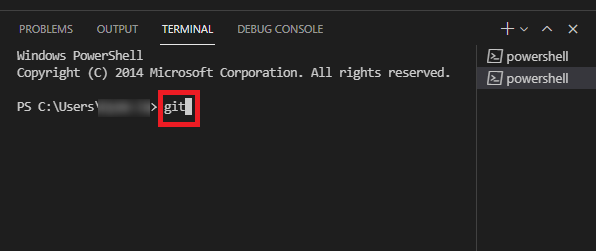
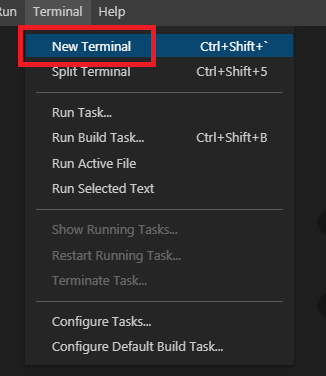
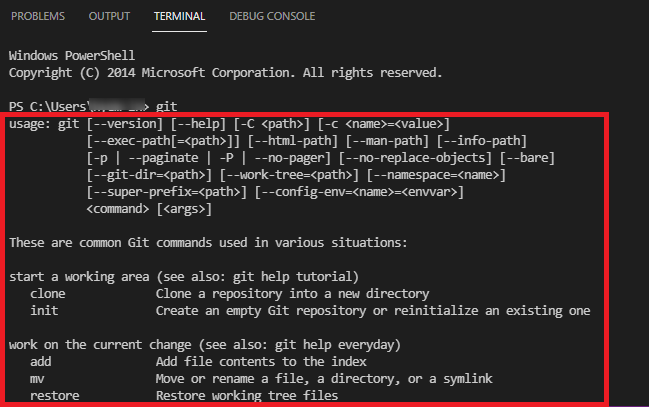
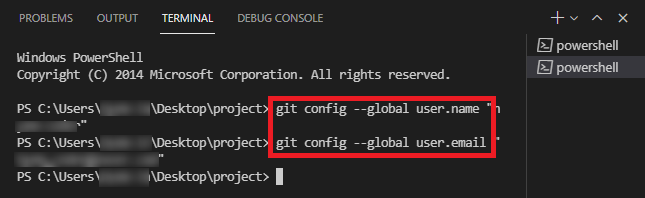
1. Github.com 가입
2. <https://git-scm.com/> -> git설치
3. Vscode
4. VS Code 실행 후 Git이 잘 설치되었지는 확인부터 해보겠다.   
     
   Terminal - New Terminal 클릭 후 화면 하단 Terminal에 "**git**"을 입력하고 Enter를 누른다.
5. 

Git이 잘 설치되었다면 Git 명령어들이 영어나 한글로 뜰 것이다.  
(만약, 뜨지 않는다면 VS Code를 껐다 다시 켜보거나 Git을 다시 설치하길 바란다.)



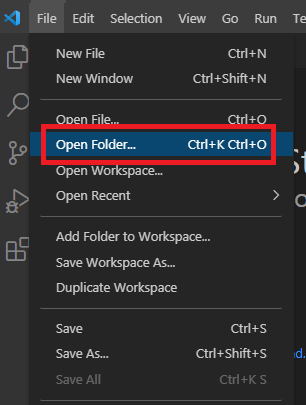
**git config --global user.name "사용자 이름"**  
**git config --global user.email "이메일"**

다른 IDE에서 git을 사용했을 경우에는 뜨지 않는 것 같지만, 만약 git계정과 관련된 팝업이 뜰 경우에는 git user 설정이 필요하다.  
  
