
소프트웨어 입문 설계

실습 6주차-2

VS Code 디버깅 및 Git 연동

김윤희 교수님

2021 봄학기

목차

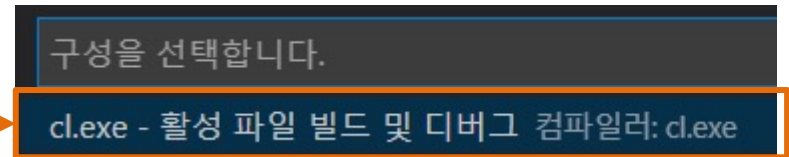
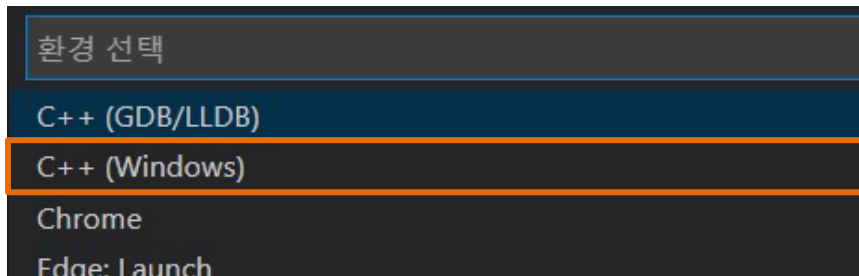
- VS Code 디버깅 및 Git 연동
- 과제 6-2 안내

실행 및 디버그 설정 (Windows)

- launch.json 생성

- Ctrl(Command) + Shift + D 입력
- 좌측의 "launch.json 파일 만들기" 클릭
- C++ (Windows) 선택
- cl.exe 선택

실행 및 디버그를 사용자 지정하려면
launch.json 파일 만들기를 수행합니다.



```
> Executing task: C/C++: cl.exe 활성 파일 빌드 <
빌드를 시작하는 중...
cl.exe /Zi /EHsc /nologo /Fe: C:\projects\hello.exe C:\projects\hello.c
hello.c
빌드가 완료되었습니다.
터미널이 작업에서 다시 사용됩니다. 닫으려면 아무 키나 누르세요.
```

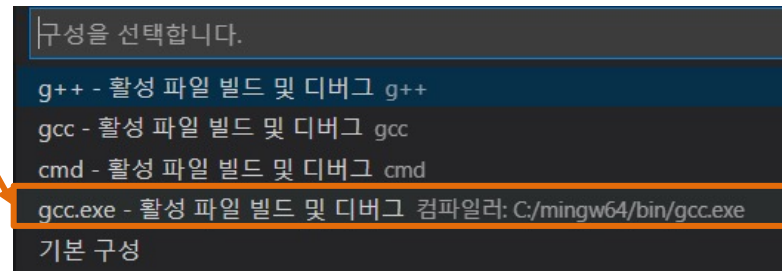
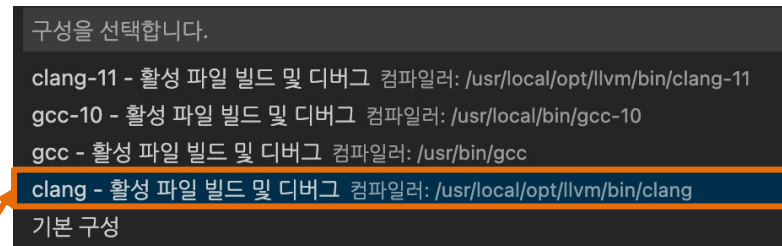
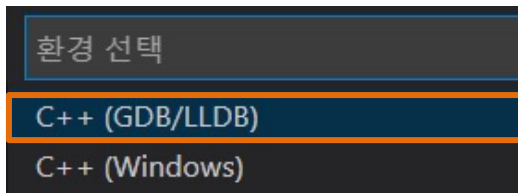
실행 및 디버그 설정 (Mac/Ubuntu)

- launch.json 생성

- Ctrl(Command) + Shift + D 입력
- 좌측의 "launch.json 파일 만들기" 클릭

실행 및 디버그를 사용자 지정하려면
launch.json 파일 만들기를 수행합니다.

