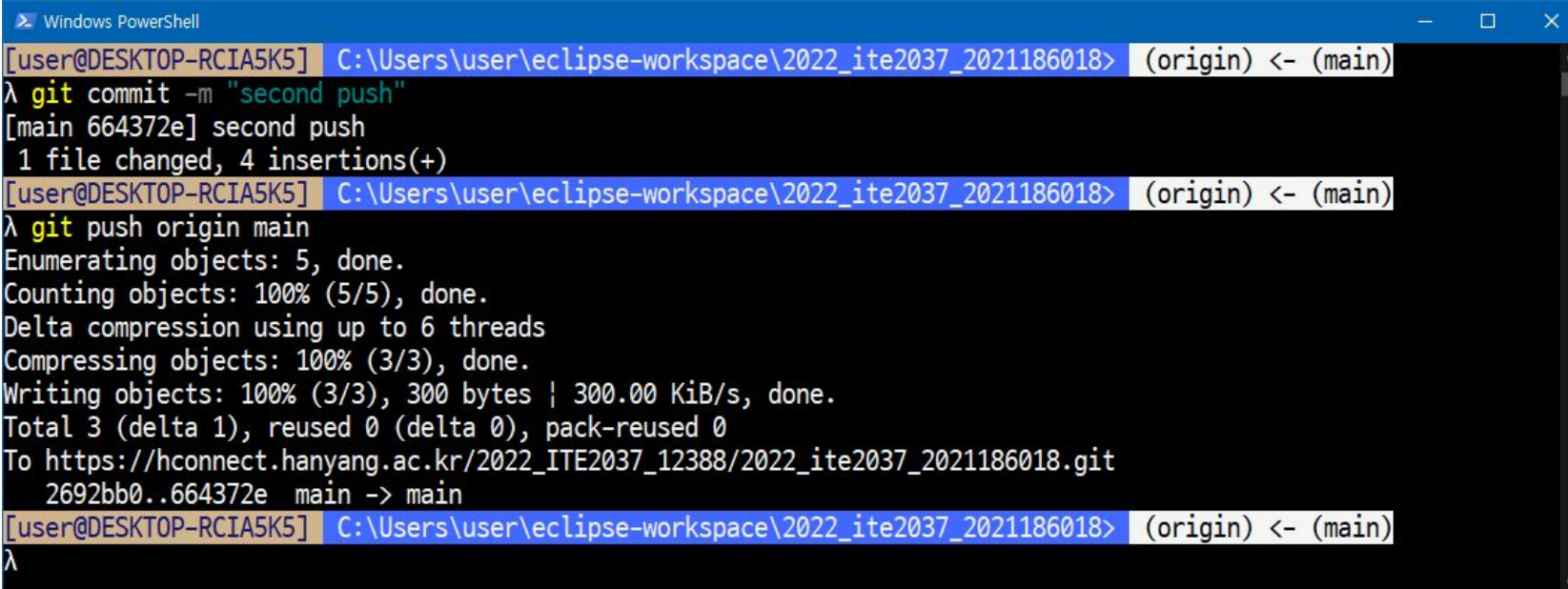
Git 기본 사용법 안내

10. 추가/수정된 파일을 커밋(Local repository에 저장)

```
$ git commit -m "first commit"
```

11. 커밋된 내용을 Remote repository로 전송

```
$ git push origin master 또는 git push origin main
```



```
Windows PowerShell
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ git commit -m "second push"
[main 664372e] second push
1 file changed, 4 insertions(+)
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 6 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 300 bytes | 300.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
To https://hconnect.hanyang.ac.kr/2022_ITE2037_12388/2022_ite2037_2021186018.git
2692bb0..664372e main -> main
[user@DESKTOP-RCIA5K5] C:\Users\user\eclipse-workspace\2022_ite2037_2021186018> (origin) <- (main)
λ
```

