

# CT MANUAL



# User Manual

2023.12 — Last update: Dec 12, 2023

Suresofttech

# Table of Contents

1. 시작하기 전에 .....	8
2. 개요 .....	9
3. 설치하기 .....	11
4. 제거하기 .....	15
5. 실행 .....	17
6. 라이선스 설정 .....	19
7. 툴체인(분석기) 설정 .....	21
7.1. 툴체인 자동 등록 .....	23
7.2. 툴체인 추가 .....	24
7.3. 툴체인 편집 .....	28
7.4. 툴체인 복제 .....	29
7.5. 툴체인 삭제 .....	30
7.6. 툴체인 내보내기 .....	31
7.7. 툴체인 가져오기 .....	33
8. 프로젝트 만들기 .....	34
8.1. 소스 파일 C/C++ 프로젝트 .....	35
8.2. 비주얼 스튜디오 프로젝트 .....	37
8.3. 기존 CodeScroll 프로젝트 정보로 C/C++ 프로젝트 생성 .....	39
8.4. 빌드 정보로부터 C/C++ 프로젝트 생성 .....	40
9. 테스트 생성 .....	42
10. 테스트 편집기 .....	45
10.1. 테스트 정보 .....	46
10.2. 테스트 케이스 .....	48
10.3. 테스트 코드 .....	49
10.4. 설정 .....	50
10.5. 테스트 매크로 .....	51
10.6. 시나리오 테스트 .....	54
11. 테스트 실행 .....	58
12. 디버그 정보 확인 .....	60
13. 함수 호출 정보 출력 .....	62
14. 결함 위치 찾기 .....	65
15. 특성 페이지 .....	67
15.1. 프로젝트 특성 .....	68
15.2. 모듈 특성 .....	78
15.3. 소스 파일 특성 .....	80

<b>16. 환경설정 .....</b>	<b>82</b>
16.1. 분석.....	83
16.2. 분석 제외 대상.....	89
16.3. 성능.....	90
16.4. 소스 파일 확장자 설정 .....	91
16.5. 언어.....	92
16.6. 테마.....	93
16.7. 테스트 .....	94
16.8. 툴체인 설정 .....	100
16.9. 편집기 .....	101
<b>17. 테스트 퍼스펙티브 .....</b>	<b>102</b>
17.1. 대시보드.....	104
17.2. 테스트 네비게이터 뷰.....	107
17.2.1. 프로젝트 커버리지 합치기.....	110
17.3. 유닛 테스트 뷰 .....	112
17.4. 통합 테스트 뷰.....	129
17.5. 커버리지 뷰 .....	136
17.6. MC/DC 뷰.....	139
17.7. 스텝 뷰 .....	141
17.8. 클래스 팩토리 뷰.....	153
17.8.1. 클래스 코드 사용하기 .....	155
17.8.2. 모의 객체 사용하기.....	159
17.9. 제어 흐름 그래프 뷰 .....	162
17.10. 함수 호출 그래프 뷰 .....	164
17.11. 오류 뷰 .....	167
17.12. 디버그 정보 뷰 .....	169
17.13. 결함 주입 뷰 .....	173
17.14. 입출력 데이터 그래프 뷰 .....	179
17.15. 콘솔 뷰 .....	181
17.16. 요구사항 뷰 .....	183
<b>18. 분석 퍼스펙티브.....</b>	<b>189</b>
18.1. 메트릭 뷰.....	190
18.2. 메트릭 차트 뷰 .....	193
18.3. 메트릭 탭 차트 뷰.....	195
18.4. 메트릭 바 차트 뷰.....	196
18.5. 메트릭 진단 차트 뷰 .....	198
18.6. 미사용 함수 뷰.....	199
18.7. 소스-헤더 연관 뷰 .....	200
18.8. 전역 변수 연관 뷰.....	202
18.9. 함수 호출 계층 뷰.....	203
18.10. 소스 코드 편집기.....	205
<b>19. 파일 메뉴 .....</b>	<b>206</b>
19.1. 모듈 생성.....	207



19.2. 워크스페이스 바꾸기 .....	208
19.3. 가져오기 .....	209
19.3.1. 요구사항 - CSV 파일에서 가져오기 .....	211
19.3.2. 요구사항 - Polarion에서 가져오기 .....	213
19.3.3. 요구사항 - V-SPICE에서 가져오기 .....	216
19.3.4. 일반 - 파일 시스템 .....	219
19.3.5. 일반 - 프로젝트 가져오기 .....	222
19.3.6. 커버리지 - 커버리지 가져오기 .....	225
19.3.7. 테스트 - 테스트 가져오기 .....	227
19.3.8. 테스트 - 테스트 코드 파일로부터 테스트 가져오기 .....	231
19.3.9. 환경 설정 - 가상 메모리 주소 .....	233
19.3.10. 환경 설정 - 톨체인 .....	235
19.4. 내보내기 .....	237
19.4.1. 메트릭 - 메트릭 내보내기 .....	239
19.4.2. 요구사항 - Polarion에 요구사항과 연결된 테스트 결과를 내보내기 .....	241
19.4.3. 요구사항 - V-SPICE에 테스트 결과를 내보내기 .....	243
19.4.4. 요구사항 - 요구사항 테스트 커버리지 보고서 .....	245
19.4.5. 일반 - 프로젝트 내보내기 .....	247
19.4.6. 커버리지 - 커버리지 내보내기 .....	249
19.4.7. 테스트 - 테스트 내보내기 .....	252
19.4.8. 테스트 - 테스트 보고서 생성 .....	255
19.4.9. 환경 설정 - 가상 메모리 주소 .....	258
19.4.10. 환경 설정 - 톨체인 .....	260
<b>20. 편집 메뉴 .....</b>	<b>261</b>
<b>21. 검색 메뉴 .....</b>	<b>262</b>
<b>22. 프로젝트 메뉴 .....</b>	<b>264</b>
<b>23. 창 메뉴 .....</b>	<b>266</b>
<b>24. 도움말 메뉴 .....</b>	<b>267</b>
<b>25. 문제 해결 .....</b>	<b>272</b>
<b>26. CT Target Plug-in .....</b>	<b>274</b>
26.1. 타겟 환경 설정하기 .....	275
26.2. C/C++ 타겟 테스트 프로젝트 .....	280
26.3. 타겟 테스트 실행 설정하기 .....	285
26.4. 타겟 로그 수집기 .....	286
26.5. 환경 설정 .....	288
<b>27. CT RTV(Remote Target Verifier) .....</b>	<b>289</b>
27.1. RTV 서버 설정 .....	290
27.2. RTV 톨체인 설정 .....	294
27.3. RTV 프로젝트 만들기 .....	297
27.4. RTV 소스 파일 새로 고치기/추가하기 .....	302
27.5. RTV 프로젝트 정보 .....	306

27.6. RTV 테스트 실행 .....	308
27.7. RTV 테스트 결과 확인 .....	310
27.8. 테스트 컴파일 .....	311
<b>28. CT RTV(Remote Target Verifier) Target Plug-in .....</b>	<b>314</b>
28.1. RTV 서버 설정 .....	315
28.2. RTV 툴체인 설정 .....	316
28.3. RTV 타겟 환경 설정 .....	317
28.4. RTV 타겟 프로젝트 만들기 .....	320
28.5. RTV 타겟 소스 파일 새로 고치기/추가하기 .....	326
28.6. RTV 타겟 프로젝트 정보 .....	327
28.7. RTV 타겟 테스트 실행 .....	329
<b>29. DISCOVERY Plug-in .....</b>	<b>330</b>
<b>30. CT CLI(Command Line Interface).....</b>	<b>332</b>
30.1. 프로젝트 생성 및 테스트 실행 .....	333
30.2. 프로젝트 가져오기 .....	339
30.3. 프로젝트 내보내기 .....	340
30.4. 분석 정보 및 VPES 데이터 내보내기 .....	341
30.5. 프로젝트 삭제 .....	343
30.6. DISCOVERY 실행하기 .....	344
30.7. 결합 위치 찾기 .....	345
30.8. 팀 프로젝트 가져오기 .....	346
<b>31. CT 팀 테스트 서버 plug-in .....</b>	<b>347</b>
31.1. CT 팀 테스트 서버 설치하기 .....	349
31.2. CT 팀 테스트 서버 실행하기 .....	354
31.3. 환경설정 .....	356
31.4. 팀 프로젝트 .....	359
31.4.1. 팀 프로젝트 생성하기 .....	360
31.4.2. 팀 프로젝트 특성 .....	362
31.4.3. 팀 프로젝트 가져오기 .....	365
31.4.4. 팀 프로젝트 변환하기 .....	368
31.5. 커밋 (Commit) .....	371
31.6. 업데이트 (Update) .....	373
31.7. 팀 테스트 .....	376
31.7.1. 팀 테스트 가져오기 .....	377
31.7.2. 팀 테스트 삭제 .....	380
<b>32. CT 팀 테스트 대시보드 .....</b>	<b>381</b>
32.1. 메인 페이지 .....	382
32.2. 프로젝트 페이지 .....	385
32.2.1. 개요 .....	387
32.2.2. 히스토리 .....	390
32.2.3. 사용자 목록 .....	393
32.2.4. 설정 .....	394

33. EULA(End-User License Agreement) ..... 395

# 1. 시작하기 전에

---

## 문서의 목적

CT 2023.12 제품의 사용법에 관한 정보를 제공합니다.



본 문서에 포함된 일부 이미지는 버전에 따라 다소 차이가 있을 수 있으나, 설명하는 기능에 해당하는 이미지는 문서 정보에 명시된 버전과 동일합니다.

## 2. 개요

---

### CT 2023.12 소개

다양한 환경에서 개발 및 수행되는 소프트웨어에 대한 단위 및 통합 테스트를 수행하는 테스트 자동화 솔루션입니다.

### CT 2023.12 지원 툴체인

CT 2023.12는 약 100종의 툴체인을 지원합니다. 자세한 내용은 본 문서 [문제 해결](#) 페이지 하단에 있는 기술 지원 연락처를 통해 문의해주시기 바랍니다.

### CT 2023.12 특징

CT 2023.12는 코드 기반 소프트웨어 단위/통합 테스트 자동화 도구로서 주요 특징은 다음과 같습니다.

#### 테스트 설계

1. 자동으로 테스트 및 테스트 데이터를 생성
2. 직관적인 테스트 설계 인터페이스
3. 다양한 형식의 테스트 데이터 가져오기 기능
4. 복잡하게 중첩된 구조의 통합 테스트 설계 가능
5. 주기 함수에 대한 시나리오 테스트 설계 가능

#### 커버리지 목표 달성

1. 구문, 분기, MC/DC, 함수 호출, 함수 커버리지 지원
2. MC/DC 목표 달성을 위한 가이드 제공
3. 소스 코드와 제어 흐름 그래프를 연동하여 커버리지 결과 확인
4. 시스템 커버리지 측정(QualityScroll COVER 제품) 후 커버되지 않은 부분만 테스트하여 빠르게 커버리지 목표 달성 가능

#### 기타

1. 시뮬레이션 및 실제 타겟 환경 테스트 지원
2. DevOps(이슈 관리 + 지속 통합 + 형상 관리)도구와 연동
3. CodeScroll 제품군과 프로젝트 설정 공유: 한 번의 프로젝트 설정으로 코딩 규칙 검사 + 실행 시간 오류 검출 + 단위/통합 테스트 가능
4. 다양한 형식의 보고서
5. 요구사항 관리 도구와 연동하여 요구사항 추적 기능 제공
6. 여러 PC에서 동시에 테스트를 진행할 수 있는 팀 테스트 기능 제공
7. Jenkins 플러그인과 연동하여 테스트 자동화할 수 있는 기능 제공

#### 표준 인증

1. IEC 61508-3 (전기/전자)

2. ISO 26262-8 (자동차)
3. IEC 60880 (원자력)
4. IEC 62279 / EN 50128 (철도)
5. DO-178C / DO-330 (항공)
6. IEC 62304 (의료 기기)

## CT 2023.12의 전체 화면

Eclipse RCP로 제작되어 자유도가 높은 창들을 제공합니다. [창] 메뉴와 각 뷰에서 사용자가 설정할 수 있습니다.

The screenshot displays the Eclipse RCP IDE interface for CT 2023.12. The interface is divided into several main areas:

- Source Code Editor (소스 코드 편집기 영역):** Located on the left, it shows the source code for the `tr_flush_bits` function in `tr_flush_block`. The code is written in C and includes comments and macros.
- Test View (테스트 뷰 영역):** Located on the right, it displays a table of test results. The table has columns for '이름' (Name), '커버리지' (Coverage), and '테스트 결과' (Test Result). The results show various test cases with their respective coverage percentages and counts.
- Test Editor (테스트 편집기 영역):** Located at the bottom right, it provides a detailed view of the test results, including a table for '테스트 구조' (Test Structure) and '테스트 정보' (Test Information).

Overlaid text labels identify these areas: '대시보드 영역' (Dashboard Area) at the top, '소스 코드 편집기 영역' (Source Code Editor Area) on the left, '테스트 뷰 영역' (Test View Area) on the right, and '테스트 편집기 영역' (Test Editor Area) at the bottom right.

## 3. 설치하기


### 설치 요구 사항

OS	Microsoft Windows 7/10/11 (64bit)
RAM	최소 4GB 이상
저장 공간	약 10GB(GCC Compiler 설치)의 여유 공간 (테스트가 수행되면 테스트 결과물로 인해 저장공간은 더 늘어납니다.)

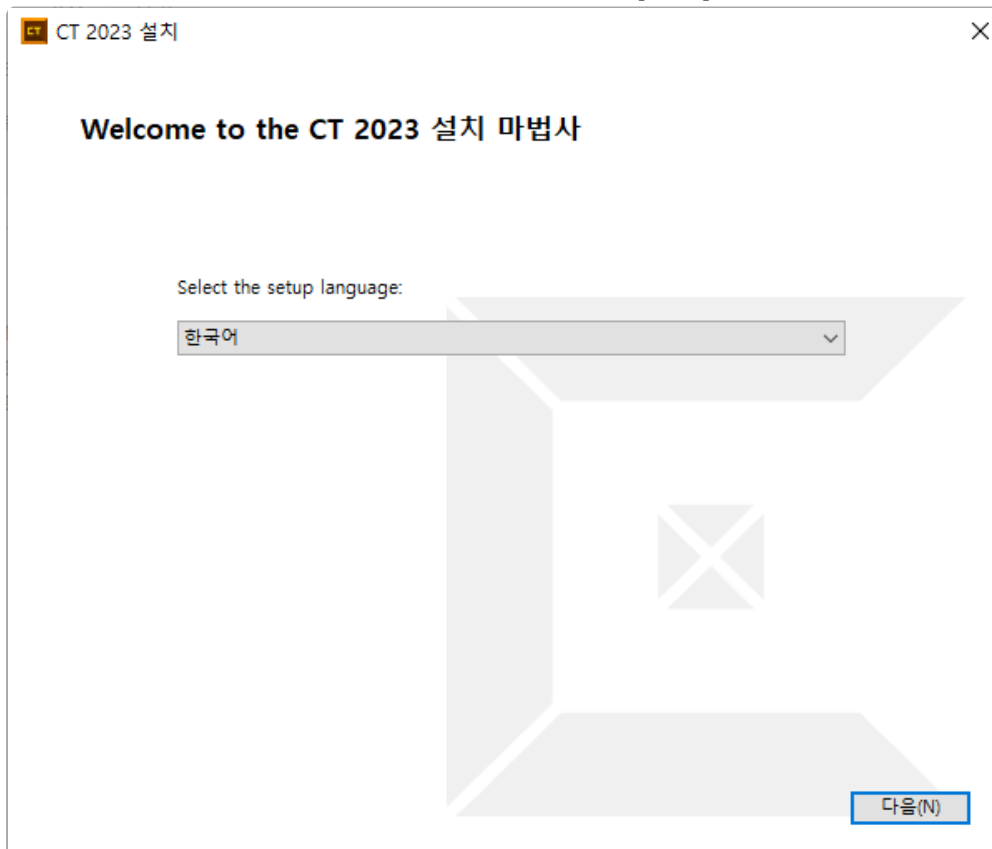
### CT 2023.12 설치하기

인스톨 패키지를 이용하여 CT 2023.12를 설치하는 방법은 다음과 같습니다.

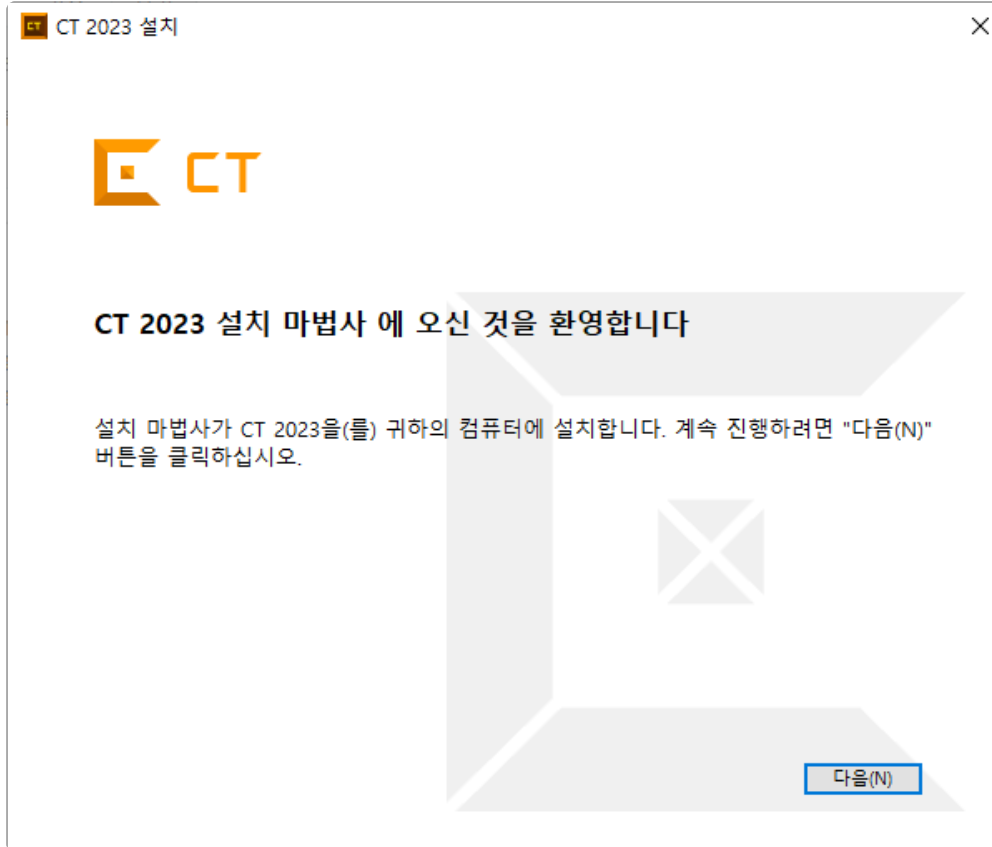
1. ct-xxxx-setup.exe 파일을 실행합니다.

이름	수정한 날짜	유형	크기
 ct-2023.12-setup	2023-12-08 오후 1:46	응용 프로그램	1,183,246...

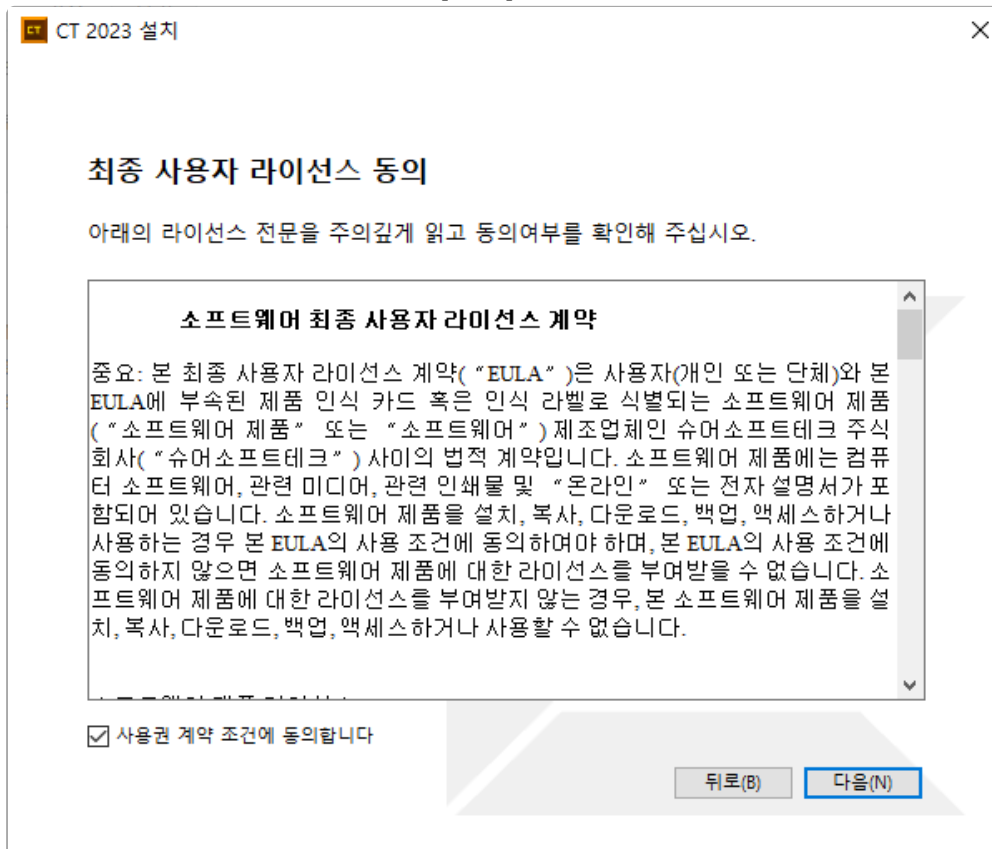
2. 설치 마법사가 실행되고 설치 언어를 선택합니다. [다음]을 클릭합니다.



3. 설치 정보를 수집합니다. [다음]을 클릭합니다.

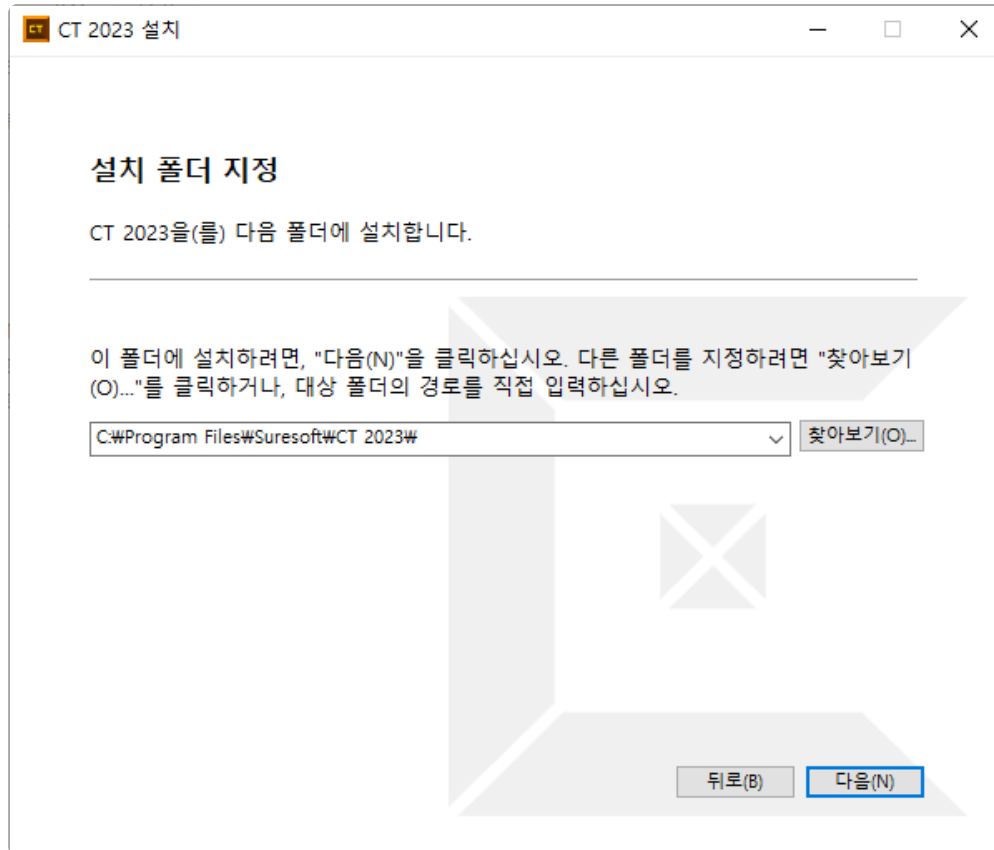


4. 최종 사용자 라이선스를 동의하고 [다음]을 클릭합니다.

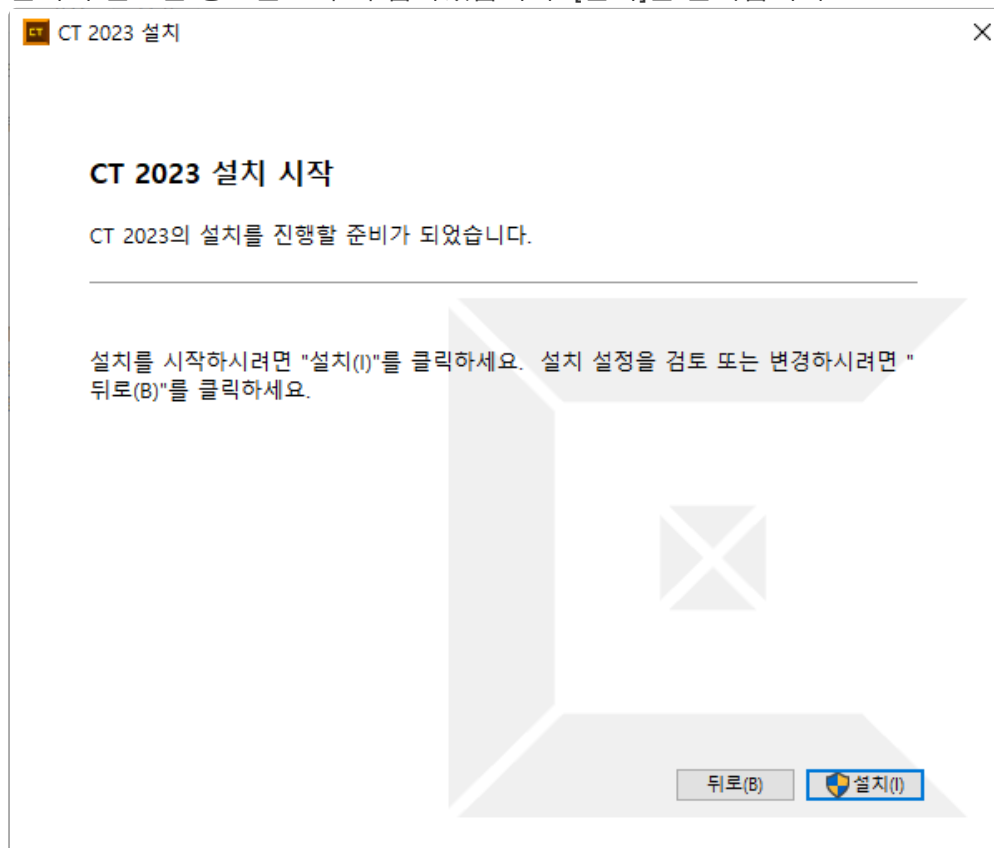


5. CT 2023.12를 설치할 경로를 설정하고 [다음]을 클릭합니다.

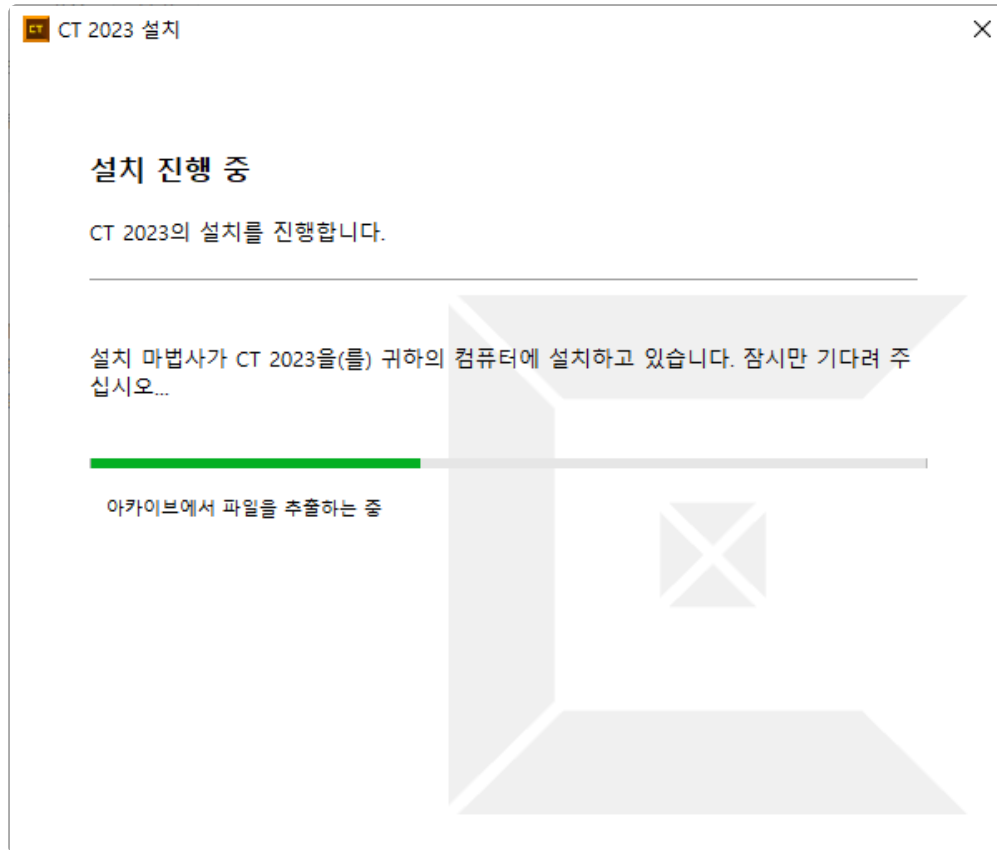




6. 설치에 필요한 정보를 모두 수집하였습니다. [설치]를 클릭합니다.



7. CT 2023.12가 설치됩니다.



8. 설치 완료 후 CT 2023.12를 바로 실행할 때는 [CT xxxx 실행]을 클릭하고 설치를 종료할 때는 [마침]을 클릭합니다.

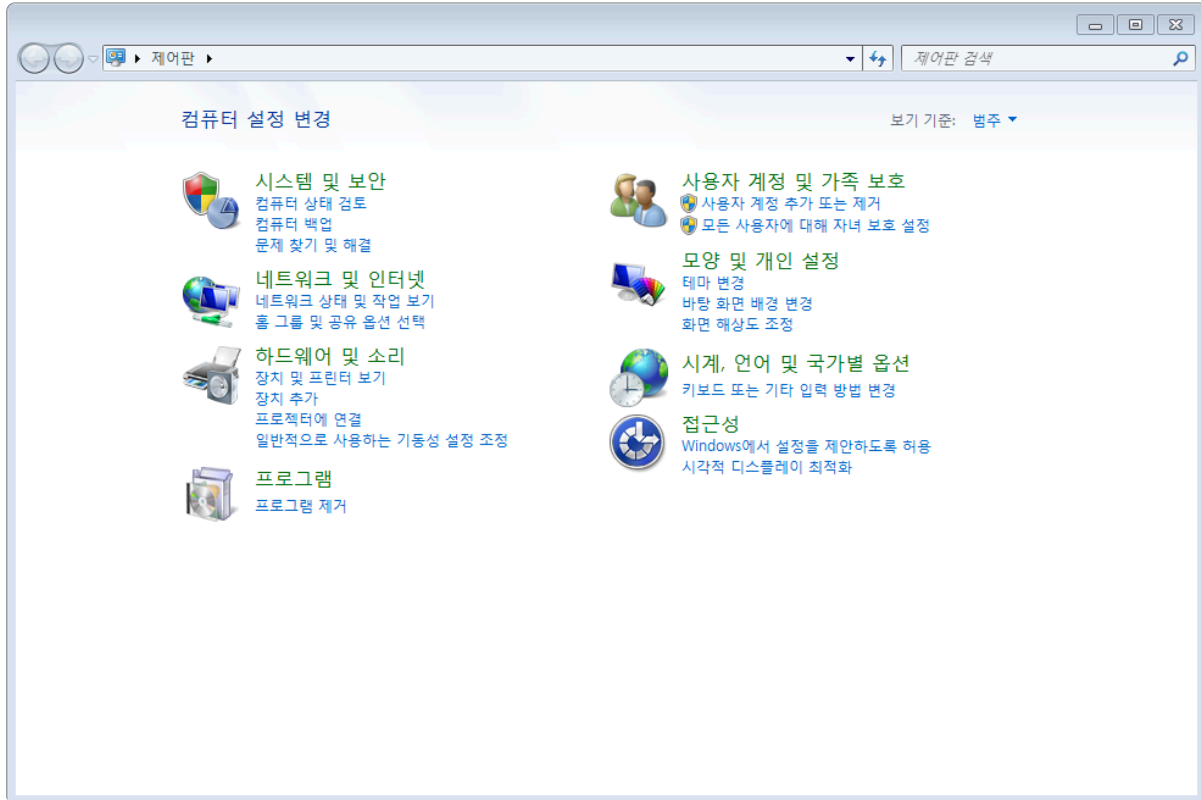


## 4. 제거하기

### CT 2023.12 제거하기

CT 2023.12를 제거하는 방법은 아래와 같습니다.

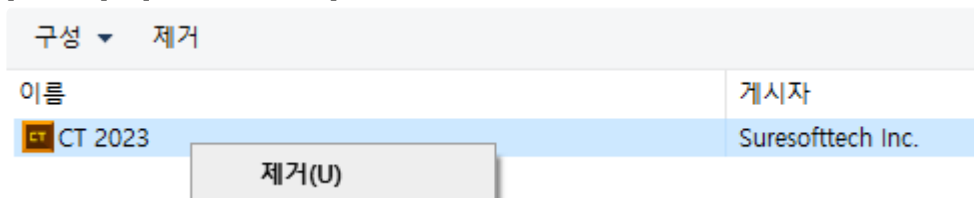
1. [제어판] -> [프로그램 제거]를 이용한 삭제



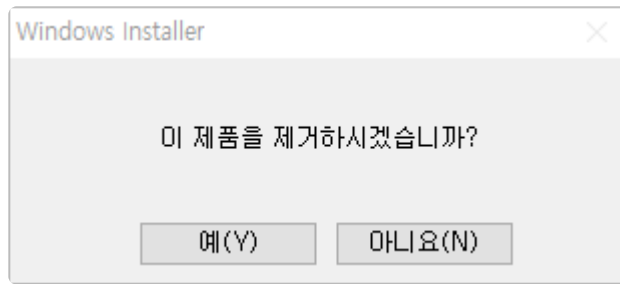
\* “CT 2023.12 제거하기”는 CT 2023.12의 동작에 필요한 파일들을 제거합니다. 사용자가 설정한 워크스페이스는 제거되지 않습니다.

### [프로그램 제거]를 이용한 삭제 과정

1. [제어판] – [프로그램 제거]를 선택합니다.



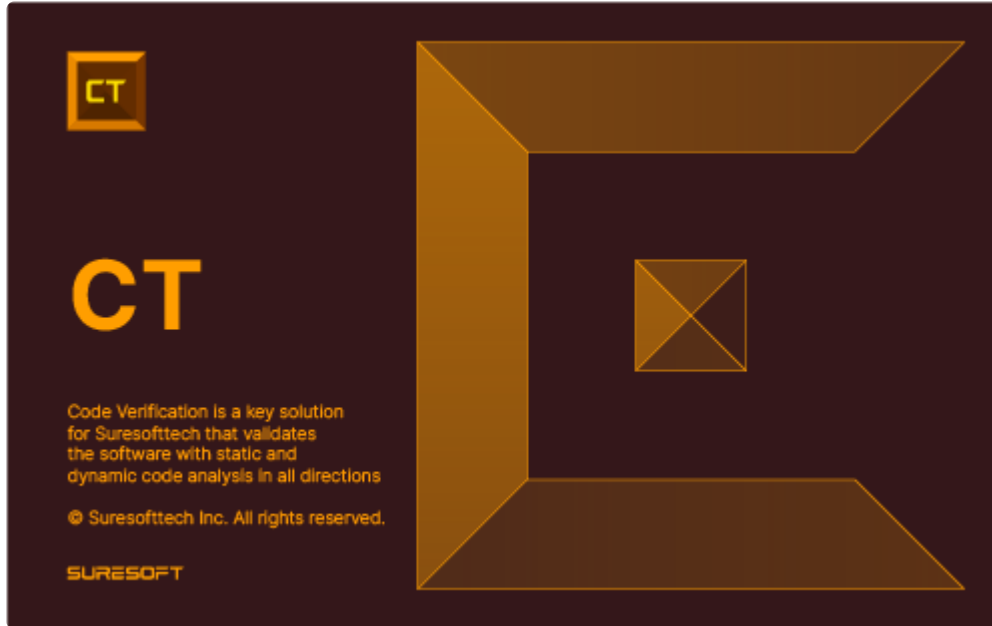
2. [예]를 클릭하여 CT 2023.12의 제거를 진행합니다.



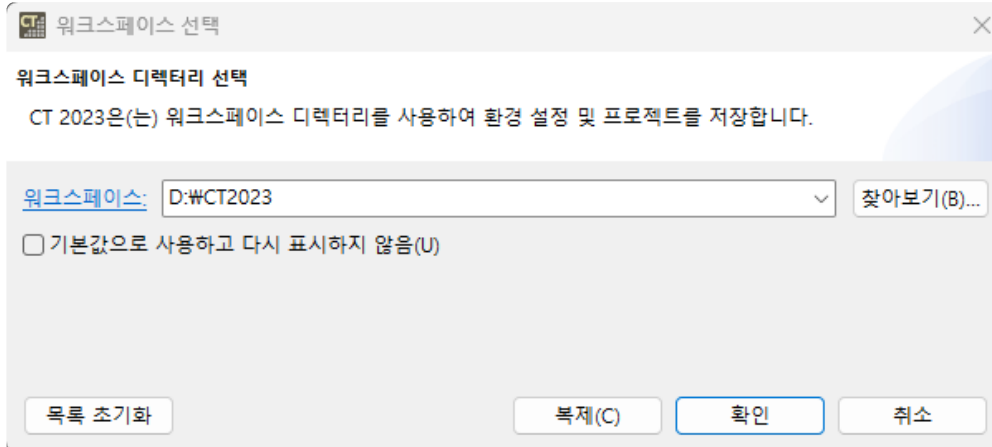
3. CT 2023.12의 제거에 필요한 정보를 수집하여 제거합니다. 제거가 완료되면 창이 닫힙니다.

## 5. 실행

바탕 화면의 단축 아이콘 혹은 [시작] -> [모든 프로그램] -> [CT xxxx]을 선택합니다.

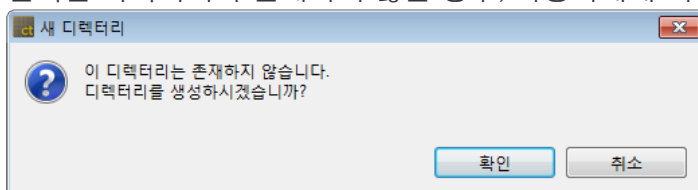


CT 2023.12의 스플래시 화면에 이어 워크스페이스(작업 공간)를 선택합니다. 워크스페이스 경로는 이미 존재하는 디렉터리를 선택하거나 새로운 디렉터리를 입력할 수 있습니다. 다음에 실행할 때 워크스페이스를 다시 선택하지 않도록 지금 선택한 워크스페이스를 기본값으로 사용할 수도 있습니다. 선택을 마치면 [확인] 버튼을 클릭합니다.

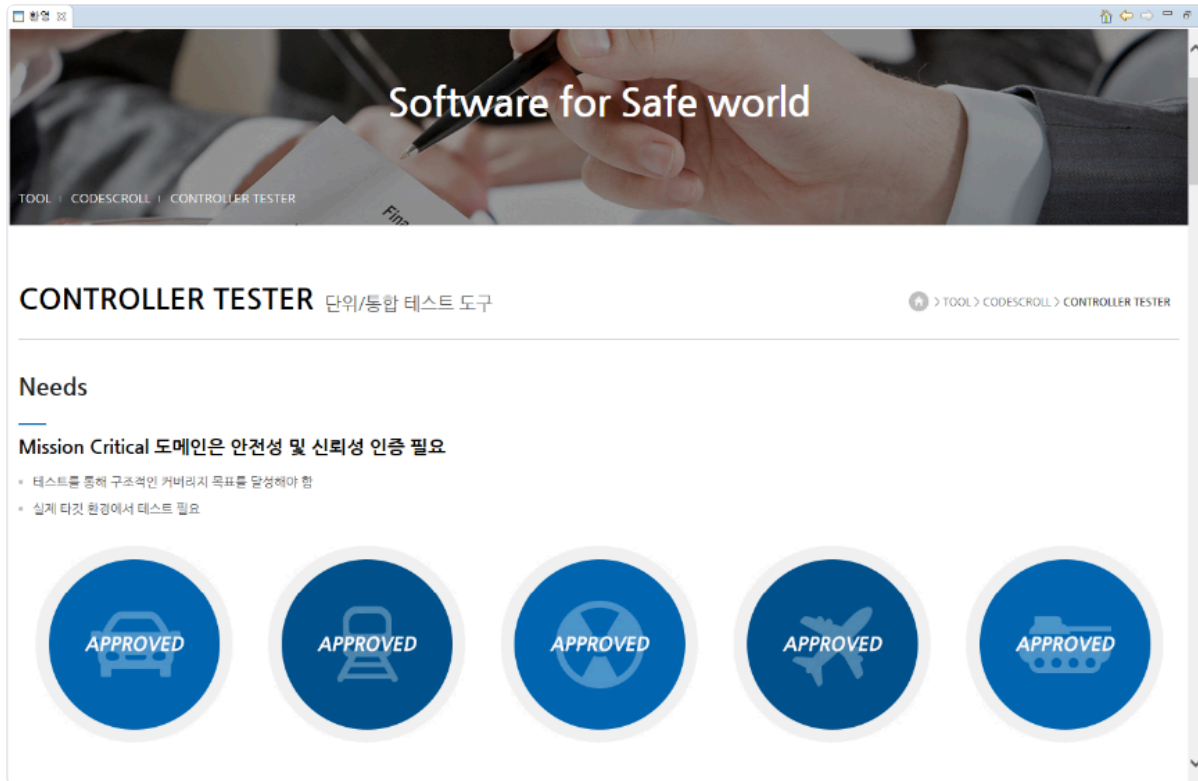


- [목록 초기화]를 선택하여 워크스페이스 목록을 삭제합니다.

선택한 디렉터리가 존재하지 않을 경우, 사용자에게 디렉터리 생성여부를 확인합니다.



설정된 워크스페이스에 디렉터리를 생성하고 도구의 환영 페이지와 프로젝트 생성 마법사를 보여 줍니다. 환영 페이지는 워크스페이스를 새롭게 생성했을 경우에만 보여주며, 이전에 워크스페이스로 설정된 경로를 다시 워크스페이스로 선택했을 경우에는 보이지 않습니다.

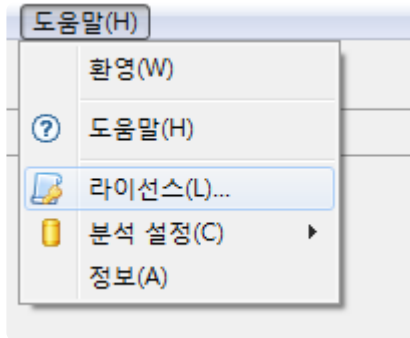


환영 페이지를 닫으면 제품 화면이 나타납니다.