Termianl에 다음과 같이 입력해주면 된다.  
**git config --global user.name "사용자 이름"**  
**git config --global user.email "이메일"**



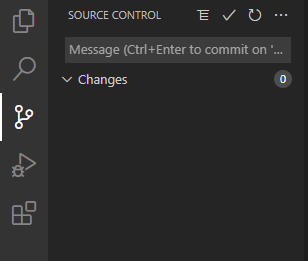
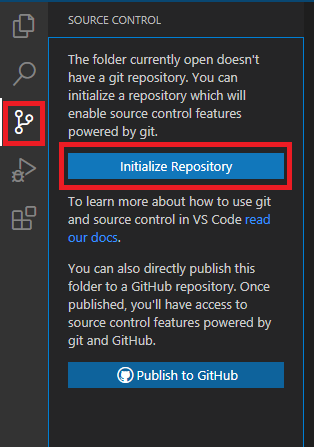
**2. 레포지토리 만들기**

※ 레포지토리를 만든다는 것은 VS Code의 파일들을 Git에게 "**버전 관리 해줘**" 라고 알려주는 것으로 git은 폴더 안에 레포지토리를 만들고 버전들을 저장한다.  
  
먼저 원하는 곳에 폴더를 만들고 VS Code에서 File - Open Folder에서 만들어둔 폴더를 선택한다.  
ex) 바탕화면에 project 폴더 생성

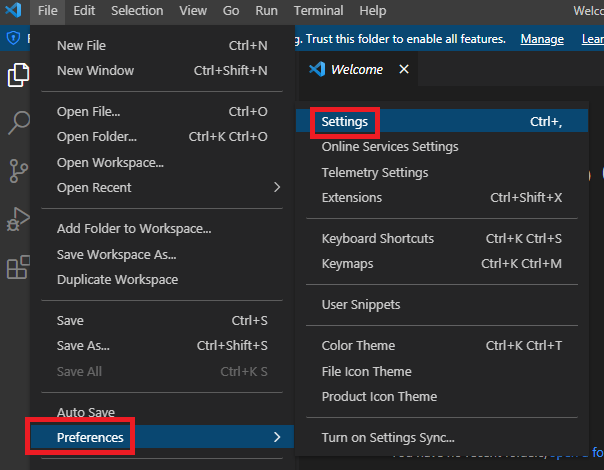


왼쪽 탭에서 Source Control - Initialize Repository 클릭하고 오른쪽 사진 화면 같이 뜨면 정상적으로 레포지토리가 생성된 것이다.

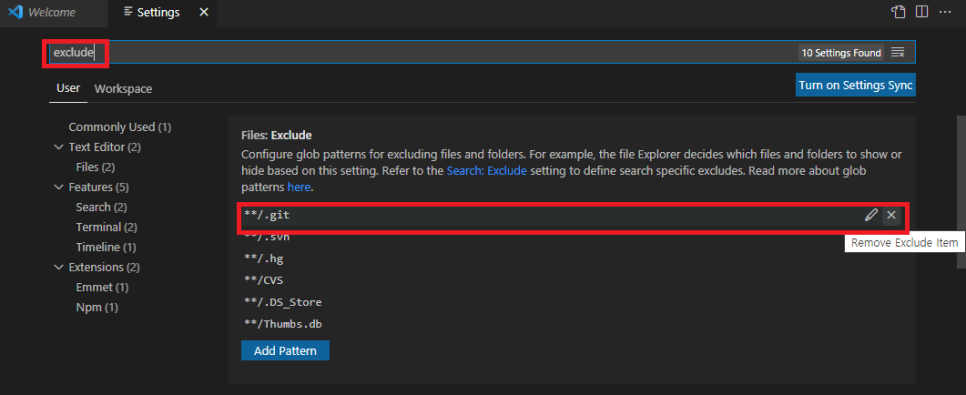
Git init



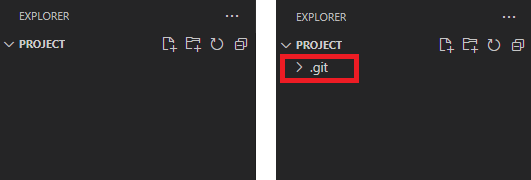
​  
VS Code 기본 설정 상 생성된 레포지토리를 볼 수 없는데, 만약 보고 싶다면  
windows 유저는 File - Preferences - Settings / mac 유저는 Code - 기본설정 - Settings 에 들어간 뒤



​  
설정 화면에서 **exclude** 라고 입력 검색 후 아래와 같이 "**\*\*/.git**"을 지워주면 된다.