Git 기본 사용법 안내

12. git push를 통해 Remote로 전송된 파일은 GitLab webpage에서 확인 가능하다

2022 ITE2037_12388 > 2022 ITE2037_2021186018 > Commits

main



2022_ite2037_2021186018

Filter by commit message



14 Mar, 2022 1 commit



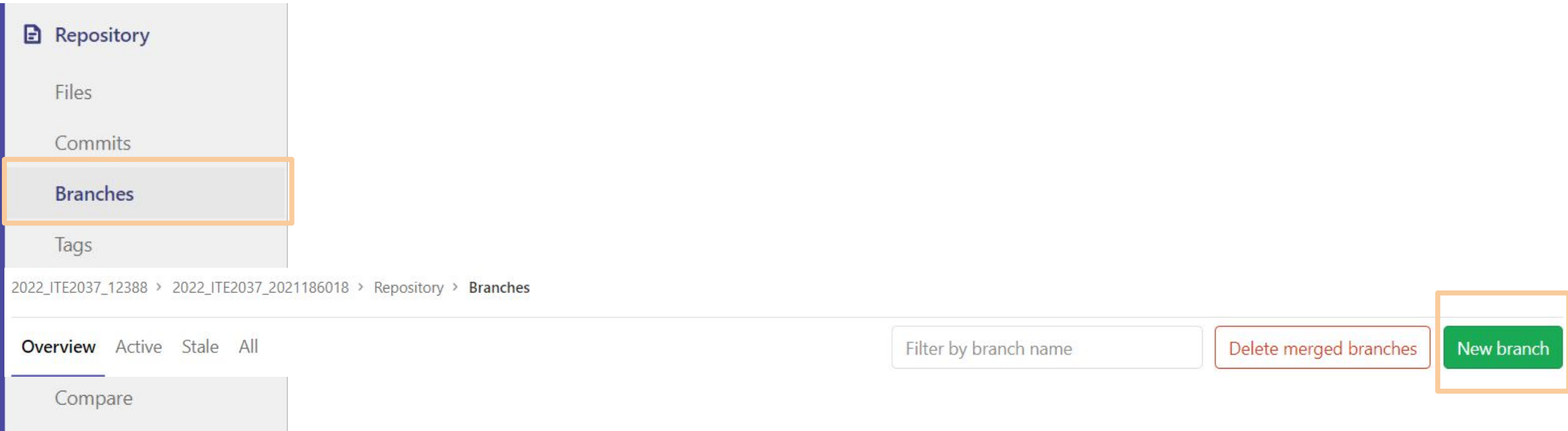
first commit

김재하 authored 16 minutes ago

2692bb02



GitLab 메인 브랜치 변경 방법



GitLab 메인 브랜치 변경 방법

Settings

General

Members

Integrations

Webhooks

Repository

CI / CD

Operations

Default Branch

Select the branch you want to set as the default for this project. All merge requests and commits will automatically be made against this branch unless you specify a different one.