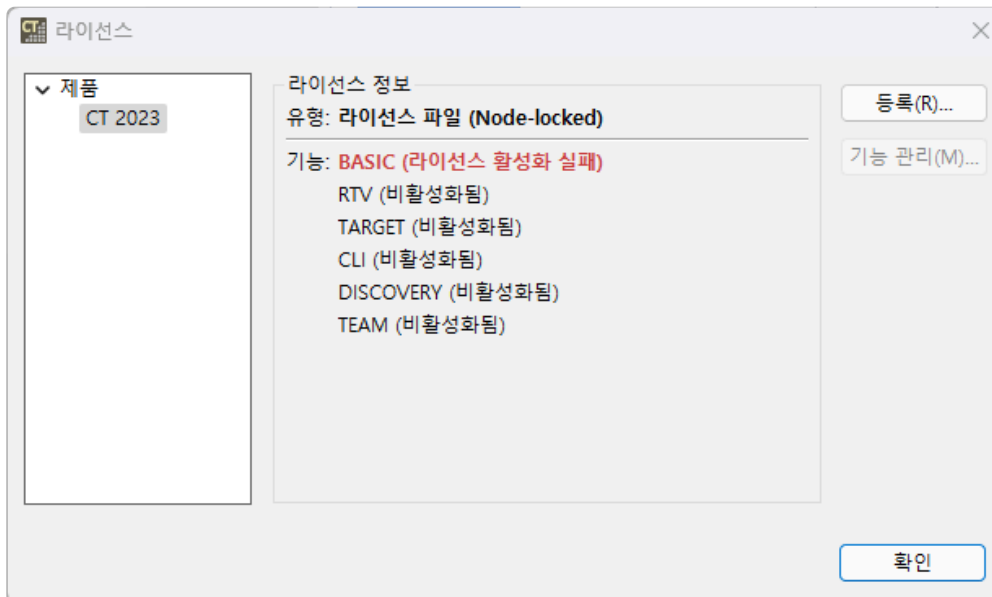
## 6. 라이선스 설정

CT 2023.12를 사용하려면 먼저 라이선스를 등록해야 합니다. 라이선스 등록 과정은 다음과 같습니다.

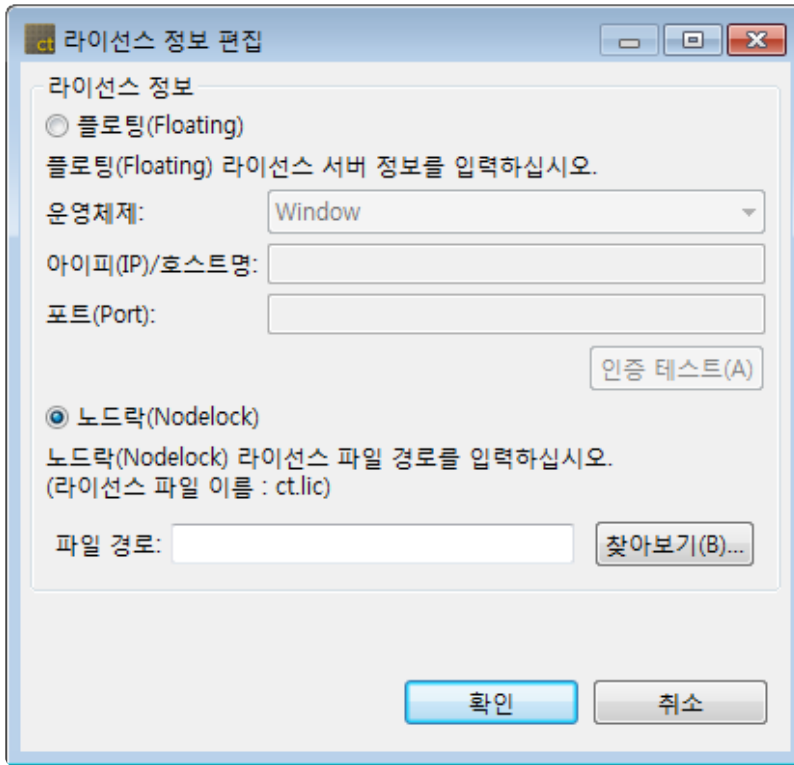
1. 메뉴의 [도움말] > [라이선스]를 선택합니다.



2. [제품] > [제품 이름]을 선택하고 [등록]을 클릭합니다.

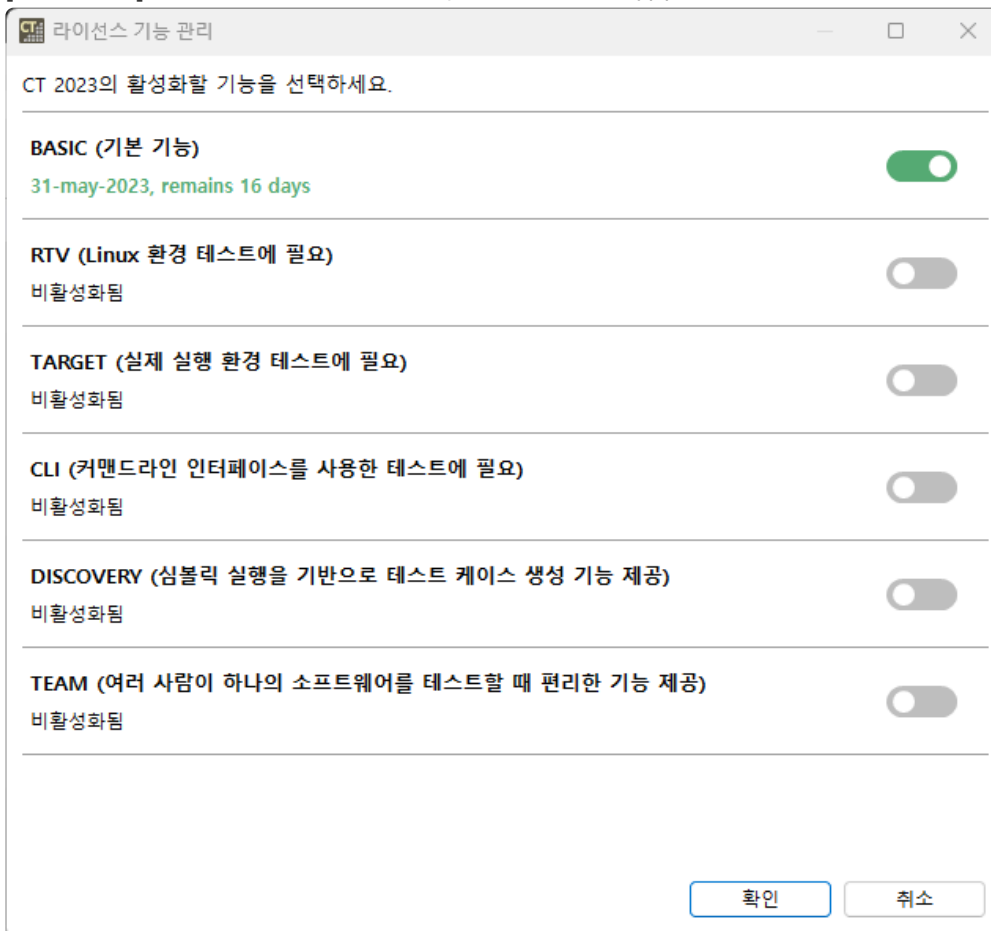


3. 제공받은 라이선스를 [라이선스 정보 편집]에 입력하고 [확인] 버튼을 클릭하여 라이선스를 등록합니다.



- 플로팅 방식: 라이선스 서버를 통해 등록합니다. 서버 정보(운영 체제, 아이피, 포트)를 입력하고, [인증 테스트] 버튼을 클릭합니다.
- 노드락 방식: 제공받은 라이선스 파일을 등록합니다.

4. [기능 관리] 버튼을 클릭하여 활성화할 기능을 선택합니다.



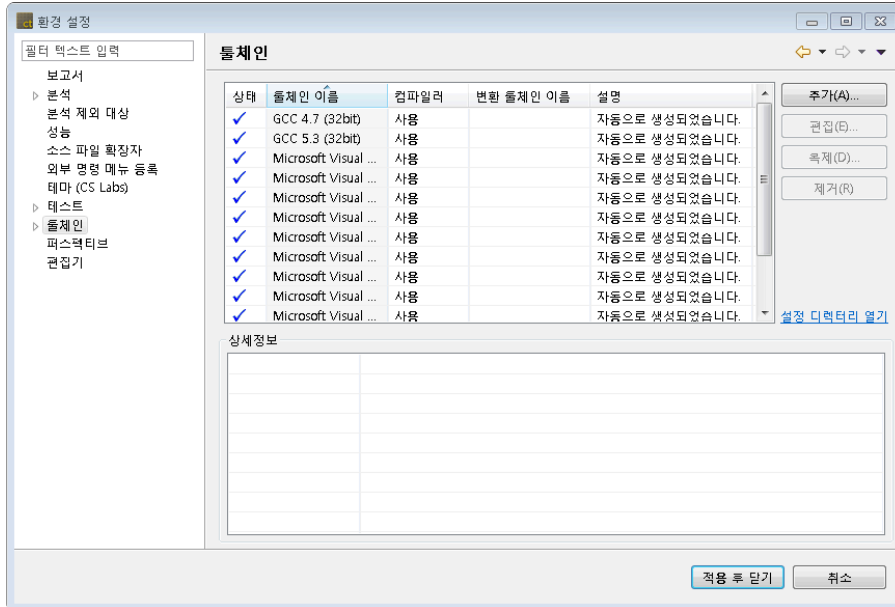


## 7. 툴체인(분석기) 설정

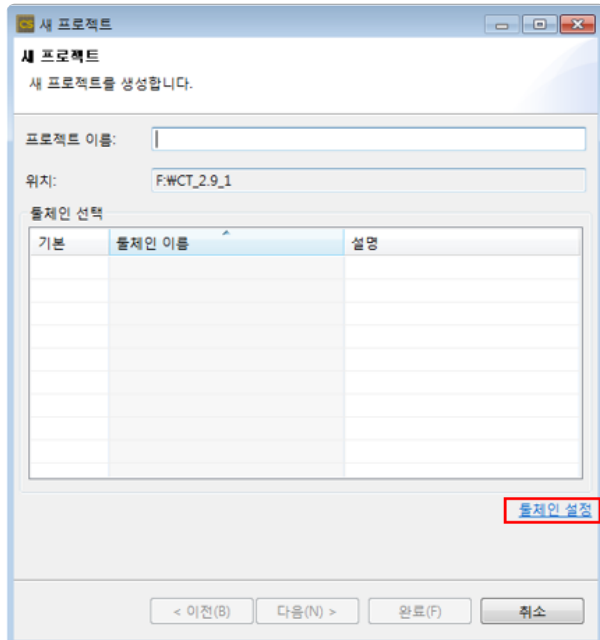
CT 2023.12를 사용하여 테스트를 수행하기 위해서는 툴체인 설정이 필요로 합니다.

테스트 대상 소스에 대한 툴체인(컴파일러 정보)이 있어야 프로젝트를 생성할 수 있습니다.

툴체인은 [창] -> [환경 설정] -> [툴체인]에서 설정할 수 있습니다.



또한 소스 파일 C/C++ 프로젝트 생성 마법사 또는 임베디드 C/C++ 프로젝트 생성 마법사 내의 [툴체인 설정]을 통해서도 설정할 수 있습니다.



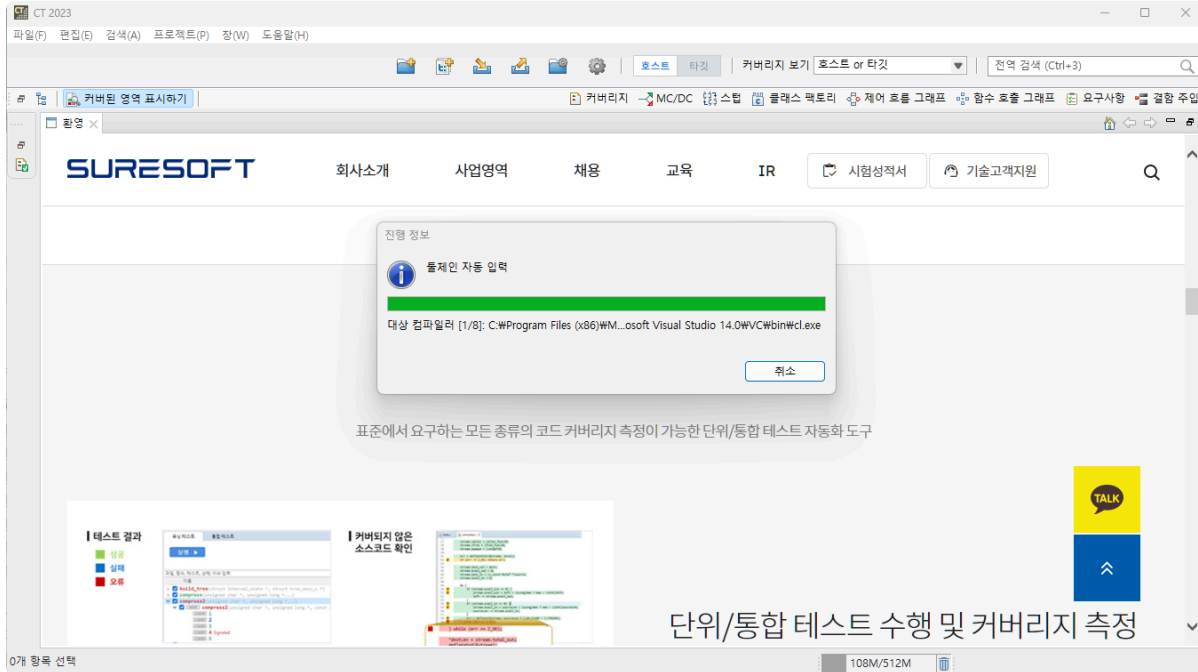
CT 2023.12가 제공하는 툴체인 설정 관련 기능은 다음과 같습니다.

- [툴체인 자동 등록](#)
- [툴체인 추가](#)
- [툴체인 편집](#)
- [툴체인 복제](#)

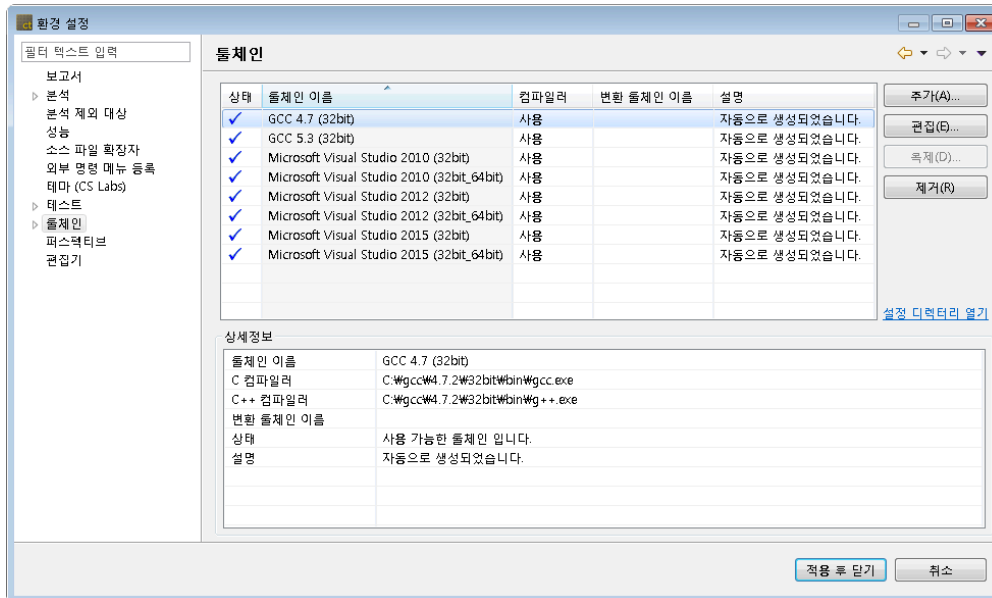
- [통체인 삭제](#)
- [통체인 내 보내기](#)
- [통체인 가져오기](#)

## 7.1. 툴체인 자동 등록

CT 2023.12 실행 시, 사용자 PC에 설치된 Visual Studio 컴파일러 정보와 CT 2023.12와 함께 설치된 GCC 컴파일러 정보를 추출하여 자동으로 툴체인을 등록합니다(등록된 툴체인은 다시 등록하지 않습니다).

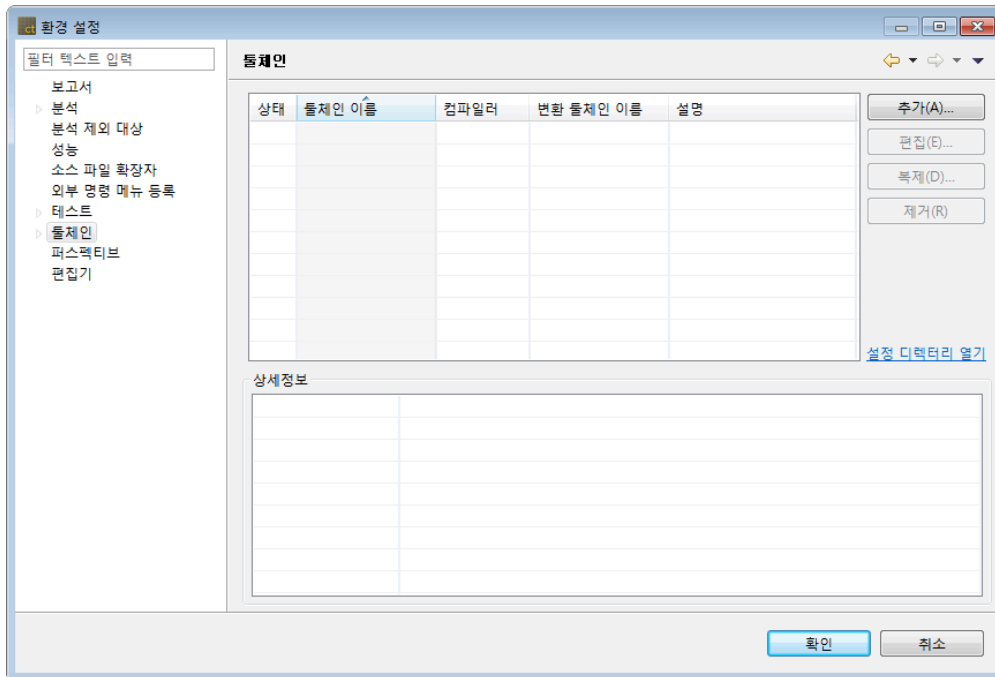


자동 등록된 툴체인은 [창] -> [환경 설정] -> [툴체인] 창에서 확인할 수 있으며, 설명에 “자동으로 생성되었습니다.” 라고 표시됩니다.



## 7.2. 툴체인 추가

1. [창] -> [환경 설정] -> [툴체인] 창에서 [추가] 버튼을 클릭합니다.



2. [툴체인 추가] 창에서 툴체인 정보를 입력합니다.

**툴체인 추가**

툴체인 정보를 설정합니다.

**툴체인 정보**

이름:

설명:

환경 정보:

분석 설정:

**C** **C++**

**1**

컴파일러:

링커:

아카이버:

시스템 헤더:

라이브러리:

입력한 컴파일러로부터 툴체인 정보를 자동으로 추출합니다.

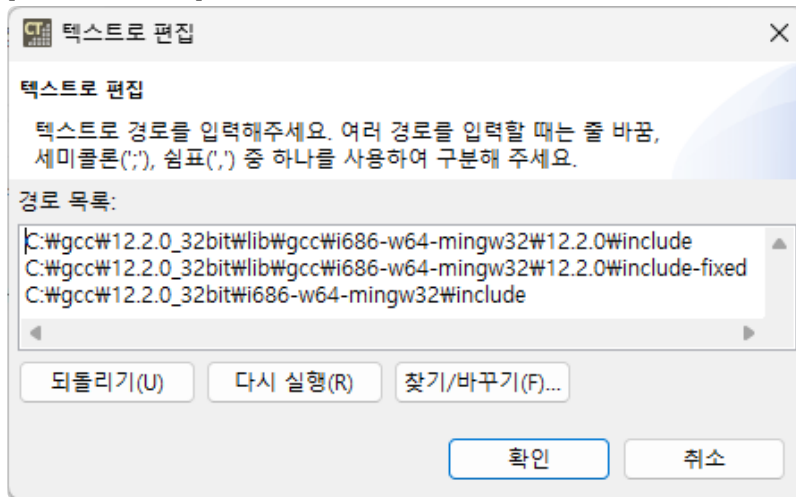
**2**

< 이전(B)    다음(N) >    완료(F)    취소

- [컴파일러] **1** 입력 후 [툴체인 정보 자동 추출] **2** 버튼을 클릭하면 입력한 컴파일러로부터 툴체인 정보를 자동으로 추출합니다.
- [컴파일러] **1** 란에는 컴파일러 경로를 찾아 입력합니다.
  - 예:
    - 비주얼 스튜디오 기반일 경우: C:\program Files (x86)\Microsoft Visual Studio 10.0\VS\bin\cl.exe
    - GCC 기반일 경우: C:\MinGW\bin\gcc.exe
    - Freescale HC(s)12용 CodeWarrior 5.x 이상인 경우: C:\Program Files (x86)\CodeWarrior\bin\cwc.exe

6) \Freescale\CWS12v5.1\Prog \chc12.exe

- [텍스트로 편집] 버튼을 선택하여 텍스트로 편집할 수 있습니다.



3. [이름]에 생성하려는 툴체인 이름을 입력합니다.
4. [설명]에 생성하려는 툴체인에 대한 설명을 입력합니다.
5. [환경 정보], [분석 설정]. [C/C++]의 입력정보들은 컴파일러로부터 정보를 추출하여 자동으로 설정되거나 사용자가 직접 입력할 수 있습니다.
6. 툴체인 정보 입력 후 [다음] 또는 [완료] 버튼을 클릭합니다.



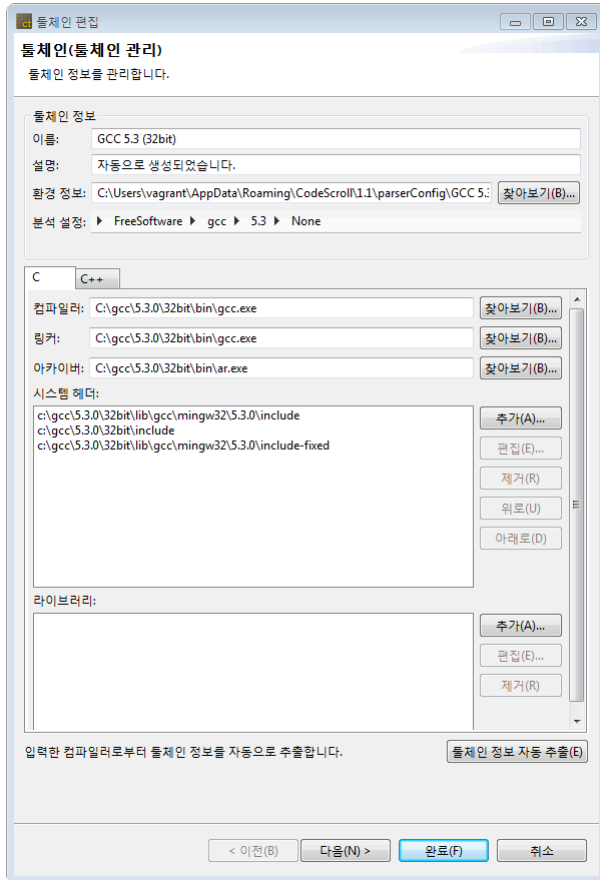
7. [다음] 버튼을 클릭하면 나오는 [통체인 세부사항 설정] 창에서는 [분석 설정]에서 선택한 분석 설정에 관련된 세부 항목들을 보여줍니다. 각 항목들은 사용자가 변경할 수 있습니다. 변경된 사항은 [미리보기]를 통해 어떻게 적용되는지 확인할 수 있습니다.



그 외 통체인 세부사항에 대한 상세 설정 방법은 본 문서 [문제 해결](#) 페이지 하단에 있는 기술지원 연락처를 통해 문의해주시기 바랍니다.

## 7.3. 툴체인 편집

1. 편집할 툴체인을 선택하고 [편집] 버튼을 클릭합니다.



2. 툴체인 정보를 편집합니다.
3. [완료] 버튼을 클릭합니다.



## 7.4. 톨체인 복제

---

1. 복제할 톨체인을 선택한 후 [복제] 버튼을 클릭합니다.
2. 변경할 톨체인 정보가 있으면 편집을 합니다.
3. [완료] 버튼을 클릭합니다.

## 7.5. 툴체인 삭제

---


삭제할 툴체인을 선택한 후 [제거] 버튼을 클릭합니다.

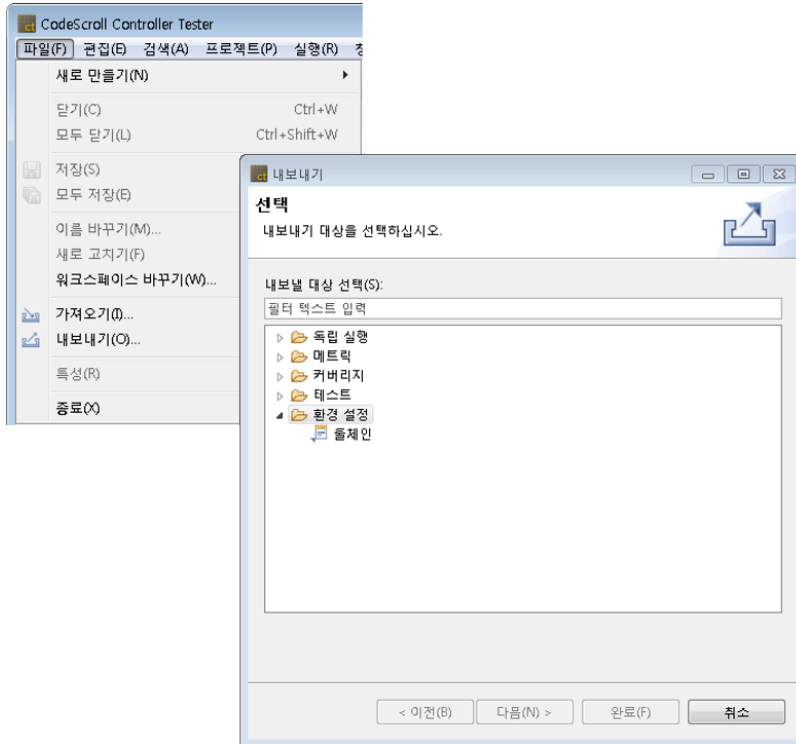


삭제하려는 툴체인을 사용하는 프로젝트가 있을 경우에, 해당 프로젝트가 정상적으로 동작하지 않게 됩니다.

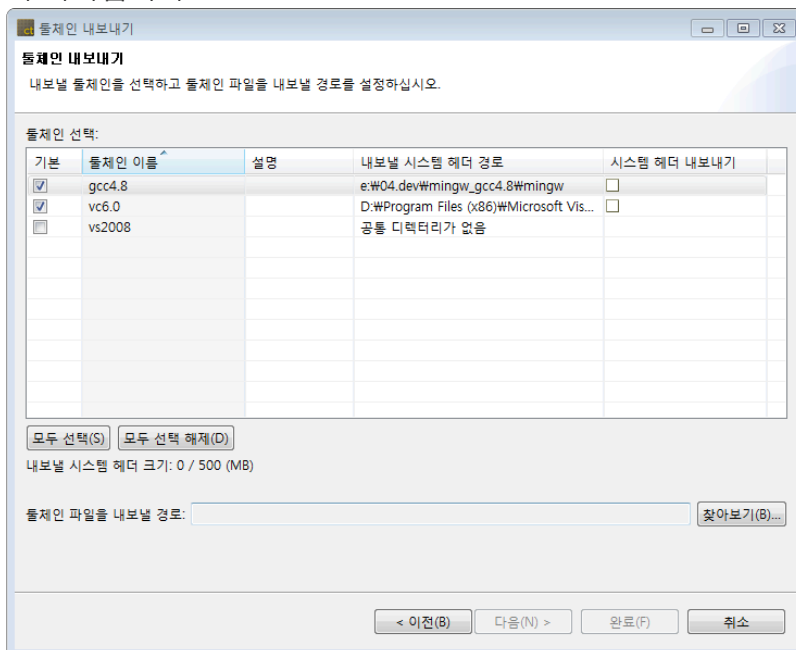
## 7.6. 툴체인 내보내기

‘내보내기’ 기능을 통해 툴체인 정보를 내보낼 수 있습니다.

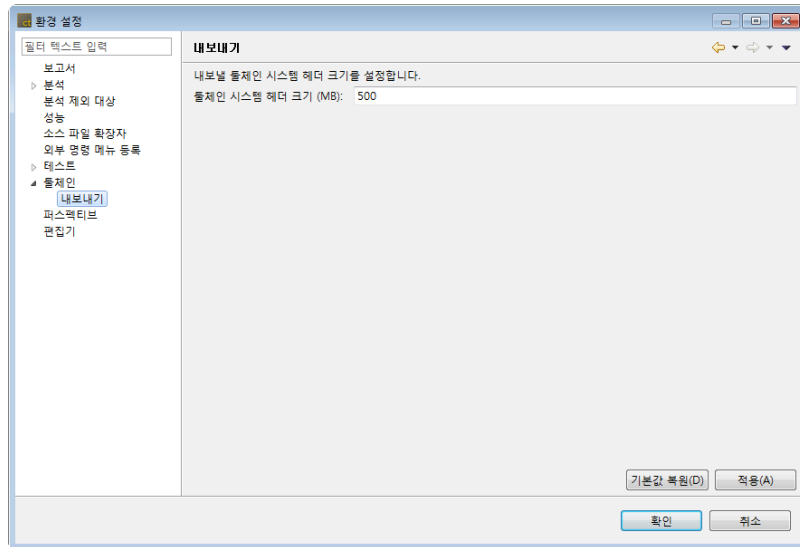
1. 메뉴의 [파일] > [내보내기] 또는 대시보드 영역의 [내보내기 ]를 클릭하여 [환경설정]의 [툴체인]을 선택합니다.



2. 내보낼 툴체인을 선택하고 [찾아보기] 버튼으로 내보낼 경로를 지정합니다. 툴체인 설정에서 시스템 헤더 파일을 설정하게 되면 [시스템 헤더 내보내기] 컬럼에 내보내기 여부를 선택할 수 있는 체크 박스가 나타납니다.




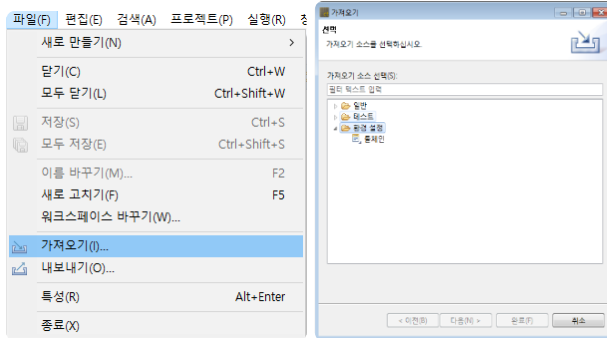
3. 메뉴의 [창] > [환경 설정] > [툴체인] > [내보내기]를 클릭하면 내보낼 시스템 헤더 크기를 설정할 수 있습니다. 시스템 헤더의 크기가 설정한 값보다 크면 시스템 헤더를 내보낼 수 없습니다.



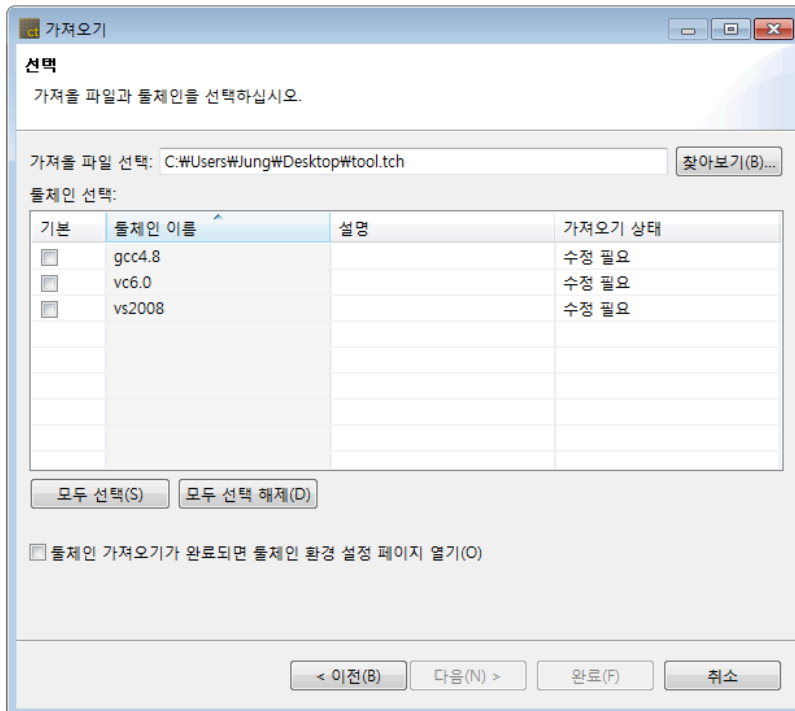
## 7.7. 툴체인 가져오기

‘가져오기’ 기능을 통해 내보낸 툴체인을 가져올 수 있습니다.

1. 메뉴의 [파일] > [가져오기] 또는 대시보드 영역의 [가져오기 ]를 선택하여 [환경 설정]의 [툴체인]을 클릭합니다.



2. 가져올 파일을 선택한 뒤에 툴체인을 선택합니다.



가져올 툴체인 이름이 기존 툴체인 이름과 중복될 경우, 가져오기 상태가 “수정 필요”가 됩니다. [다음] 버튼을 클릭하여 나오는 [문제해결] 대화상자에서 가져올 툴체인의 이름을 변경할 수 있습니다.

## 8. 프로젝트 만들기

### 프로젝트 만들기 전 확인 사항

CT 2023.12는 소스 코드를 실제 수행하여 오류를 검출하는 도구이므로, 테스트 대상 소스 코드를 수행 가능한 코드로 만들 수 있어야 합니다. 즉, 빌드가 가능한 소스 코드여야 정상적인 테스트를 진행할 수 있습니다.

프로젝트를 생성하여 테스트를 진행하기 전에 대상 소프트웨어의 개발 환경을 고려한 ‘툴체인’ 설정 과정이 필요합니다. 자세한 사항은 [툴체인 설정](#) 을 참고하시기 바랍니다.

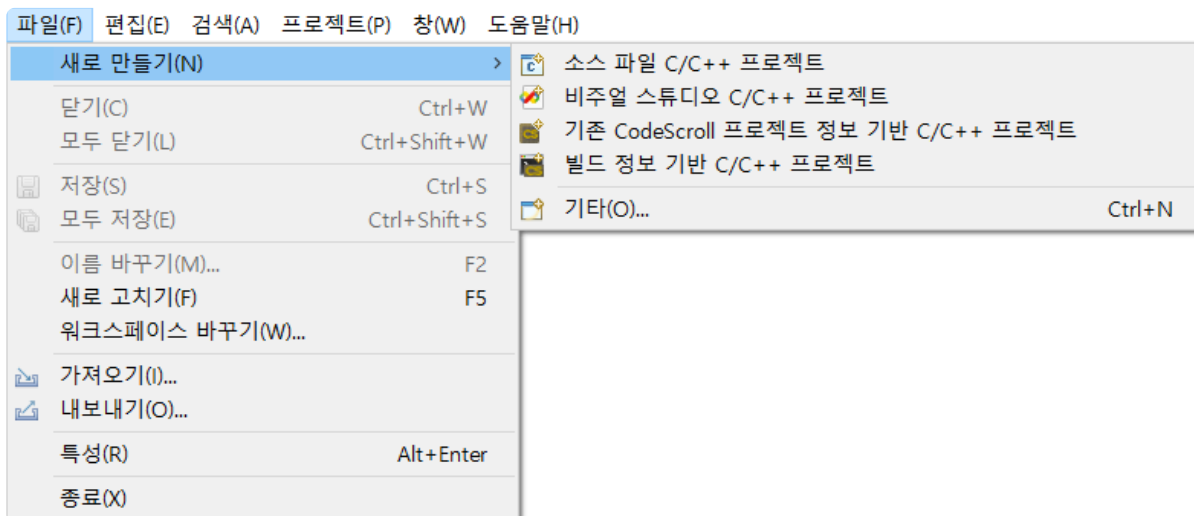
빌드 과정에서 링크할 대상 객체가 존재하지 않을 경우에는 CT 2023.12가 자동으로 스텝을 생성합니다. 하지만 자동 테스트 스텝 생성 기능은 테스트 수행에 제약 사항이 있을 수 있으므로, 적절한 테스트 전략을 사전에 준비하는 것이 좋습니다.

또한, 프로젝트의 이름은 100자를 넘을 수 없습니다.

### 프로젝트 생성

CT 2023.12 프로젝트를 생성하기 위해 [프로젝트 생성 마법사]를 실행합니다.

[파일] -> [새로 만들기] 또는 해당 단축 아이콘을 클릭합니다.



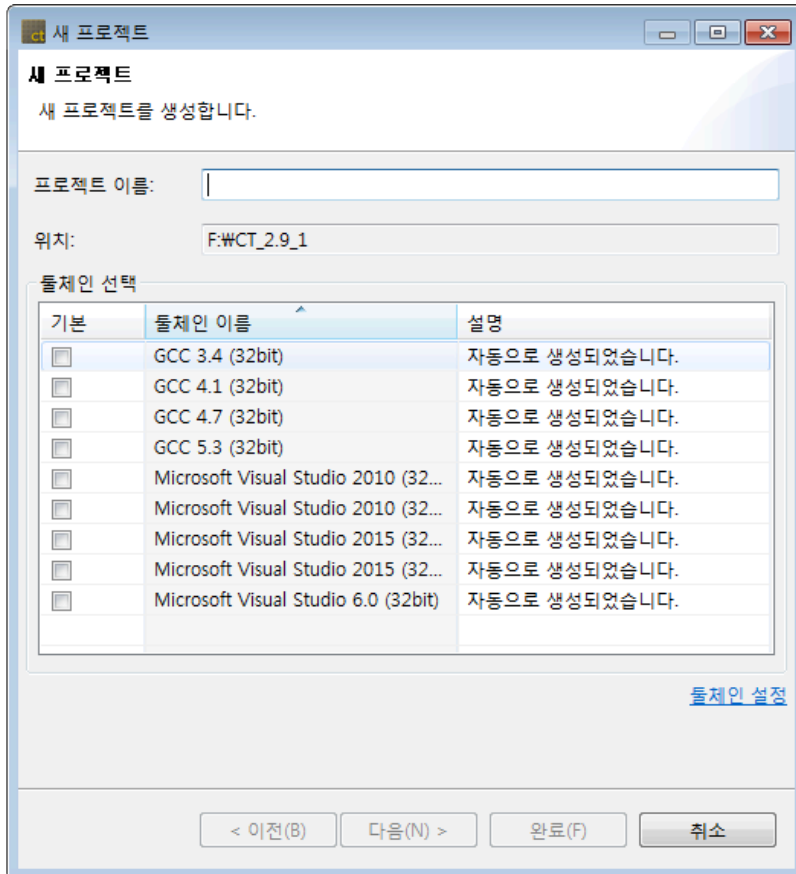
[프로젝트 생성 마법사]는 다음과 같은 종류의 프로젝트 생성을 제공합니다.

- [소스 파일 C/C++ 프로젝트](#)  
소스 파일을 직접 선택하여 프로젝트 생성
- [비주얼 스튜디오 프로젝트](#)  
비주얼 스튜디오 dsw, sln, vcxproj, vcproj 파일을 이용해서 프로젝트 생성
- [기존 CodeScroll 프로젝트 정보로 C/C++ 프로젝트 생성](#)  
기존에 생성된 CodeScroll 프로젝트를 이용해서 프로젝트 생성
- [빌드 정보로부터 C/C++ 프로젝트 생성](#)  
makefile 및 빌드 스크립트를 이용해서 프로젝트 생성

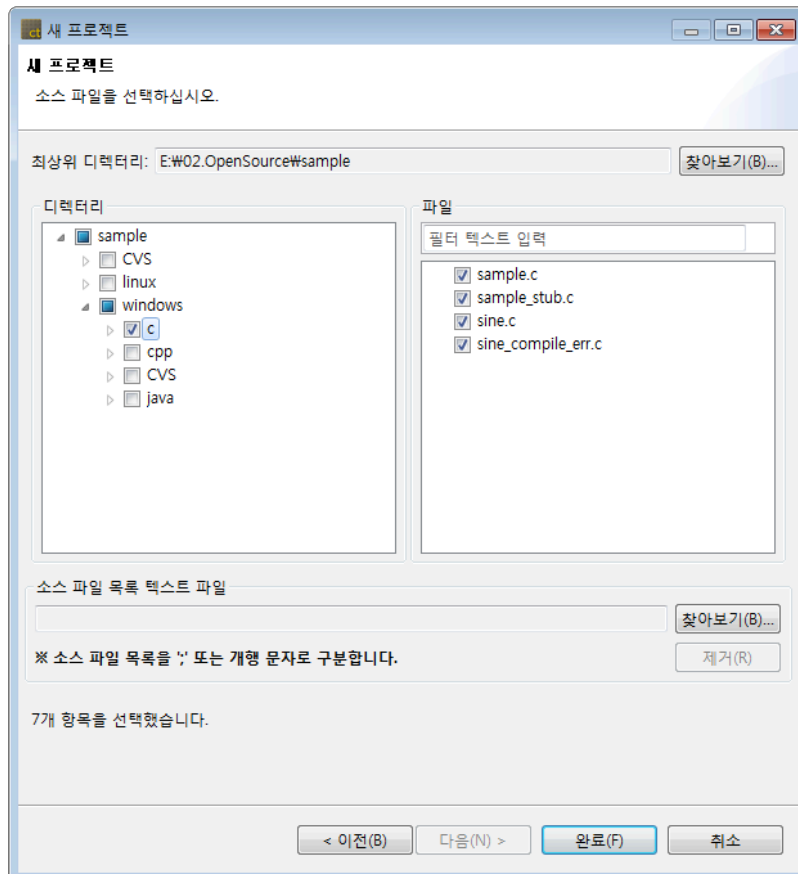
## 8.1. 소스 파일 C/C++ 프로젝트

C/C++ 소스 코드 파일만으로 프로젝트를 생성할 경우 선택합니다. 소스 코드 빌드에 필요한 정보를 사용자가 직접 입력해야 합니다.

1. [프로젝트 이름]에 프로젝트 이름을 입력합니다.



2. [툴체인 선택]에서 해당 프로젝트에서 사용할 툴체인을 선택합니다. 생성된 툴체인이 없다면 [툴체인 설정]을 통해 툴체인을 생성합니다. 자세한 사항은 [툴체인 설정](#) 을 참고하시기 바랍니다.
3. [다음] 버튼을 클릭하여 다음 화면으로 이동합니다. [완료] 버튼을 클릭 하였을 경우에는 소스 파일이 포함되지 않는 빈 프로젝트가 생성됩니다.



- 소스 파일을 직접 선택하거나, 파일 경로가 쓰인 텍스트 파일을 통해 선택할 수 있습니다.

#### 4. 소스 파일을 직접 선택

- [찾아보기...]를 클릭하여 아래 디렉터리 화면에 보일 [최상위 디렉터리]를 지정합니다.
  - 최상위 디렉터리는 특별한 경우를 제외하고, 선택하려는 소스 파일을 담고 있는 디렉터리보다 한 단계 높은 수준의 디렉터리를 지정합니다.
- 왼쪽에 보이는 [디렉터리] 화면에서 프로젝트 생성에 사용할 소스 파일이 있는 디렉터리를 선택합니다.
- 선택한 디렉터리에 있는 파일이 오른쪽에 보이는 [파일] 화면에 나타납니다. 추가할 파일을 체크합니다.