- C++ (GDB/LLDB) 선택
 - Mac: clang 선택
 - Ubuntu: gcc.exe 선�ک



디버깅

- 디버깅(debugging)





- 프로그램 내 오류를 발견하고 그 원인을 규명하는 작업
- 특정 시점에서의 프로그램 상태(변수에 담긴 값 등)를 점검

- 방법

- 실행 및 디버그 메뉴 진입
- 대상 코드 내에서 실행 직전에 멈출 위치(중단점, breakpoint)를 지정
- "디버깅 시작" 버튼을 클릭하거나 F5 키를 입력해 디버깅 시작

- 상단 디버깅 관련 버튼 모음을 활용해 흐름 제어

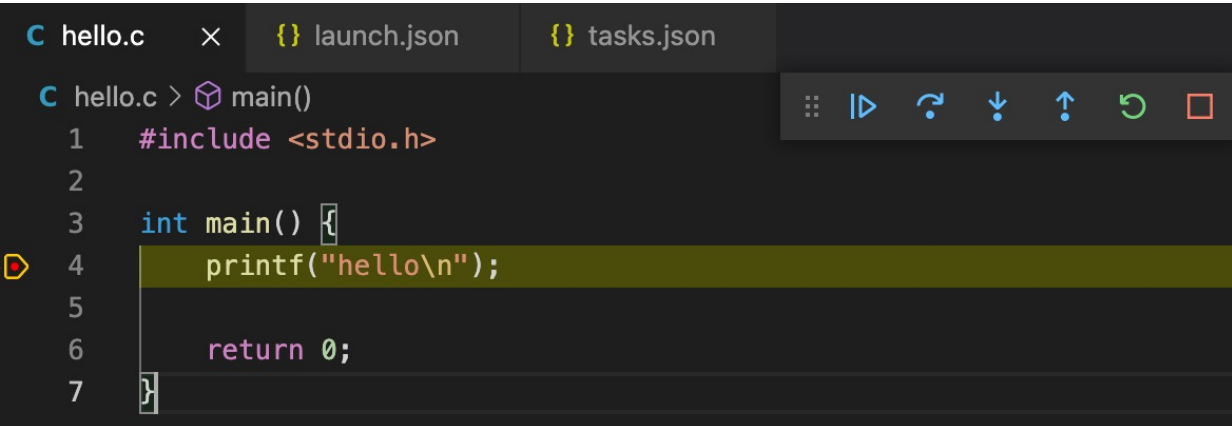


-  계속(F5): 현재 중단된 위치에서 이어서 실행
-  단위 실행(F10): 현재 위치의 코드 한 단만 실행 후 다시 중단
-  다시 시작(Ctrl/Command + Shift + F5): 디버깅 처음부터 재 시작
-  중지(Shift + F5): 디버깅 중단

디버깅

- 방법

- 대상 코드 내에서 실행 직전에 멈출 위치(중단점, breakpoint)를 지정
 - 특정 코드 라인 번호 좌측 공간을 클릭 시 중단점을 나타내는 붉은 색 점이 추가됨
 - "계속" 버튼(F5) 클릭 시 현재 강조된 코드 부분부터 이어서 실행
- 상단 디버깅 관련 버튼 모음을 활용해 흐름 제어



The screenshot shows a code editor with three tabs: 'hello.c', 'launch.json', and 'tasks.json'. The 'hello.c' tab is active, displaying a C program. The code is as follows:

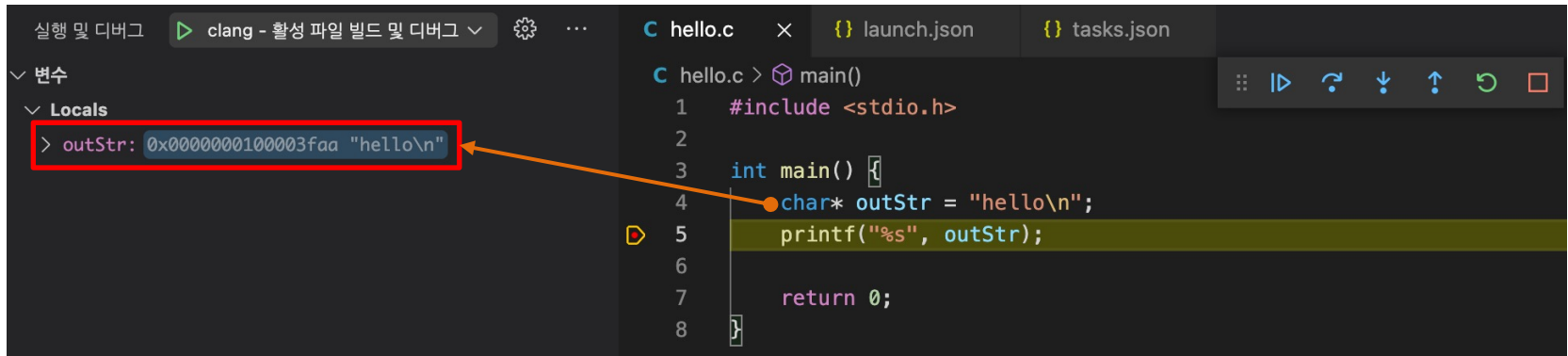
```
C hello.c > main()
1  #include <stdio.h>
2
3  int main() {
4      printf("hello\n");
5
6      return 0;
7  }
```

A red arrow labeled '중단점' (Breakpoint) points to a red dot on the left margin of line 4, indicating where a breakpoint has been set. The line containing the breakpoint is highlighted in yellow. To the right of the code, a toolbar contains several icons: a grid icon, a play icon, a refresh icon, a downward arrow icon, an upward arrow icon, a circular arrow icon, and a square icon.