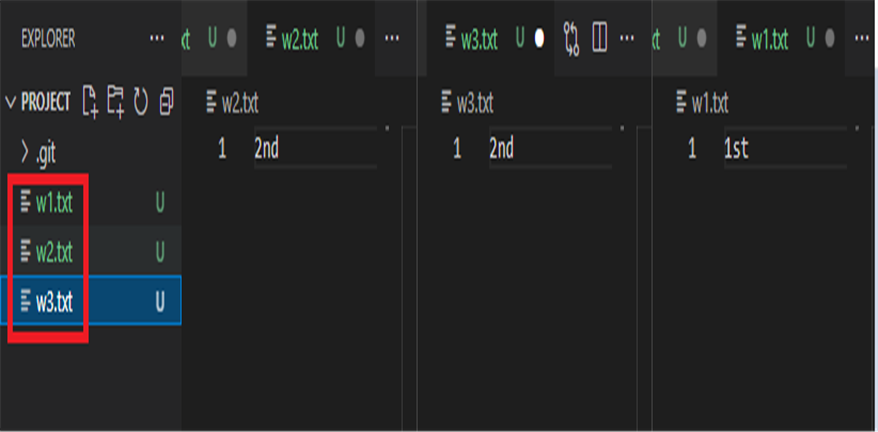
​그러면 EXPLORER 탭에 **.git**파일이 추가된 것을 확인할 수 있다.



​

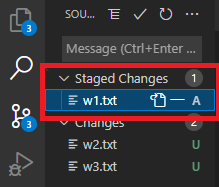
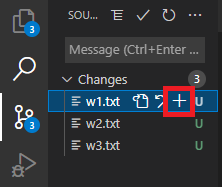
## 3. 버전 만들기​

각각 1st, 2nd, 2nd라는 내용이 입력된 w1, w2, w3라는 제목의 3개의 txt파일이 있다.  
이 3개의 파일을 버전 관리하려고 하는데 버전1에는 w1 파일만 버전2에는 w2, w3 파일도 포함시킬려고 한다.

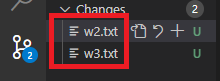
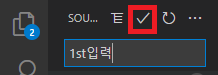


#### ① 버전1 만들기

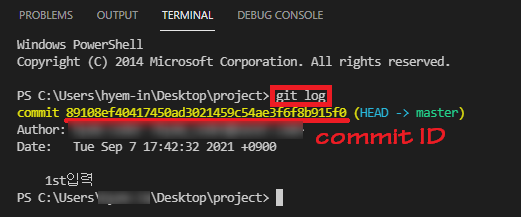
Source Control 탭에서 w1 파일의 +버튼을 누르면 commit 대기 상태인 Staged Changes 영역에  w1 파일이 옮겨진다.



commit Message에 수정 내역을 입력해주고 commit 버튼을 누르면 w1 파일이 사라진 것을 통해 성공적으로 버전이 만들어졌음을 알 수 있다.



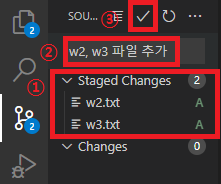
만들어진 버전을 보기 위해서는 Terminal - New Terminal에서 **git log** 를 입력하면 버전의 고유 식별자인 40글자의 commit ID와 함께 데이터 작성자, 작성 일자, 시간 등을 볼 수 있다.



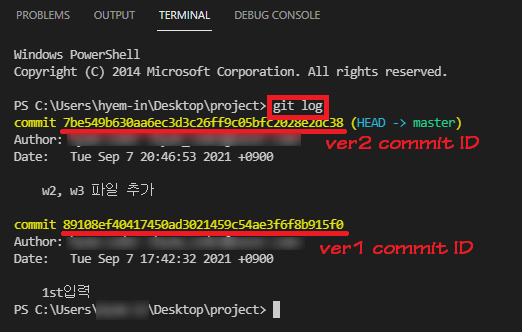
※commit ID는 데이터 내용, commit 메시지, 유저 이름 등을 바탕으로 생성

#### ② 버전2 만들기

버전2에는 w2 파일, w3 파일을 추가하려 한다.  
  