#### 5. 파일 경로가 쓰인 텍스트 파일을 통해 선택

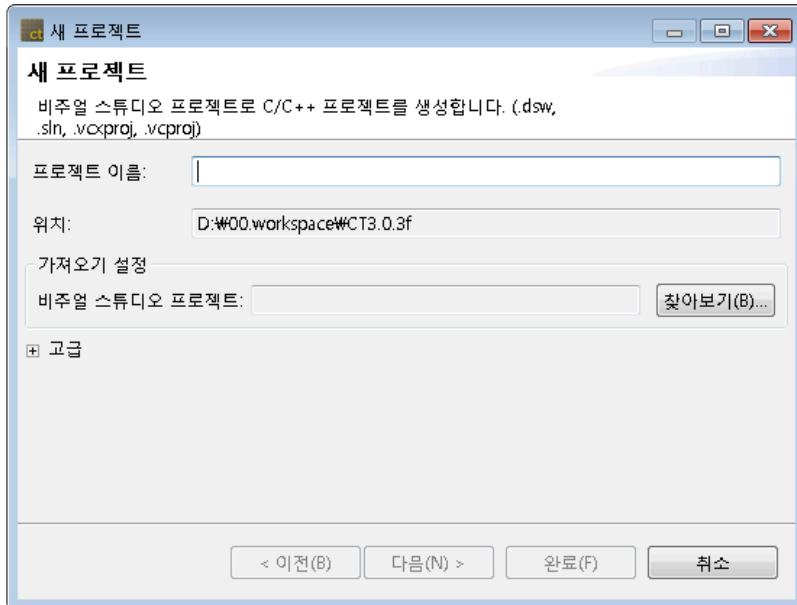
- 아래에 보이는 [소스 파일 목록 텍스트 파일] 화면에서 [찾아보기...]를 클릭하여 소스 파일 목록 (파일 절대 경로)이 기록된 텍스트 파일을 선택합니다.
- [제거] 버튼을 클릭하면 지정한 텍스트 파일과 파일을 통해 선택된 소스 파일을 선택 해제할 수 있습니다.

#### 6. 모든 설정이 완료된 후 [완료] 버튼을 클릭하여 프로젝트를 생성합니다.

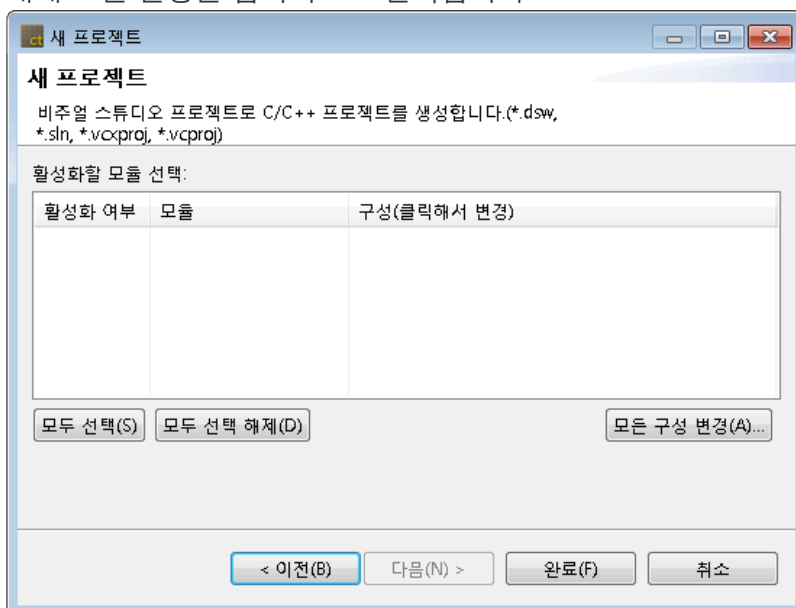


## 8.2. 비주얼 스튜디오 프로젝트

마이크로소프트 비주얼 스튜디오 프로젝트 파일(.dsw, .sln, .vcxproj, .vcproj)로 프로젝트를 생성할 때 선택합니다. 특별한 경우를 제외하고, 빌드에 필요한 정보는 자동으로 입력되므로 사용자가 별도로 지정할 내용이 없습니다.



1. [프로젝트 이름]에 프로젝트 이름을 입력합니다.
2. [가져오기 설정] -> [비주얼 스튜디오 프로젝트] 에 가져오기 위한 비주얼 스튜디오 프로젝트 파일을 지정합니다.
3. [고급] -> [환경 설정 스크립트 파일]은 등록하는 툴체인이 사용하는 컴파일러가 특정 환경 변수를 사용하는 경우, 컴파일러를 호출하기 전에 반드시 설정해야 하는 값들이 저장된 스크립트 파일을 설정합니다.
4. [다음]을 클릭하여 다음 화면으로 이동합니다. [완료]를 클릭하면 가져온 비주얼 스튜디오 프로젝트에 대해 모듈 설정은 임의적으로 선택됩니다.

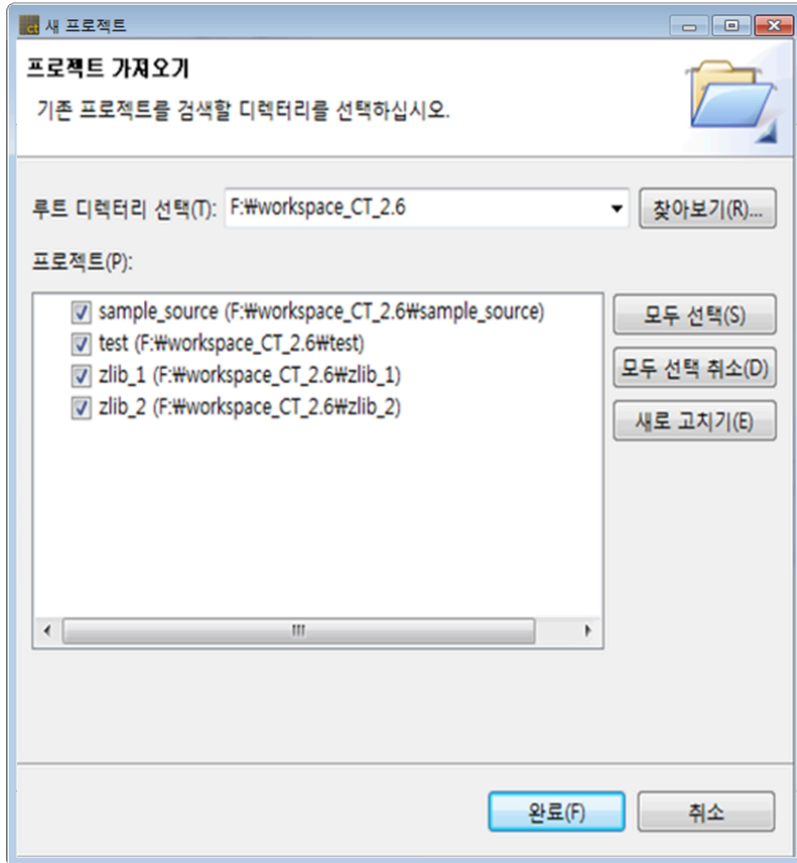


5. 비주얼 스튜디오 프로젝트에 포함된 모듈들 중에서 활성화 시킬 모듈을 선택합니다.

6. 각 모듈들이 갖는 구성을 선택합니다.
  - [모든 구성 변경]은 모듈에 대해 구성을 일괄적으로 변경할 수 있습니다.
7. 모든 설정이 완료된 후 [완료]버튼을 클릭하여 프로젝트를 생성합니다.

## 8.3. 기존 CodeScroll 프로젝트 정보로 C/C++ 프로젝트 생성

기존 CodeScroll 프로젝트를 임포트하여 해당 정보로 새로운 프로젝트를 생성합니다. 기존 프로젝트에 대한 여러 프로젝트 설정(소스 파일, 분석 제외/포함, 컴파일 플래그)을 그대로 구성하여 같은 정보의 새 프로젝트를 쉽게 생성할 수 있습니다.

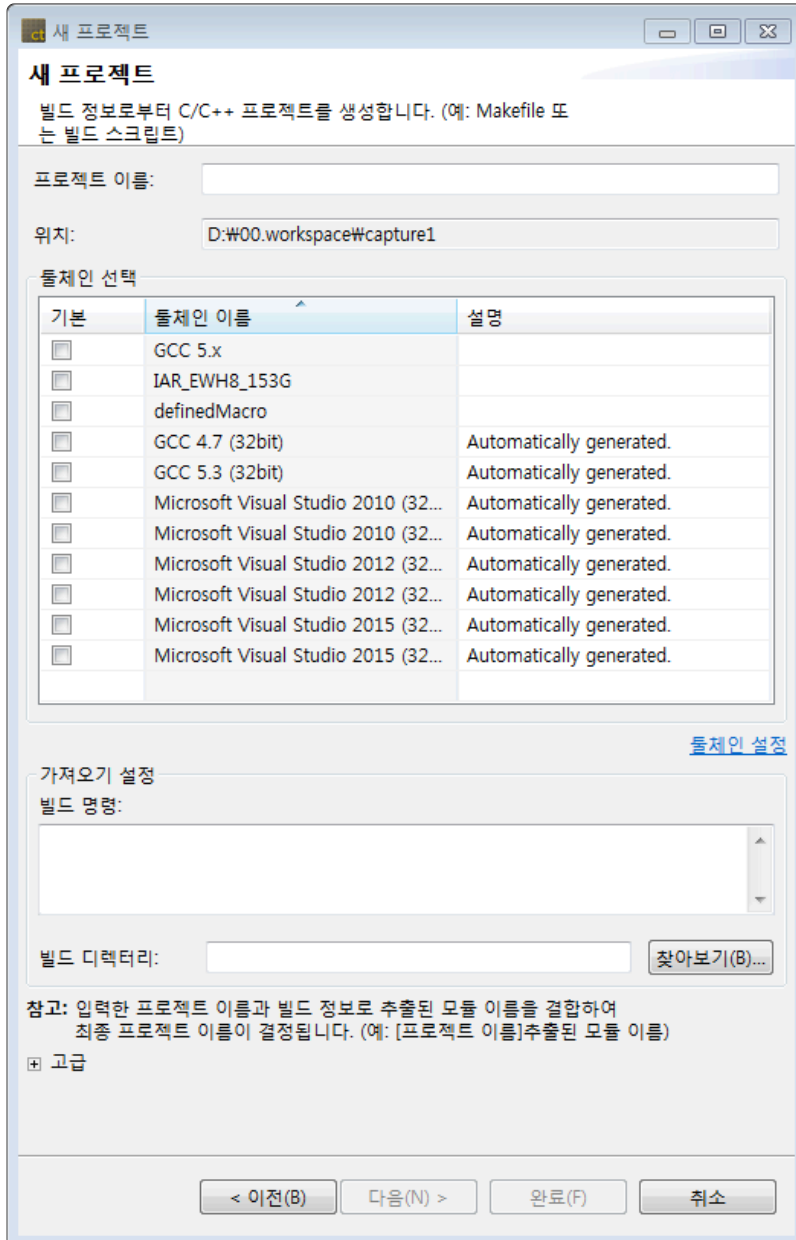


1. [찾아보기...] 버튼을 이용하여 CodeScroll 작업공간이나 CodeScroll 프로젝트 경로를 선택합니다.
2. 선택한 경로에 존재하는 프로젝트가 [프로젝트] 목록에 표시됩니다.
  - CodeScroll 작업공간을 선택했을 경우, 하위의 모든 프로젝트들이 표시됩니다.
3. [완료]를 클릭하여 프로젝트를 생성합니다.

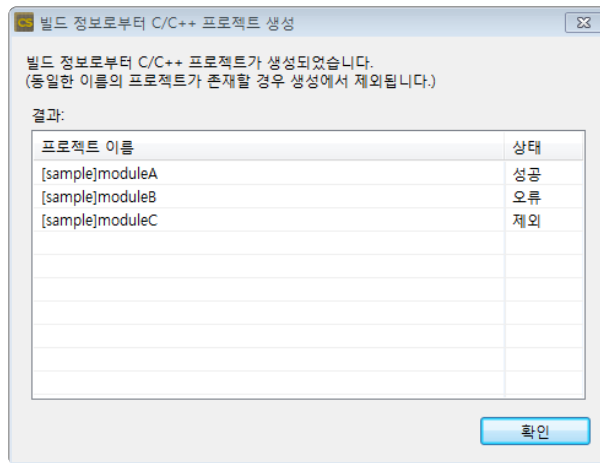
## 8.4. 빌드 정보로부터 C/C++ 프로젝트 생성

빌드 정보로부터 C/C++ 프로젝트를 생성할 때 선택합니다. 빌드 정보로 추출된 모듈의 수만큼 프로젝트가 생성됩니다.

1. [프로젝트 이름]에 프로젝트 이름을 입력합니다. 입력한 프로젝트의 이름과 빌드 정보로 추출된 모듈 이름을 결합하여 최종 프로젝트 이름이 결정됩니다. (예: [프로젝트 이름]추출된 모듈 이름)



2. 빌드에 사용하는 컴파일러와 동일한 툴체인을 선택합니다.
3. [가져오기 설정] -> [빌드 명령]에 빌드 시 필요한 명령어를 입력합니다.
  - 예) make all / make clean
4. [가져오기 설정] -> [빌드 디렉터리]에서 makefile이 저장되어 있는 경로를 지정해 줍니다.
5. [고급] -> [환경 설정 스크립트 파일]은 등록하는 툴체인이 사용하는 컴파일러가 특정 환경 변수를 사용하거나 환경 설정 스크립트 파일을 필요로 하는 프로젝트일 경우, 컴파일러를 호출하기 전에 반드시 설정해야 하는 값들이 저장된 스크립트 파일을 지정합니다.
6. 모든 설정을 완료하고 [완료]를 클릭하여 프로젝트를 생성합니다.
7. 빌드 후 프로젝트 생성이 완료되면 결과 화면이 나타납니다.



- 성공: 프로젝트 생성이 완료.
- 오류: 프로젝트 생성 중 오류로 인해 생성하지 못한 경우.
- 제외: 워크스페이스 내에 같은 프로젝트 이름이 존재하는 경우.

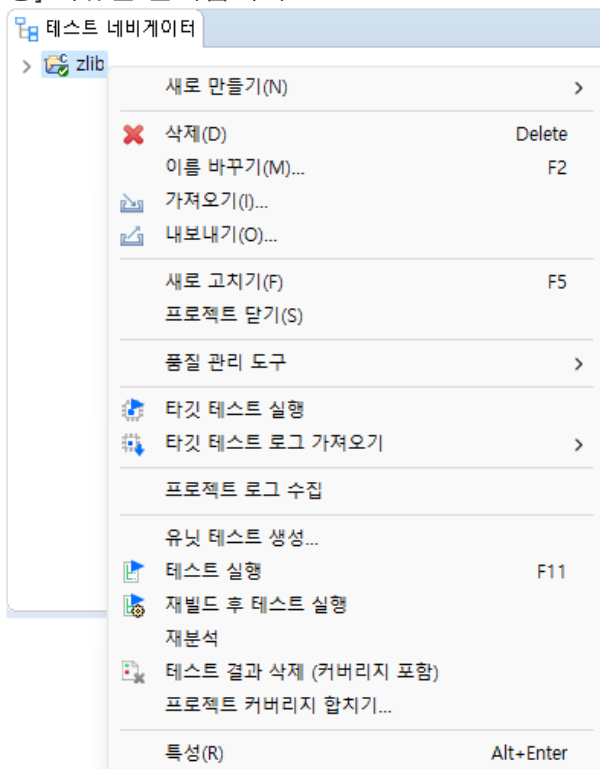
## 9. 테스트 생성

소스 코드의 함수를 테스트하기 위해서 테스트 데이터와 테스트 코드를 생성합니다. 각 함수 단위로 개별적인 테스트를 만들 수 있습니다.

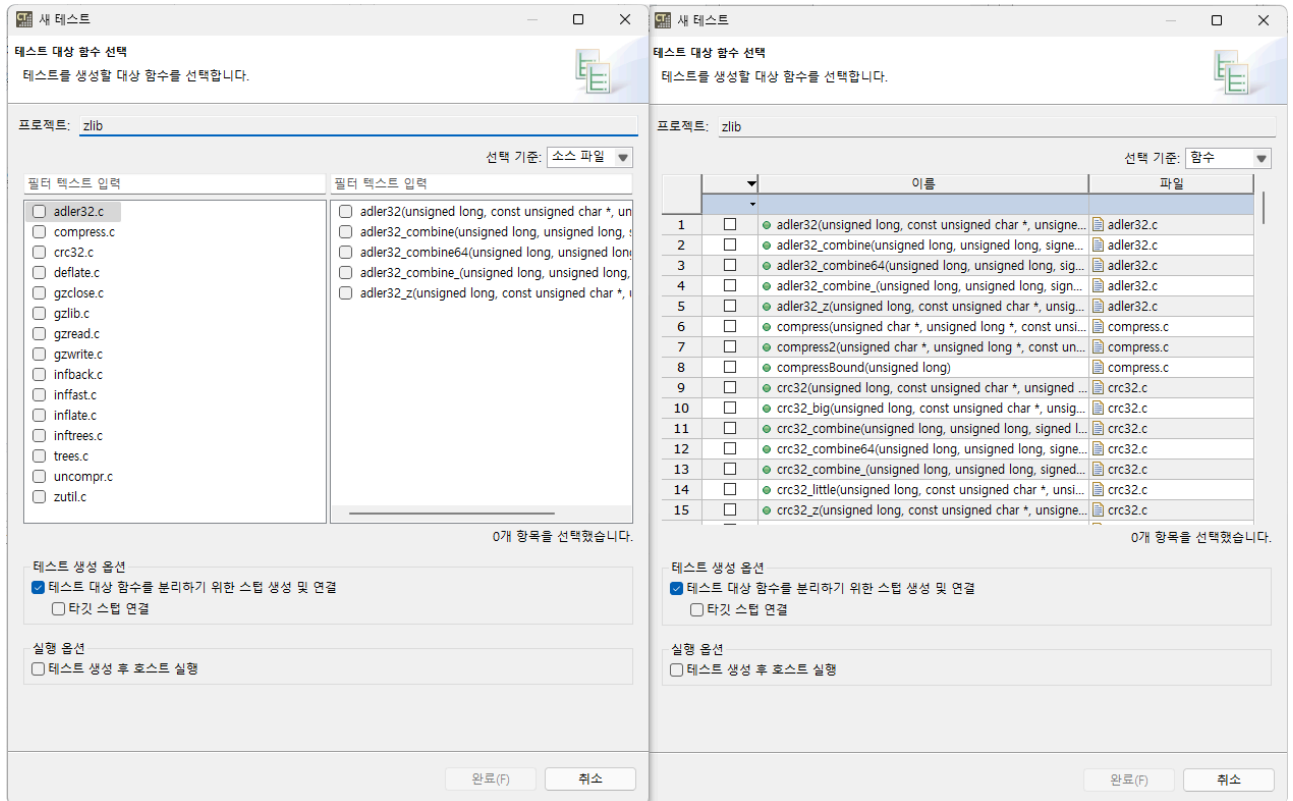
### 유닛 테스트 생성

사용자 소스의 각 함수에 대해 개별적으로 테스트 데이터와 테스트 코드를 생성합니다.

1. 유닛 테스트를 생성할 프로젝트 또는 프로젝트에 포함된 모듈을 선택하고 우클릭 후 [유닛 테스트 생성] 메뉴를 클릭합니다.



2. 분석이 끝나면 [테스트 대상 함수 선택]에서 유닛 테스트를 수행할 대상 함수를 선택합니다.



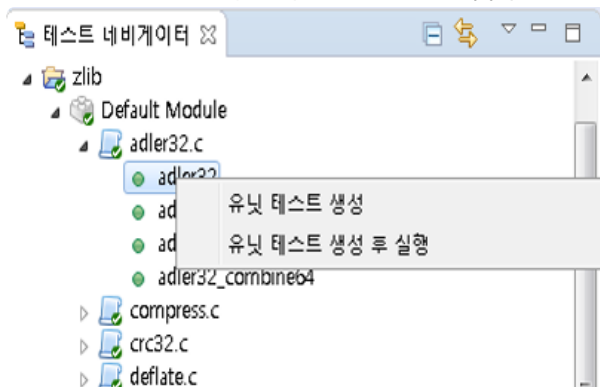
- [필터 텍스트 입력]을 이용해서 [테스트 대상 함수]에 대해 필터링할 수 있습니다.
- [선택 기준] 콤보 박스를 이용해서 소스 파일 또는 함수 기준으로 [테스트 대상 함수]를 선택할 수 있습니다.
- 우클릭을 이용하여 손쉽게 [테스트 대상 함수]를 전체 선택 및 해제할 수 있습니다.
- [테스트 생성 옵션]이 선택되면 테스트 대상 함수를 분리하기 위한 스텝이 생성되며 테스트의 호스트 스텝과 연결됩니다.
  - [타겟 스텝 연결]이 선택되면 생성된 스텝이 타겟 스텝과 연결됩니다.
- [테스트 생성 후 실행]이 선택되면 테스트 데이터 준비 후 테스트를 실행합니다.

3. [완료]를 클릭하여 유닛 테스트를 생성합니다.

## 유닛 테스트 생성 후 실행

[유닛 테스트 생성] 기능을 사용하지 않고, 유닛 테스트 생성 및 테스트 수행을 한 번에 진행할 수 있는 [유닛 테스트 생성 후 실행] 기능을 제공합니다.

- 테스트 네비게이터에서 테스트 대상 함수를 선택한 후 우클릭합니다.



- [유닛 테스트 생성 후 실행] 메뉴를 선택하면 대상 함수에 대한 테스트를 생성한 후, 테스트를 수행함

니다.

## 통합 테스트 생성

툴바 메뉴에서 통합 테스트를 생성할 수 있습니다.

유닛 테스트    통합 테스트

실행 ▶

함수 호출 커버리지

(98 / 0 / 0) 98    41.0% (130/317)

파일, 테스트, 상태, 이슈 입력

이름	결과	커버리지
> <input checked="" type="checkbox"/> INTEGRATION_0	(98 / 0 / 0) 98	0.0% (0/46)

통합 테스트 이름은 자동으로 부여되며 [이름 바꾸기] 컨텍스트 메뉴를 이용해 변경할 수 있습니다.

유닛 테스트    통합 테스트

실행 ▶

함수 호출 커버리지

(98 / 0 / 0) 98    41.0% (130/317)

파일, 테스트, 상태, 이슈 입력

이름	결과	커버리지
> <input checked="" type="checkbox"/> INTEGRATION_0	(98 / 0 / 0) 98	0.0% (0/46)

이름 바꾸기

새 이름: INTEGRATION\_0

확인    취소

하위 통합 테스트로 붙여넣기

삭제 Delete

이름 바꾸기... F2

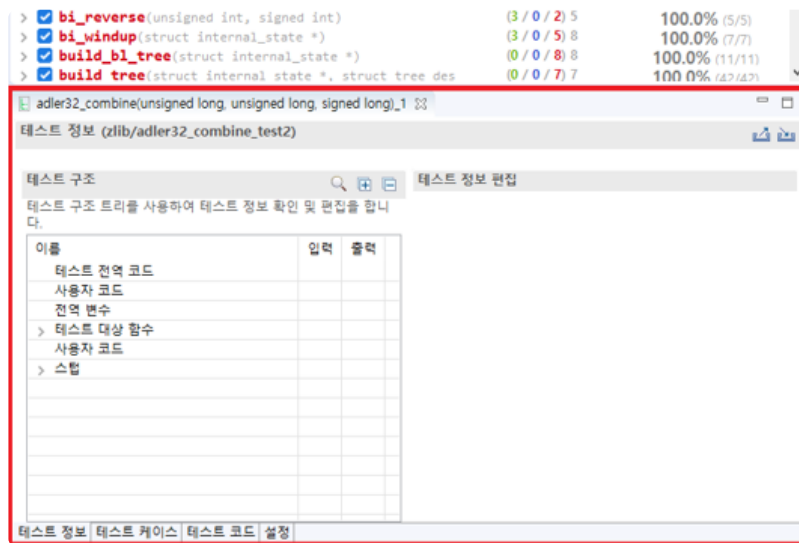
테스트 선택

테스트 선택 해제



## 10. 테스트 편집기

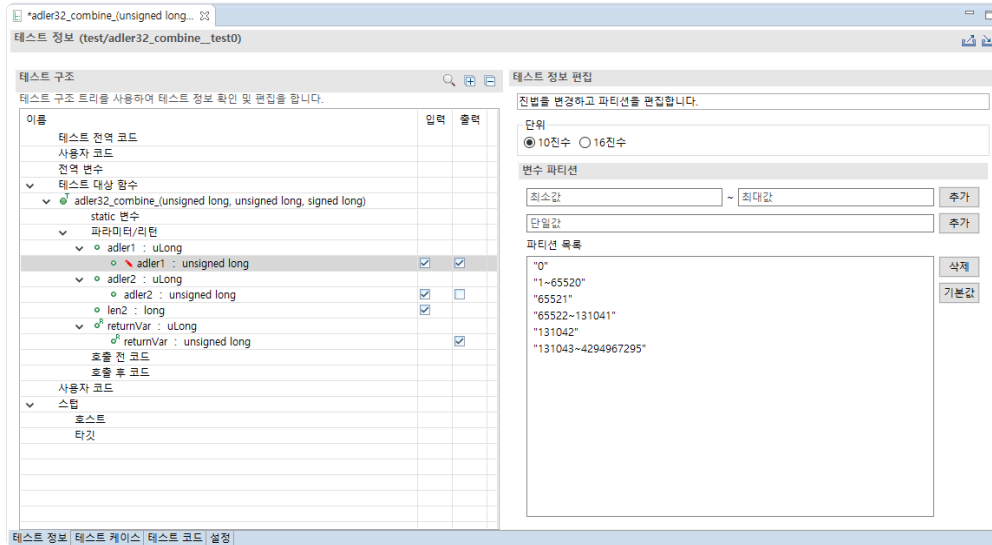
테스트 편집기는 각 테스트 뷰의 테스트나 테스트 케이스를 더블 클릭하여 열 수 있으며, 유닛/통합 테스트 뷰 하단에 위치합니다.




✿ 편집기의 위치는 사용자가 변경할 수 있으며, 독립된 창에서도 열 수 있습니다.

## 10.1. 테스트 정보

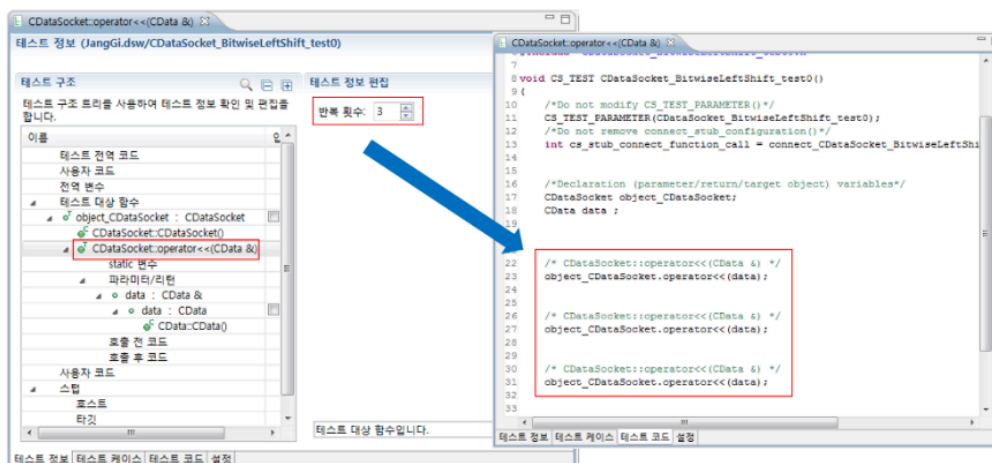
테스트 정보 탭은 테스트 코드를 테스트 구조 트리로 나타내어 사용자가 테스트 코드를 쉽게 수정할 수 있도록 제공하는 기능입니다.



- 사용자 코드를 테스트 코드에 삽입할 수 있습니다.
- 테스트 대상 함수와 관련이 있는 전역변수를 제어할 수 있습니다.
- 함수의 파라미터 및 리턴 값에 대한 입출력을 제어할 수 있습니다.
- 테스트 대상 함수를 호출하기 직전에 수행할 코드를 입력할 수 있습니다.
- 변경된 항목은  마커가 표시됩니다. 변경 사항을 저장하면 마커는 사라집니다.

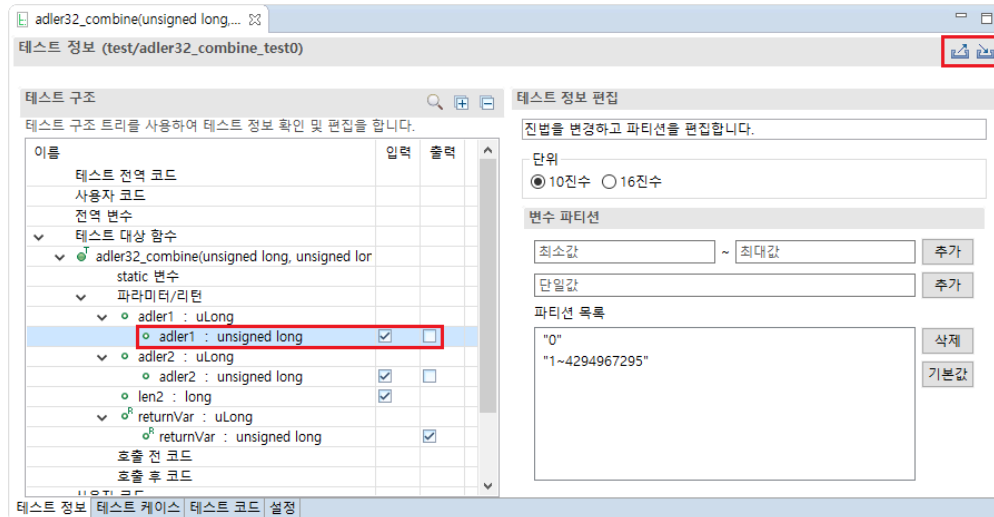
## 함수 호출 반복

테스트 대상 함수의 반복 횟수를 지정할 수 있습니다.



## 변수 파티션 내보내기/가져오기

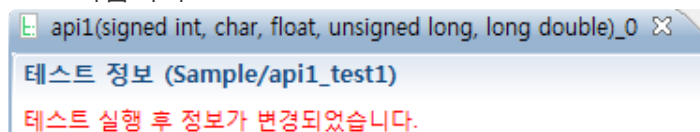
테스트 정보 탭에서 변수를 선택하여 변수 파티션 정보를 가져오고 내보낼 수 있습니다.



메뉴	설명
변수 파티션 내보내기	선택한 변수의 파티션 정보를 내보내기
변수 파티션 가져오기	선택한 변수에 파티션 정보를 가져오기

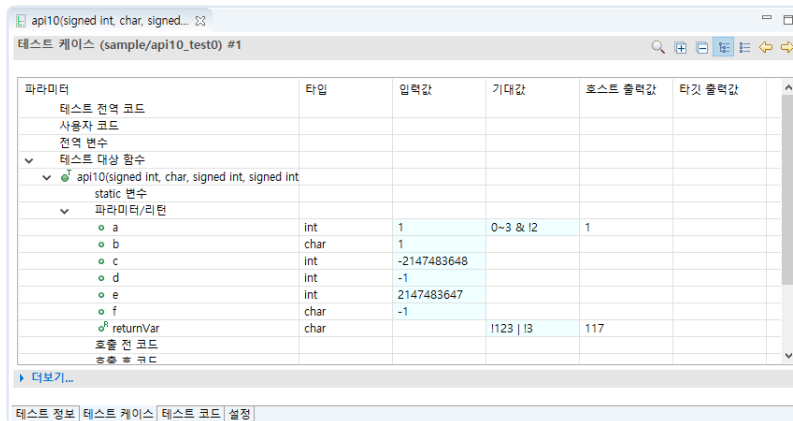
## 변경 내용 감지

이미 실행이 완료된 테스트 정보를 수정한 뒤 저장하면 상단에 “테스트 실행 후 정보가 변경되었습니다.”라고 표시됩니다.



## 10.2. 테스트 케이스

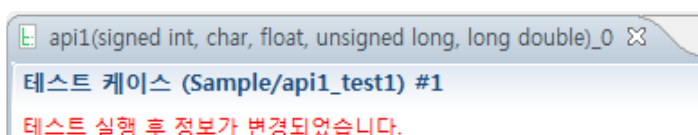
테스트 케이스 탭은 테스트 대상 함수의 입력값과 실행 결과의 기대값을 편집하는 기능을 제공합니다.



- 테스트 구조 트리에서 입력으로 지정한 변수에 대한 입력값을 입력할 수 있습니다.
  - 입력값이 없는 경우, 해당 변수의 값은 **'undefined value'**이며, 도구에서는 사용자 편의를 위해 입력 사용으로 체크된 변수 중 아무 값도 입력되지 않은 변수는 **0**(문자열일 경우 공백)으로 초기화합니다.
- [트리로 보기], [테이블로 보기] 툴바 메뉴로 테스트 구조를 테이블과 트리로 표현할 수 있습니다.
- [테스트 케이스 일괄 적용] 컨텍스트 메뉴로 해당 테스트 케이스의 입력값과 기대값을 같은 테스트 내 다른 테스트 케이스에 한 번에 적용할 수 있습니다.
- 테스트 구조 트리에서 출력으로 지정한 변수는 테스트를 수행하고 난 후, 결과값을 확인할 수 있습니다.
- 편집 가능한 셀은 옅은 하늘 색으로 표시됩니다.
- 기대값과 호스트/타겟 출력값이 다른 경우 파란색으로 표시됩니다.
- 호스트와 타겟의 출력값이 다른 경우 노란색으로 표시됩니다.
- 기대값에 ~(범위)와 논리 연산자 !(not), &(and), |(or)를 사용할 수 있습니다.(단, 문자열 형태의 데이터에 대해서는 |(or)연산자를 사용할 수 없습니다.)
- 다음과 같은 단축키를 제공합니다.
  - 되돌리기 : Ctrl + Z
  - 행 이동: 방향키(↑, ↓)
  - 편집 가능한 셀 포커스: Enter
  - 편집 가능한 셀 포커스 아웃: Esc
  - 열(편집 가능한 셀)이동: Tab(오른쪽), Shift + Tab(왼쪽)
  - 테스트 케이스 이동: Ctrl + ← / →

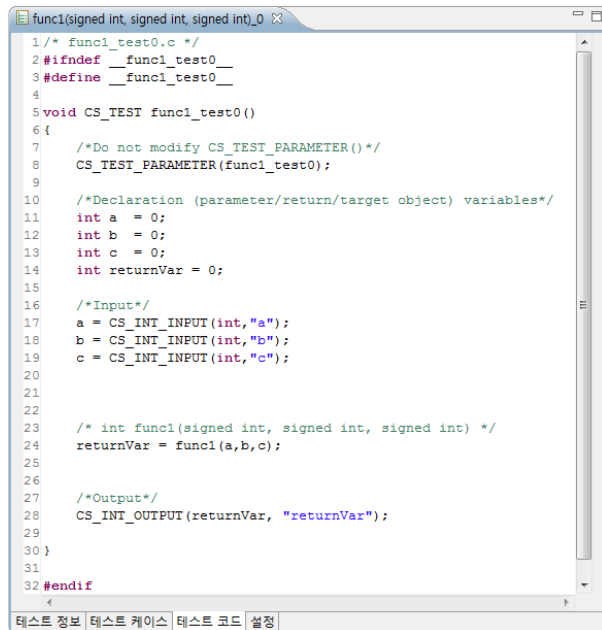
## 변경 내용 감지

이미 실행이 완료된 테스트 케이스를 수정하면 상단에 “테스트 실행 후 정보가 변경되었습니다.”라고 표시됩니다.



## 10.3. 테스트 코드

테스트 코드는 테스트 정보 창의 테스트 코드 탭을 클릭하여 확인할 수 있습니다.



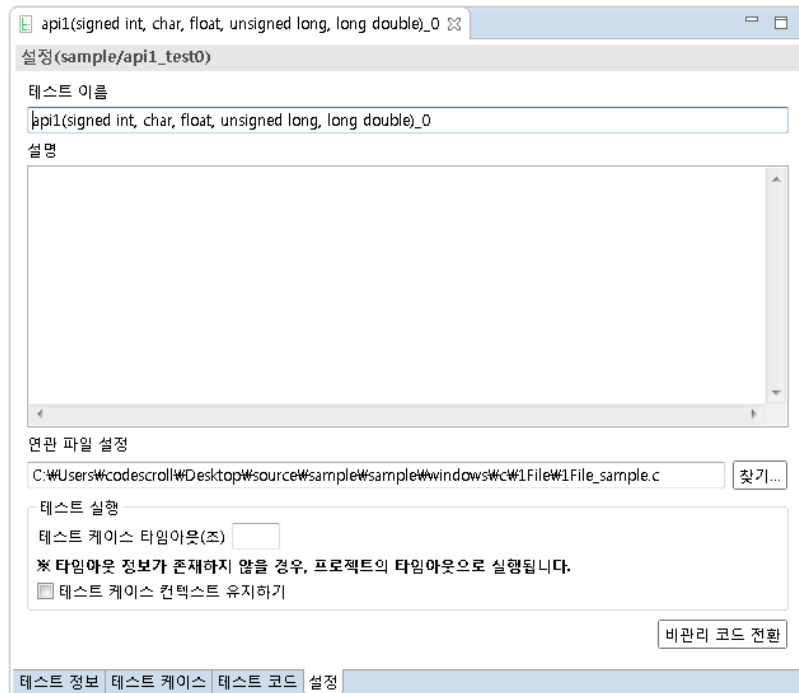
```

1 /* func1_test0.c */
2 #ifndef __func1_test0__
3 #define __func1_test0__
4
5 void CS_TEST func1_test0()
6 {
7     /*Do not modify CS_TEST_PARAMETER()*/
8     CS_TEST_PARAMETER(func1_test0);
9
10    /*Declaration (parameter/return/target object) variables*/
11    int a = 0;
12    int b = 0;
13    int c = 0;
14    int returnVar = 0;
15
16    /*Input*/
17    a = CS_INT_INPUT(int, "a");
18    b = CS_INT_INPUT(int, "b");
19    c = CS_INT_INPUT(int, "c");
20
21
22
23    /* int func1(signed int, signed int, signed int) */
24    returnVar = func1(a,b,c);
25
26
27    /*Output*/
28    CS_INT_OUTPUT(returnVar, "returnVar");
29 }
30
31
32 #endif
  
```

- 사용자가 테스트 구조 트리를 편집한 후 저장하면 그 변경 사항은 자동으로 테스트 코드에 반영됩니다.
- 테스트 대상 함수의 반환 타입이 primitive 타입일 경우 반환 값을 출력하는 매크로가 자동으로 생성됩니다.
- 사용자가 직접 테스트 코드를 수정하려면 해당 테스트를 비관리 코드로 전환해야 합니다.
- 테스트 수행 함수의 형식은 반환 타입과 입력 파라미터가 void 타입이어야 합니다.

## 10.4. 설정

설정 탭에서는 유닛 테스트에 대한 여러 설정을 할 수 있습니다.



- 테스트 이름을 수정한 후 저장하면, 유닛 테스트 뷰에 나타나는 테스트 이름이 변경됩니다.
- 테스트에 대한 설명을 입력할 수 있습니다.
- 연관 파일 설정으로 테스트 대상 함수가 속한 파일을 선택할 수 있습니다.
- 테스트에 대한 타임아웃 설정을 할 수 있습니다.
- 테스트에 대한 컨텍스트 유지 여부를 선택할 수 있습니다.
- 사용자가 테스트 코드를 직접 작성하려면 [비관리 코드 전환]을 해야 합니다.

## 10.5. 테스트 매크로

CT 2023.12는 여러 가지 매크로를 제공하여 사용자가 테스트 코드를 보다 쉽게 작성할 수 있도록 도와줍니다. 제공되는 매크로는 편집기에서 단축키(Ctrl + Spacebar)를 사용하여 확인 및 작성할 수 있습니다.

### ASSERT 매크로

조건식을 검사하여 성공/실패 여부를 [테스트 케이스 탭]에 출력합니다.

매크로 형태	매개 변수	예제
CS_ASSERT(_b)	_b: 조건식	CS_ASSERT(val!=1);
CS_ASSERT_MSG(_B, _msg)	_B: 조건식 _msg: 조건식이 거짓일 경우 출력할 메시지	CS_ASSERT_MSG(val==1, "val is not 1!");

### 출력 매크로

특정 변수의 값을 [테스트 케이스 탭]에 출력합니다.

매크로 형태	매개 변수	예제
CS_INT_OUTPUT(_v, _s) CS_UINT_OUTPUT(_v, _s) CS_FLT_OUTPUT(_v, _s) CS_STR_OUTPUT(_v, _s)	_v: 값을 출력할 변수 _s: [테스트 케이스 탭]에 표시할 이름	CS_INT_OUTPUT(int_var, "int_var_name"); CS_UINT_OUTPUT(unsigned_int_var, "unsigned_int_var_name"); CS_FLT_OUTPUT(float_var, "float_var_name"); CS_STR_OUTPUT(string_var, "string_var_name");

### 입력 매크로

테스트 수행 함수에 테스트 데이터를 전달하는 매크로입니다.

매크로 형태	매개 변수	예제
CS_INT_INPUT(_t, _s) CS_UINT_INPUT(_t, _s) CS_FLT_INPUT(_t, _s) CS_STR_INPUT(_t, _s)	_t: 변수의 타입 _s: [테스트 케이스 탭]에 표시할 이름	CS_INT_INPUT(int, "int_var_name"); CS_UINT_INPUT(unsigned int, "unsigned_int_var_name"); CS_FLT_INPUT(float, "float_var_name"); CS_STR_INPUT(char*, "string_var_name");

## 주소 관련 매크로

변환된 소스 코드에서 임베디드 상의 주소에 직접 값을 입력하거나 가져오는 부분이 있는 경우 로컬 컴퓨터에서 수행 시 정상 동작을 하지 않을 수 있습니다. 이런 경우 주소 관련 매크로를 사용하여 가상의 주소에 값을 입력하고 출력할 수 있습니다.

매크로 형태	설명
CS_VIRTUAL_ADDR(_b,_e)	주소(_b)부터 주소(_e)까지의 공간을 생성합니다.
CS_ADDR_ASSIGN(_t,_a,_v) CS_ADDR_SET(_t,_a,_v)	주소(_a)에 타입(_t)의 값(_v)을 지정합니다.
CS_ADDR_GET(_t,_a)	주소(_a)에서 타입(_t)의 값을 가져옵니다.
CS_ADDR_PTR(_a)	주소(_a)를 포인터 타입으로 반환합니다.
CS_VIRTUAL_ADDR_CLEAR()	생성한 메모리 공간을 제거합니다.

### 주소 관련 매크로 예제

```
// 0xFFE40000U에서부터 100만큼의 가상 메모리 공간 생성
CS_VIRTUAL_ADDR(0xFFE40000U, 0xFFE40000U+100);

// 0xFFE40000U에 int 타입의 값 10을 할당
CS_ADDR_ASSIGN(int ,0xFFE40000U, 10);
CS_ADDR_SET(int ,0xFFE40000U, 10);

// 변수 a에 0xFFE40000U의 값을 할당
int a = CS_ADDR_GET(int ,0xFFE40000U);

// 변수 ptr에 0xFFE40000U의 주소를 할당
int* ptr = CS_ADDR_PTR(0xFFE40000U);

// 생성한 가상 메모리 공간 삭제
CS_VIRTUAL_ADDR_CLEAR();
```

\* 주소 관련 매크로에 대한 자세한 내용은 User Guides의 [가상 주소 사용 가이드](#) 페이지를 참고하시기 바랍니다.