디버깅

- 방법

- 좌측에서 각 변수명에 대한 메모리 상의 주소 및 값 확인 가능

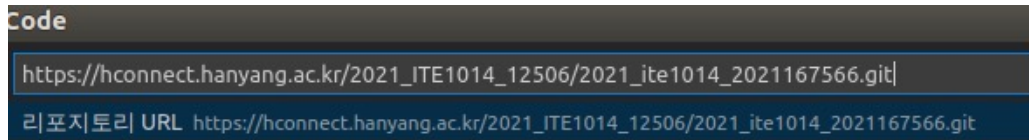


Git 연동

- **Repository 복제**

- VS Code 실행 후 좌측 첫번째 아이콘 클릭
- 리포지토리 복제 클릭

- clone할 주소 입력

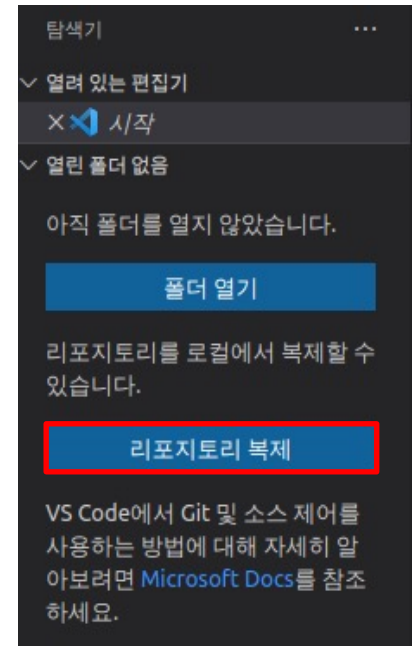


Code

`https://hconnect.hanyang.ac.kr/2021_ITE1014_12506/2021_ite1014_2021167566.git`

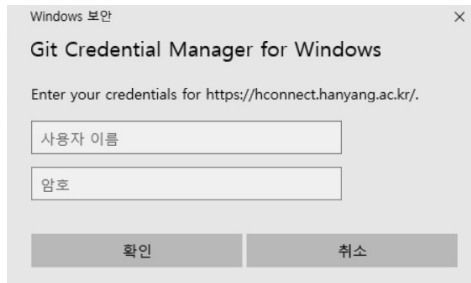
리포지토리 URL `https://hconnect.hanyang.ac.kr/2021_ITE1014_12506/2021_ite1014_2021167566.git`