버전1을 만들었을 때와 같이 Source Control 탭에서 w2, w3 파일을 Staged Changes 영역에 올리고 commit 메시지를 입력하고 commit 버튼을 클릭한다.

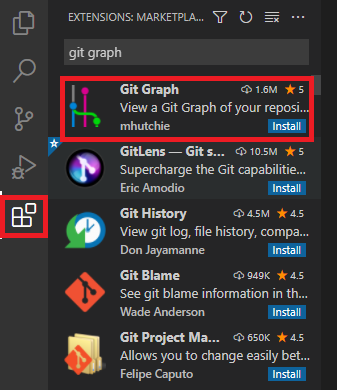


​  
git log 명령을 통해 버전2가 성공적으로 생성되었을을 확인할 수 있다.

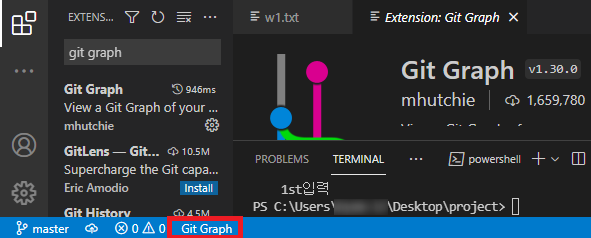


앞서 VS Code에서 깃으로 버전관리하는 법을 알아보았는데, 아무래도 Termianl에서 Git 명령어를 입력하는 방법이다보니 GUI 환경에 익숙한 다수의 사람들에게는 조금 어렵게 느껴졌을 수도 있다. 특히나 코딩을 이제 처음 접한 분들이라면 더욱더 그럴 것 같다.  
  
그래서 이번에는 **git graph**라는 확장기능을 이용해 GUI 환경에서 버전 관리 환경을 볼 수 있는 방법을 소개시켜주고자 한다.

왼쪽의 Extentions 탭에서 **git graph** 라고 검색하고 파란색 Install 버튼을 눌러 설치를 한다.

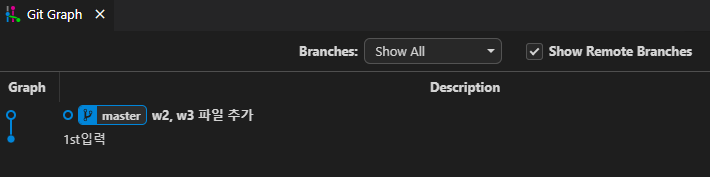


​  
설치가 끝나면 화면 왼쪽 하단에 Git Graph 탭이 생기는데, 그 탭을 클릭한다.

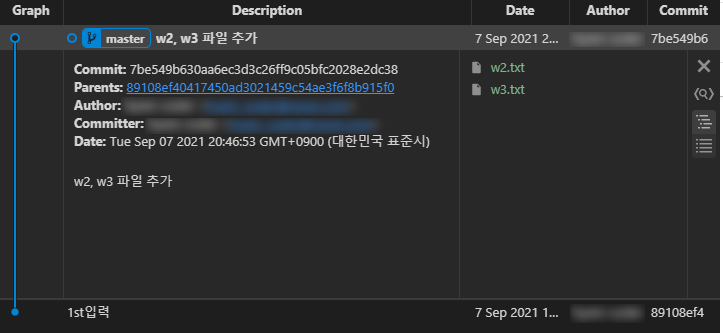


​

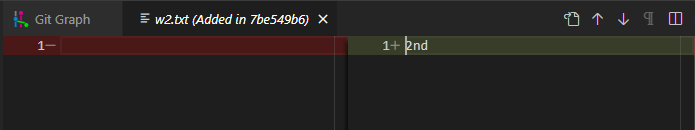
그러면 앞선 포스팅에서 만들어 둔 버전2를 확인할 수 있다.  
- 버전1 : 1st입력  
- 버전2 : w2, w3 파일 추가



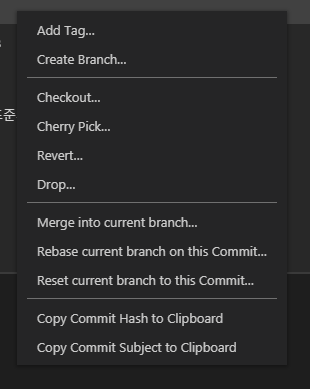
​  
각 버전의 commit 메시지를 클릭하면 git log 명령을 통해 볼 수 있던 commit 정보를 확인할 수 있다.