## 기타 매크로

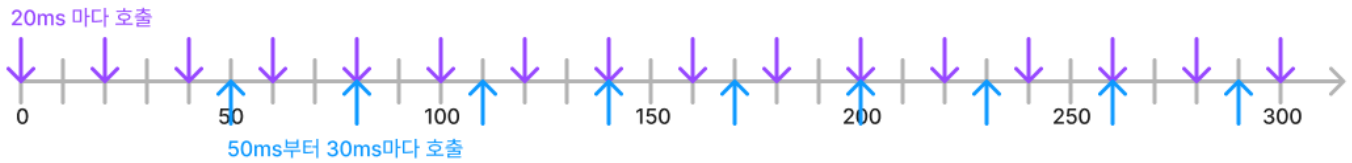
매크로 형태	매개 변수	설명	예제
CS_LOG(_msg)	_msg: 로그 메시지	사용자 로그 출력	CS_LOG("User Log");



CS_TESTCASENO()		현재 수행중인 테스트 케이스 의 번호 반환	long long testCaseNum = CS_T ESTCASENO ( ) ;
-----------------	--	----------------------------	---

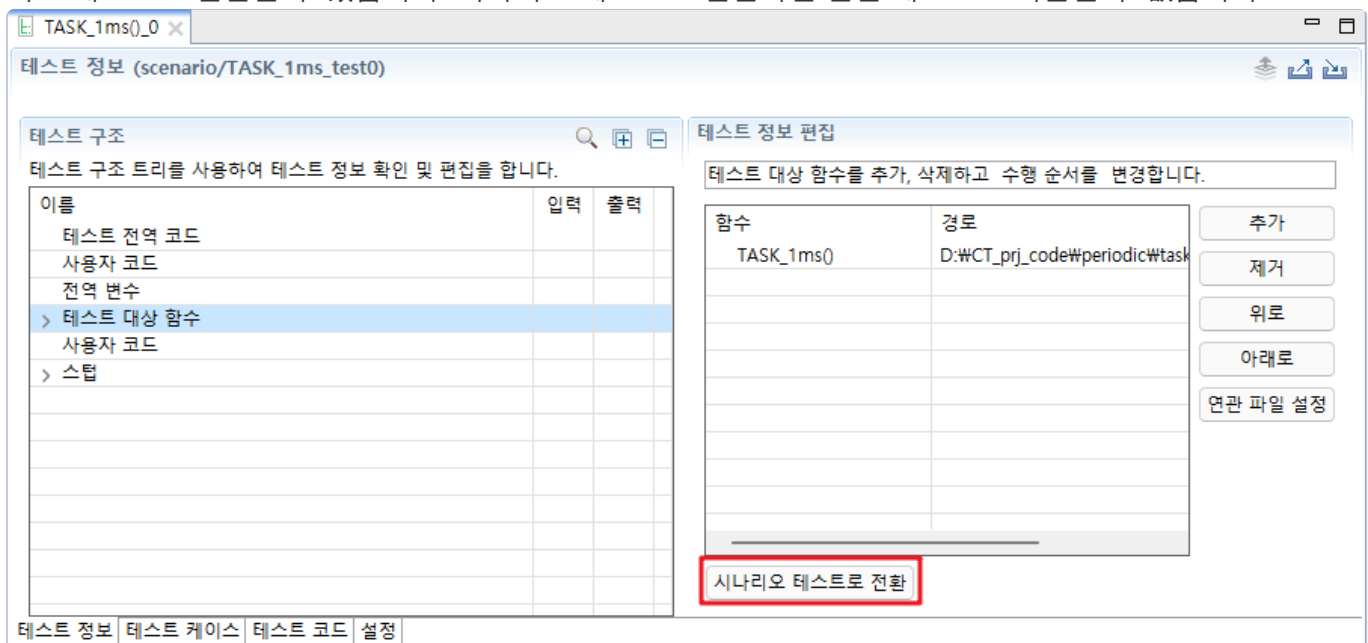
## 10.6. 시나리오 테스트

CT 2023.12는 다양한 시나리오를 테스트할 수 있도록 시나리오 테스트 기능을 제공합니다. CT 2023.12의 시나리오 테스트 기능을 이용하여 주기적으로 호출되는 여러 함수를 테스트할 수 있습니다.



### 시나리오 테스트로 전환하기

[테스트 편집기] > [테스트 정보] > [테스트 대상 함수] > [시나리오 테스트로 전환]을 선택하여 테스트를 시나리오 테스트로 전환할 수 있습니다. 시나리오 테스트로 전환하면 일반 테스트로 되돌릴 수 없습니다.



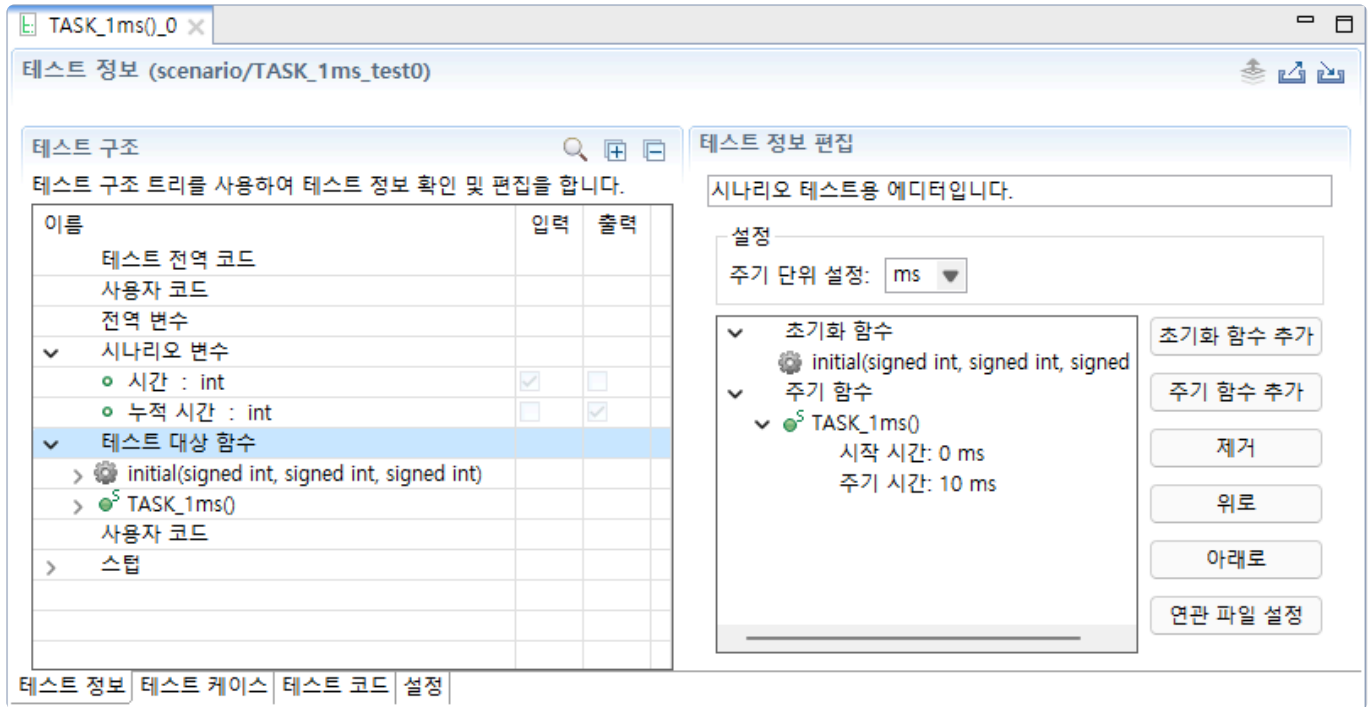
- 시나리오 테스트로 전환할 수 없는 경우
  - C 프로젝트가 아닌 경우
  - [테스트 대상 함수]에 주기 함수로 사용할 수 없는 함수가 있는 경우
    - 주기 함수로 사용할 수 있는 함수는 반환형이 void이고 매개변수가 없는 경우입니다.

### 시나리오 테스트 설계

시나리오 테스트로 전환한 후, [테스트 편집기]의 [테스트 정보], [테스트 케이스] 탭에서 시나리오 테스트를 설계할 수 있습니다. 시나리오 테스트는 테스트 케이스의 컨텍스트가 유지됩니다.

#### 시나리오 테스트 설계 - [테스트 정보] 탭

시나리오 테스트로 전환하면 [테스트 정보] 탭에 시나리오 변수가 추가되며 [테스트 대상 함수]의 에디터가 변경됩니다.

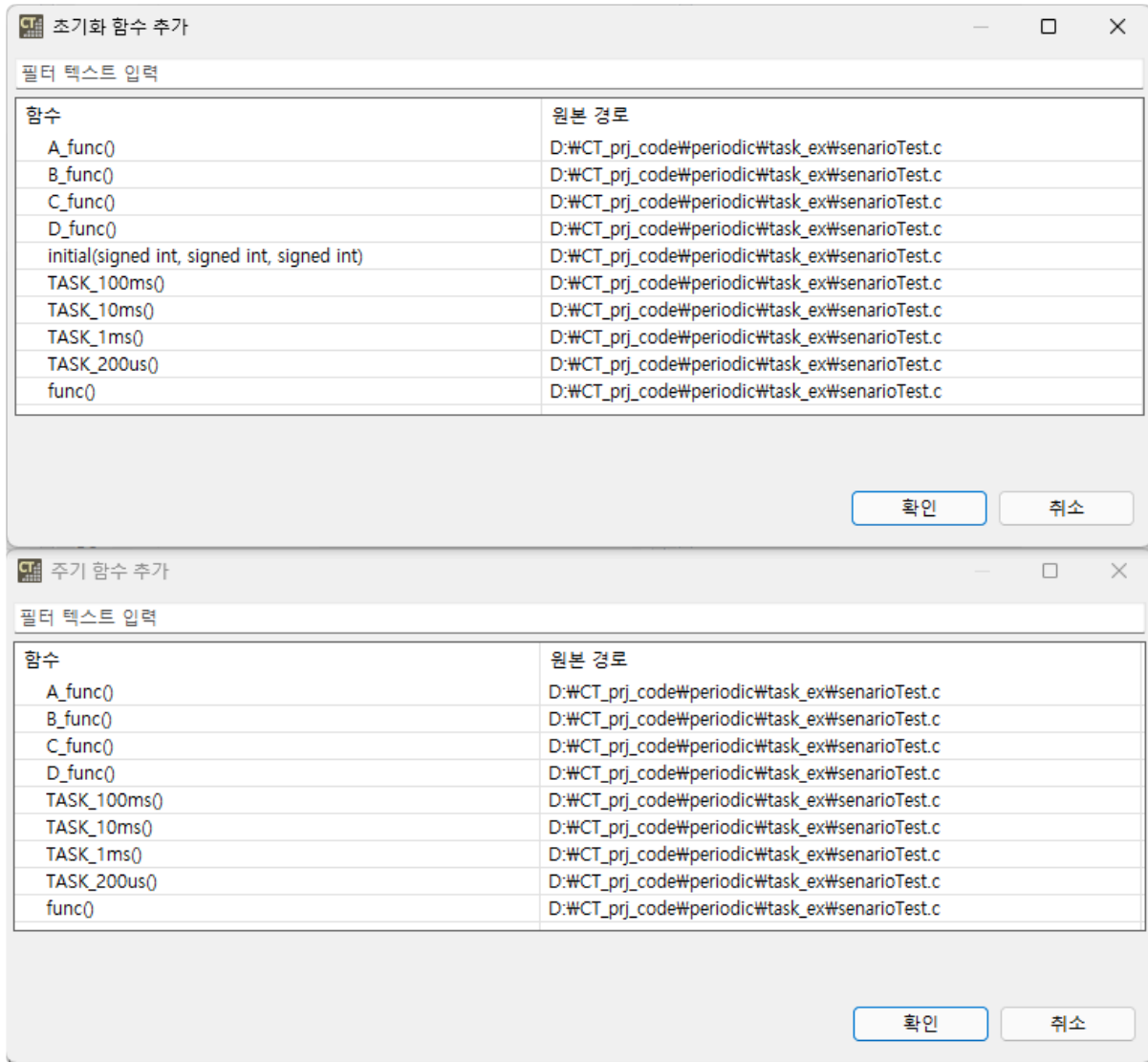


### 시나리오 테스트의 아이콘

아이콘	설명
	초기화 함수
	주기 함수

### 테스트 대상 함수

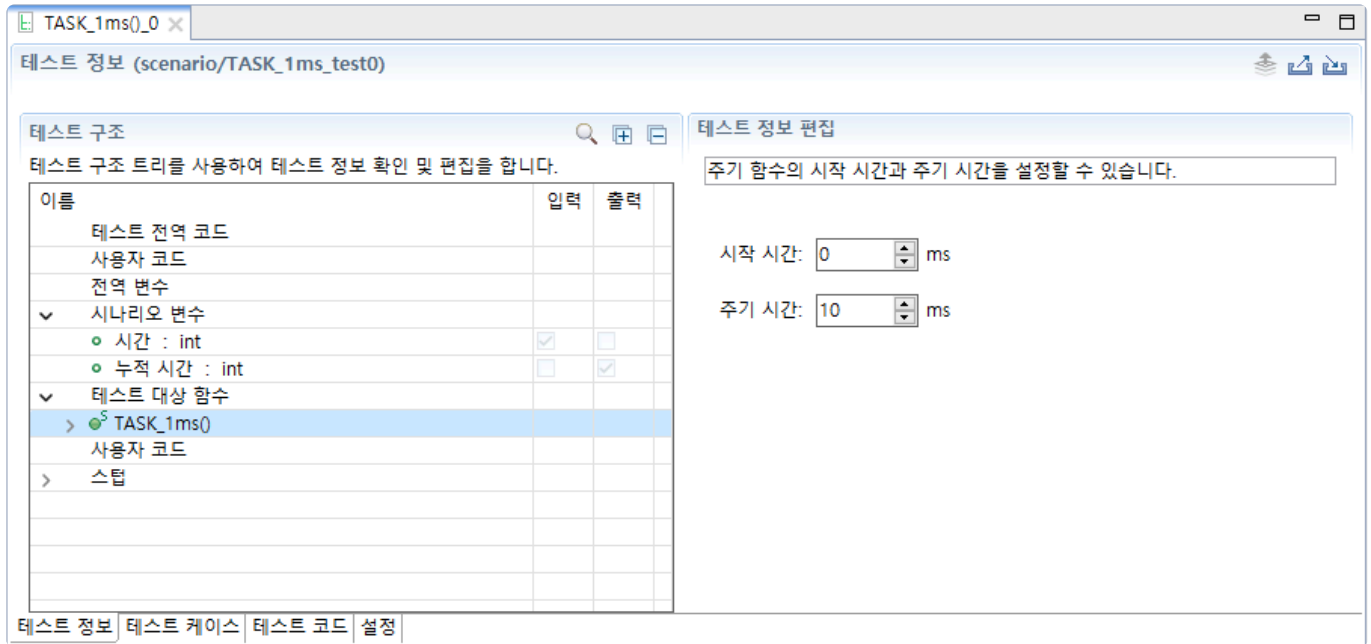
- 초기화 함수
  - 주기가 시작되기 전에 한 번 호출되는 함수
  - 전역 변수 초기화 등 테스트를 수행하기 전에 미리 실행하여야 하는 함수를 추가하여 사용할 수 있습니다.
- 주기 함수
  - 일정한 시간 간격을 두고 반복적으로 호출되는 함수
  - 반환형이 void이고 파라미터가 없는 경우, 주기 함수로 사용할 수 있습니다.
  - 사용자 코드, 함수 호출 전/후 코드를 활용하여 다양한 설계가 가능합니다.
- 주기 시간
  - 함수가 반복 호출되는 시간 간격
  - [테스트 케이스] 탭에서 시나리오 변수의 시간에 적절한 값을 넣어 테스트 수행 시간을 조절할 수 있습니다.
- 주기 단위 설정
  - [테스트 편집기]와 테스트 보고서에서 표현할 주기의 단위를 ms,  $\mu$ s, ns로 설정할 수 있습니다. 프로젝트 특성에서 주기 단위의 기본값을 변경할 수 있습니다.
- 초기화 함수 추가/주기 함수 추가



- [초기화 함수 추가]를 선택하면 모든 함수가 목록에 나타납니다.
- [주기 함수 추가]를 선택하면 주기 함수로 사용할 수 있는 함수만 목록에 나타납니다.
- 주기 함수를 추가하는 경우, 시작 시간은 0, 주기 시간은 10으로 설정됩니다.
- 제거
  - 초기화 함수와 주기 함수를 삭제합니다. 주기 함수는 2개 이상일 경우 삭제가 가능합니다.
- 위로/아래로
  - 초기화 함수와 주기 함수를 위/아래로 움직여 호출 순서를 변경합니다.
- 연관 파일 설정
  - 해당 함수가 포함된 소스 코드를 설정하거나 변경할 수 있습니다.

## 함수

주기 함수를 선택하여 시작 시간과 주기 시간을 설정할 수 있습니다.

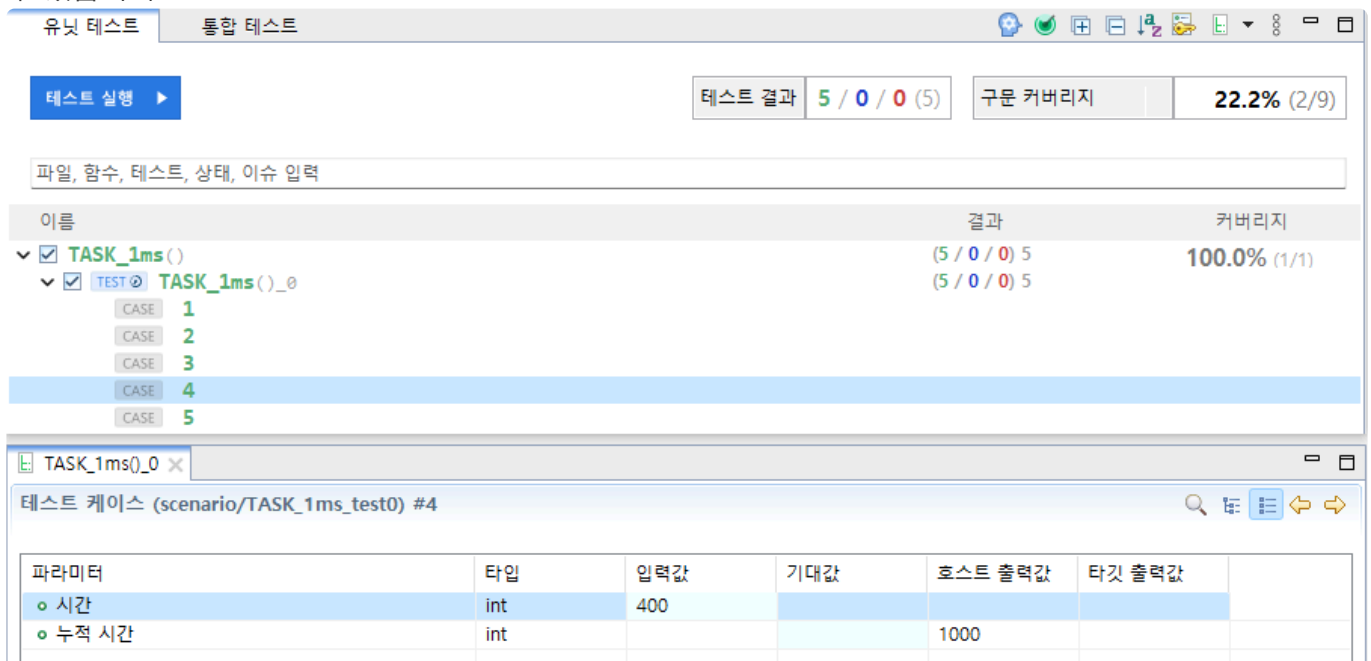


## 시나리오 변수

- 시간: 테스트 케이스별 실행 시간을 입력받는 변수
- 누적 시간: 시나리오 테스트가 실행된 누적 시간을 출력하는 변수

## 시나리오 테스트 설계 - [테스트 케이스] 탭

시나리오 테스트의 [테스트 케이스] 탭에서 시나리오 변수를 확인할 수 있습니다. [시간]의 입력값에 테스트 케이스별 실행 시간을 입력하고 테스트를 실행하면 실제로 실행된 시간을 [누적 시간]의 출력값으로 확인할 수 있습니다.



시나리오 테스트에 대한 자세한 내용은 User Guide의 [\[시나리오 테스트 사용 가이드\]](#) 페이지를 참고하시기를 바랍니다.

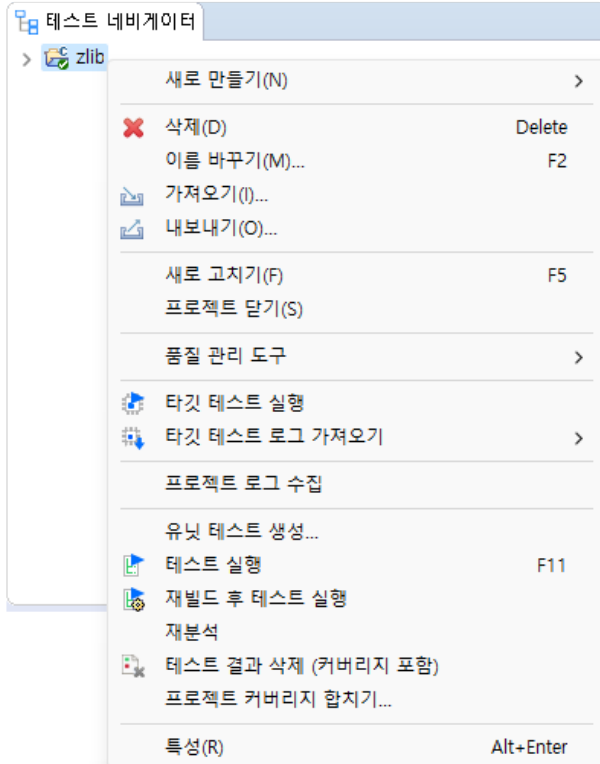
# 11. 테스트 실행

테스트는 다양한 방법으로 실행 가능합니다.

## 전체 테스트 실행

[유닛 테스트] 뷰와 [통합 테스트] 뷰에서 선택한 테스트를 실행합니다.

1. 테스트를 실행할 프로젝트를 선택, 우클릭 후 [테스트 실행]

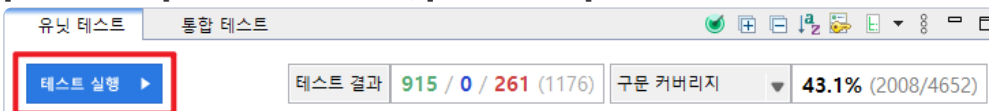


2. 테스트 진행 상태를 확인할 수 있는 진행 정보 창이 열리며, 테스트가 완료되면 각 뷰를 통해 결과를 확인할 수 있습니다.

## 유닛 테스트 실행

[유닛 테스트] 뷰에서 선택한 테스트를 수행합니다.

1. [유닛 테스트] 뷰의 대시보드에서 [테스트 실행] 버튼 클릭



2. 테스트 진행 상태를 확인할 수 있는 진행 정보 창이 열리며, 테스트가 완료되면 각 뷰를 통해 결과를 확인할 수 있습니다.

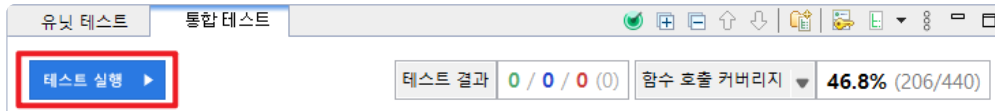
✿ 특정 테스트 케이스만 수행하려면 실행하려는 테스트 케이스를 선택한 후 마우스 우클릭하

여 [테스트 케이스 실행] 메뉴를 클릭합니다.

## 통합 테스트 실행

[통합 테스트] 뷰에서 선택한 테스트를 수행합니다.

1. [통합 테스트] 뷰의 대시보드에서 [테스트 실행] 버튼 클릭

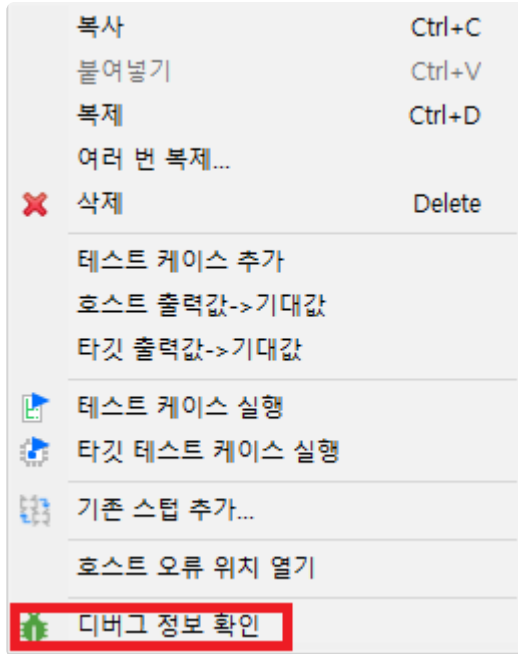


2. 테스트 진행 상태를 확인할 수 있는 진행 정보 창이 열리며, 테스트가 완료되면 각 뷰를 통해 결과를 확인할 수 있습니다.

## 12. 디버그 정보 확인

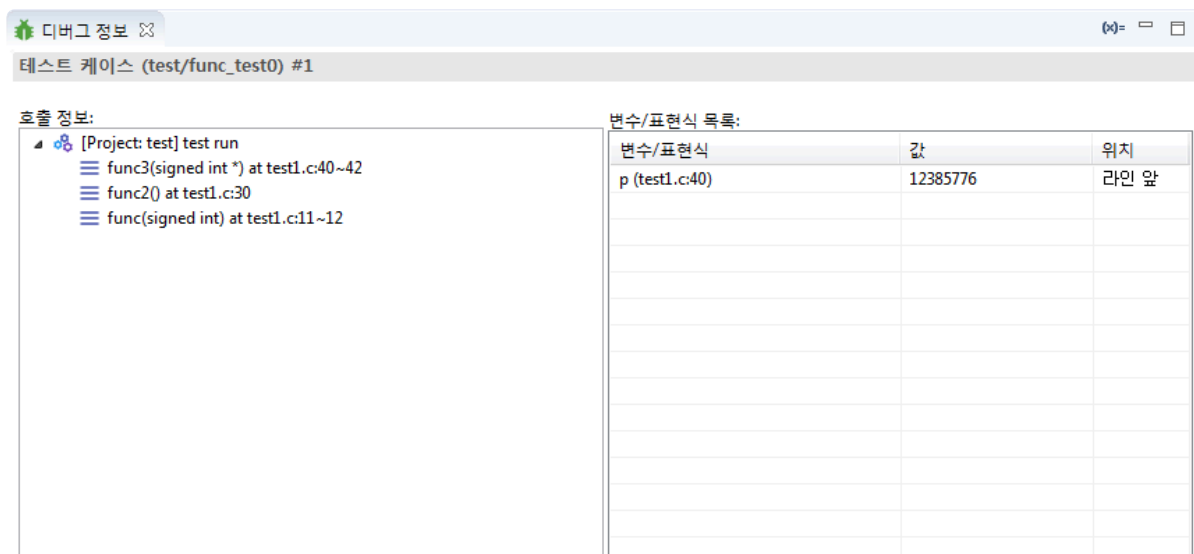
테스트 케이스 실행 중 오류가 발생한 경우, 오류의 원인을 파악하기 위해 디버그 정보 확인 기능을 사용할 수 있습니다.

[디버그 정보 확인] 기능은 유닛 테스트 뷰에서 테스트 케이스의 컨텍스트 메뉴로 확인할 수 있습니다.



✿ 실행되지 않은 테스트 케이스는 [디버그 정보 확인]을 수행하기 전에 자동으로 실행합니다.

[디버그 정보 확인]이 수행되면, [디버그 정보 뷰]에 디버그 정보가 표시됩니다.



[디버그 정보 확인]을 수행한 테스트 케이스에는 (디버그 정보 포함) 텍스트가 표시됩니다.



▼

☒ test()

▼

☒ test test()\_0

case

1

Signaled

(디버그 정보 포함)

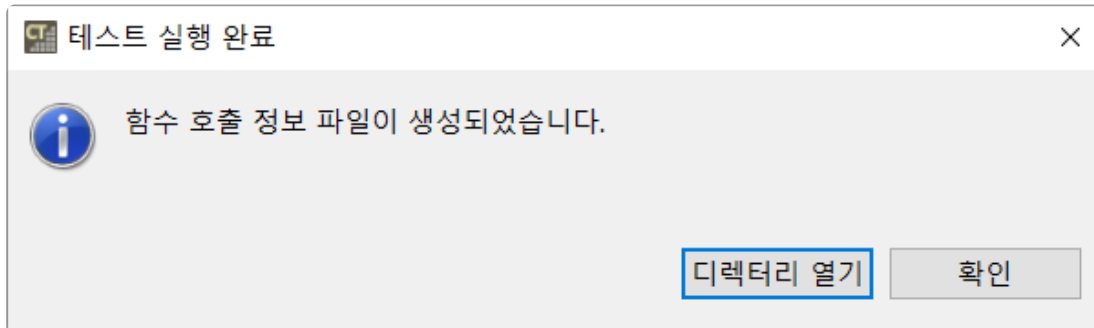
## 13. 함수 호출 정보 출력

[테스트 케이스별 함수 호출 정보 출력] 기능으로 테스트 케이스를 실행할 때 호출된 함수를 순서대로 출력할 수 있습니다.

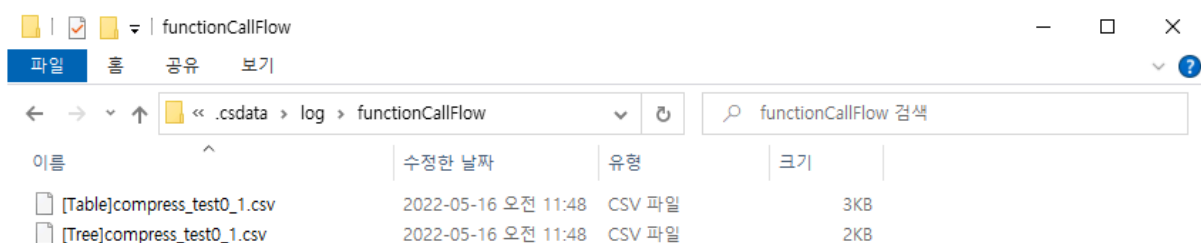
1. [유닛 테스트 뷰]의 테스트 케이스를 오른쪽 클릭하여 [테스트 케이스별 함수 호출 정보 출력]을 선택합니다.



2. [테스트 케이스별 함수 호출 정보 출력]이 수행되면, [테스트 실행 완료] 창이 뜹니다.



3. [테스트 실행 완료] 창에서 [디렉터리 열기] 버튼을 클릭하면 함수 호출 정보 파일이 있는 디렉터리가 열립니다.



✿ 함수 호출 정보 파일은 [출력 형태] 고유한 테스트 이름\_테스트 케이스 번호.csv 형식으로 출력됩니다.

- 출력에 포함되는 함수 정보는 함수의 이름과 매개 변수이며, 함수 호출 정보 파일은 두 가지 형태로 출력됩니다.
  - 테이블 형태
    - 함수 호출 관계가 2개의 열로 기록됩니다.
    - 첫번째 열에는 호출하는 함수, 두번째 열에는 호출되는 함수가 기록됩니다.
  - 트리 형태
    - 함수 호출 정보를 호출 깊이에 따라 여러 개의 열로 나타냅니다.
    - 호출 위치가 다르면 다른 행에 정보를 표시합니다.
- 동일한 함수가 두 번 이상 반복해서 호출되면 축약하여 기록합니다.
  - 반복 호출 횟수는 함수 뒤에 [called 호출 횟수 times]를 붙여 나타냅니다.
  - ex. 같은 함수가 5번 연속으로 호출될 경우, 첫번째 함수 뒤에 [called 5 times]가 붙습니다.

## 출력 예시

```
// example code
void callee() { /*doing something*/ }
void callee2() { /*doing something*/ }
void func1() { callee(); callee2(); }
void func2() { callee(); callee(); callee(); }
void caller() { func1(); func2(); }
```

- 테이블 형태

caller-function	callee-function
caller()	func1()
func1()	callee()
func1()	callee2()
caller()	func2()
func2()	callee() [called 3 times]

- 트리 형태

caller()	func1()	callee()
		callee2()
	func2()	callee() [called 3 times]

! 생성자, 소멸자, 연산자 오버로딩 함수와 정의가 없는 함수(ex. 시스템 함수)는 출력에서 제외

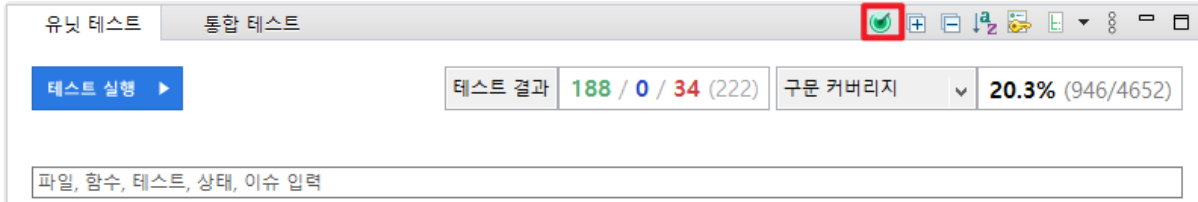
외됩니다. try-catch문, 삼항 연산자 등에서 호출되는 함수는 제대로 출력되지 않을 수 있습니다.

## 14. 결함 위치 찾기

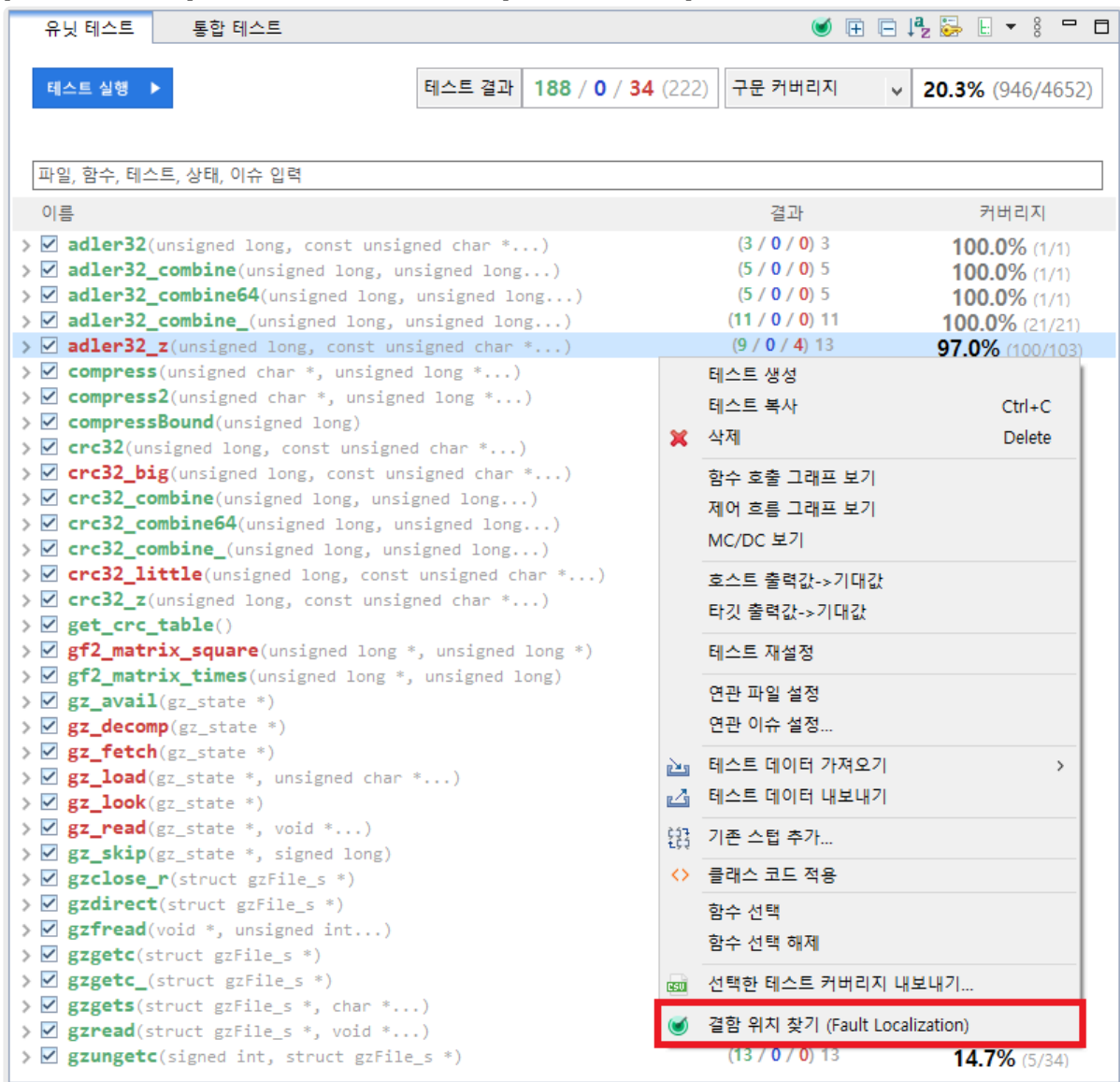
결함 위치 찾기는 오류 또는 실패한 테스트 케이스로부터 결함 위치를 찾아냅니다.

1. 결함 위치 찾기를 실행 방법은 네 가지 방법이 있습니다.

- [테스트 뷰]의 [툴바]에서 [결함 위치 찾기] 아이콘을 선택

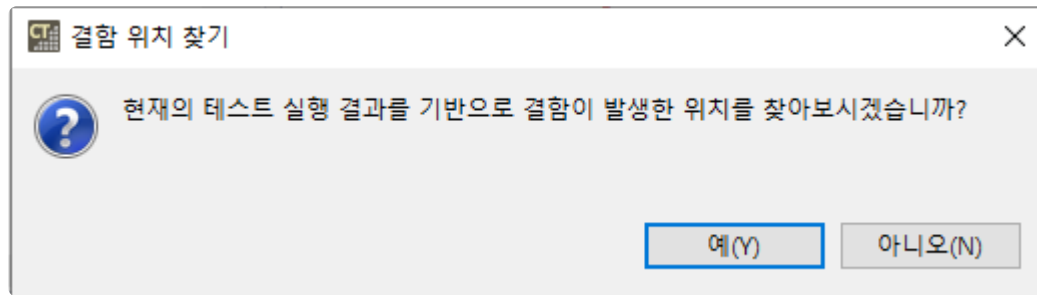


- [테스트 뷰 영역]의 테스트 함수를 우클릭, [결함 위치 찾기] 선택

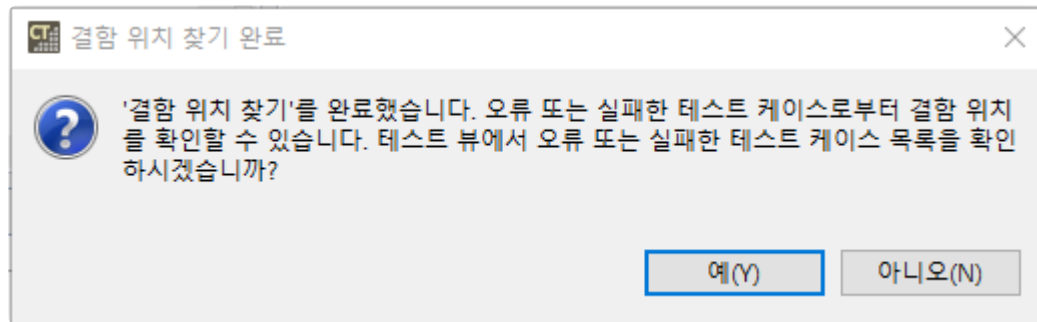


- [테스트 뷰 영역]의 테스트를 우클릭하여, [결함 위치 찾기] 선택
- [테스트 뷰 영역]의 테스트 케이스를 우클릭하여, [결함 위치 찾기] 선택

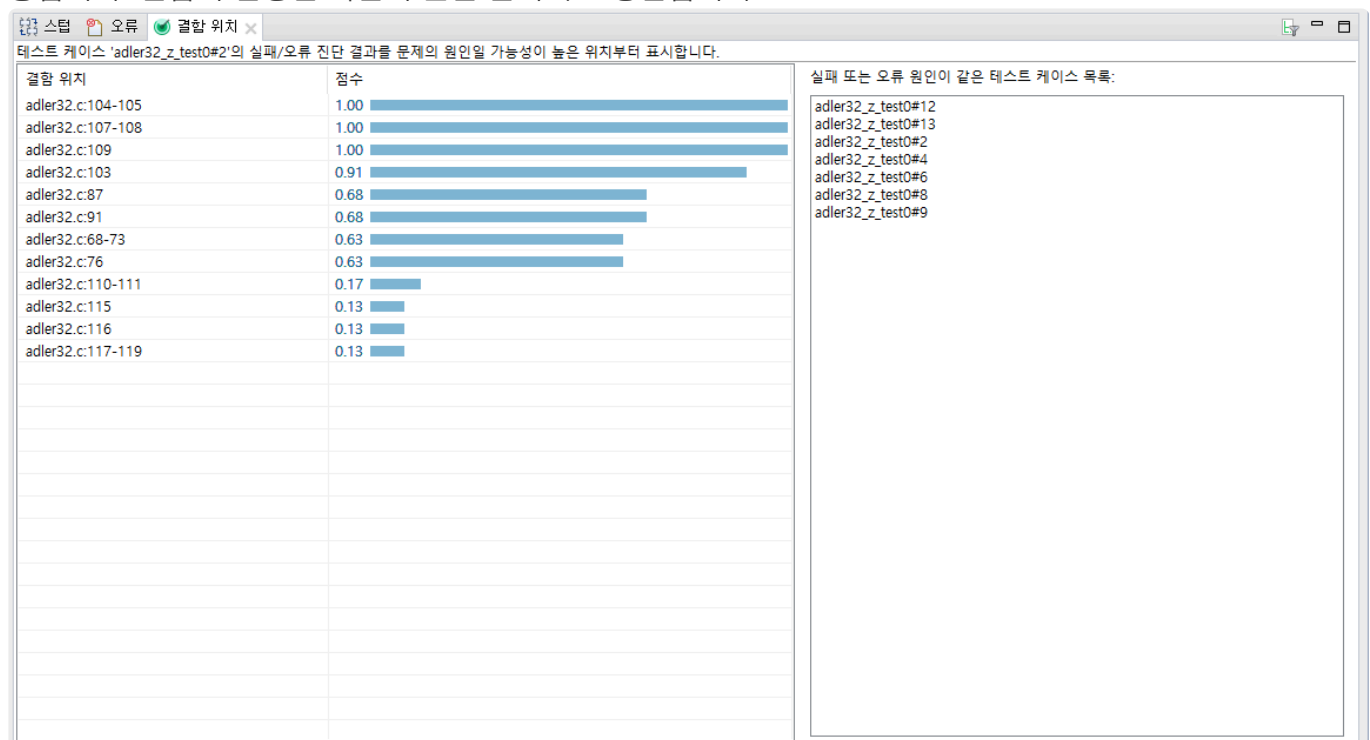
2. [결함 위치 찾기]를 클릭하면 [결함 위치 찾기] 대화상자가 뜹니다. [예]를 누르면 결함 위치 찾기를 진행합니다.



3. 결함 위치 찾기가 완료되면 [결함 위치 찾기 완료] 대화상자가 뜹니다. [예] 버튼을 눌러 오류 또는 실패한 케이스만 모아 볼 수 있습니다.



4. 결함 위치 찾기를 진행한 테스트 케이스를 더블 클릭하면, [결함 위치 뷰]에서 결함 위치에 대한 정보를 제공합니다. 결함이 발생할 확률이 높은 순서대로 정렬됩니다.



## 15. 특성 페이지

---

CT 2023.12는 프로젝트, 모듈, 소스 파일의 특성을 설정할 수 있는 페이지를 제공합니다.

특성 페이지 종류는 다음과 같습니다.

- [프로젝트 특성](#)
- [모듈 특성](#)
- [소스 파일 특성](#)

## 15.1. 프로젝트 특성

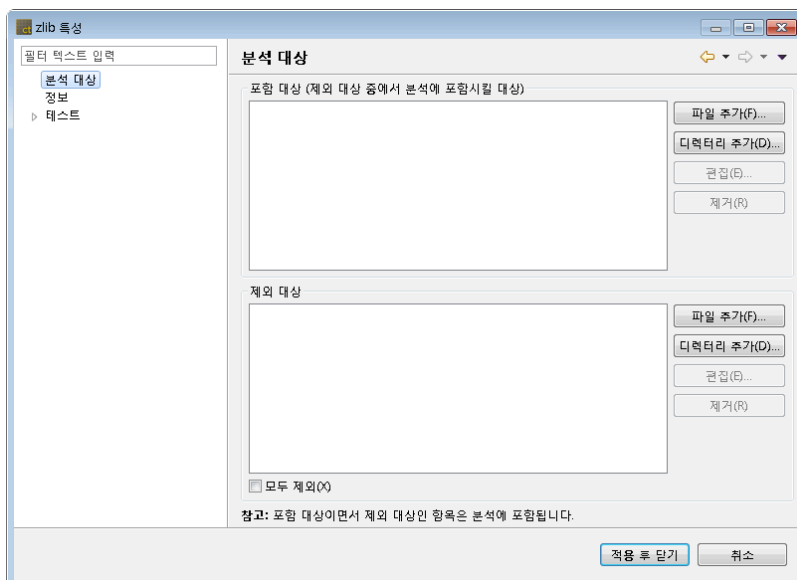
생성된 프로젝트의 특성을 변경할 수 있습니다.