- 복제할 폴더 선택 - 리포지토리 위치 선택 클릭

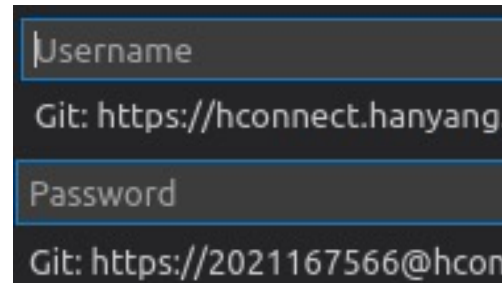


Git 연동

- GitLab 계정 정보 입력
 - 학번 및 비밀번호를 각각 입력

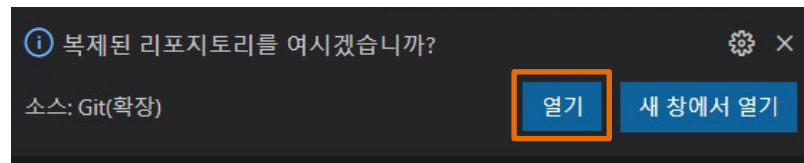


Windows



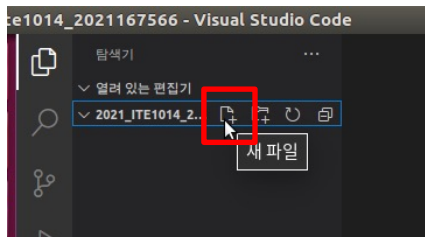
Ubuntu

- 복제된 레포지토리 "열기" 버튼 클릭



테스트 코드 작성

- Test 코드 작성
 - 새 파일 만들기
 - 파일 이름 작성
 - 테스트 코드 작성 후 저장

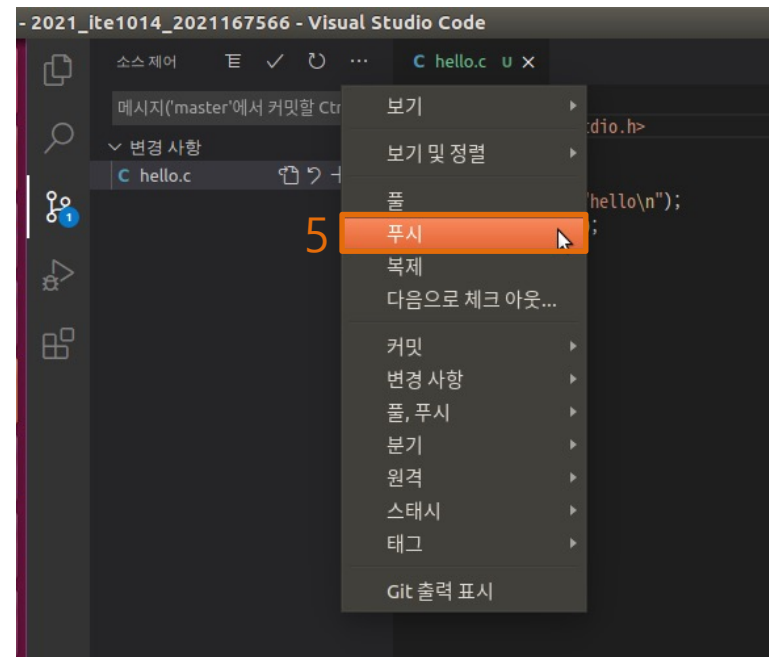
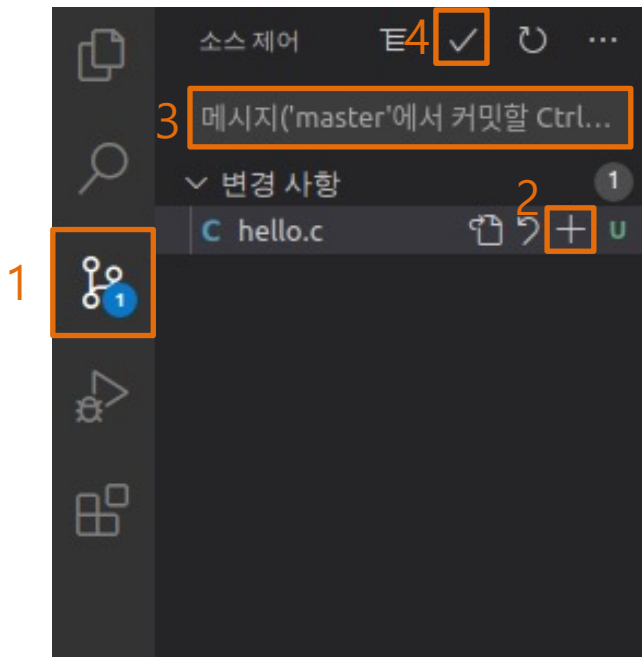


```
C hello.c u x
C hello.c > main()
1  #include <stdio.h>
2
3  int main(){
4      printf("hello\n");
5      return 0;
6  }
```

Git 푸시


- Git 서버에 변경 사항 커밋/푸시

1. 좌측 세번째 아이콘 클릭
2. 커밋할 파일들마다 마우스를 오버해 우측의 + 버튼 클릭
3. Commit message 작성 ex) first commit
4. 체크 아이콘 클릭
5. 푸시 클릭



Git 푸시

- GitLab 프로젝트 페이지에서 푸시된 내역 확인 가능

Name	Last commit	Last update
 hello.c	first commit	28 seconds ago

과제 6-2

- 제출 안내
 - **제출 기한: 4월 9일 (금) 23시 59분** (git push 완료 시점 기준)
 - 프로젝트 하위에 <과제번호>/<문제번호>/<답안 파일> 구조를 갖도록 제출
- 프로젝트 구조 예시

```
+ 2021_ITE1014_2021000001
+ 6-2/
+ 1/
+   - 1.c
+ 2/
+   - 2.c
+ 3/
+   - 3.c
```

과제 6-2

- 1번 문항
 - 구구단의 단수를 입력 받음
 - 해당하는 단수를 모두 출력
 - 반드시 `while` 반복문만 사용
- 제출 파일: 1.c

(실행 예)

4↵

4*1=4

4*2=8

4*3=12

4*4=16

4*5=20

4*6=24

4*7=28

4*8=32

4*9=36

과제 6-2

- 2번 문항
 - 구구단의 단수를 입력 받음
 - 해당하는 단수를 모두 출력
 - 반드시 for 반복문만 사용
- 제출 파일: 2.c

(실행 예)

4↵

4*1=4

4*2=8

4*3=12

4*4=16

4*5=20

4*6=24

4*7=28

4*8=32

4*9=36

과제 6-2

- 3번 문항
 - 정수를 하나 입력 받음
 - 실행 예와 같이 별(*) 출력
 - 반드시 중첩 반복문 사용
- 제출 파일: 3.c

(실행 예 1)

5↵

```
*  
* *  
* * *  
* * * *  
* * * * *
```

(실행 예 2)

3↵

```
*  
* *  
* * *
```

수고하셨습니다.