​  
위 사진 우측에 보이는 파일을 클릭하면 코드 변경 사항도 확인가능하다.



그리고 오늘 포스팅에는 없었지만 git을 사용하다보면 명령 코멘드로 처리해야할 것을 git graph를 사용하면 우클릭만 하면 gui로 사용할 수 있으니 참고하면 좋겠다.



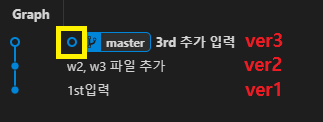
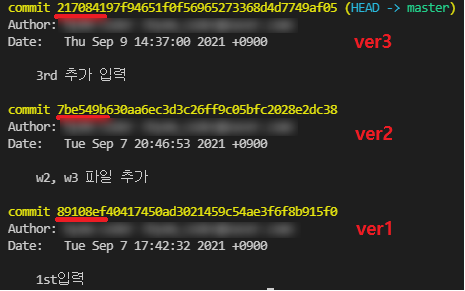
**[[Git]깃 버전 이동하기 - git checkout](https://hyemco.tistory.com/43)**

이번에는 이미 만들어 둔 여러 버전 중 원하는 버전으로 이동하는 법에 대해 알아보고자 한다.  
  
git을 사용하는 이유 중 하나가 버전 관리인데 만약 버전100까지 만들어 둔 상황에서 버그를 발견하여 버전81의 상태로 돌리고 싶은 경우에 사용된다.

※ 왼쪽(혹은 위 쪽) 사진은 git log 오른쪽(혹은 아래 쪽) 사진은 GIT GRAPH를 이용한 것이니 참고 바란다.

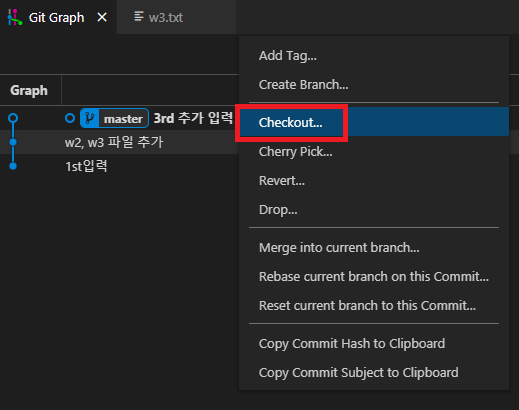
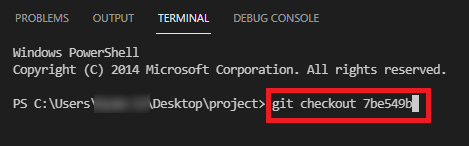
**1. 과거로 돌아가기**

현재 3개의 버전이 있다.

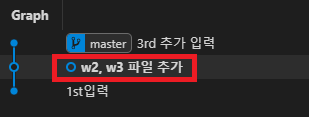
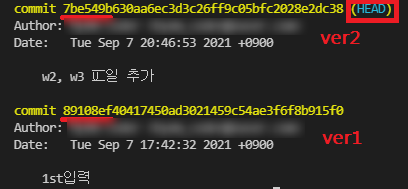


(참고로 GIT GRAPH에서 **HEAD**는 노란 네모 박스 안의 파란 원으로 **HEAD → master**는 **master** 주위의 파란 테두리로 확인할 수 있다.)

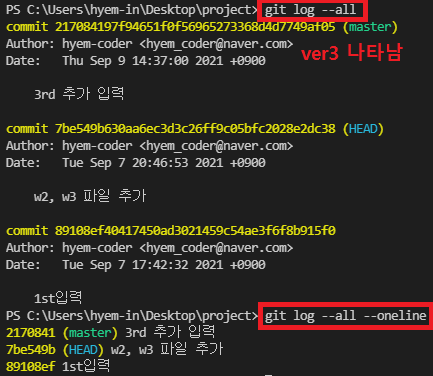
이 때, 버전2로 돌아가 수정을 하고 싶다고 가정하자.여기서 버전2로 돌아간다는 것은 버전2가 만들어졌던 시점의 Staging Area로 Working Directory가 변경된다는 뜻이다.  
  