[프로젝트] 메뉴의 [특성]을 선택하거나 [테스트 네비게이터 뷰]에서 프로젝트를 우클릭하여 [특성]을 선택합니다.

### 분석 대상

해당 프로젝트를 분석할 때 제외 대상과 포함 대상을 설정합니다. 포함 대상은 제외 대상 중에서 분석에 포함시킬 대상을 입력합니다.

✿ 선택한 디렉터리 또는 파일만 분석하려면, 분석할 대상을 포함 대상에 추가하고 [모두 제외]를 체크합니다.

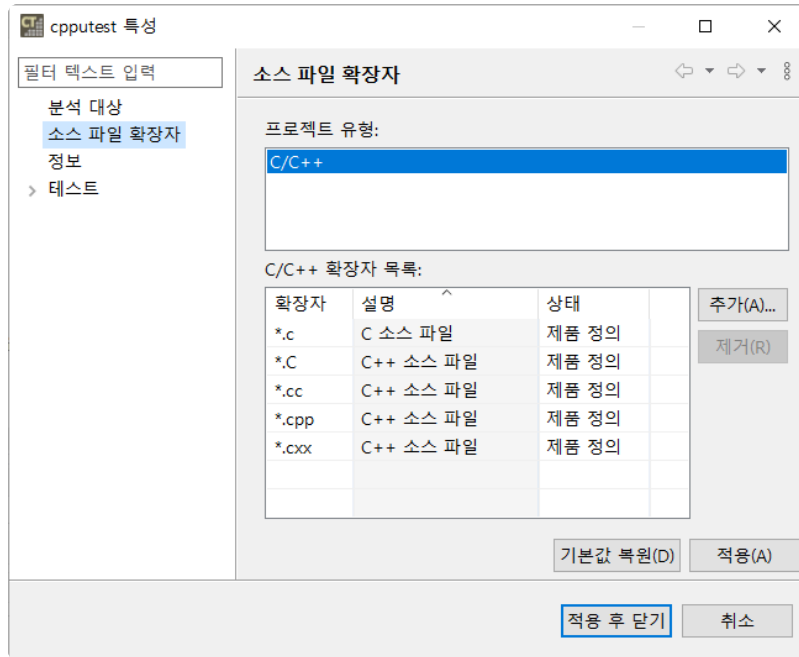


- 분석 포함 대상 추가
  1. [파일 추가] 또는 [디렉터리 추가] 버튼을 클릭합니다.
  2. 분석할 파일 또는 디렉터리를 선택합니다.
  3. [열기] 또는 [확인] 버튼을 클릭합니다.
- 분석 제외 대상 추가
  - [파일 추가] 또는 [디렉터리 추가] 버튼을 클릭합니다.
  - 분석 제외할 파일 또는 디렉터리를 선택합니다.
  - [열기] 또는 [확인] 버튼을 클릭합니다.

### 소스 파일 확장자

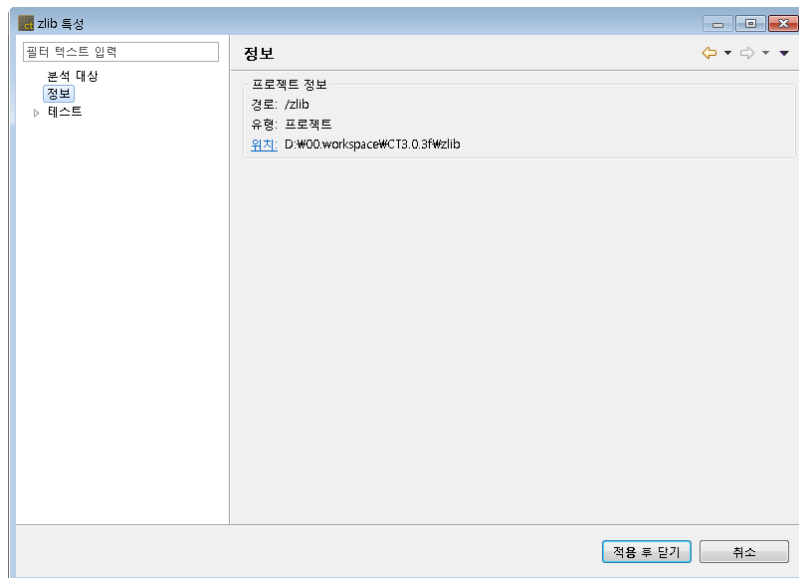
소스 파일 확장자를 정의할 수 있습니다.





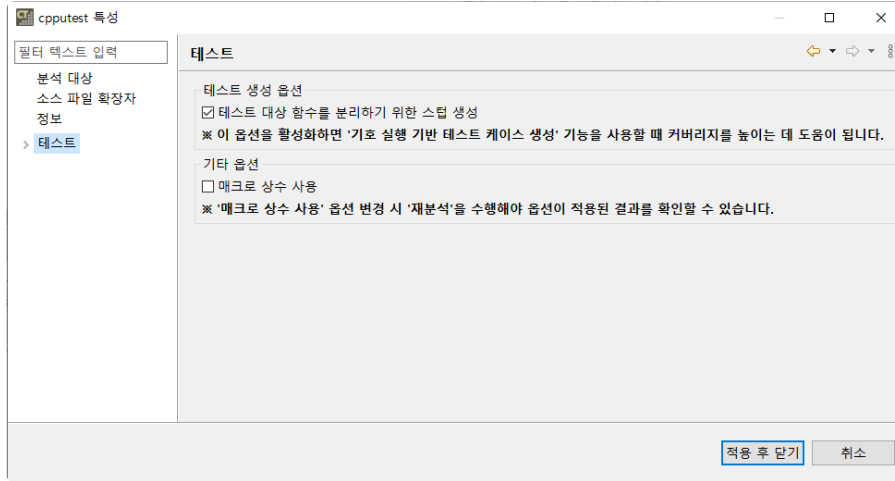
## 정보

프로젝트에 대한 간략한 정보(경로, 유형, 위치)를 보여줍니다.



## 테스트

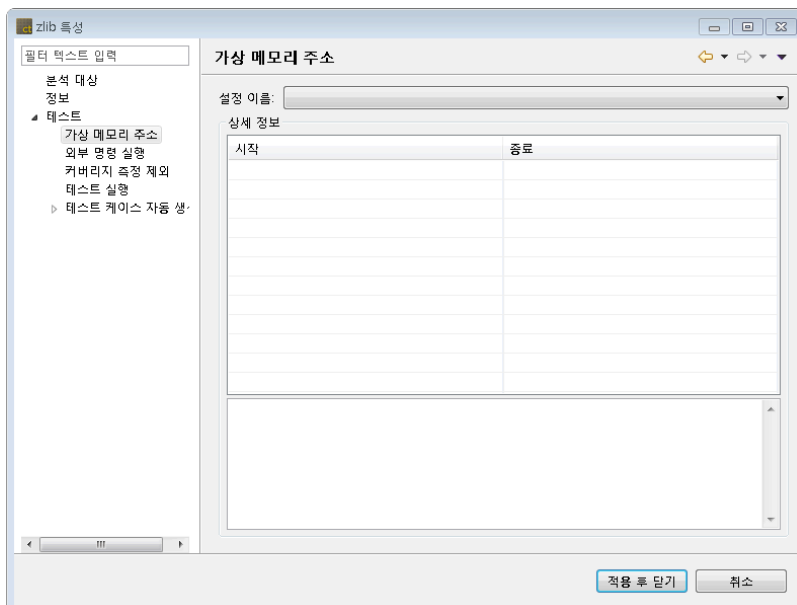
테스트 관련 옵션을 설정할 수 있습니다.



## 가상 메모리 주소

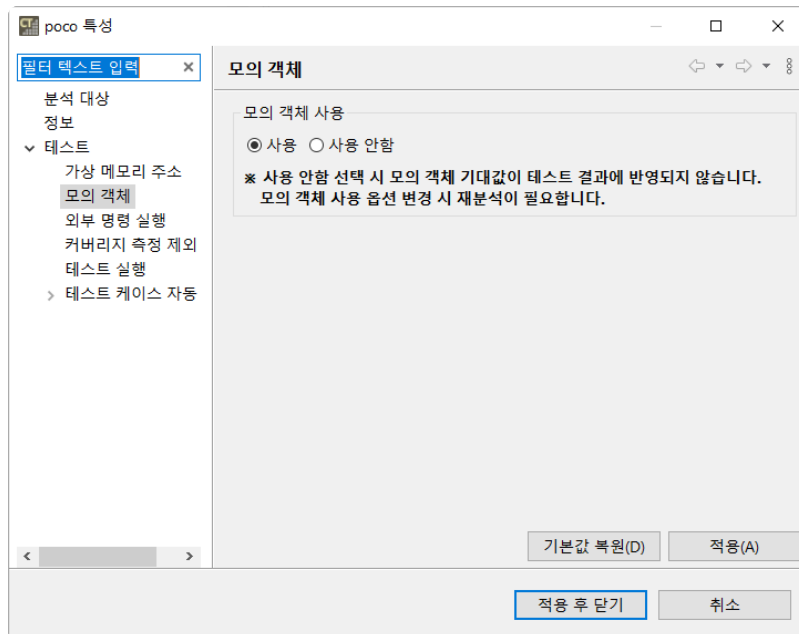
가상 메모리 주소 환경설정에서 관리되는 메모리 설정 정보를 선택할 수 있습니다.

프로젝트의 툴체인이 가상 메모리를 사용할 수 없는 툴체인인 경우(ex. 호스트 툴체인) 경고가 표시됩니다.



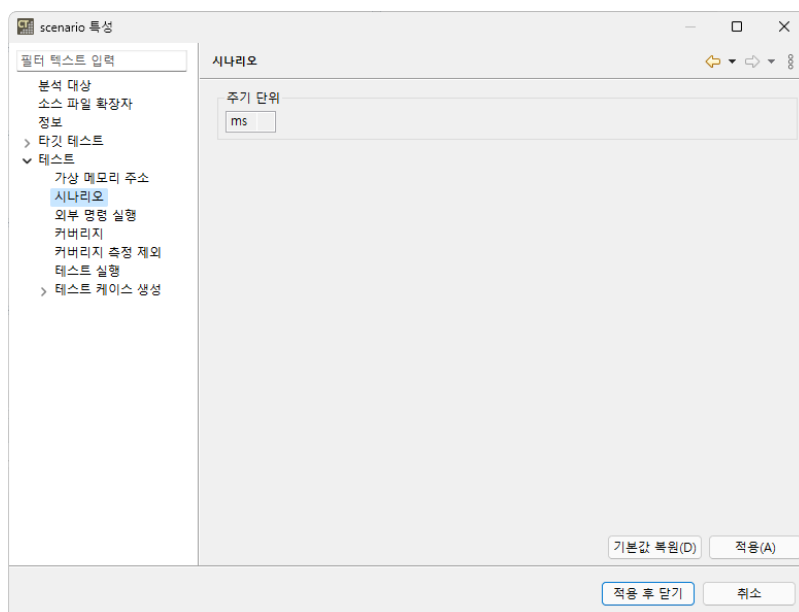
## 모의 객체

C++ 프로젝트인 경우, 모의 객체 사용 여부를 설정합니다. 모의 객체를 사용할 수 있는 툴체인(GCC 6.0 이상, Visual Studio 2015 이상)인 경우 기본으로 [사용]이 선택됩니다.



## 시나리오

C 프로젝트의 시나리오 테스트에서 사용하는 주기 단위의 기본값을 설정할 수 있습니다.

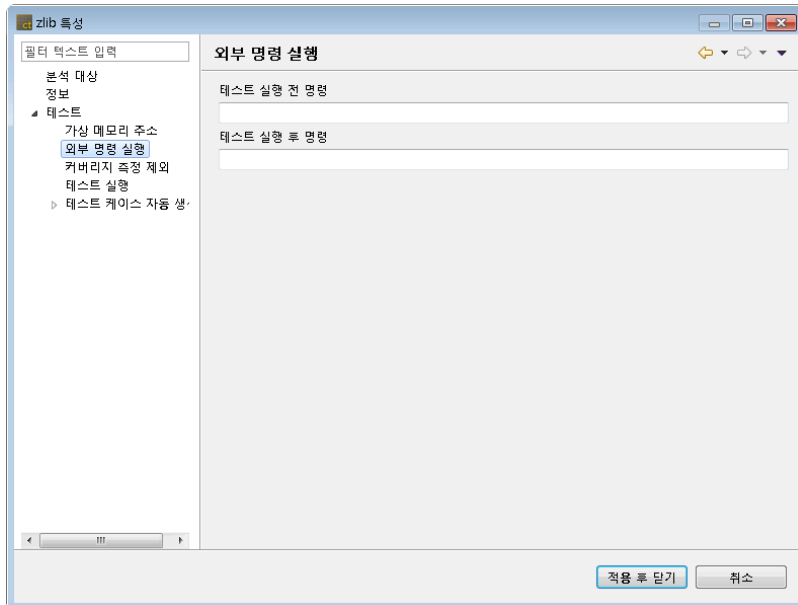


## 외부 명령 실행

테스트 실행 전, 후 외부 명령어를 입력할 수 있습니다.

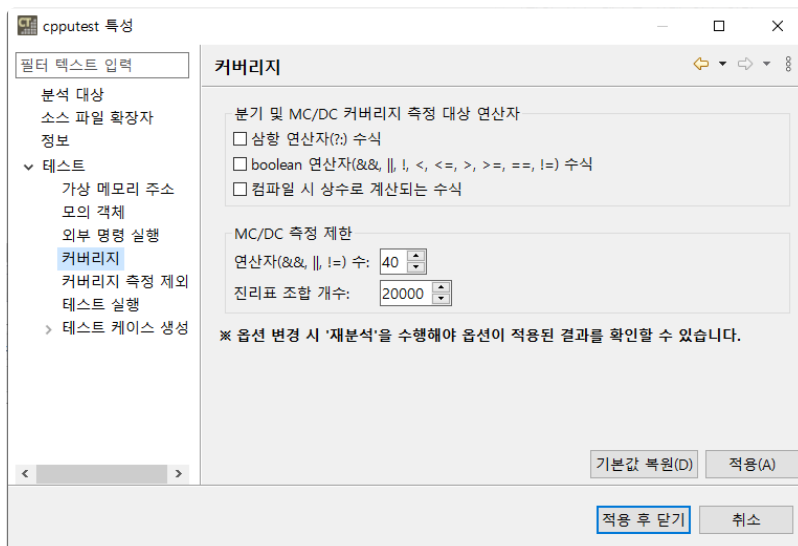
[테스트 실행 전 명령]에는 테스트를 수행하기 전 외부 명령을 수행 할 명령어 또는 배치 파일을 입력할 수 있습니다.

[테스트 실행 후 명령]에는 테스트를 수행한 후 외부 명령을 수행 할 외부명령어 또는 배치 파일을 입력할 수 있습니다.



## 커버리지

커버리지 옵션을 설정할 수 있습니다.

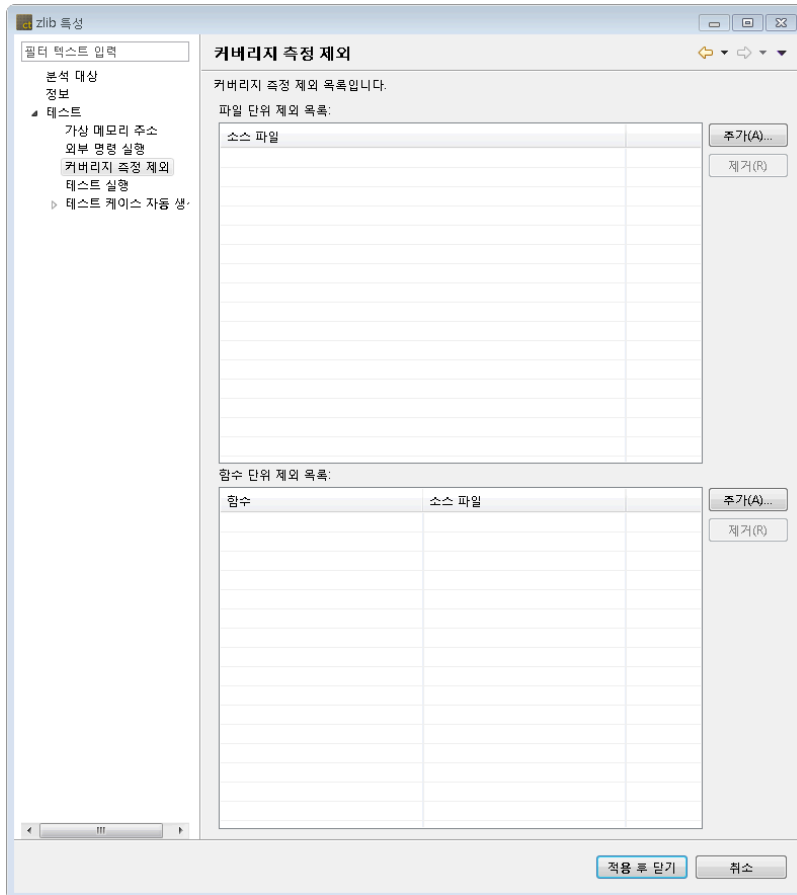


옵션에 대한 자세한 설명은 [환경설정 > 테스트](#) 페이지의 커버리지 탭을 확인하시기 바랍니다.

## 커버리지 측정 제외

파일 단위 또는 함수 단위로 커버리지 제외 항목을 설정할 수 있습니다.

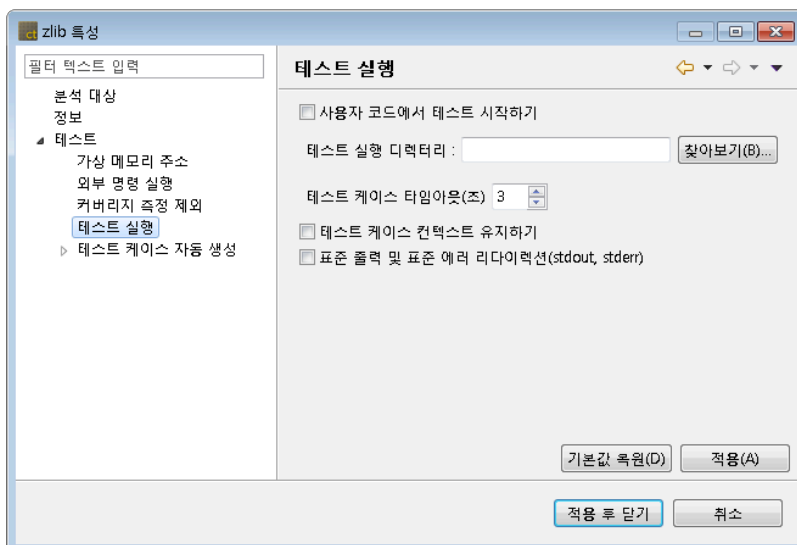
테스트 실행을 통해 제외된 함수의 커버리지 정보가 측정 대상에서 제외되었음을 확인할 수 있습니다.



## 테스트 실행

생성된 테스트를 워크스페이스가 아닌 사용자가 지정한 디렉터리에서 수행합니다. 이 경우 해당 테스트 대상이 사용하는 라이브러리나 헤더파일들을 워크스페이스로 복사하거나 링크 설정을 변경하는 작업을 하지 않아도 됩니다.

[사용자 코드에서 테스트 시작하기]의 체크 박스를 클릭합니다.  
테스트가 수행될 홈 디렉터리를 선택한 후 [적용]버튼을 클릭합니다.



[테스트 케이스 타임아웃] 테스트 대상 함수가 테스트 케이스를 실행할 때 타임 아웃으로 판정하는 기준이 되는 시간을 설정합니다. 테스트 수행 시간이 테스트 케이스 타임아웃 시간을 초과하면 해당 테스트 케이스

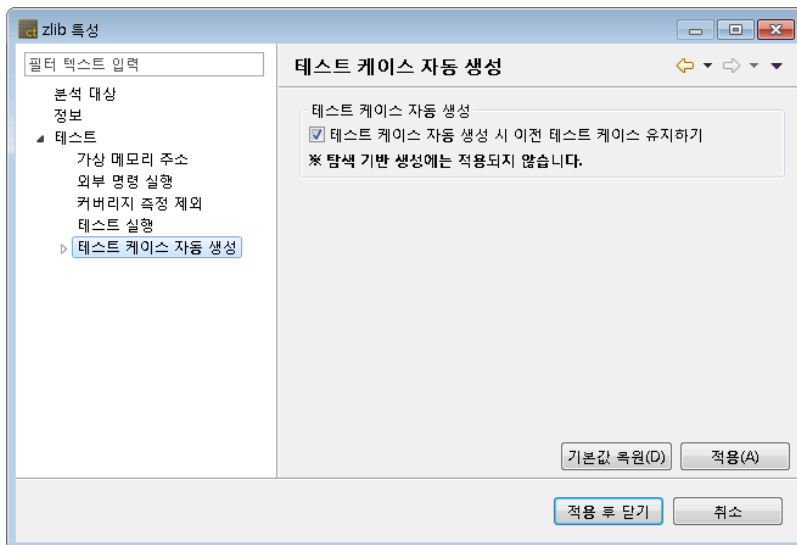
의 테스트는 종료되고 결과에 타임아웃으로 리포트 됩니다.

[테스트 케이스 컨텍스트 유지하기]는 하나의 실행기로 모든 테스트 케이스를 수행하는 기능입니다. 이전 테스트 수행 결과(static, 전역변수)가 테스트 수행 중 유지되며 테스트 케이스 결과값은 독립적으로 나타나지 않습니다. 이로 인해 테스트 케이스 결과값의 선, 후 관계가 분명해집니다.

[표준 출력 및 표준 에러 리다이렉션(stdout, stderr)]는 각 테스트 케이스별 로그파일에 표준 출력과 표준 에러를 기록하는 기능입니다.

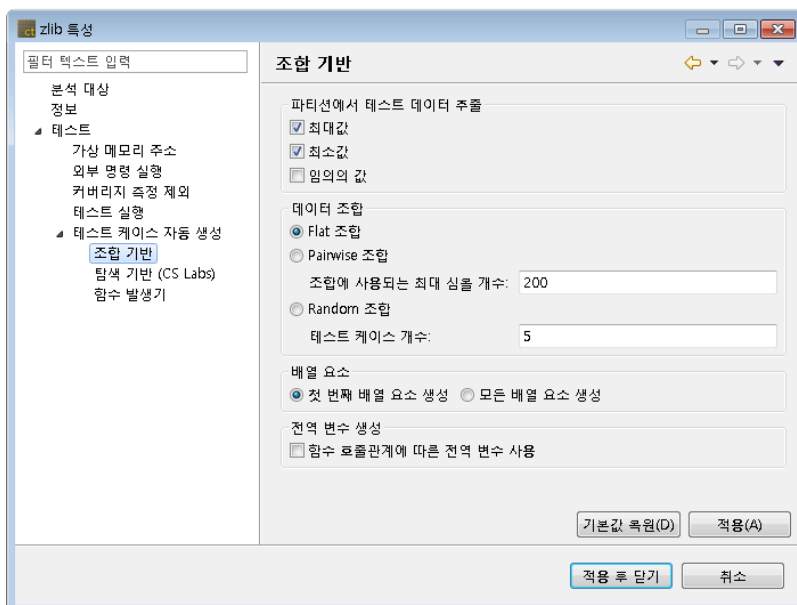
## 테스트 케이스 생성

테스트 케이스 자동 생성 시 이전 테스트 케이스 유지하기를 설정할 수 있습니다.



## 조합 기반

변수 파티션이 범위일 경우 테스트 데이터를 추출하는 방법을 설정할 수 있습니다.



체크박스를 체크하여 원하는 데이터의 추출 방법을 선택합니다. 이때 각 값의 의미는 다음과 같습니다.

추출 방법	설명
최대값	파티션 구간의 최대값 선택
최소값	파티션 구간의 최소값 선택
임의의 값	파티션 구간 중 임의의 값 선택

데이터 추출의 방법은 복수 선택이 가능합니다.

모두 체크 해제 시 기본으로 최대값, 최소값으로 자동 설정됩니다.

데이터 조합은 Flat 조합, Pairwise 조합과 Random 조합을 지원합니다. 조합의 의미는 다음 표와 같습니다.

값	설명
Flat	테스트 데이터 개수가 가장 많은 변수를 기준으로 단순 조합
Pairwise	최소의 개수로 서로소가 되도록 테스트 케이스를 조합하여 제한적이지만 최대의 커버리지 달성이 가능한 방식 조합에 사용되는 최대 심볼 개수 만큼 Pairwise로 수행하고 심볼 개수를 초과하면 Flat으로 생성(기본 200)
Random	입력 파라미터에 해당하는 변수 파티션의 최소값과 최대값 사이의 임의의 값으로 사용자가 입력한 테스트 케이스의 개수만큼 테스트 데이터를 조합(기본 5)

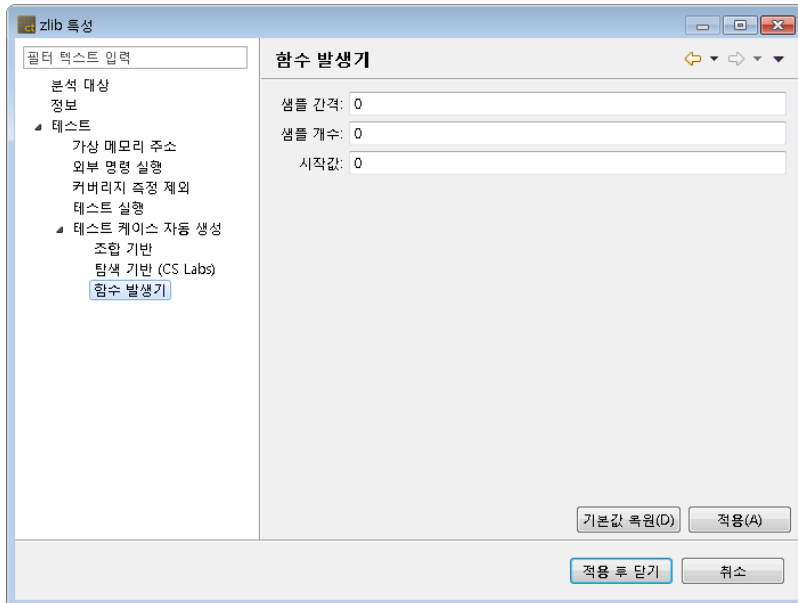
배열 요소는 각 배열 요소에 대한 테스트 데이터의 생성 여부를 선택하는 항목입니다.

값	설명
첫 번째 배열 요소 생성	배열 사이즈에 상관없이 첫 번째 배열 요소만 생성
모든 배열 요소 생성	배열 사이즈만큼 배열 요소 생성

전역변수 생성은 테스트를 생성할 때 테스트 대상 함수로부터 호출되는 모든 함수들에서 사용하는 전역 변수를 모두 생성하는 옵션입니다.

## 함수 발생기

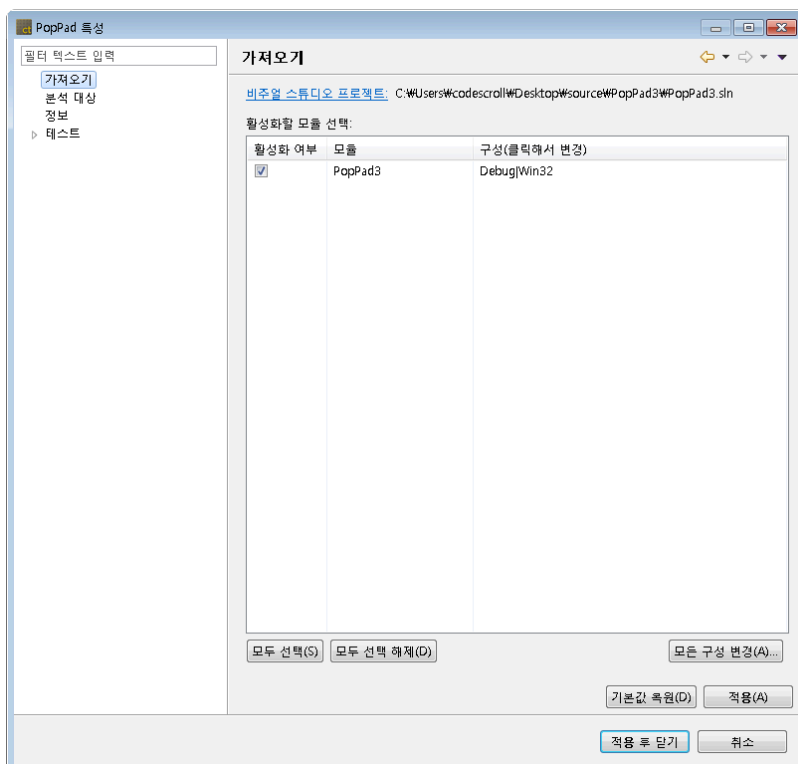
함수 발생기를 사용하여 테스트 케이스를 자동으로 생성하는 경우에 각 함수가 공통으로 가지는 공통 설정 값을 지정합니다.



설정	설명
샘플 간격	함수로부터 샘플링할 샘플의 간격
샘플 개수	함수로부터 샘플링할 샘플의 개수 (테스트 케이스 개수)
시작 값	함수가 시작되는 기본 값으로 해당 값을 기준으로 값을 생성

## 가져오기 설정

비주얼 스튜디오 프로젝트로 생성한 프로젝트



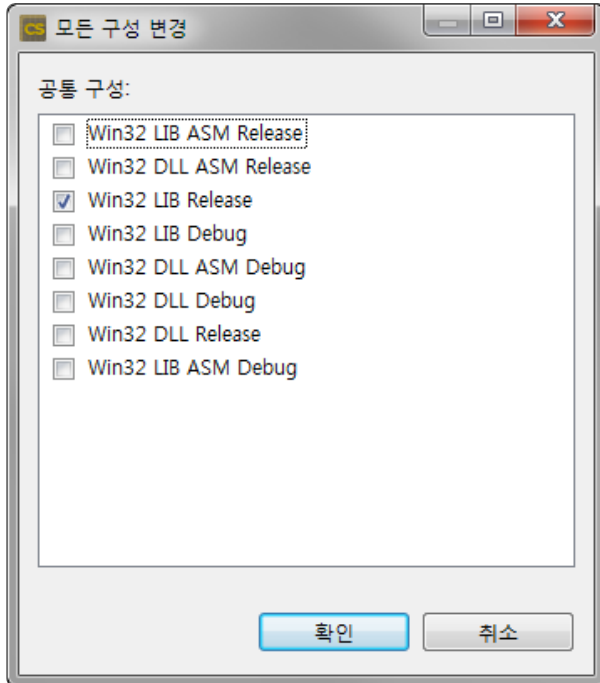


프로젝트를 생성하는데 사용한 비주얼 스튜디오 프로젝트의 경로와 선택된 프로젝트에서 활성화된 모듈을 확인할 수 있습니다.

각 모듈의 활성화 여부와 구성을 변경할 수 있습니다.

[모든 구성 변경] 메뉴를 통해 모든 모듈의 구성을 일괄 변경할 수 있습니다.

1. [모든 구성 변경] 버튼을 클릭하면, 모든 모듈에 공통으로 포함된 구성들 중에서 한가지를 선택할 수 있는 대화 상자가 표시됩니다.

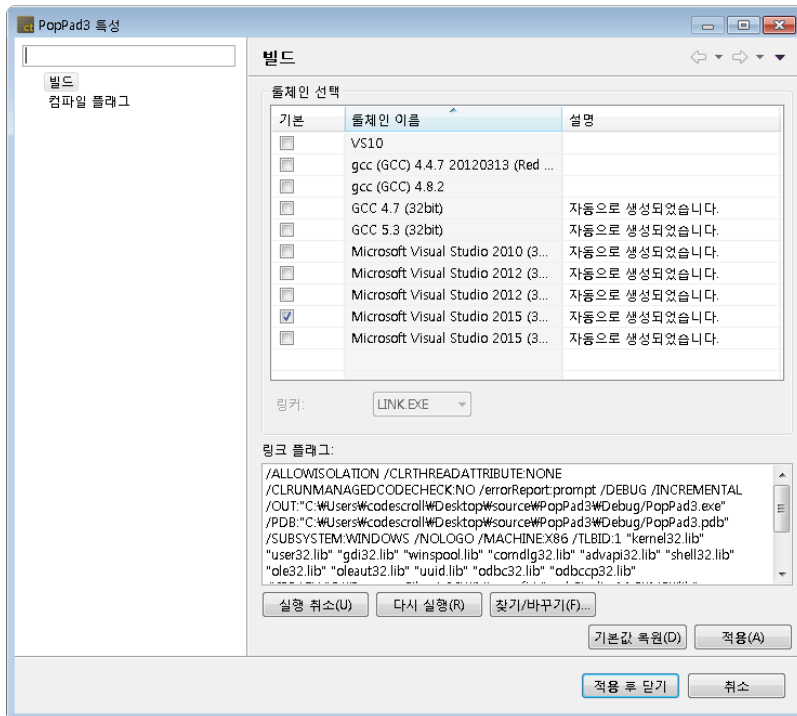


2. 모든 모듈에 공통으로 적용할 구성을 선택한 뒤 [확인] 버튼을 클릭합니다.

## 15.2. 모듈 특성

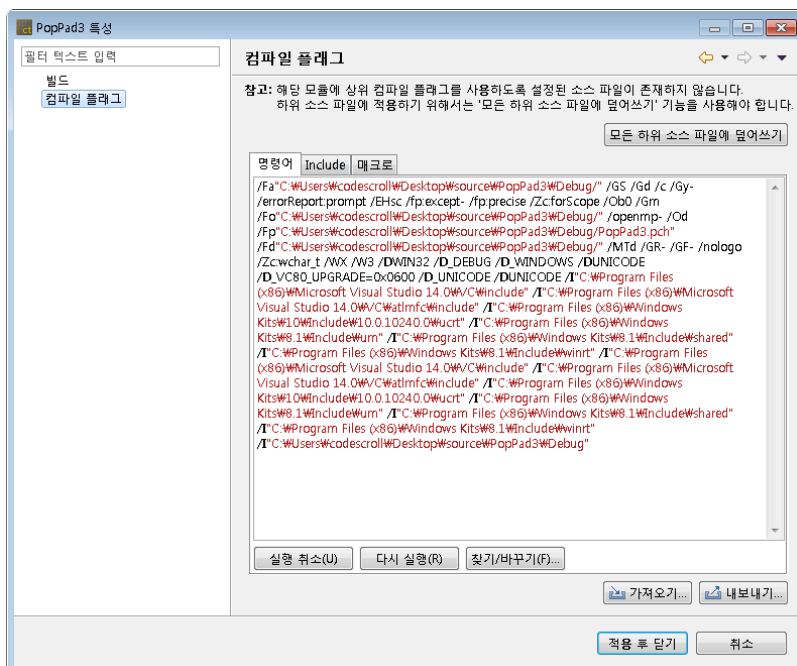
### 빌드

모듈에 대한 툴체인 관련 정보(툴체인, 링커)와 링크 플래그를 설정합니다. 툴체인을 변경하면 모듈 하위에 있는 모든 소스 파일에 설정된 툴체인도 변경됩니다.



### 컴파일 플래그 설정

해당 모듈에 대한 컴파일 플래그 정보를 설정합니다.

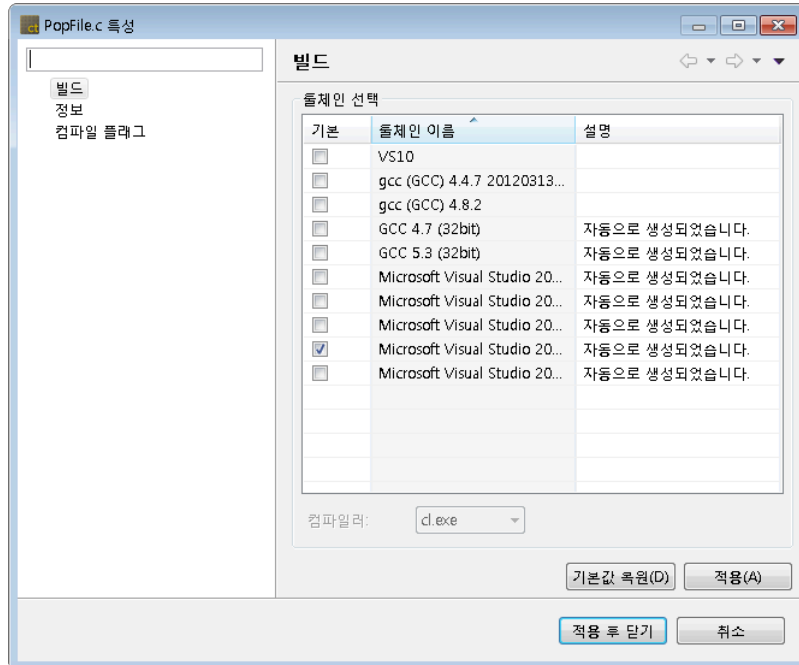


- [명령어]탭에 컴파일 플래그를 입력합니다. [Include] 탭과 [매크로] 탭을 이용하면 컴파일 플래그를 쉽게 입력할 수 있습니다.
- [모든 하위 소스 파일에 덮어쓰기]는 입력된 컴파일 플래그 정보가 해당 모듈 하위에 있는 모든 소스 파일의 컴파일 플래그 정보를 덮어씁니다.
- [가져오기]는 외부에 있는 컴파일 플래그 정보를 가져옵니다. [내보내기]는 현재 컴파일 플래그 정보를 외부에 저장합니다.
  - 가져오기
    1. [가져오기] 버튼을 클릭합니다.
    2. 파일열기 다이얼로그로부터 컴파일 플래그 정보가 입력된 파일(.cf 파일)을 선택합니다.
  - 내보내기
    1. [내보내기] 버튼을 클릭합니다.
    2. 파일저장 다이얼로그로부터 컴파일 플래그 정보가 입력될 위치와 파일명을 입력 후 저장합니다.

## 15.3. 소스 파일 특성

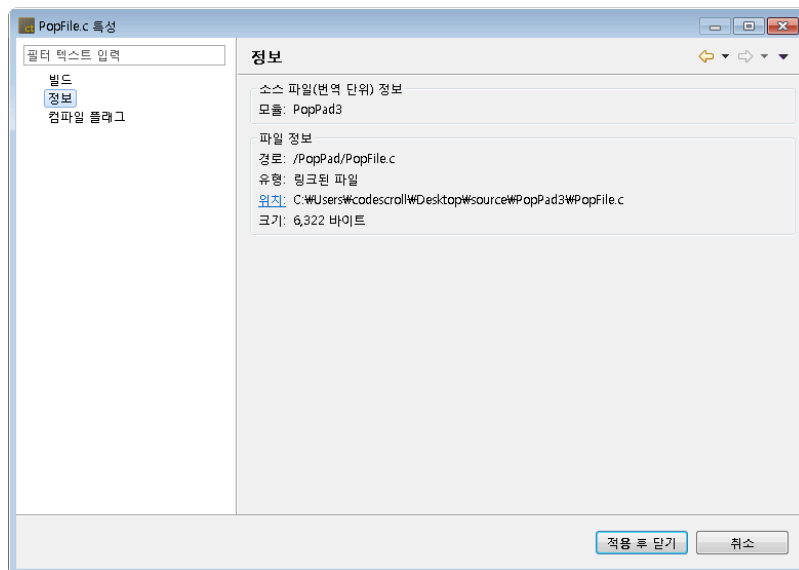
### 빌드

소스 파일에 대한 툴체인 관련 정보(툴체인, 컴파일러)를 설정합니다.



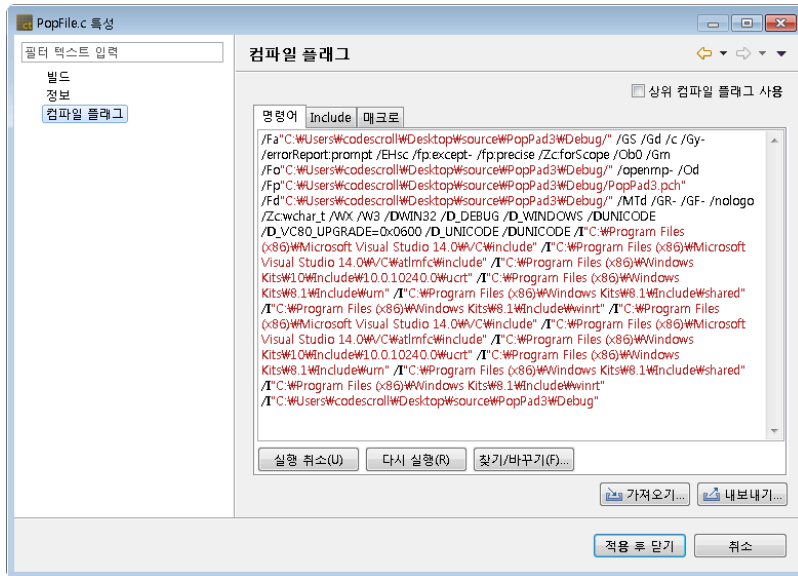
### 정보

소스 파일이 포함된 모듈 정보와 파일 정보를 보여줍니다.



### 컴파일 플래그

소스 파일에 대한 컴파일 플래그 정보를 설정합니다.



- [상위 컴파일 플래그 사용] 체크버튼을 선택하면 컴파일 플래그 정보를 변경할 수 없고, 상위 모듈의 컴파일 플래그 정보로 덮어쓰게 됩니다. 반대로 선택해제 하면 현재 컴파일 플래그 정보를 변경할 수 있습니다.
- [가져오기]는 외부에 있는 컴파일 플래그 정보를 가져옵니다. [내보내기]는 현재 컴파일 플래그 정보를 외부에 저장합니다.
  - 가져오기
    1. [가져오기] 버튼을 클릭합니다.
    2. 파일열기 다이얼로그로부터 컴파일 플래그 정보가 입력된 파일(.cf 파일)을 선택합니다.
  - 내보내기
    1. [내보내기] 버튼을 클릭합니다.
    2. 파일저장 다이얼로그로부터 컴파일 플래그 정보가 입력될 위치와 파일명을 입력 후 저장합니다.

## 16. 환경설정

---

환경 설정 메뉴에서는 도구 전체에 적용되고 있는 현재 설정을 확인하거나 변경할 수 있습니다.

CT 2023.12가 제공하는 환경설정의 종류는 다음과 같습니다.

- [분석](#)
- [분석 제외 대상](#)
- [성능](#)
- [소스 파일 확장자 설정](#)
- [언어](#)
- [테스트](#)
- [툴체인 설정](#)
- [편집기](#)

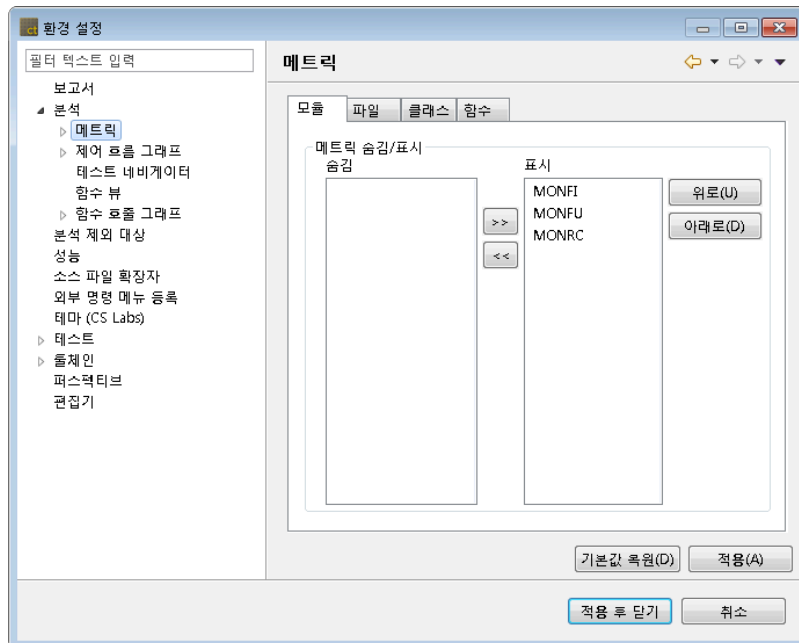
## 16.1. 분석

분석 정보에 관련된 설정을 확인하거나 변경할 수 있습니다.