Collapse

Default Branch

main

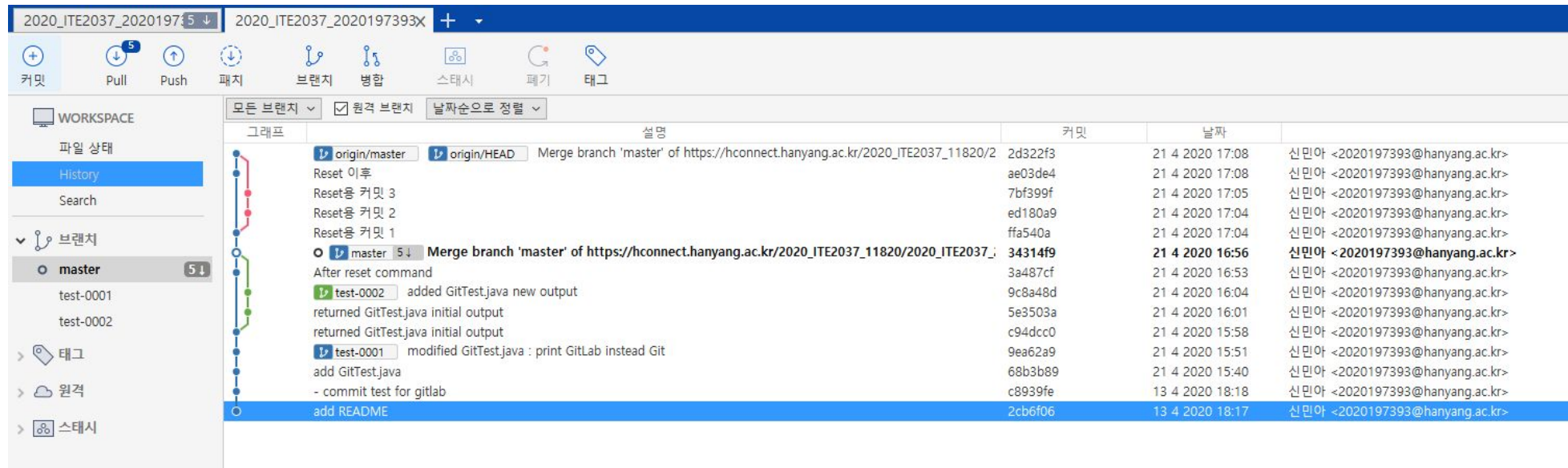
main

master

test

Git GUI Tool

- Branch 현황, 현재 작업 위치, 작업 이력 등을 GUI로 편하게 보거나 별도의 command 없이 add, commit, merge, branch를 간편하게 수행
- 예시 : SourceTree



실습

- 2주차, 3주차 과제 업로드
 - git clone, add, commit, push 명령어를 이용하여 gitlab에 업로드
- 이후의 과제 제출은 gitlab을 이용해서 제출
 - 다음과 같은 디렉토리 구조로 제출

```
2022_ite2037_{학번} └─ lab02
                    └─ lab03
                    └─ ⋮
                    └─ assignment1
```


(번외) Windows Powershell 꾸미기

1. notepad \$PROFILE -> Microsoft.PowerShell_profile.ps1 파일 수정
2. <https://superuser.com/questions/1259900/how-to-colorize-the-powershell-prompt>
3. 예시 코드

```
function Write-BranchName() {
    Try {
        $gitbranch = git branch
        $splited = $gitbranch.Split("*")
        return $splited
    }
    Catch{
        $nobranch = "none"
        return $nobranch
    }
}
function Write-RemoteName() {
    Try {
        $gitremote = git remote
        return $gitremote
    }
    Catch{
        $noremote = "none"
        return $noremote
    }
}
function prompt {
    $returnval1 = Write-BranchName
    $returnval2 = Write-RemoteName
    if($returnval2){
        "$([char]27)[38;2;25;25;112m$([char]27)[48;2;210;180;140m["$env:USERNAME+"@"$env:COMPUTERNAME+ "] $([char]27)[0m$([char]27)[38;2;240;240;240m$([char]27)[48;2;65;105;255m " + (Get-Location)
+ "> $([char]27)[0m$([char]27)[38;2;10;10;10m$([char]27)[48;2;245;245;245m (" + ($returnval2) +") <- (" + ($returnval1[1].Trim()) +")$([char]27)[0m`n$([char]27)[38;2;255;255;255m λ $([char]27)[0m "
    }
    else{
        "$([char]27)[38;2;240;240;240m$([char]27)[48;2;34;139;34m["$env:USERNAME+"@"$env:COMPUTERNAME+ "] $([char]27)[0m$([char]27)[38;2;240;240;240m$([char]27)[48;2;0;123;255m " + (Get-Location)
+ "> $([char]27)[0m " + "`n$([char]27)[38;2;255;255;255m λ $([char]27)[0m "
    }
}
```