상단 탭 Terminal - New Terminal에서 git checkout commid ID 를 입력한다.GIT GRAPH에서는 버전2 위에서 우클릭한 후 checkout 버튼을 누른다.



그러면 버전2로 **HEAD**가 이동한 것을 볼 수 있다 (워킹 디렉토리도 변경된다.)  
GIT GRAPH도 HEAD가 master를 가리키지 않아 master의 파란 테두리가 사라진 것을 볼 수 있다.



​  
+  
이 때, cmd상에는 버전3에서 작업한 것은 나타나지 않게 되는데 이는 "git log"가 **HEAD가 가리키는 버전 + 그 버전의 Parent 버전들**만을 보여주기 때문이다.  
  
모든 버전을 다 확인하고 싶다면 아래 명령어를 입력하면 된다.  
**git log --all**  
**git log -- all --oneline** (한줄로 확인)

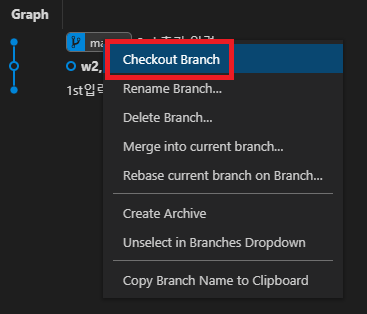


​

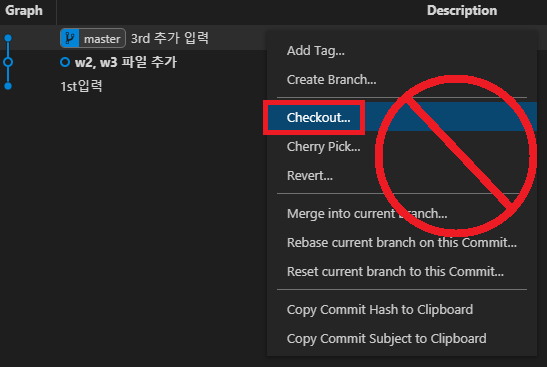
**2. 현재로 돌아가기**

이동한 버전에서 수정을 다 마쳤다면 최신 버전(버전3)에도 반영하기위해 현재(=최신 버전)로 돌아와야 될 것이다.  
  
현재로 돌아오는 방법도 동일하다. **git checkout master**를 입력하거나 master 박스 안에서 우클릭 - **CheckOut Branch**를 누른다.

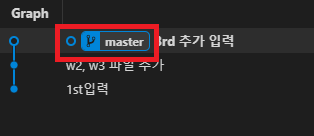
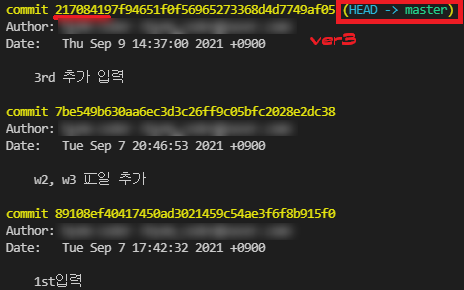
https://blog.kakaocdn.net/dn/lJlyi/btroWnwMouT/abX5nIsZH1OE1xvbpyxv0K/img.png



​  
+  
여기서 주의할 점은 GIT GRAPH에서 git checkout master를 할 때 master 박스를 벗어나 우클릭해 Checkout하지 않도록 하는 것이다



​git log와 GIT GRAPH를 확인해보면 **HEAD**가 **master**를 가리키는 것으로 최신 버전으로 돌아온 것을 알 수 있다. (GIT GRAPH는 master 주위에 파란색 테두리가 생긴 것으로 알 수 있다.)



**요약**

과거(=이전 버전)로 돌아가기  **git checkout commit ID**현재(=최신 버전)로 돌아가기  **git checkout master**모든 버전 다보기  **git log --all**모든 버전 한줄로 보기  **git log --all --oneline**