### 메트릭

메트릭 데이터를 보여주는 모든 뷰에서 어떤 메트릭을 보이거나 숨길지 여부와 보이는 순서를 설정할 수 있습니다.

툴팁 메시지로 선택한 메트릭에 대한 설명을 볼 수 있습니다.

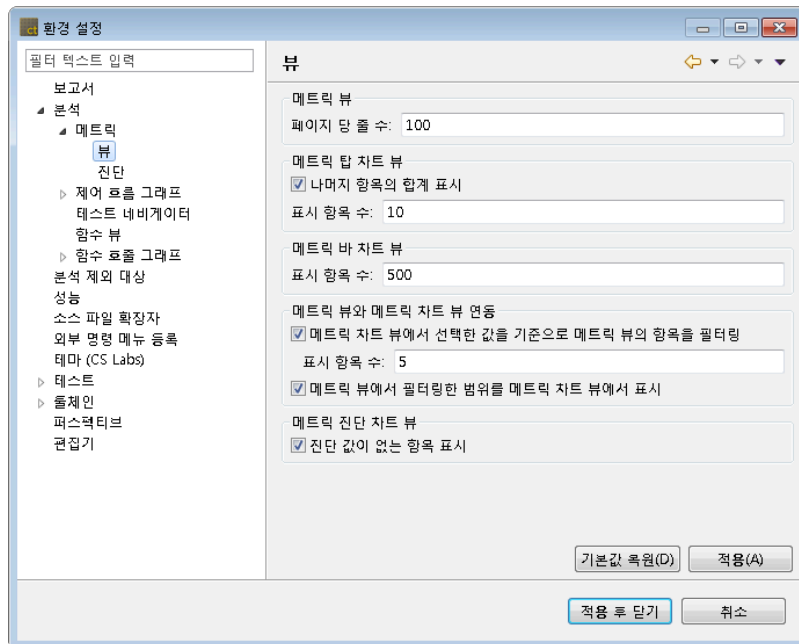


메트릭을 선택한 뒤 [ >> ] 또는 [ << ] 버튼으로 숨김 목록이나 표시 목록으로 이동할 수 있습니다.

표시 목록에 있는 메트릭을 선택한 뒤 [위로] 또는 [아래로] 버튼으로 메트릭이 보이는 순서를 변경할 수 있습니다.

### 메트릭 뷰

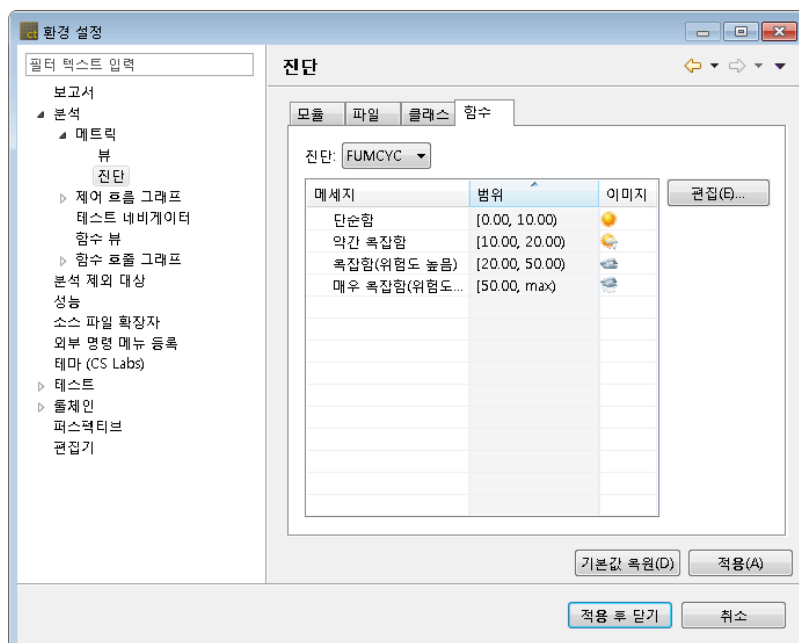
메트릭 데이터를 보여주는 뷰들의 설정을 변경할 수 있습니다.



## 진단

메트릭에 대한 진단 정보를 확인하거나 설정할 수 있습니다.

선택된 메트릭 값이 각각의 범위에 포함될 경우에 대한 진단 메시지와 진단 이미지(메트릭 뷰에 표시)를 보여줍니다.



진단 정보를 수정하려면 [편집] 버튼을 클릭합니다.

선택한 메트릭의 진단 정보를 편집할 수 있는 대화상자가 나타납니다.



진단 편집 'FUCYC'

진단 단계: 4

이미지 설정: Sun4 정렬 순서 변경(C)

단계1

메세지: 단순함

범위: 0.00 ~ 10.00 이미지: ●

단계2

메세지: 약간 복잡함

범위: 10.00 ~ 20.00 이미지: ●

단계3

메세지: 복잡함(위험도 높음)

범위: 20.00 ~ 50.00 이미지: ●

단계4

메세지: 매우 복잡함(위험도 매우 높음)

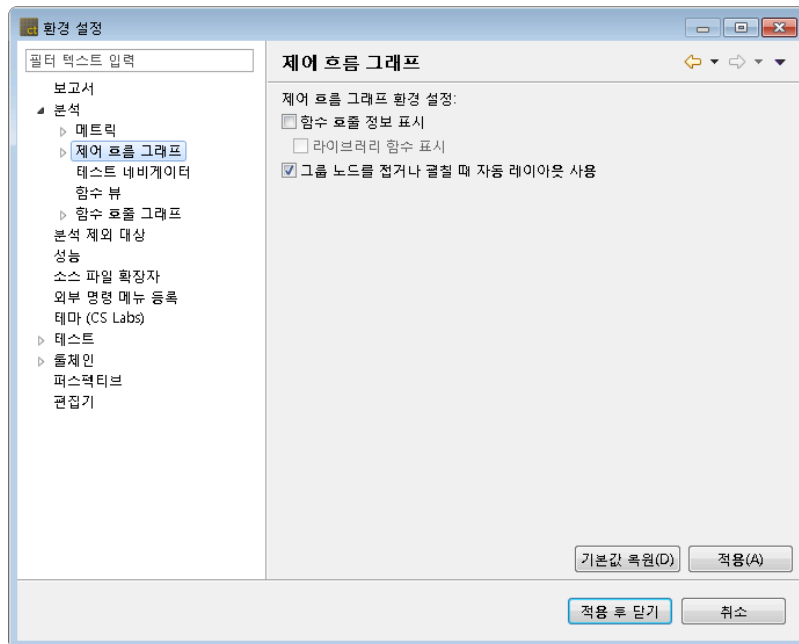
범위: 50.00 ~ 최대 이미지: ●

확인 취소

1. 진단 단계(2~5)를 설정합니다.
2. 메트릭 뷰에서 보여줄 진단 이미지를 설정합니다.
3. [정렬 순서 변경] 버튼을 누르면 진단 이미지의 정렬 순서가 변경됩니다.
4. 각 단계별로 진단 메시지와 진단 범위를 입력합니다.
5. [확인] 버튼을 클릭합니다.

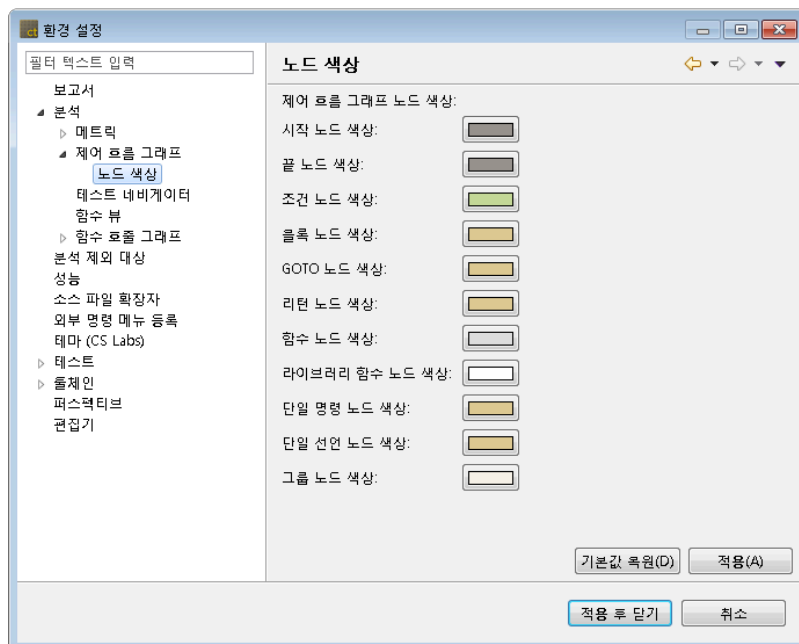
## 제어 흐름 그래프

제어 흐름 그래프 뷰에서 보이는 제어 흐름 그래프에 대한 설정을 변경할 수 있습니다.



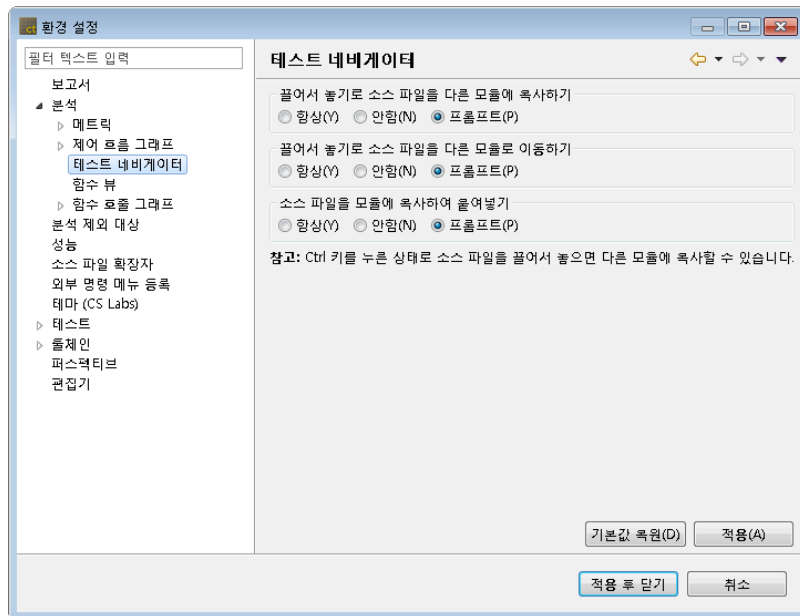
## 노드 색상

제어 흐름 그래프에 보이는 노드의 색상을 변경할 수 있습니다.

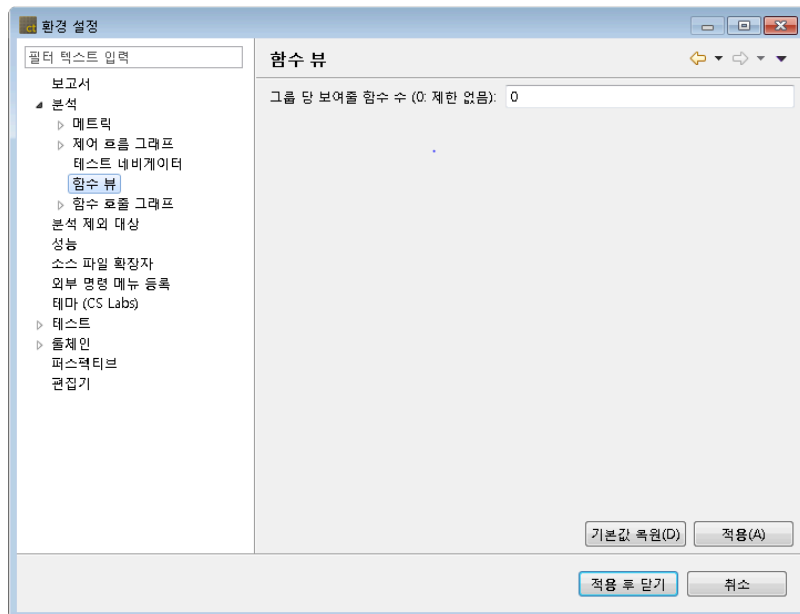


## 테스트 네비게이터

테스트 네비게이터 뷰에서 소스 파일을 복사하거나 이동하는 각각의 작업에 대해, 계속할지를 항상 묻거나 묻지 않도록 설정할 수 있습니다.

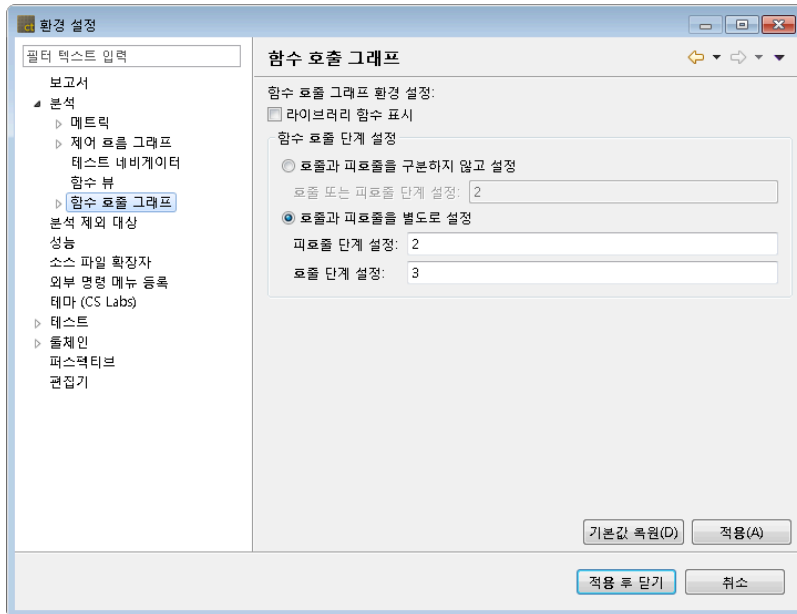


## 함수 뷰



## 함수 호출 그래프

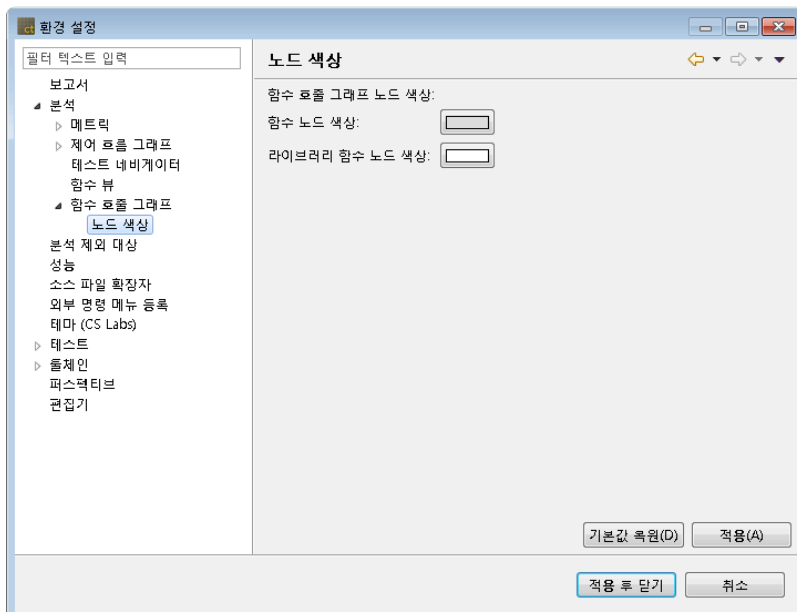
함수 호출 그래프 뷰에서 보이는 함수 호출 그래프의 설정을 변경할 수 있습니다.



[라이브러리 함수 표시] 체크박스는 라이브러리 함수를 표시할 것인지 아닌지를 설정할 수 있습니다.

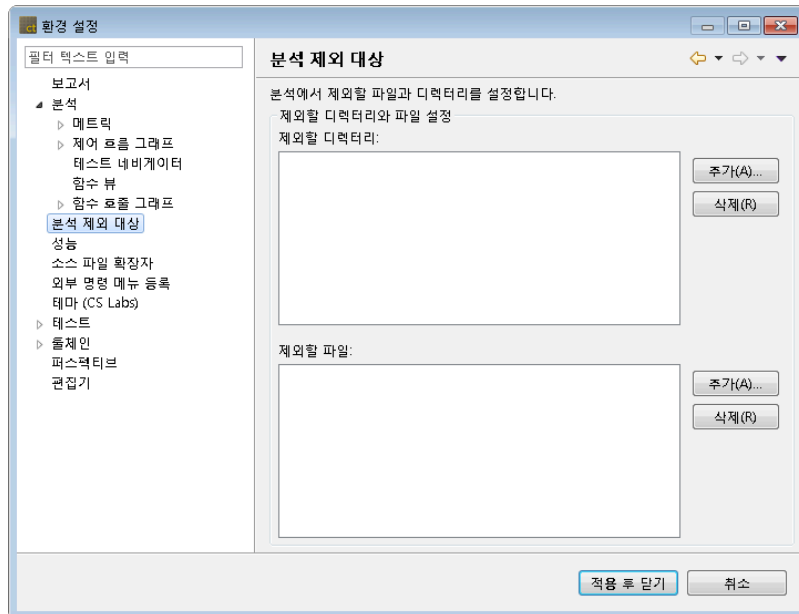
함수 호출 단계 설정에서 '호출과 피호출을 구분하지 않고 설정'의 값을 '2'로 설정하면 선택한 함수를 중심으로 호출과 피호출 함수 모두에 대해 한 단계의 호출 정보를 보여줍니다(선택한 함수를 호출하거나 선택한 함수가 호출한 함수까지 보여짐).

만일, 호출과 피호출을 별도로 설정하고자 한다면 하위 옵션을 선택하여 각각의 단계 값을 지정할 수 있습니다. 함수 호출 그래프에 보이는 노드의 색상을 변경할 수 있습니다.



## 16.2. 분석 제외 대상

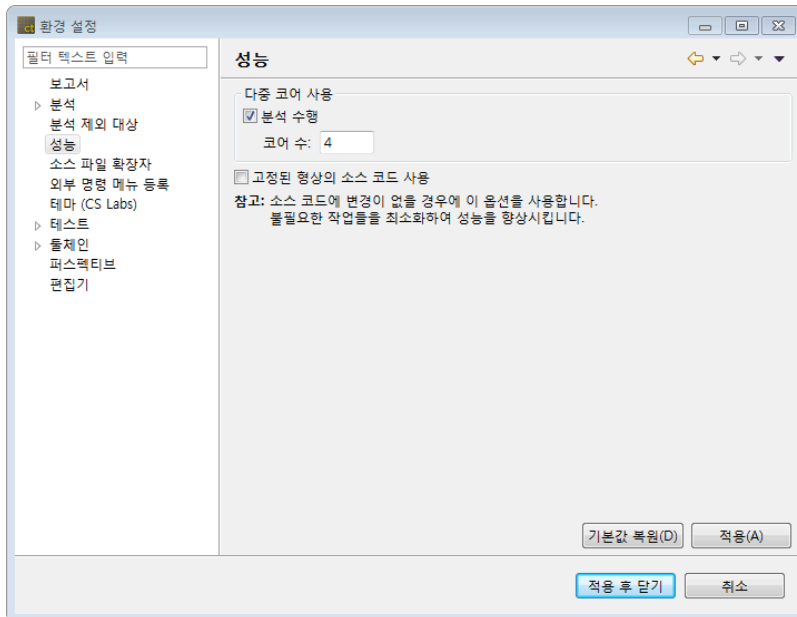
분석에서 제외할 디렉터리와 파일을 설정합니다.



- 분석 제외 대상 추가
  1. [추가] 버튼을 클릭합니다.
  2. 제외할 디렉터리 또는 파일을 선택합니다.
  3. [확인] 또는 [열기] 버튼을 클릭합니다.
- 분석 제외 대상 삭제
  1. 삭제할 디렉터리 또는 파일을 선택합니다.
  2. [삭제] 버튼을 클릭합니다.

## 16.3. 성능

사용자 환경에 맞게 작업 성능을 향상시킬 수 있는 설정을 제공합니다.



### 다중 코어 사용

[다중 코어 사용] 옵션은 프로젝트 분석을 수행할 때 다중 코어를 사용하여 성능을 향상시킵니다(CPU 자원이 충분하지 않은 PC에서는 다른 작업의 성능에 영향을 미칠 수 있습니다).

- 분석 수행: 소스 코드를 분석할 때 사용할 코어 수를 입력합니다(입력할 수 있는 코어 수: 2개 ~ 실행 환경의 최대 코어 수).

### 고정된 형상의 소스 코드 사용

[고정된 형상의 소스 코드 사용] 옵션은 소스 코드 형상에 변경이 없다고 가정하고, 불필요한 작업(소스 코드 수정 감지 및 재분석 등)을 최소화하여 작업 성능을 향상시킵니다.

PC 성능이 매우 느리고, 테스트 수행 중 소스 코드 형상이 절대로 변경되지 않는 경우에만 이 옵션을 사용하기 바랍니다(옵션이 활성화된 상태에서 소스 코드 또는 분석 설정이 변경되면, 예상하지 못한 문제가 발생할 수 있습니다).

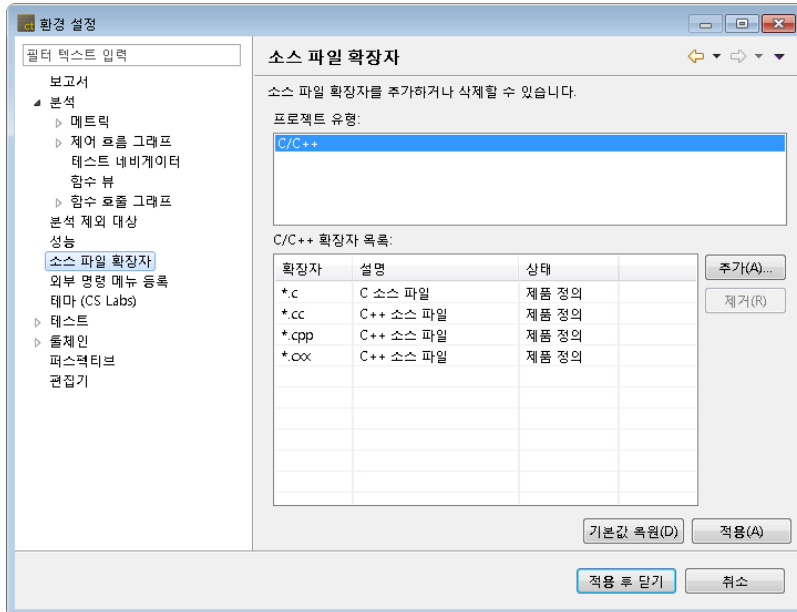


#### 편집기 - [수정 감지] 옵션과 차이점

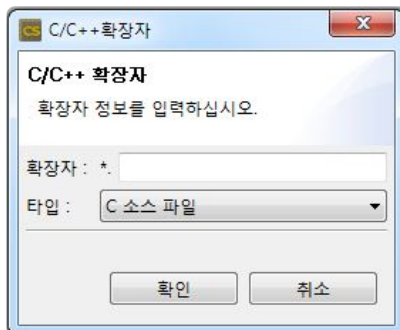
편집기의 [수정 감지] 옵션을 활성화하는 경우 [분석]을 수행하면 수정된 소스 코드로 분석하지만, 분석의 [고정된 형상의 소스 코드 사용] 옵션을 활성화하면 소스 코드를 수정해도 [재분석]하기 전까지 수정된 소스 코드를 분석하지 않습니다. 또한 편집기 - [수정 감지] 옵션은 무시됩니다.

## 16.4. 소스 파일 확장자 설정

프로젝트 유형별로 소스 파일로 사용할 확장자를 설정합니다.



- 소스 파일 확장자 추가
  1. 프로젝트 유형을 선택합니다.
  2. [추가] 버튼을 클릭합니다.
  3. 확장자를 입력하고 타입을 선택한 후 [확인] 버튼을 클릭합니다.



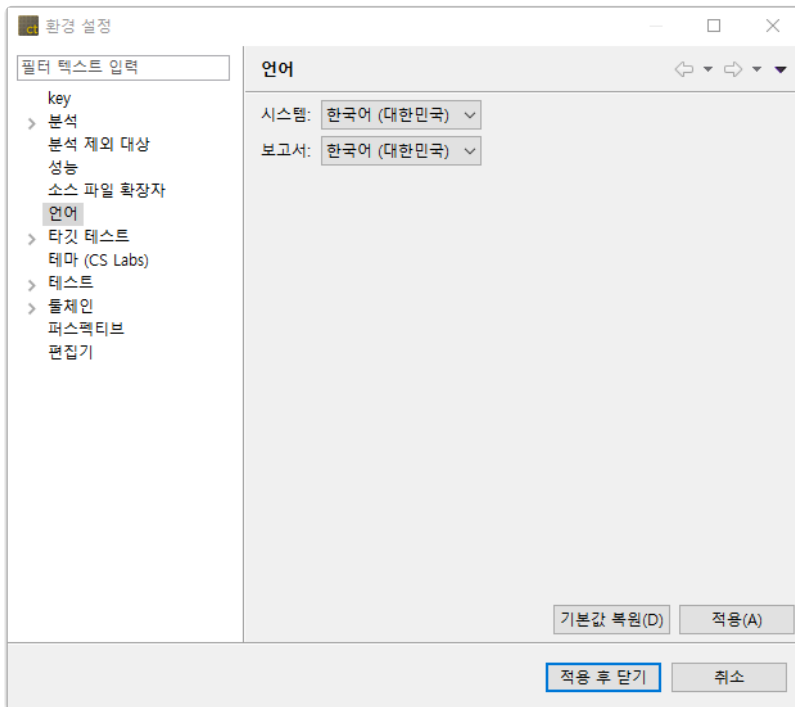
- 소스 파일 확장자 삭제
  1. 프로젝트 유형을 선택합니다.
  2. 삭제할 확장자를 선택합니다.
  3. [삭제] 버튼을 클릭합니다.

✿ 소스 파일 확장자 설정은 프로젝트 설정을 우선하여 적용됩니다.

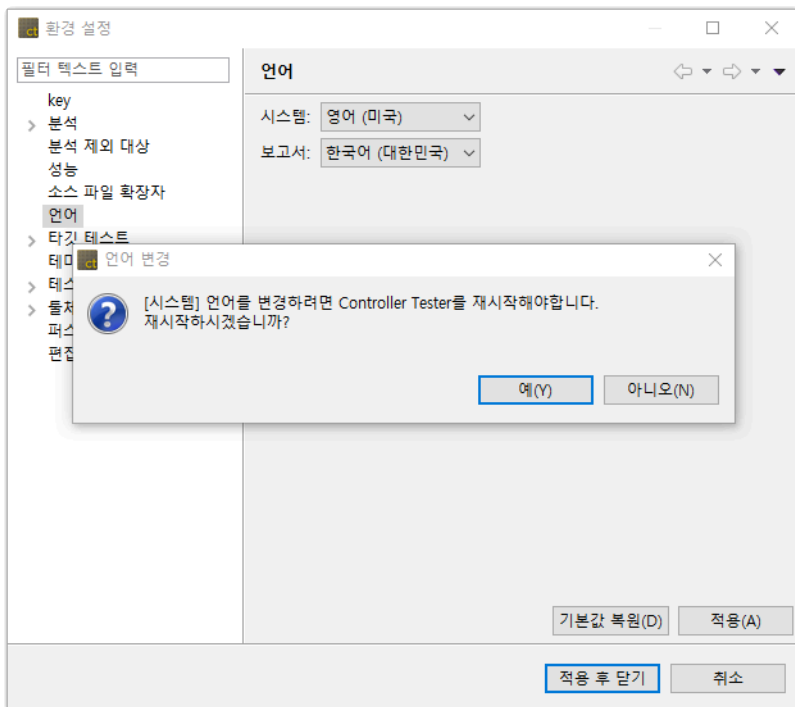
## 16.5. 언어

도구와 보고서에 표시할 언어를 설정합니다.

[시스템], [보고서] 각각의 드롭다운 메뉴에서 언어를 선택할 수 있습니다.



언어 변경 후 [적용] 버튼을 클릭하면 CT 2023.12을 재시작을 묻는 대화창이 열립니다.



✿ 시스템 언어를 변경하면 보고서 언어도 자동으로 변경됩니다. 보고서 언어를 시스템 언어와 다르게 설정하려면, 재시작 후에 보고서 언어를 변경하시기 바랍니다.

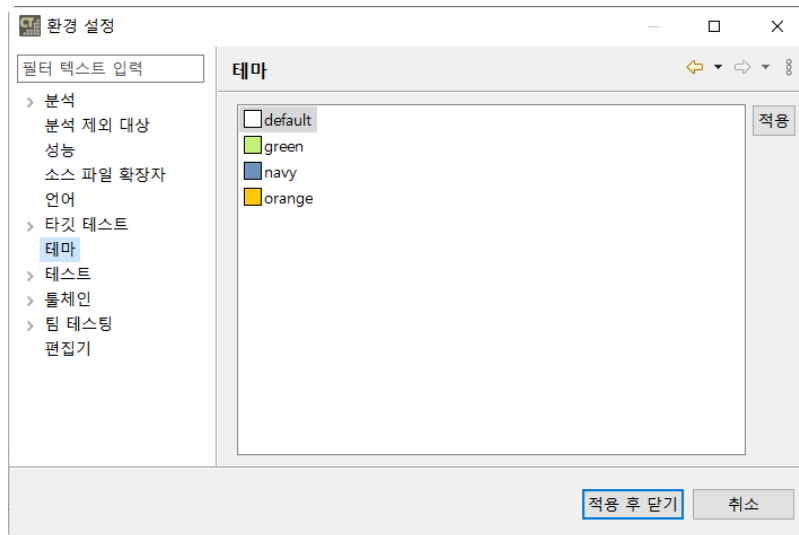


## 16.6. 테마

CT 2023.12의 테마를 변경할 수 있는 설정을 제공합니다.

네 가지 옵션 중 하나를 선택하고 [적용]을 누릅니다.

테마를 적용하려면 CT 2023.12를 재실행해야 합니다.

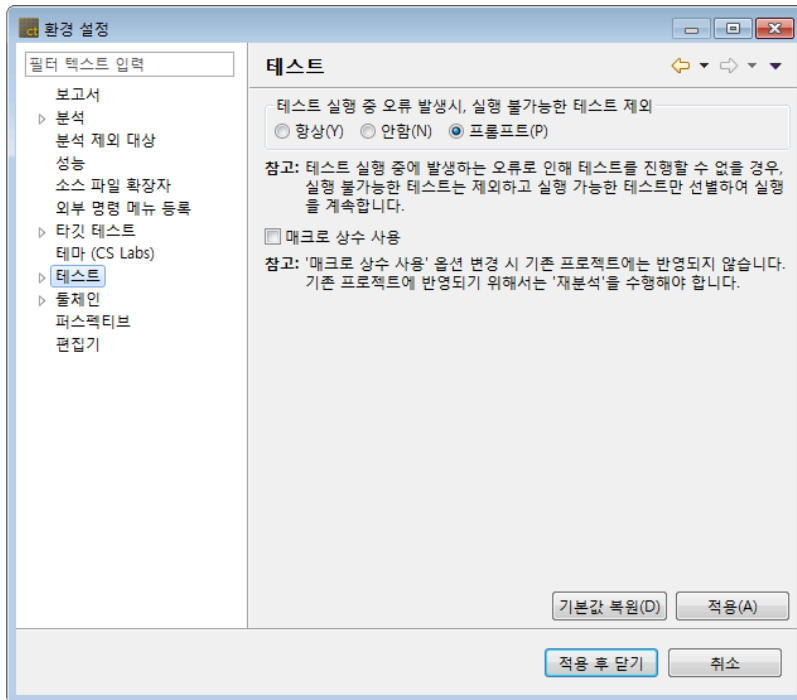


## 16.7. 테스트

테스트 편집 및 실행 관련 설정을 제공합니다.

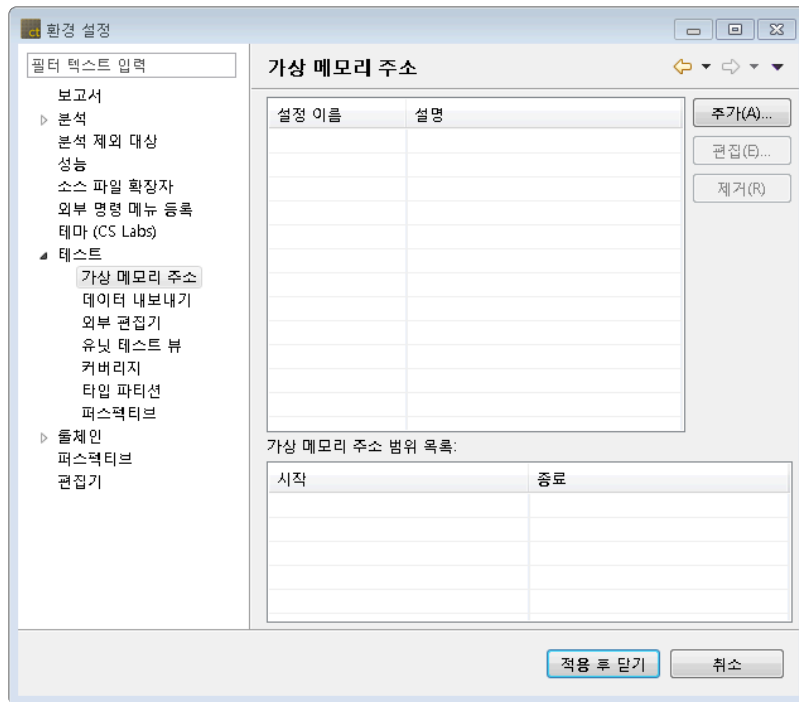
[테스트 실행 중 오류 발생시, 실행 불가능한 테스트 제외] 옵션은 테스트 실행 중에 발생하는 오류(예: 테스트 대상 소스 코드가 변경된 경우)로 인해 테스트를 진행할 수 없을 경우, 실행 불가능한 테스트는 제외하고 실행 가능한 테스트만 선별하여 실행할 수 있도록 설정합니다.

[매크로 상수 사용] 옵션은 소스 파일에 정의된 매크로 상수를 이용하여 테스트를 편집(예: 배열 인덱스 및 테스트 케이스의 입력/기대값에 매크로 상수 사용)할 수 있도록 설정합니다(※ 이 설정은 변환 프로젝트인 경우에만 적용됩니다).

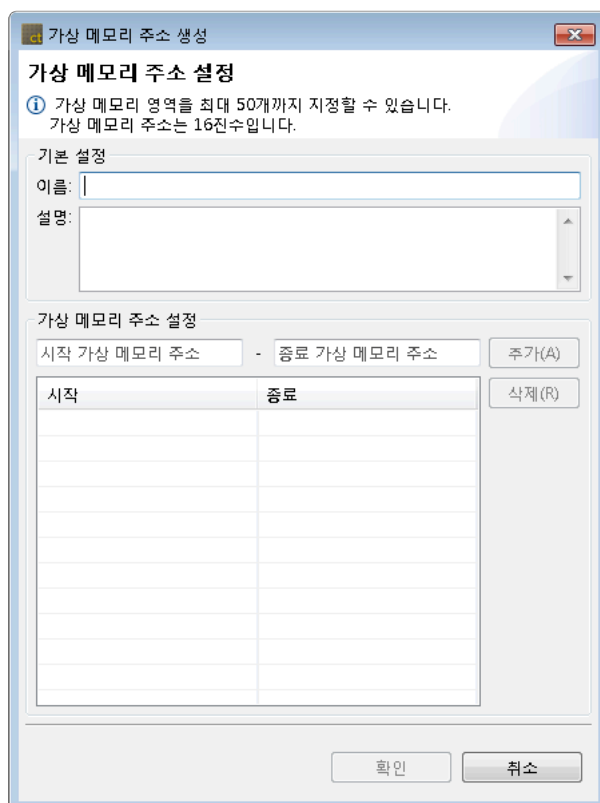


## 가상 메모리 주소

임베디드 환경 테스트를 위한 메모리 설정을 제공합니다. 프로젝트 특성에서 사용할 메모리 주소 그룹을 관리할 수 있습니다.

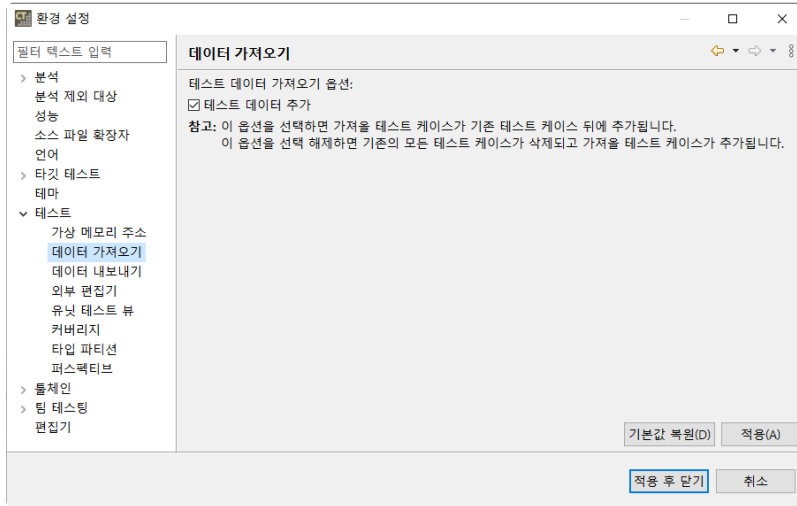


위 화면에서는 설정된 가상 메모리 주소 정보를 확인할 수 있습니다. [추가...] 버튼을 클릭하여 가상 메모리 주소를 추가할 수 있습니다.



가상 메모리 주소 영역은 최대 50개까지 지정할 수 있고, 16진수로 입력할 수 있습니다.

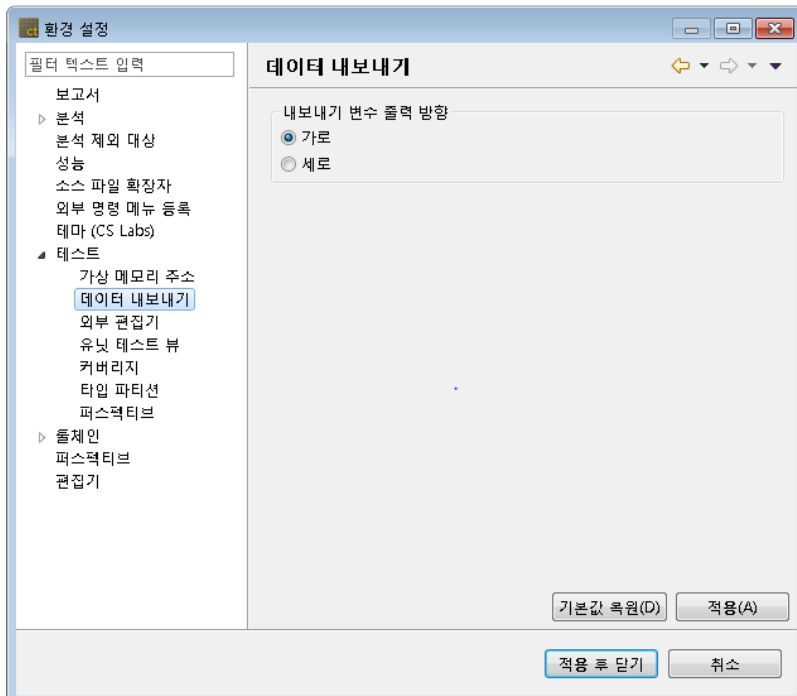
## 데이터 가져오기



[테스트 데이터 추가] 옵션을 체크하면 가져올 테스트 케이스가 기존 테스트 케이스 뒤에 추가됩니다. 이 옵션을 선택 해제하면 기존의 모든 테스트 케이스가 삭제되고 가져올 테스트 케이스가 추가됩니다.

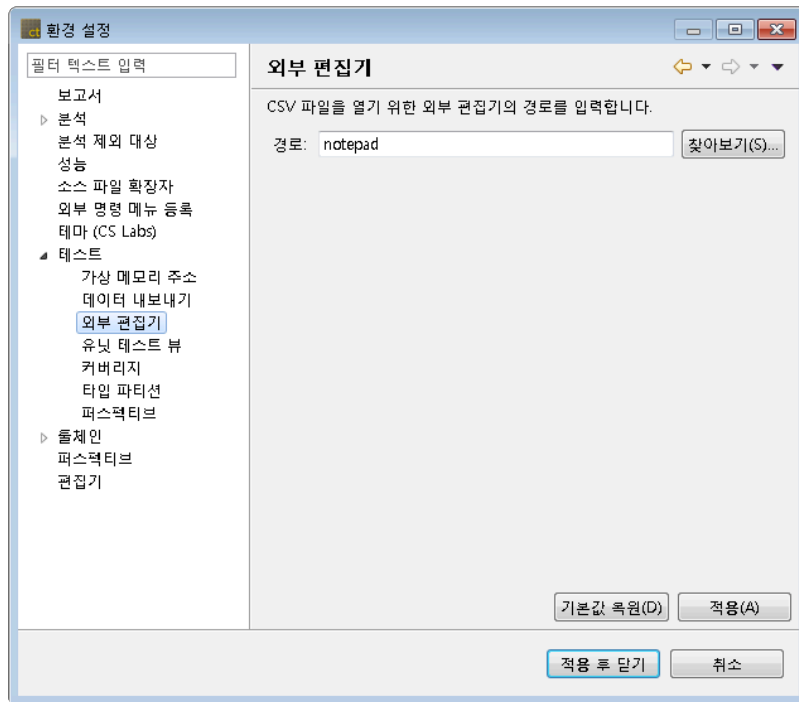
## 데이터 내보내기

데이터 내보내기 수행 시 변수의 출력 방향을 가로 또는 세로로 설정할 수 있습니다. 내보내기 변수 출력 방향을 선택한 후 [적용] 버튼을 클릭합니다.



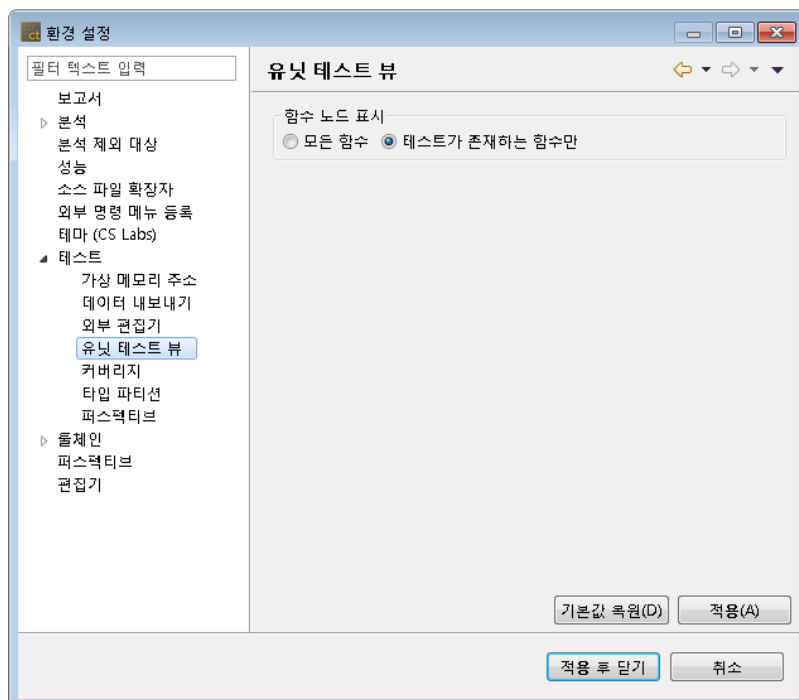
## 외부 편집기

CSV 파일을 열기 위한 외부 편집기를 설정할 수 있습니다. 외부 편집기 경로를 입력한 후 [열기] 버튼을 클릭합니다.



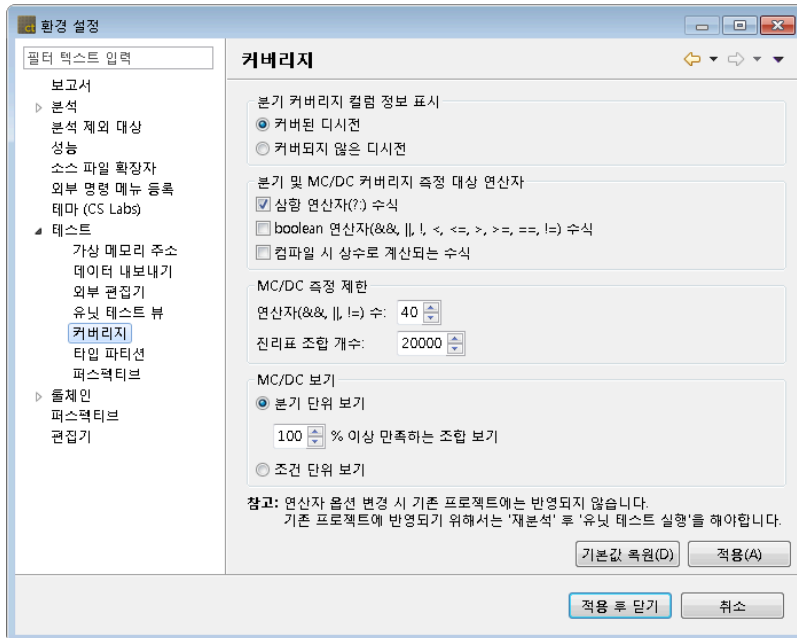
## 유닛 테스트 뷰

UI/UX가 개선된 테스트 퍼스펙티브에 포함된 유닛 테스트 뷰의 함수 노드 표시에 관해 설정할 수 있습니다.



## 커버리지

분기 커버리지 측정 및 커버 여부 표시에 관해 설정할 수 있습니다.



[분기 커버리지 컬럼 정보 표시] 라디오 버튼은 에디터 왼쪽 컬럼에 보여질 마커의 정보를 선택할 수 있도록 합니다.

- 소스 코드 에디터에 보여지는 T/F 마커 옵션
- 커버된 디시전: 커버된 디시전의 T/F 표시
- 커버되지 않은 디시전: 커버되지 않은 디시전의 T/F 표시

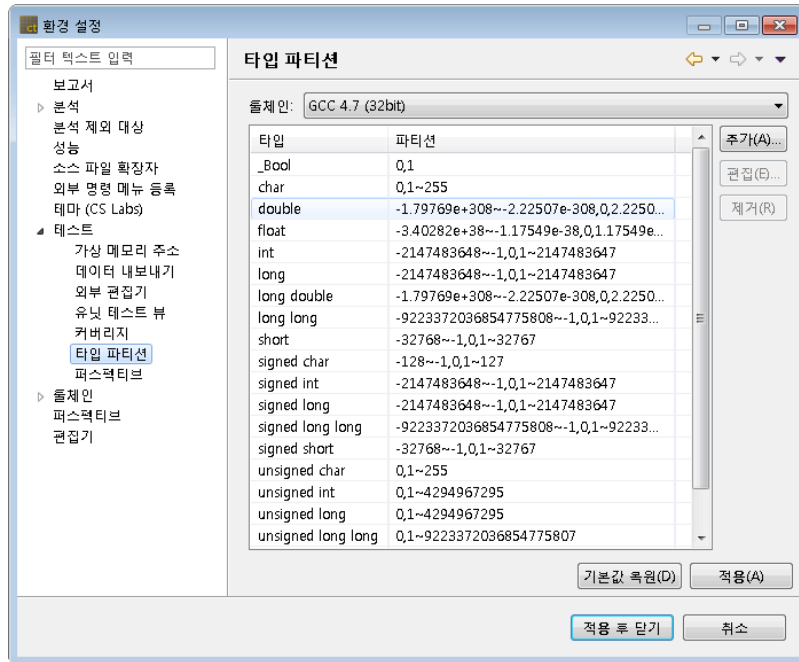
[분기 및 MC/DC 커버리지 측정 대상 연산자] 체크 박스 버튼은 분기 커버리지 측정 대상을 설정합니다. '재 분석' 후 '유닛 테스트 실행'을 해야 해당 설정이 적용 됩니다.

- 삼항 연산자(?:) 수식: 삼항 연산자 수식 분기 커버리지 측정
- boolean 연산자(&, ||, !=, <=, >, >=, ==, !=) 수식: boolean 연산자 수식 분기 커버리지 측정
- 모두 체크 해제하면 if, for, while, do-while, switch 문만 분기 커버리지 측정

! '삼항 연산자(?:) 수식' 설정 변경 시 COVER의 [툴체인 설정] > [측정 기준]과 동일하게 설정 하여야 COVER와 커버리지를 공유할 수 있습니다. 자세한 사항은 User Guides의 [삼항 연산자 설정에 따른 커버리지 가져오기](#) 페이지를 참고하시기 바랍니다.

## 타입 파티션

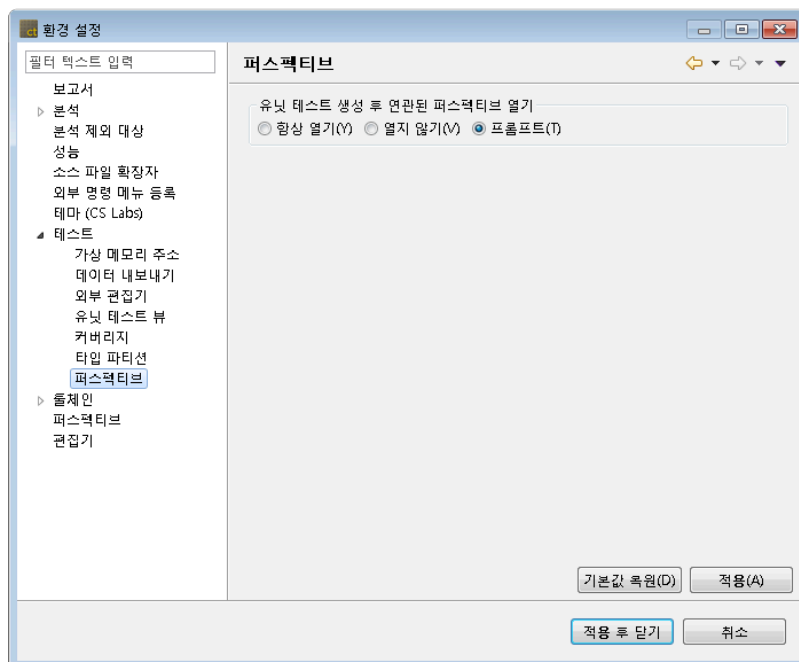
각 툴체인에 따른 기본 타입 파티션을 편집 및 제공되는 기본 값으로 복원할 수 있습니다. 툴체인 콤보 박스에서 수정할 툴체인을 선택하고 파티션을 편집 후 [적용] 버튼을 클릭합니다.



## 퍼스펙티브

테스트 생성 후 연관된 퍼스펙티브 열기에 대해 설정할 수 있습니다.

퍼스펙티브 항상 열기, 열지 않기, 프롬프트 중 하나를 선택한 후 [적용] 버튼을 클릭합니다.



## 16.8. 툴체인 설정

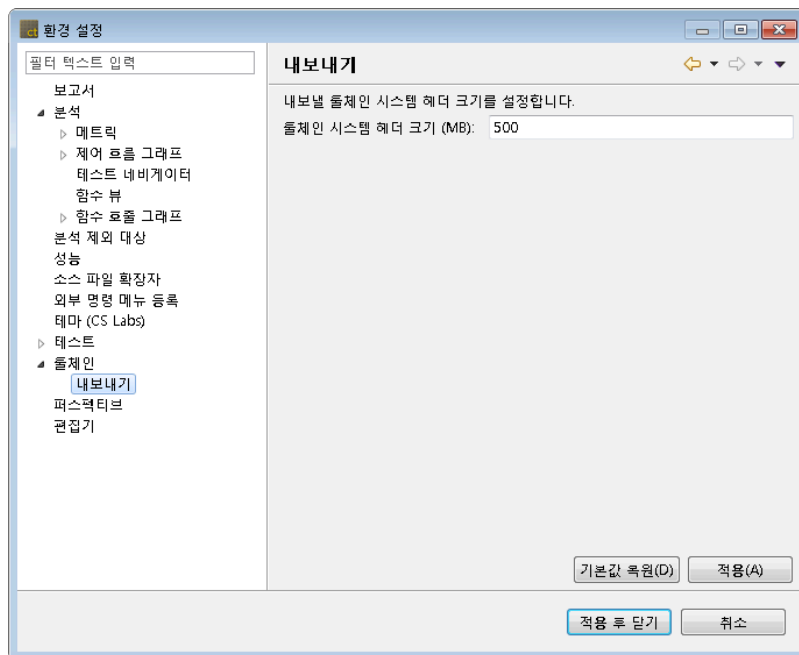
도구에서 사용할 툴체인에 대한 정보를 설정합니다.

테스트 대상 소스에 대한 툴체인(컴파일러 정보)이 있어야 프로젝트를 생성하거나 분석할 수 있습니다.

툴체인 설정에 대한 상세 사용법은 제품 매뉴얼의 [“툴체인\(분석기\) 설정”](#) 장을 통해 확인할 수 있습니다.

### 툴체인 내보내기

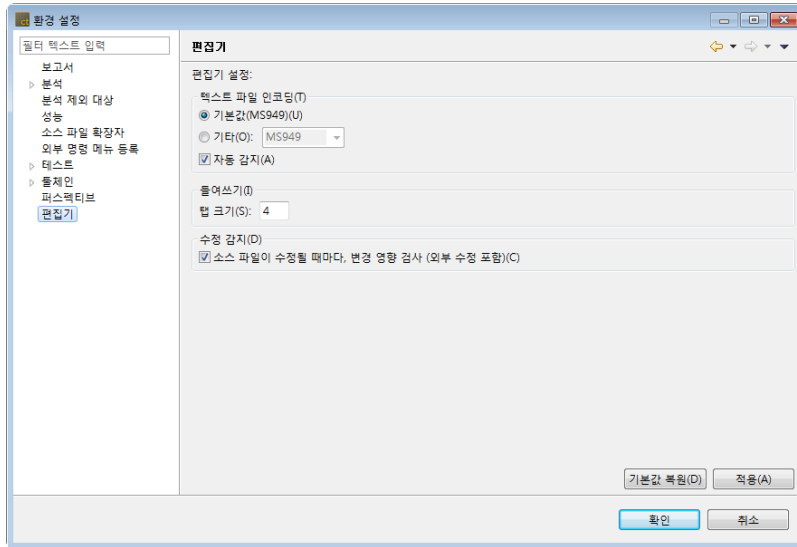
내보낼 툴체인의 시스템 헤더 크기를 설정합니다. 설정한 크기 보다 큰 시스템 헤더는 내보낼 수 없습니다.





## 16.9. 편집기

편집기와 관련된 설정을 변경할 수 있습니다.



### 텍스트 파일 인코딩

텍스트 파일을 편집기에서 열 때 사용할 인코딩을 설정할 수 있습니다. '자동 감지' 옵션이 켜져 있으면, 텍스트 파일의 인코딩을 자동으로 감지합니다(자동으로 감지할 수 없는 경우 설정된 인코딩으로 열림).

### 들여쓰기

탭 크기를 변경하면, 편집기에 보이는 탭 크기가 변경됩니다.

### 수정 감지

수정 감지 옵션이 켜져 있으면, 소스 파일이 수정될 때마다, 변경 영향을 검사합니다.

✿ 디스크 속도가 느린 경우, 이 옵션을 해제하면 성능이 향상됩니다.

# 17. 테스트 퍼스펙티브

테스트 퍼스펙티브는 시험 검증 작업의 흐름을 유지하고 목표에 집중할 수 있도록 필요한 정보만 노출하는 UI를 제공합니다.

Eclipse RCP로 제작되어 자유도가 높은 화면을 제공합니다. [창] 메뉴와 각 뷰에서 사용자가 설정할 수 있습니다.



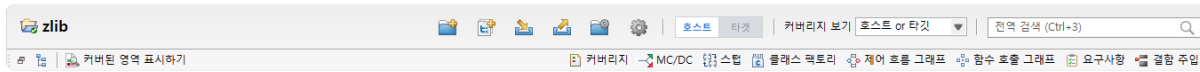
테스트 퍼스펙티브를 구성하는 요소는 다음과 같습니다.

- [대시보드](#)
- [테스트 네비게이터 뷰](#)
- [유닛 테스트 뷰](#)
- [통합 테스트 뷰](#)
- [커버리지 뷰](#)
- [MC/DC 뷰](#)
- [스텝 뷰](#)
- [클래스 팩토리 뷰](#)
- [제어 흐름 그래프 뷰](#)
- [함수 호출 그래프 뷰](#)
- [오류 뷰](#)
- [디버그 정보 뷰](#)
- [결합 주입 뷰](#)
- [입출력 데이터 그래프 뷰](#)
- [콘솔 뷰](#)
- [요구사항 뷰](#)

기본으로 포함되지 않는 뷰는 상단 메뉴의 [창] > [뷰 표시] > [기타...] 에서 열 수 있습니다.

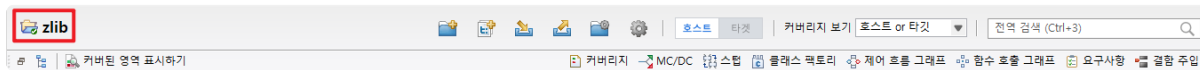
# 17.1. 대시보드

대시보드는 자주 사용되는 기능이나 현재 진행중인 테스트와 관련된 요약정보를 제공합니다.



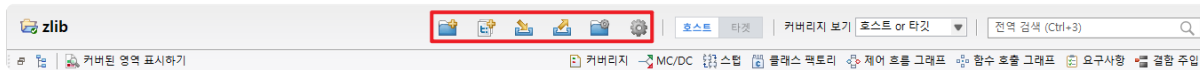
## 프로젝트 이름

[테스트 네비게이터 뷰]에서 선택한 프로젝트의 이름을 표시합니다.



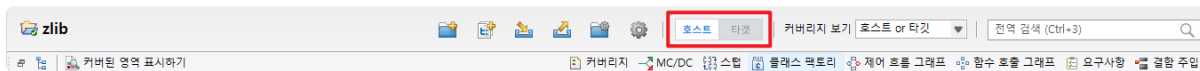
## 북마크 메뉴

[프로젝트 생성], [테스트 생성], [가져오기], [내보내기], [프로젝트 특성], [환경 설정] 기능을 사용할 수 있습니다.



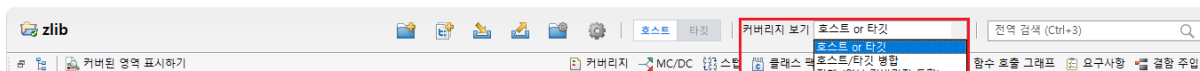
## 호스트/타겟 설정 메뉴

테스트 환경을 호스트 또는 타겟으로 변경할 수 있습니다.



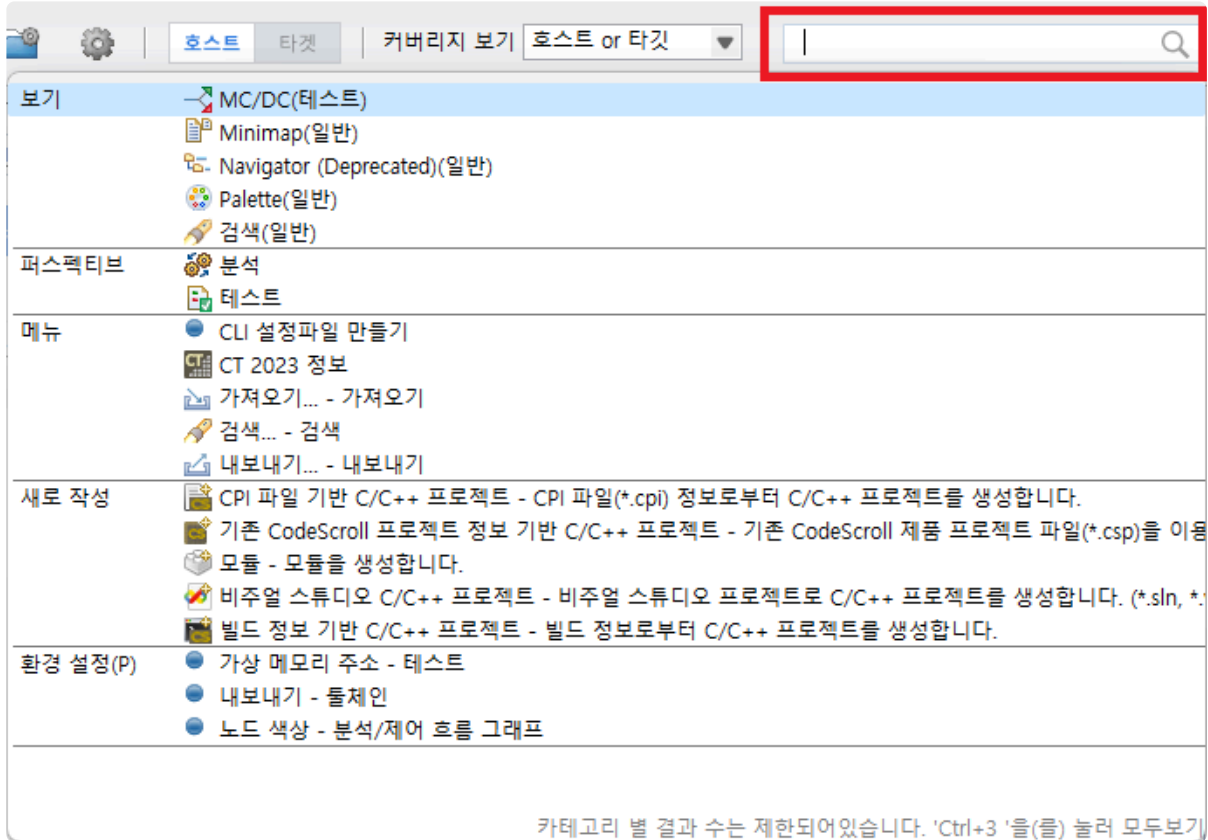
## 커버리지 관련 메뉴

[호스트 or 타겟], [호스트/타겟 병합], [전체 (외부 커버리지 통합)] 커버리지 보기 기능을 사용할 수 있습니다.

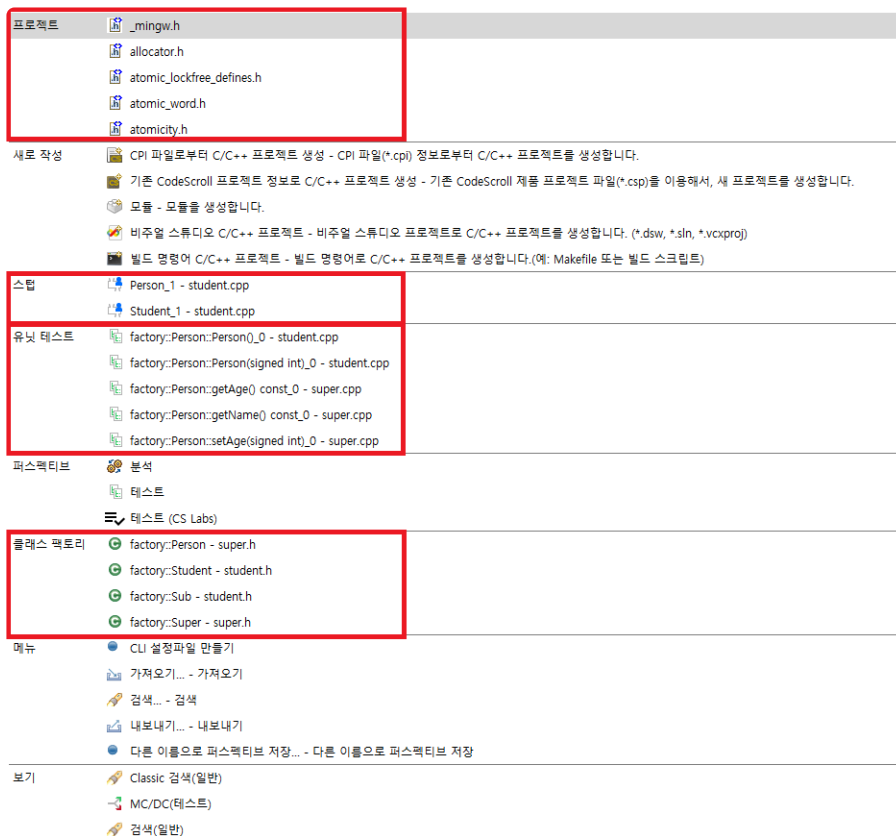


## 전역 검색

전역 검색은 새로 작성, 보기, 편집, 메뉴, 특성, 환경 설정, 프로젝트 리소스(소스 파일, 함수) 테스트, 스텝, 클래스 팩토리 등을 키워드로 검색할 수 있습니다. 전역 검색 기능은 전역 검색 창을 선택하거나, 단축키(Ctrl + 3)를 통해 사용할 수 있습니다.

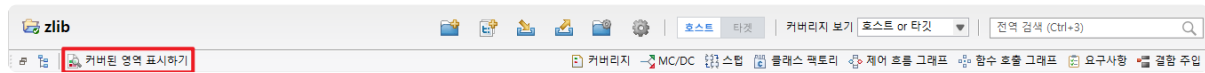


프로젝트의 리소스(소스 파일, 함수 등), 테스트, 스텝, 클래스 팩토리 는 분석된 프로젝트를 선택하면 검색할 수 있습니다.



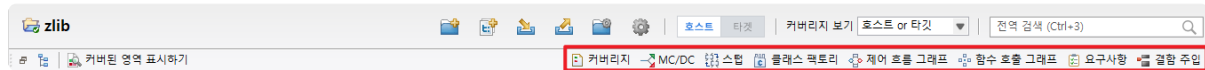
## 커버된 영역 표시하기

[소스 코드 편집기]에 커버된 영역을 표시합니다.



## 테스트 관련 뷰 열기/닫기 메뉴

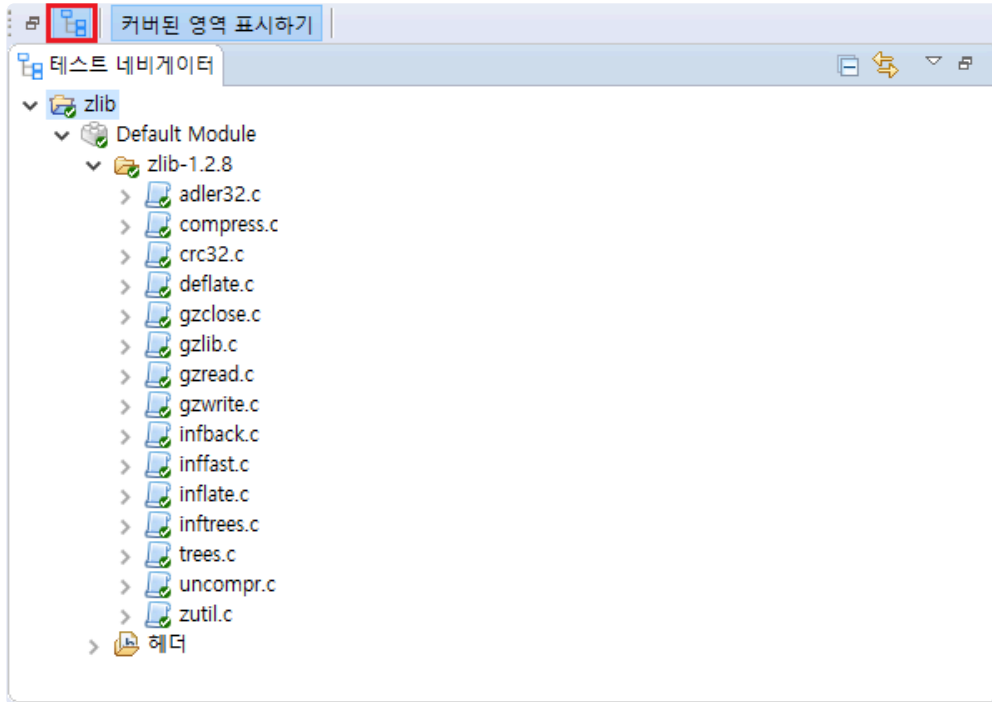
테스트와 관련이 있는 뷰를 열고 닫을 수 있습니다.



## 17.2. 테스트 네비게이터 뷰

[테스트 네비게이터 뷰]는 워크스페이스에 속한 프로젝트와 프로젝트 하위 테스트 모델들의 계층구조를 보여줍니다.

영역을 차지하지 않고 최소화 상태를 유지하는 것이 기본 설정입니다. 대시보드의 [테스트 네비게이터] 메뉴 선택 시, [테스트 네비게이터 뷰]는 [소스 코드 편집기] 위에 위치합니다.



### 아이콘 설명

아이콘	설명(*: 분석 이후 생성됨)
	열린 프로젝트
	닫힌 프로젝트
	모듈
	소스 파일
	*전역 변수
	*함수
	*헤더 폴더
	*헤더 파일
	*시스템 헤더 파일

## 툴바 메뉴

메뉴	설명
 다른 뷰와 연결	다른 뷰에서 선택한 항목을 강조
 모두 접기	모든 트리 노드를 접음

## 아이콘 오버레이 설명

아이콘 오버레이	설명
	분석됨
	분석 이후 변경됨

## 소스 파일을 다른 모듈에 복사하기

소스 파일을 다른 모듈에 복사하려면, Ctrl 키를 누른 상태로 복사할 소스 파일을 끌어서 대상 모듈에 놓습니다.

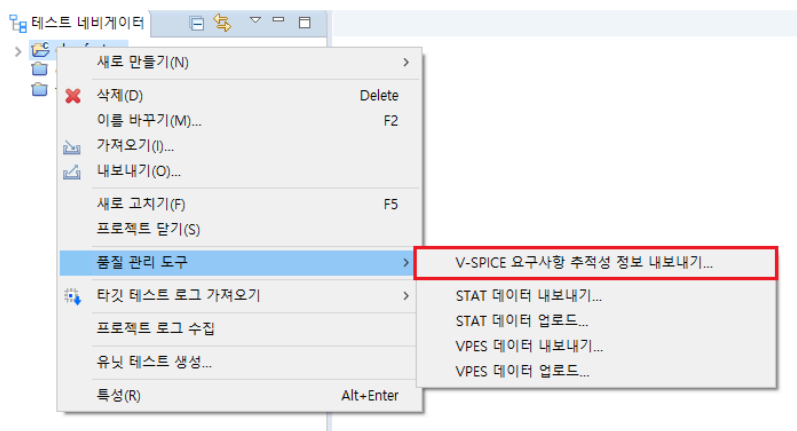
다른 방법은, 복사할 소스 파일을 오른쪽 클릭한 뒤에 [복사]를 선택하고, 대상 모듈을 오른쪽 클릭한 뒤에 [붙여넣기]를 선택하는 것입니다(바로 가기 키인 Ctrl + C 및 Ctrl + V 사용 가능).

## 소스 파일을 다른 모듈로 이동하기

소스 파일을 다른 모듈로 이동하려면, 이동할 소스 파일을 끌어서 대상 모듈에 놓습니다.

## V-SPICE 요구사항 추적성 정보 내보내기

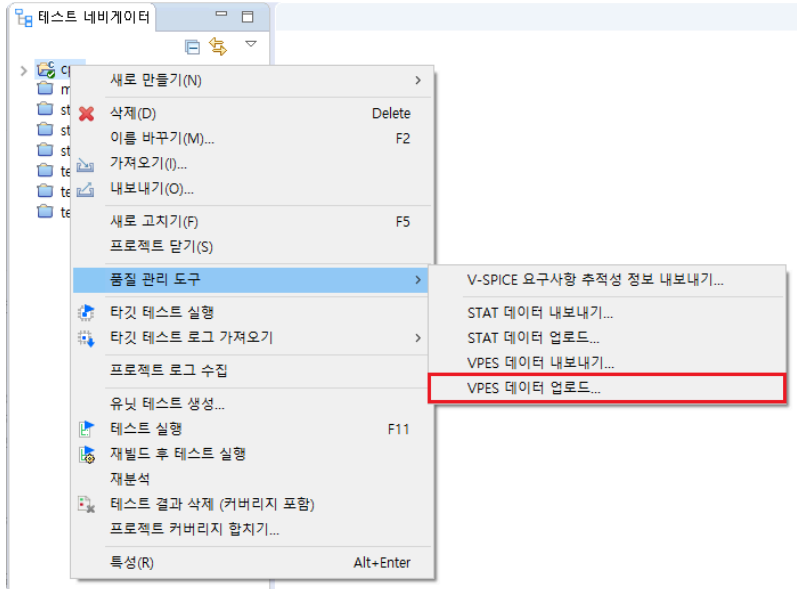
요구사항과 테스트 간의 연결 정보를 V-SPICE 형식에 맞는 xml 파일로 내보낼 수 있습니다.



프로젝트를 선택하고 우클릭하면 메뉴에 [품질 관리 도구] > [V-SPICE 요구사항 추적성 정보 내보내기...] 기능이 있습니다.



## VPES에 데이터 업로드

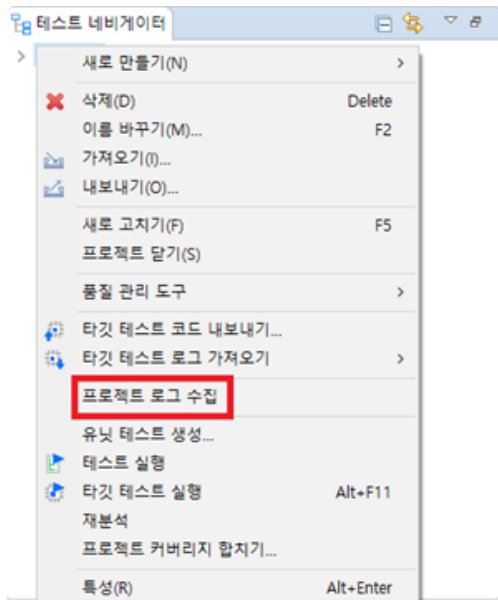


프로젝트를 선택하고 우클릭하면 메뉴에 [품질 관리 도구] -> [VPES에 데이터 업로드...] 기능이 있습니다. 이 기능에 대해서는 [기술지원](#) 연락처를 통해 문의하시기 바랍니다.

## 프로젝트 로그 수집

CT 2023.12 사용 중 남기는 로그를 프로젝트 단위로 일괄 수집하여 보내는 기능입니다.

CT 2023.12 발생하는 오류로 인해 기술지원에 문의할 때 로그가 수집된 압축 파일을 같이 보내주시면 문제를 더욱 빨리 해결할 수 있습니다.



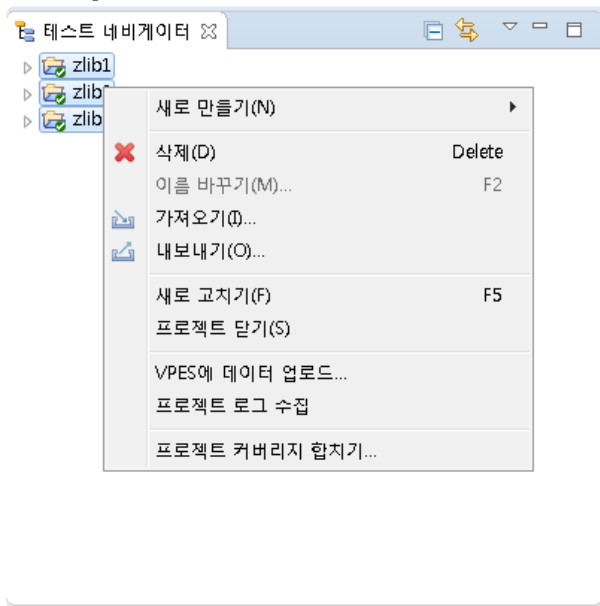
프로젝트를 선택하고 우클릭하면 메뉴에 [프로젝트 로그 수집] 기능이 있습니다.

## 17.2.1. 프로젝트 커버리지 합치기

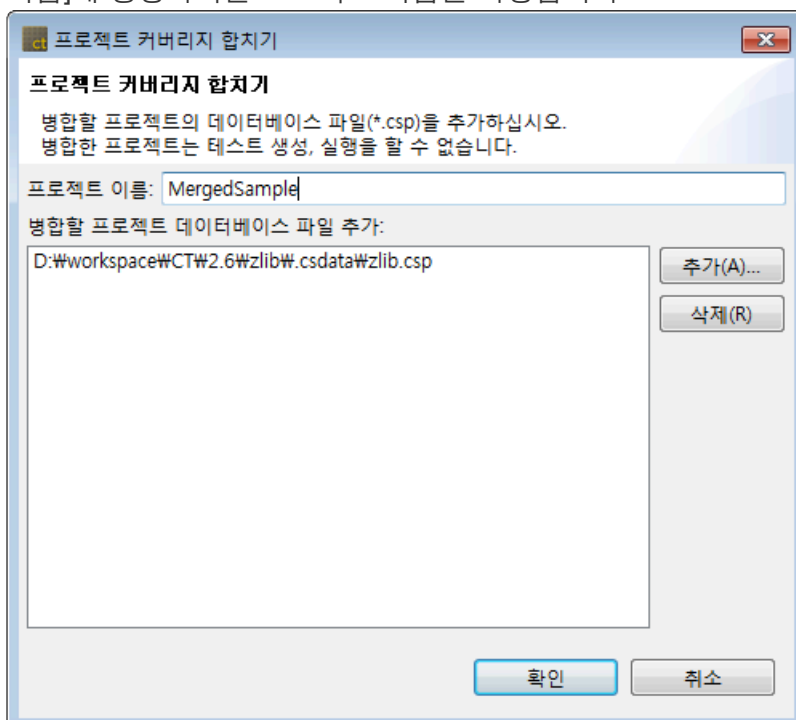
프로젝트 크기가 너무 커서 한 번에 테스트 실행을 하기 어려운 경우, 여러 개의 프로젝트로 나누어 테스트를 실행할 수 있습니다. [프로젝트 커버리지 합치기] 기능은 하나의 프로젝트를 다수의 프로젝트로 나누어 테스트를 실행한 후, 이 프로젝트들의 테스트 실행 결과를 병합하여 커버리지 결과를 전체적으로 보여줍니다.

### 프로젝트 커버리지 합치기

1. 커버리지 결과를 병합하려는 프로젝트를 모두 선택한 후, 마우스를 우 클릭하여 [프로젝트 커버리지 합치기]를 선택합니다.

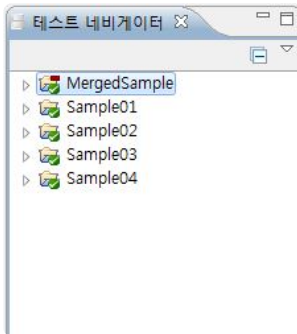


2. [프로젝트 커버리지 합치기] 메뉴를 선택하면 프로젝트의 이름을 작성하는 창이 열립니다. [프로젝트 이름]에 생성하려는 프로젝트 이름을 작성합니다.



- [추가] 버튼을 클릭하면, 워크스페이스에 포함되지 않은 프로젝트를 추가할 수 있습니다.

3. [확인] 버튼을 클릭하면 테스트 네비게이터 뷰에 프로젝트 결과가 병합된 새로운 프로젝트가 생성된 것을 확인할 수 있습니다. 병합된 프로젝트는 우측 상단에 빨간색 마크가 추가되어 표시됩니다.



4. 프로젝트 결과가 병합된 프로젝트를 선택 후 커버리지 뷰를 확인하면 전체(test01, test02, test03, test04) 커버리지를 확인할 수 있습니다.

커버리지

'Sample01' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	function1()	75.00% (...)	50.00% (...)	0.00% (0...)	N/A	Y
2	function2()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
3	function3()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
4	function4()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
합계		16.66% (...)	10.00% (...)	0.00% (0...)	N/A	25.00...

타겟 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

커버리지

'Sample02' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	function1()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
2	function2()	83.33% (...)	50.00% (...)	0.00% (0...)	N/A	Y
3	function3()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
4	function4()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
합계		13.88% (...)	5.00% (1...)	0.00% (0...)	N/A	25.00...

타겟 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

커버리지

'Sample03' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	function1()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
2	function2()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
3	function3()	60.00% (...)	33.33% (...)	0.00% (0...)	N/A	Y
4	function4()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
합계		16.66% (...)	10.00% (...)	0.00% (0...)	N/A	25.00...

타겟 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

커버리지

'Sample04' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	function1()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
2	function2()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
3	function3()	0.00% (0...)	0.00% (0...)	0.00% (0...)	N/A	N
4	function4()	50.00% (...)	25.00% (...)	0.00% (0...)	N/A	Y
합계		16.66% (...)	10.00% (...)	0.00% (0...)	N/A	25.00...

타겟 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

커버리지

'MergedSample' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	function1()	75.00% (...)	50.00% (...)	0.00% (0...)	N/A	Y
2	function2()	83.33% (...)	50.00% (...)	0.00% (0...)	N/A	Y
3	function3()	60.00% (...)	33.33% (...)	0.00% (0...)	N/A	Y
4	function4()	50.00% (...)	25.00% (...)	0.00% (0...)	N/A	Y
합계		63.88% (...)	35.00% (...)	0.00% (0...)	N/A	100.0...

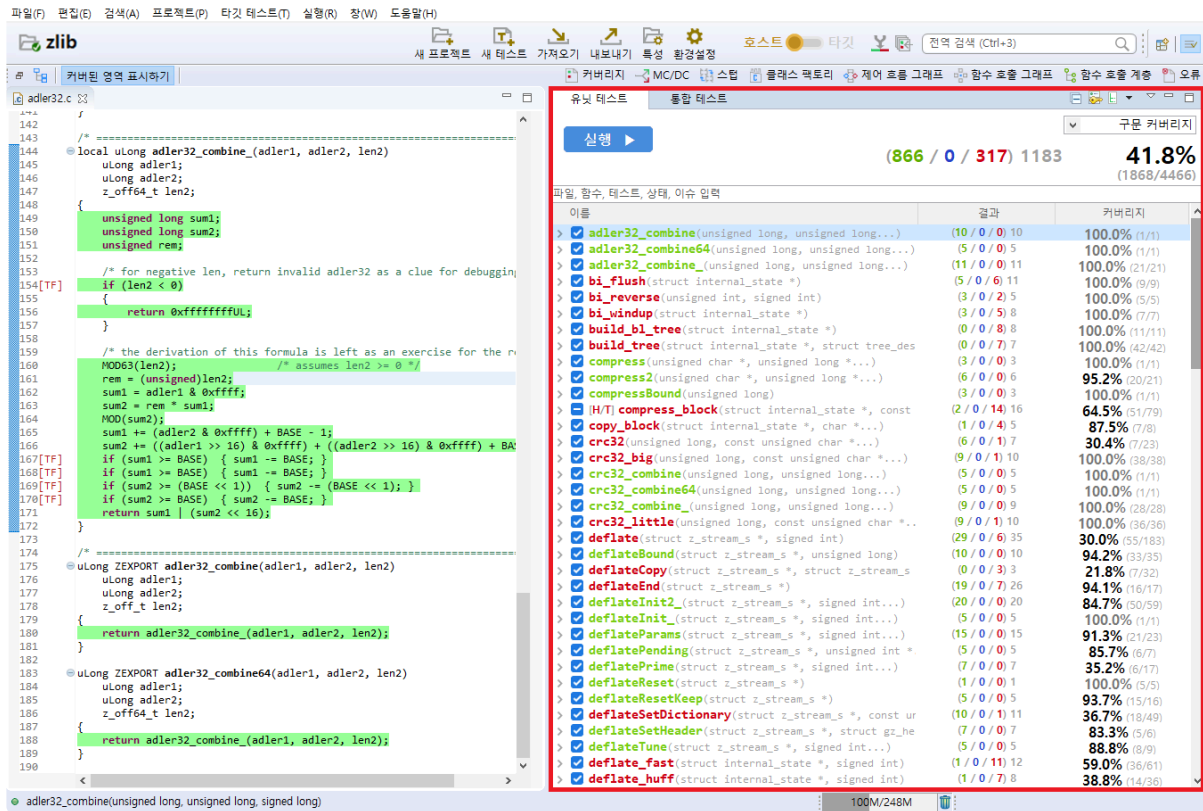
타겟 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.



[프로젝트 커버리지 합치기] 기능은 병합할 프로젝트 내에 두 개 이상의 다른 함수가 동일한 함수 이름을 가진 경우에는 정상적으로 수행되지 않을 수 있습니다.


## 17.3. 유닛 테스트 뷰

[유닛 테스트 뷰]는 기본으로 화면 오른쪽에 위치합니다. [유닛 테스트 뷰]는 테스트와 커버리지 정보를 통합하여 하나의 뷰로 제공합니다.

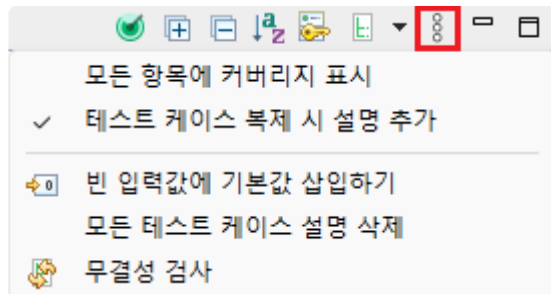


### [유닛 테스트 뷰]의 툴바 메뉴

툴바 아이콘	설명
	결함 위치 찾기
	모두 펼치기
	모두 접기
	고유한 테스트 이름으로 정렬하기
	고유한 테스트 이름으로 보기
	합계
	실패/오류
	실패
	오류
	성공
	함수 변경

 실행 보장 안됨	테스트 실행 보장 안되는 테스트만 보기
 호스트/타겟 결과 다름	호스트/타겟 결과가 다른 테스트만 보기

## [유닛 테스트 뷰]의 풀다운 메뉴



메뉴 이름	설명
모든 항목에 커버리지 표시	모든 항목(함수, 테스트, 테스트 케이스)에 커버리지 표시
테스트 케이스 복제 시 설명 추가	복제한 테스트 케이스에 원본 테스트 케이스 번호 표시
빈 입력값에 기본값 삽입하기	테스트 케이스에 입력값이 없는 경우 기본값 삽입
모든 테스트 케이스 설명 삭제	테스트 케이스에 입력된 설명을 일괄 삭제
무결성 검사	테스트와 스텝의 무결성 검사 후 필요 시 테스트 재설정

- 빈 입력값에 기본값 삽입하기

변수 종류	기본값
문자열	(빈 입력값)
문자열이 아닌 변수	0

✿ [테스트 재설정]에 대한 자세한 설명은 User Guide 2023.12의 [\[소스 코드 변경 후 테스트 재설정\]](#) 페이지를 참고하시기 바랍니다.

## [유닛 테스트 뷰] 대시보드

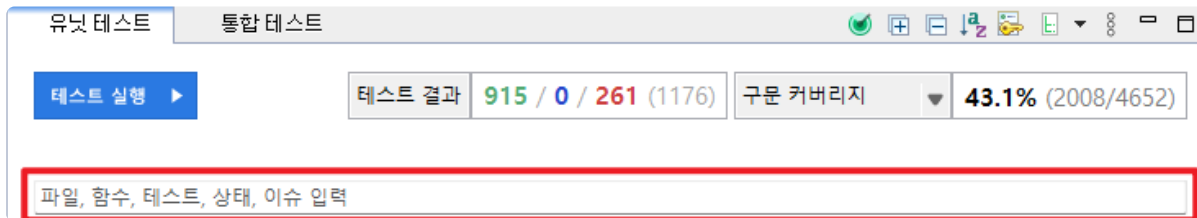


메뉴	설명
----	----

<b>테스트 실행</b> ▶	유닛 테스트 실행 버튼
<b>테스트 결과</b> 915 / 0 / 261 (1176)	테스트 케이스의 성공, 실패, 오류, 총합 표현
<b>구문 커버리지</b> ▼	[유닛 테스트 뷰]에 표시할 커버리지 종류(구문, 분기, MC/DC, 함수 호출, 함수) 선택
<b>43.1%</b> (2008/4652)	선택한 종류의 전체 커버리지 표현

## 검색

[유닛 테스트 뷰]에서는 파일, 함수, 테스트, 상태(성공, 실패, 오류), 이슈의 이름을 통해 함수, 테스트, 테스트 케이스를 검색할 수 있습니다.



## 상태 검색 키워드

[유닛 테스트 뷰]에서는 상태 검색 키워드를 통해 표시되는 테스트를 필터링할 수 있습니다. 검색 가능한 키워드의 추천 및 자동완성 기능을 제공합니다.

키워드	설명
%TEST_SUCCESS%	테스트 성공
%TEST_FAILURE% %TEST_ERROR%	테스트 실패/오류
%TEST_FAILURE%	테스트 실패
%TEST_ERROR%	테스트 오류
%TEST_HAS_NOT_FUNCTION%	함수 변경
%TEST_NOT_GUARANTEE%	실행 보장 안됨
%TEST_RESULT_DIFFERENT%	호스트/타겟 결과 다름

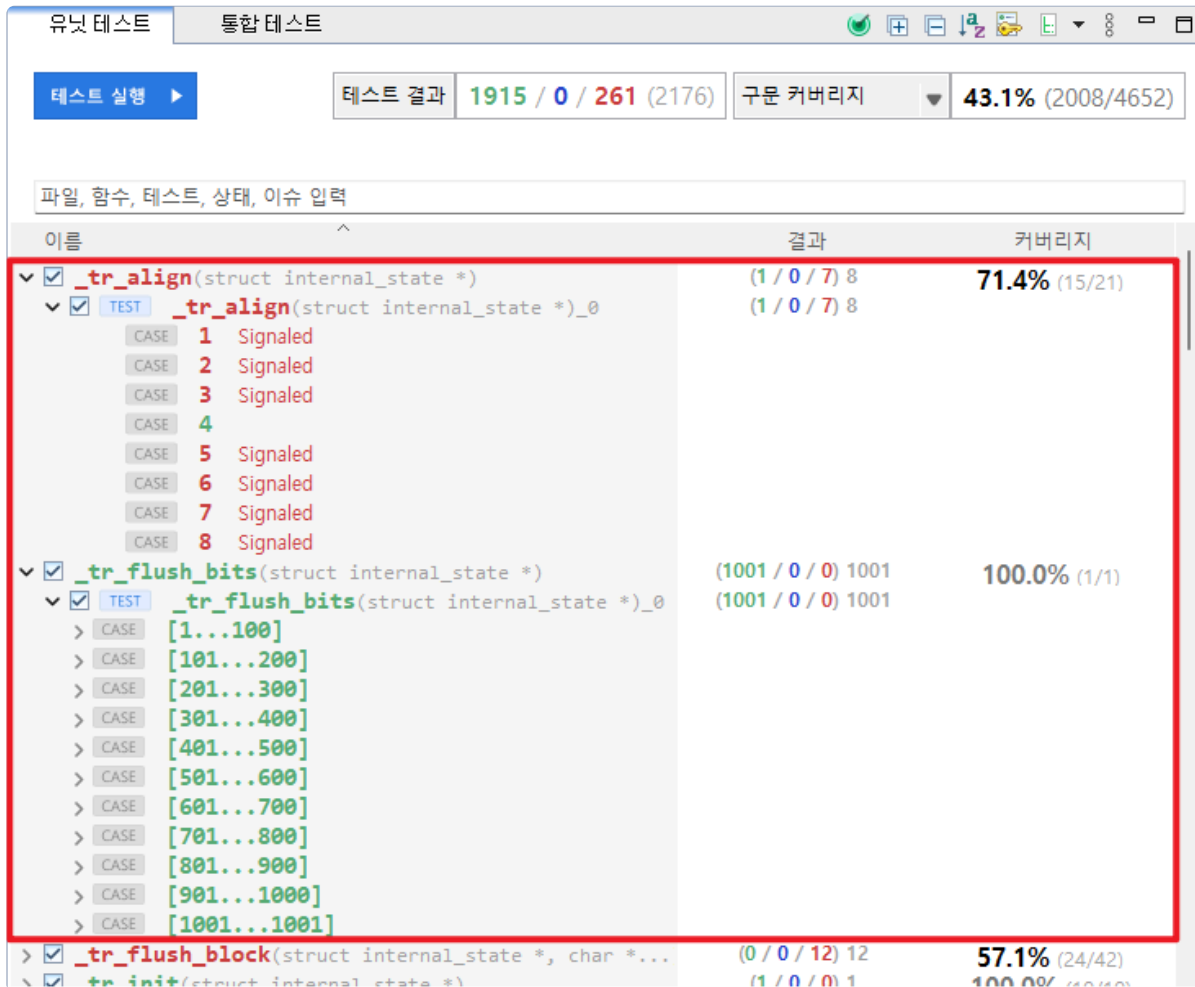


‘실행 보장 안됨’은 테스트 코드를 생성했지만 내부적으로 정상 실행이 될 지 여부를 보장할 수 없는 경우입니다. 특정하기 어려운 타입을 파라미터나 반환값으로 사용하는 경우입니다.

## [유닛 테스트 뷰]의 구조

[유닛 테스트 뷰]는 [함수] > [테스트] > [테스트 케이스]의 계층 구조로 표현합니다. 하나의 테스트에 100개

이상의 테스트 케이스가 존재할 경우, 100개씩 묶어서 그룹으로 표현합니다.



## [유닛 테스트 뷰]의 항목 아이콘

항목 아이콘	설명
아이콘 없음	함수
TEST	테스트
TEST	시나리오 테스트
CASE	테스트 케이스, 테스트 케이스 그룹

## [유닛 테스트 뷰]의 항목 상태 표현

[유닛 테스트 뷰]의 함수, 테스트, 테스트 케이스는 테스트 실행 여부 및 실행 결과를 색상으로 표현합니다.

색상	설명
녹색	함수/테스트: 하위에 모든 테스트 케이스의 실행 결과가 성공인 경우 테스트 케이스: 실행 결과가 성공인 경우
파란색	함수/테스트: 하위에 실행 결과가 실패인 테스트 케이스가 존재하고 오류인 테스트 케이스는 존재하지 않음

	테스트 케이스: 실행 결과가 실패인 경우
붉은 색	함수/테스트: 하위에 실행 결과가 오류인 테스트 케이스가 존재함 테스트 케이스: 실행 결과가 오류인 경우
주황 색	함수/테스트: 하위에 모든 테스트 케이스를 실행하지 않았고 실행 보장 안 됨 테스트 케이스가 존재함 테스트 케이스: 실행 보장 안 됨
검정 색	함수/테스트: 하위에 모든 테스트 케이스를 실행하지 않음 테스트 케이스: 실행하지 않음

## 함수

함수에서는 함수의 이름, 커버리지, 테스트 케이스 수행 결과(성공, 실패, 오류, 합계)를 제공합니다.

✿ 각 함수의 커버리지는 함수 간 호출 관계에 따라 다른 함수의 테스트 수행 결과가 반영될 수 있습니다.

함수를 더블 클릭하여 해당 함수의 위치를 [소스 코드 편집기]에서 확인할 수 있습니다.

유닛 테스트 통합 테스트

실행 ▶

(1819 / 0 / 282) 2101 **43.4%**  
(1942/4471)

파일, 함수, 테스트, 상태, 이슈 입력

이름	결과	커버리지
✓ <input checked="" type="checkbox"/> <b>_tr_align</b> (struct internal_state *)	(1 / 0 / 7) 8	<b>71.4%</b> (15/21)
✓ <input checked="" type="checkbox"/> test <b>_tr_align</b> (struct internal_state *)_0 case 1 Signaled case 2 Signaled case 3 Signaled case 4 Signaled case 5 Signaled case 6 Signaled case 7 case 8 Signaled	(1 / 0 / 7) 8	
✓ <input checked="" type="checkbox"/> <b>_tr_flush_bits</b> (struct internal_state *)	(1004 / 0 / 0) 1004	<b>100.0%</b> (1/1)
✓ <input checked="" type="checkbox"/> test <b>_tr_flush_bits</b> (struct internal_state *)_0 > case [1...100] > case [101...200] > case [201...300] > case [301...400] > case [401...500] > case [501...600] > case [601...700] > case [701...800] > case [801...900] > case [901...1000] > case [1001...1004]	(1004 / 0 / 0) 1004	
> <input checked="" type="checkbox"/> <b>_tr_flush_block</b> (struct internal_state *, char *...	(0 / 0 / 12) 12	<b>57.1%</b> (24/42)
> <input checked="" type="checkbox"/> <b>_tr_init</b> (struct internal_state *)	(1 / 0 / 0) 1	<b>100.0%</b> (10/10)
> <input checked="" type="checkbox"/> <b>_tr_stored_block</b> (struct internal_state *, char *...	(0 / 0 / 8) 8	<b>100.0%</b> (11/11)
> <input checked="" type="checkbox"/> <b>_tr_tally</b> (struct internal_state *, unsigned int...)	(0 / 0 / 3) 3	<b>22.2%</b> (2/9)
> <input checked="" type="checkbox"/> <b>adler32</b> (unsigned long, const unsigned char *...)	(8 / 0 / 5) 13	<b>98.0%</b> (101/103)
> <input checked="" type="checkbox"/> <b>adler32_combine</b> (unsigned long, unsigned long...)	(5 / 0 / 0) 5	<b>100.0%</b> (1/1)



## 함수의 컨텍스트 메뉴



컨텍스트 메뉴	설명
테스트 생성	선택된 함수의 테스트 생성
테스트 복사	선택된 함수의 테스트 복사
삭제	선택된 함수의 테스트 및 케이스 삭제
함수 호출 그래프 보기	선택된 함수의 함수 호출 그래프 보기
제어 흐름 그래프 보기	선택된 함수의 제어 흐름 그래프 보기
MC/DC 보기	선택된 함수의 MC/DC 보기
호스트 출력값 -> 기대값	기대값에 호스트 출력값을 붙여넣기
테스트 재설정	테스트 정보 재설정
연관 파일 설정	해당 테스트가 속한 파일 선택
연관 이슈 설정	선택된 테스트를 이슈 관리 도구의 이슈와 연관
테스트 데이터 가져오기	로컬에 저장된 테스트 데이터 가져오기
테스트 데이터 내보내기	로컬로 테스트 데이터 내보내기

기존 스텝 추가	선택된 테스트에 스텝 추가
클래스 코드 적용	<a href="#">[클래스 팩토리 뷰]</a> 에서 생성된 클래스 코드를 함수에 적용하는 기능
함수 선택	마우스로 선택된 모든 함수의 실행 체크박스를 선택으로 변경
함수 선택 해제	마우스로 선택된 모든 함수의 실행 체크박스를 해제로 변경
선택한 테스트 커버리지 내보내기	선택한 테스트 커버리지를 선택한 경로로 내보내는 기능
결함 위치 찾기	오류 또는 실패한 테스트 케이스로부터 결함 위치 확인



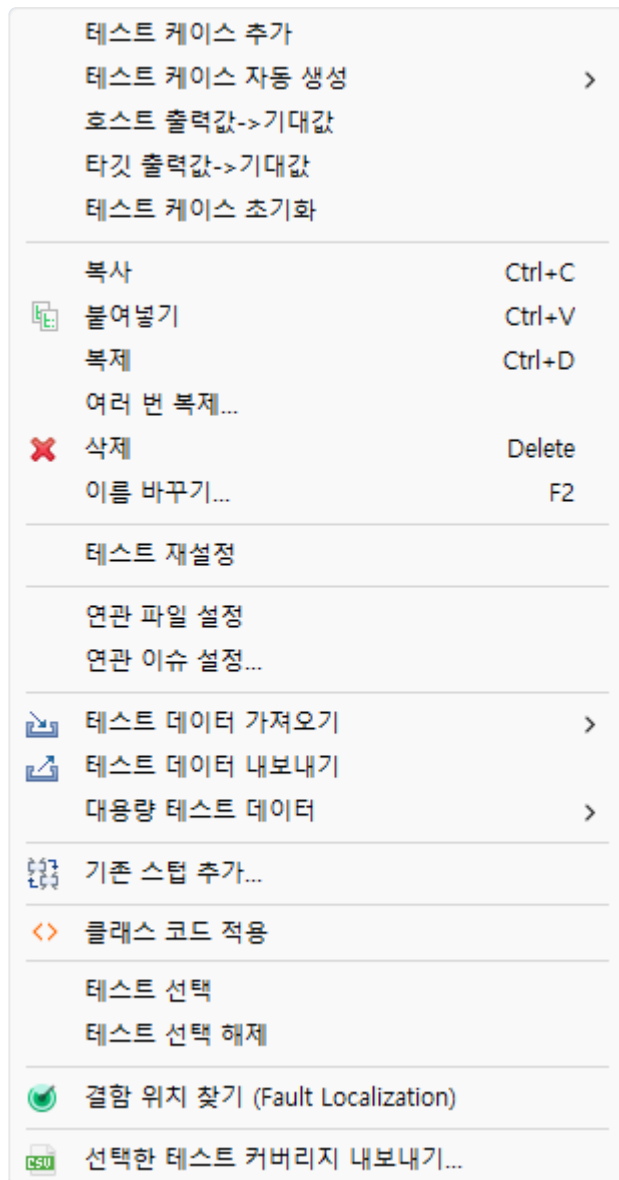
[테스트 재설정]에 대한 자세한 설명은 User Guide 2023.12의 [\[소스 코드 변경 후 테스트 재설정\]](#) 페이지를 참고하시기 바랍니다.

## 테스트

테스트는 커버리지, 테스트 케이스 수행 결과(성공, 실패, 오류, 합계)를 제공합니다.

테스트를 더블 클릭하여 해당 [\[테스트 편집기\]](#) 를 열 수 있습니다.

## 테스트의 컨텍스트 메뉴



컨텍스트 메뉴	설명
기호 실행 기반 테스트 케이스 생성	기호 실행을 이용해 테스트 케이스 생성
테스트 케이스 추가	선택된 유닛 테스트에 테스트 케이스 추가
테스트 케이스 자동 생성	다양한 방법으로 테스트 케이스를 생성
호스트 출력값 -> 기대값	기대값에 호스트 출력값을 붙여넣기
테스트 케이스 초기화	모든 테스트 케이스 삭제
복사	테스트 및 케이스 복사
붙여넣기	테스트 및 케이스 붙여넣기
복제	테스트 및 케이스 복제
여러 번 복제	테스트 및 케이스 입력한 수만큼 복제

삭제	테스트 및 케이스 삭제
이름 바꾸기	테스트 이름 변경
테스트 재설정	테스트 정보 재설정
연관 파일 설정	해당 테스트가 속한 파일 선택
연관 이슈 설정	선택된 테스트를 이슈 관리 도구의 이슈와 연관
테스트 데이터 가져오기	로컬에 저장된 테스트 데이터 가져오기
테스트 데이터 내보내기	로컬로 테스트 데이터 내보내기
대용량 테스트 데이터	로컬로 내보내거나 사용자가 작성한 파일을 대상 테스트에 테스트 케이스로 등록하는 기능
기존 스텝 추가	선택된 테스트에 스텝 추가
클래스 코드 적용	<a href="#">[클래스 팩토리 뷰]</a> 에서 생성된 클래스 코드를 테스트에 적용하는 기능
테스트 선택	마우스로 선택된 모든 테스트의 실행 체크박스를 선택으로 변경
테스트 선택 해제	마우스로 선택된 모든 테스트의 실행 체크박스를 해제로 변경
결함 위치 찾기	테스트 케이스의 결함위치 보기
선택한 테스트 커버리지 내보내기	선택한 테스트 커버리지를 선택한 경로로 내보내는 기능

- 테스트 데이터 가져오기 규칙

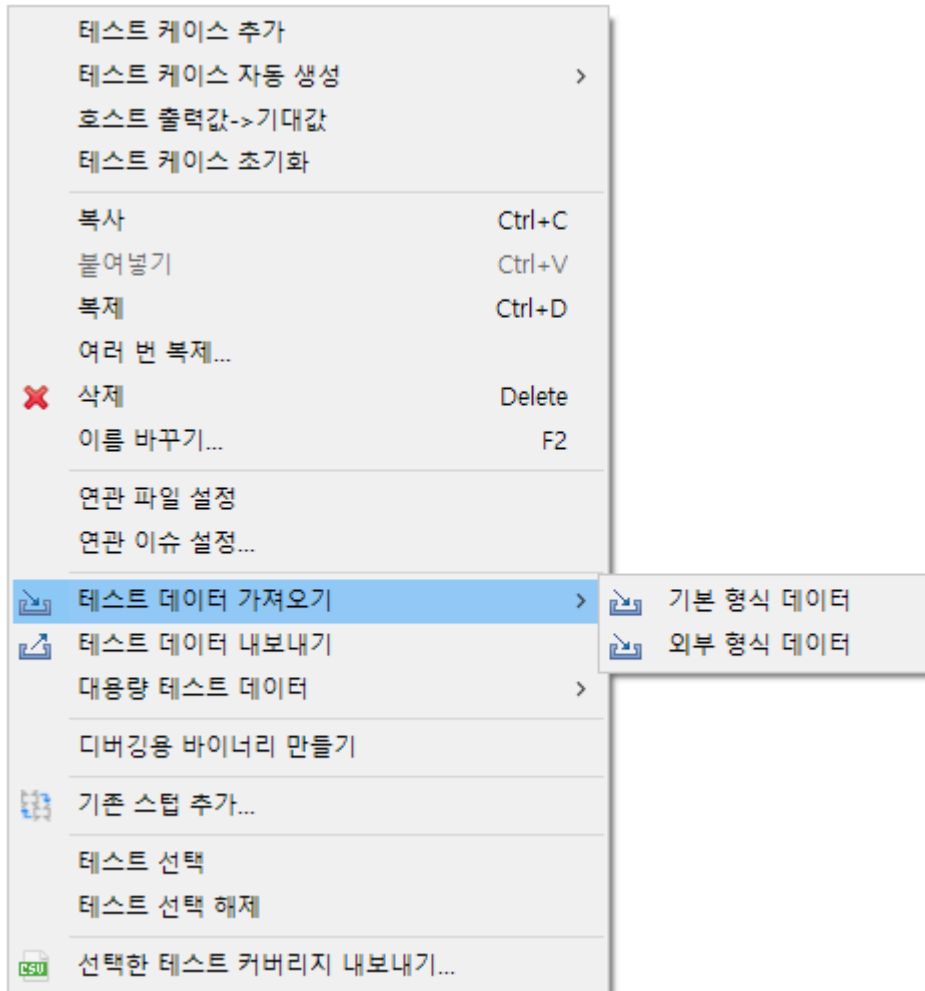
테스트 데이터를 가져올 때 해당 테스트의 이름을 가지고 넘버링 된 다수의 테스트 데이터를 선택하면 해당 파일을 병합하여 가져옵니다.



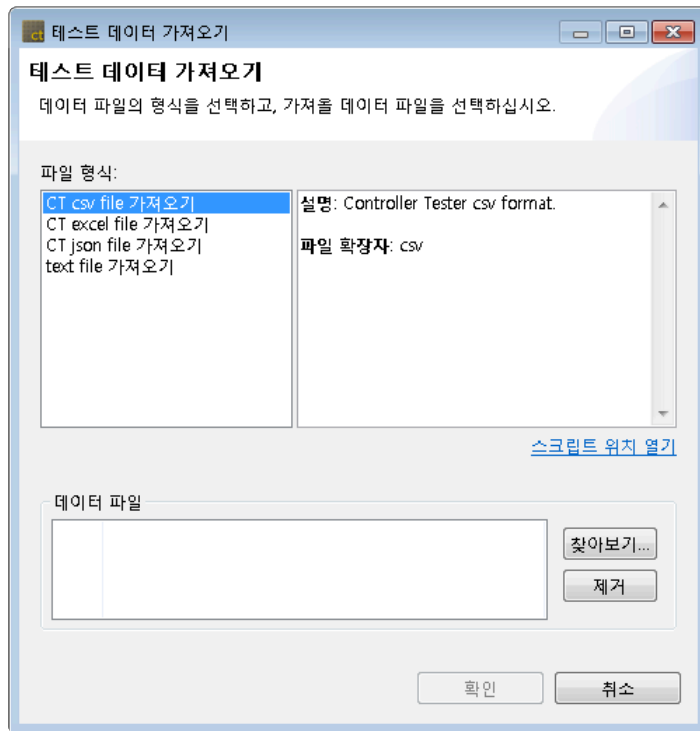
[테스트 재설정]에 대한 자세한 설명은 User Guide 2023.12의 [\[소스 코드 변경 후 테스트 재설정\]](#) 페이지를 참고하시기 바랍니다.

## 테스트 데이터 가져오기

다양한 형식(csv, xlsx, txt, json)의 테스트 데이터를 가져올 수 있습니다.

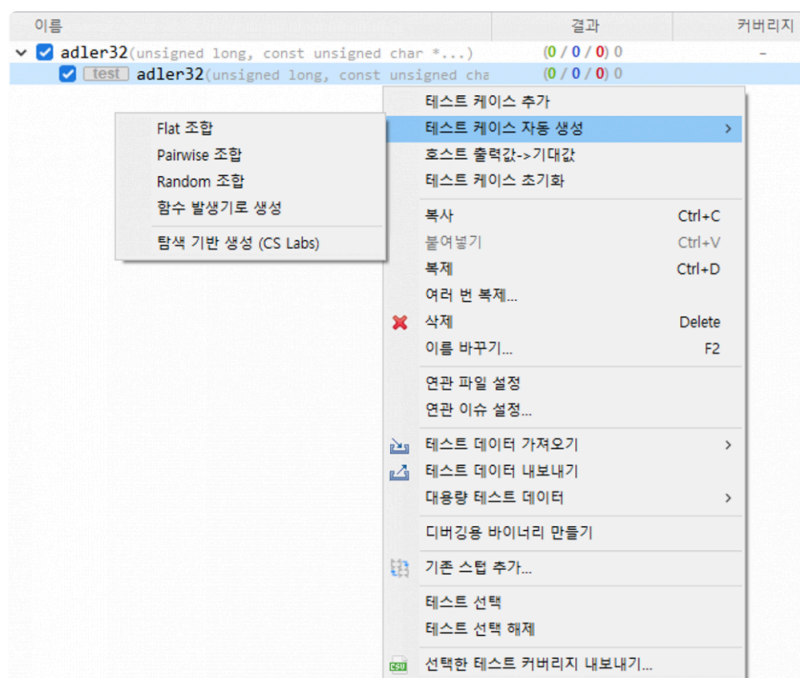


1. [유닛 테스트 뷰]의 컨텍스트 메뉴 중 [테스트 데이터 가져오기]를 클릭 후, [기본 형식 데이터], [외부 형식 데이터] 중 하나를 선택합니다.
2. [기본 형식 데이터]를 선택할 경우, CT 2023.12에서 내보낸 형식(csv)의 테스트 데이터를 가져옵니다.
3. [외부 형식 데이터]를 선택할 경우, 다양한 형식(csv, xlsx, txt, json)의 파일로부터 테스트 데이터를 가져옵니다.



## 테스트 케이스 자동 생성

Flat/Pairwise/Random 조합, 함수 발생기로 생성, 탐색 기반 생성 (CS Labs)를 사용하여 테스트 케이스를 생성합니다.

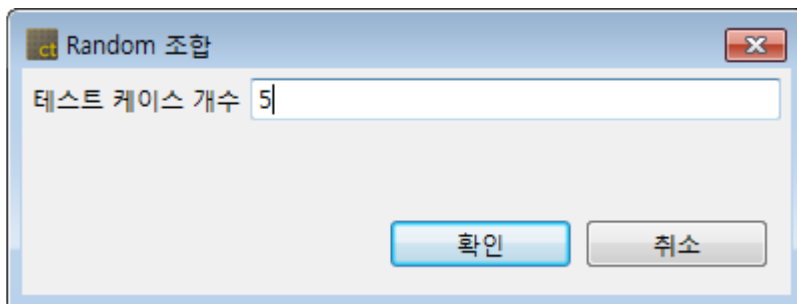


1. [유닛 테스트 뷰] 또는 [통합 테스트 뷰]의 컨텍스트 메뉴 중 [테스트 케이스 자동 생성]을 클릭 후, [Flat 조합], [Pairwise 조합], [Random 조합], [함수 발생기로 생성], [탐색 기반 생성 (CS Labs)] 중 한 가지 테스트 케이스 자동 생성 방식을 선택합니다.

방법	설명
Flat	테스트 데이터 개수가 가장 많은 변수를 기준으로 단순 조합

Pairwise	각 선출된 파라미터의 데이터가 자신 이외의 파라미터 데이터와 적어도 한 번씩은 쌍을 이루도록 조합 ※ 변수 파티션 개수는 2보다 크고 52보다 작아야 함
Random	입력 파라미터의 변수 파티션의 최소값과 최대값 사이의 임의의 값을 사용자가 정의한 테스트 케이스 수만큼 테스트 데이터 조합
함수 발생기로 생성	6가지 함수 유형(Ramps, Random, Range, Sine, Toggle, Single Value)을 사용하여 테스트 케이스 생성
탐색 기반 생성 (CS Labs)	국지 탐색(Local search) 알고리즘인 AVM (Alternating Variable Method) 기법을 통해 테스트 케이스를 생성 ※ 현재는 C 언어만 지원

2. [Random 조합]을 선택할 경우, 사용자로부터 테스트 케이스 개수를 입력 받을 수 있도록 Random 조합 창을 보여줍니다.



3. [함수 발생기로 생성]을 선택할 경우, 함수 유형{Ramps, Sine, Random, Toggle, Range, Single value, None)을 선택할 수 있고, 함수 발생기 설정을 통해 선택된 함수의 설정 값을 변경할 수 있습니다. 프로젝트의 특성 창의 [테스트] > [테스트 케이스 자동 생성] > [함수 발생기] 에서 함수 발생기 정보를 설정할 수 있습니다.





Fall samples	Hold 샘플 값에서 Pre/Post 샘플 값으로 떨어지는 시간
Post delay	Pre/Post 샘플 값이 지속되는 시간
Pre/Post delay value	Pre/Post delay 샘플 값
Hold value	Hold 샘플 값

### c. Random 함수

최소값(Min)과 최대값(Max) 사이에서 랜덤한 값을 생성하는 함수입니다.

설정	설명
Min	랜덤 범위 최소값
Max	랜덤 범위 최대값

### d. Range 함수

타입 최소값과 최대값 사이에 일정한 간격(Step Size)만큼 증가 혹은 감소하는 값을 생성하는 함수입니다.

설정	설명
Step size	증가 혹은 감소 값 크기
Hold	Step size가 유지되는 횟수
Rising	Step size만큼 상승되는 함수 유형 예) Step size가 30일 경우, 0,30,60.....
Falling	size만큼 감소되는 함수 유형 예) Step size가 30일 경우, 100,70,40.....
Alternate	함수의 결과값이 타입 최대값 혹은 최소값을 만난 경우 상승 혹은 감소로 바뀌는 함수 유형 예) Step size가 30이고 최대값 혹은 최소값이 0~100인 경우, 0,30,60,90,60,30,0.....

### e. Sine 함수

Sine값을 생성하는 함수 입니다. 샘플 개수가 함수의 주기보다 큰 경우는 함수가 재귀적으로 호출됩니다.

설정	설명
Amplitude	Sine 함수의 진폭
Period	Sine 함수의 주기
Phase	Sine 함수의 위상
Offset	Sine 함수의 offset

## f. Toggle 함수

FirstValue와 SecondValue를 반복적으로 생성하는 함수입니다.

설정	설명
First value	Toggle함수에서 반복되는 첫 번째 값
Second value	Toggle함수에서 반복되는 두 번째 값

## g. SingleValue 함수

일정한 단일 값만 반환하는 함수입니다.

설정	설명
Value	생성할 값

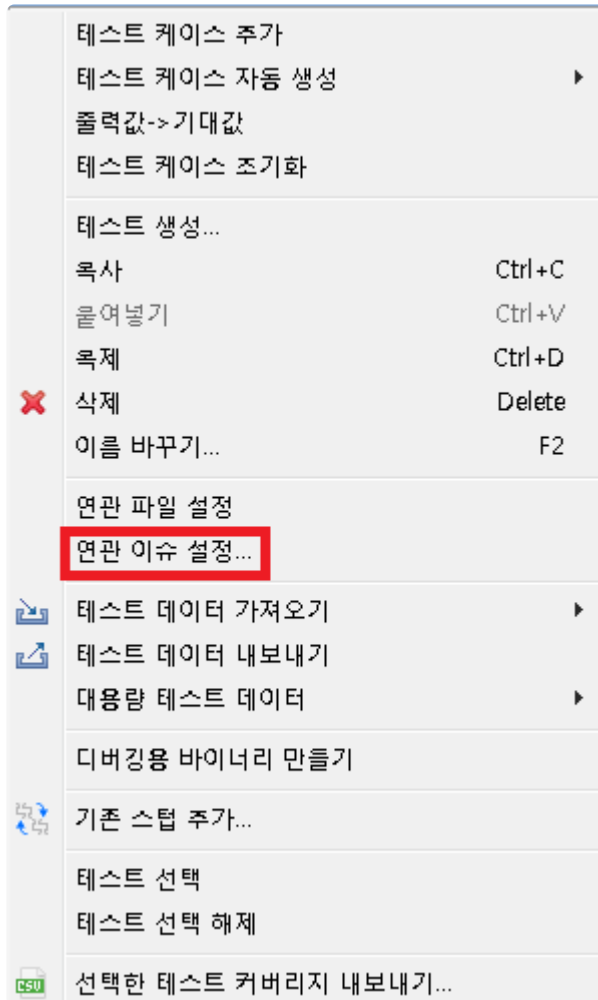
## h. None

함수를 생성하지 않습니다.

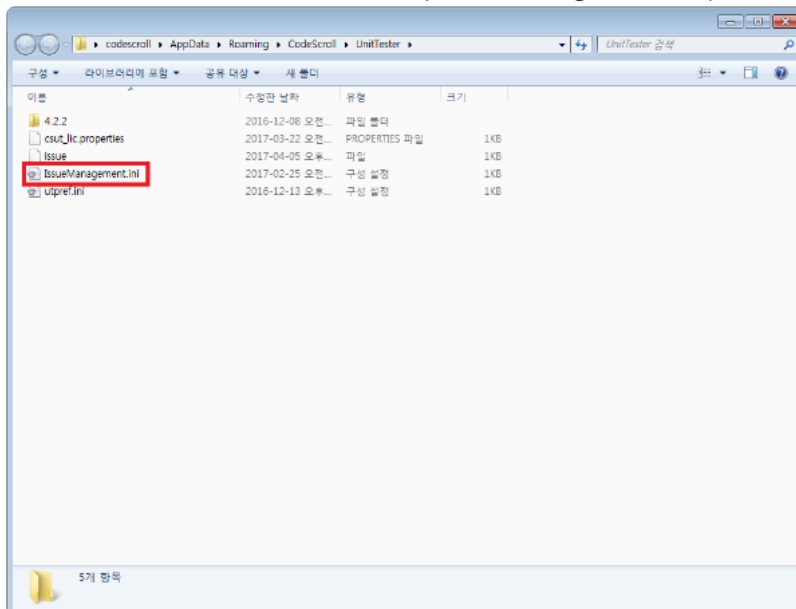
## 연관 이슈 설정

선택된 테스트를 등록된 이슈 관리 도구의 이슈와 연관시키는 기능을 제공합니다. CT 2023.12에서 지원하는 이슈 관리 도구의 종류는 JIRA, Trac, Redmine, Mantis, Bugzilla 입니다.

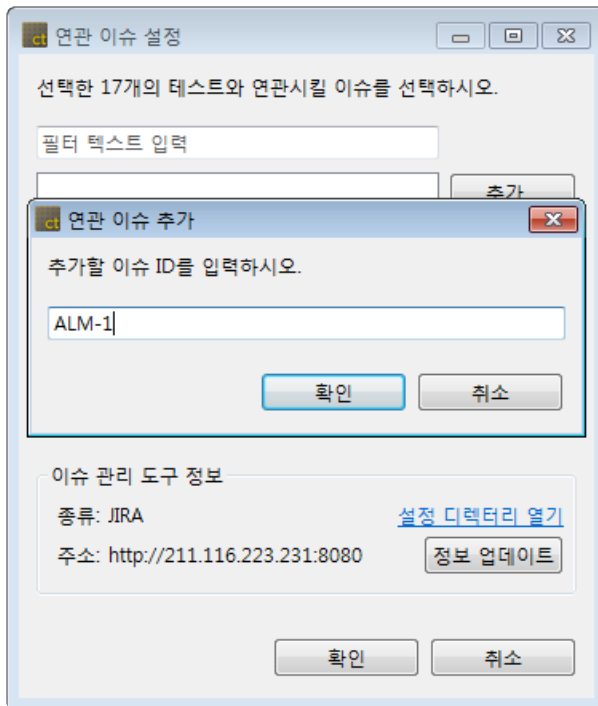
1. [유닛 테스트 뷰]의 컨텍스트 메뉴 중 [연관 이슈 설정]을 클릭합니다.



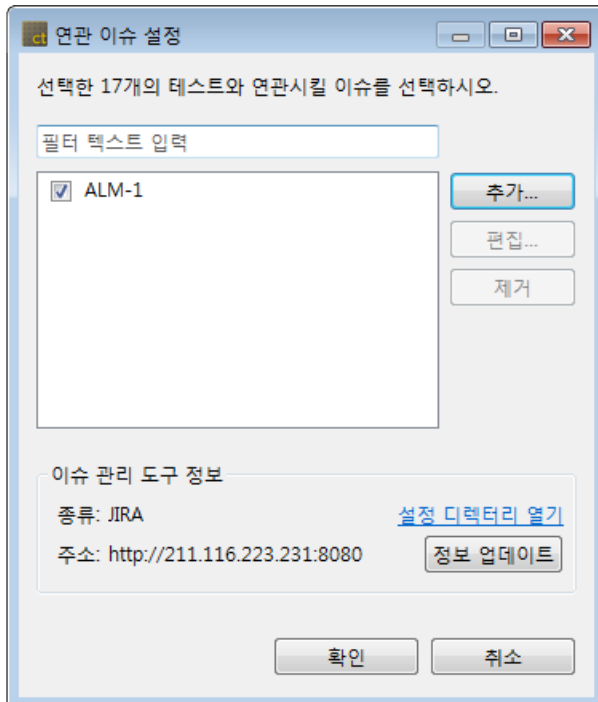
2. 이슈 관리 도구 정보를 설정 파일(IssueManagement.ini)에 입력합니다.



3. 테스트와 연관 시키려는 이슈를 추가합니다.

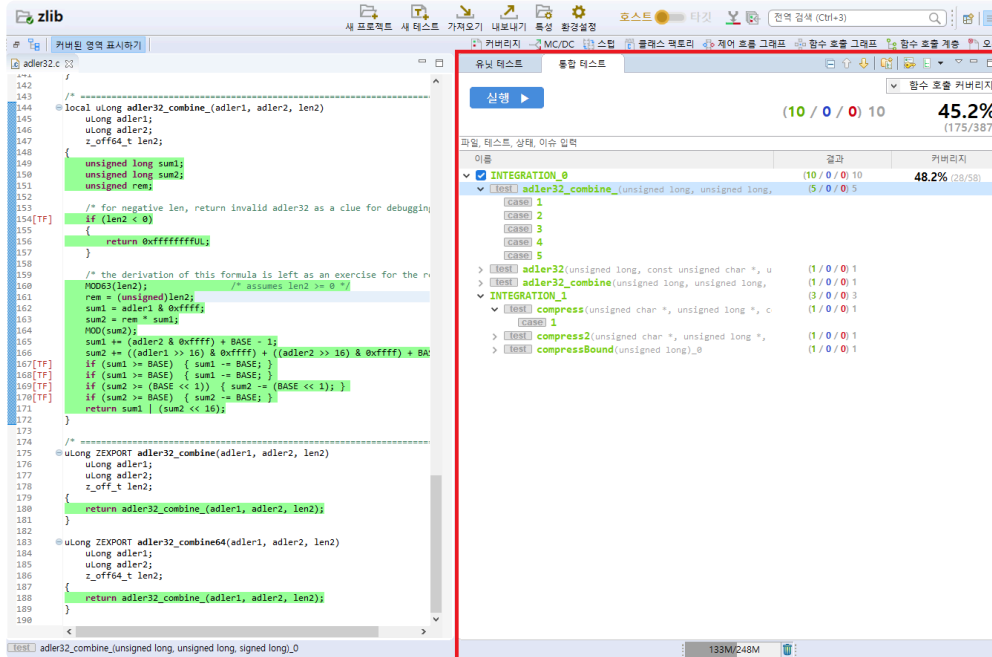


4. 체크된 이슈 목록을 확인한 뒤, [확인]을 클릭합니다.





## 17.4. 통합 테스트 뷰

[통합 테스트 뷰]에서는 여러 유닛 테스트를 묶을 수 있는 단위인 통합 테스트를 제공합니다. 통합 테스트로 묶인 유닛 테스트들은 컨텍스트 유지 하여 순서대로 실행합니다.

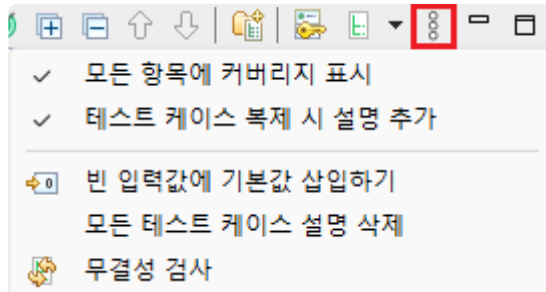


### [통합 테스트 뷰]의 툴바 메뉴

툴바 아이콘	설명
	결함 위치 찾기
	모두 펼치기
	모두 접기
	테스트 위로 이동
	테스트 아래로 이동
	생성
	고유한 테스트 이름으로 보기
	합계
	실패/오류
	실패
	오류
	성공
	함수 변경

 실행 보장 안됨	테스트 실행 보장 안되는 테스트만 보기
 호스트/타겟 결과 다름	호스트/타겟 결과가 다른 테스트만 보기

## [통합 테스트 뷰]의 풀다운 메뉴



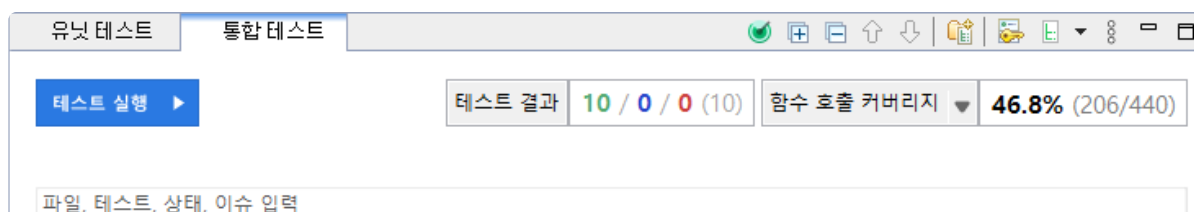
메뉴 이름	설명
모든 항목에 커버리지 표시	모든 항목(통합 테스트, 테스트, 테스트 케이스)에 커버리지 표시
테스트 케이스 복제 시 설명 추가	복제한 테스트 케이스에 원본 테스트 케이스 번호 표시
빈 입력값에 기본값 삽입하기	테스트 케이스에 입력값이 없는 경우 기본값 삽입
모든 테스트 케이스 설명 삭제	테스트 케이스에 입력된 설명을 일괄 삭제
무결성 검사	테스트와 스텝의 무결성 검사 후 필요 시 테스트 재설정

- 빈 입력값에 기본값 삽입하기

변수 종류	기본값
문자열	(빈 입력값)
문자열이 아닌 변수	0

✿ [테스트 재설정]에 대한 자세한 설명은 User Guide 2023.12의 [\[소스 코드 변경 후 테스트 재설정\]](#) 페이지를 참고하시기 바랍니다.

## [통합 테스트 뷰] 대시보드

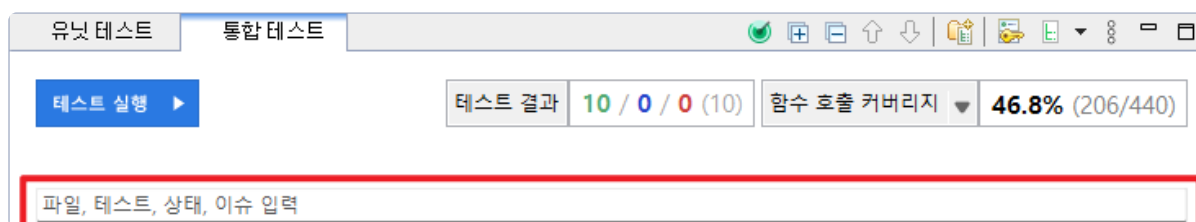


메뉴	설명
----	----

<b>테스트 실행</b> ▶	통합 테스트 실행 버튼
<b>테스트 결과</b> 915 / 0 / 261 (1176)	테스트 케이스의 성공, 실패, 오류, 총합 표현
구문 커버리지 ▼	[통합 테스트 뷰]에 표시할 커버리지 종류(구문, 분기, MC/DC, 함수 호출, 함수) 선택
<b>43.1%</b> (2008/4652)	선택한 종류의 전체 커버리지 표현

## 검색

[통합 테스트 뷰]는 파일, 함수, 테스트, 상태(성공, 실패, 오류), 이슈의 이름을 통해 통합 테스트, 테스트, 테스트 케이스를 검색할 수 있습니다.



## 상태 검색 키워드

[통합 테스트 뷰]에서는 상태 검색 키워드를 통해 표시되는 테스트를 필터링할 수 있습니다. 검색 가능한 키워드의 추천 및 자동완성 기능을 제공합니다.

키워드	설명
%TEST_SUCCESS%	테스트 성공
%TEST_FAILURE% %TEST_ERROR%	테스트 실패/오류
%TEST_FAILURE%	테스트 실패
%TEST_ERROR%	테스트 오류
%TEST_HAS_NOT_FUNCTION%	함수 변경
%TEST_NOT_GUARANTEE%	실행 보장 안됨
%TEST_RESULT_DIFFERENT%	호스트/타겟 결과 다름

## [통합 테스트 뷰]의 구조

[통합 테스트 뷰]는 [통합 테스트] > [테스트] > [테스트 케이스]의 계층 구조로 표현합니다.

유닛 테스트

통합 테스트

테스트 실행

테스트 결과 10 / 0 / 0 (10) 함수 호출 커버리지 46.8% (206/440)

파일, 테스트, 상태, 이슈 입력

이름	결과	커버리지
<div> <div> <div> <div>INTEGRATION_0</div> <div> <div>TEST</div> <div>adler32_combine_(unsigned long, unsigned long,</div> <div> <div>CASE</div> <div>1</div> <div>CASE</div> <div>2</div> <div>CASE</div> <div>3</div> <div>CASE</div> <div>4</div> <div>CASE</div> <div>5</div> </div> </div> </div> </div> </div>	(10 / 0 / 0) 10 (5 / 0 / 0) 5	47.7% (32/67) N/A N/A N/A N/A N/A
> TEST	adler32_combine(unsigned long, unsigned long, s	(1 / 0 / 0) 1 100.0% (1/1)
> TEST	adler32(unsigned long, const unsigned char *, ur	(1 / 0 / 0) 1 100.0% (1/1)
INTEGRATION_1	(3 / 0 / 0) 3	46.9% (31/66)
<div> <div> <div> <div>compress2(unsigned char *, unsigned long *, c</div> <div> <div>CASE</div> <div>1</div> </div> </div> </div> </div>	(1 / 0 / 0) 1	46.1% (30/65) 46.1% (30/65)
> TEST	compressBound(unsigned long)_0	(1 / 0 / 0) 1 N/A
> TEST	compress(unsigned char *, unsigned long *, coi	(1 / 0 / 0) 1 46.9% (31/66)

## [통합 테스트 뷰]의 항목 아이콘

항목 아이콘	설명
아이콘 없음	통합 테스트
TEST	테스트
TEST	시나리오 테스트
TEST	전역 변수 테스트
CASE	테스트 케이스, 테스트 케이스 그룹

## [통합 테스트 뷰]의 항목 상태 표현

[통합 테스트 뷰]의 통합 테스트, 테스트, 테스트 케이스는 테스트 실행 여부 및 실행 결과를 색상으로 표현합니다.

색상	설명
녹색	통합 테스트/테스트: 하위에 모든 테스트 케이스의 실행 결과가 성공인 경우 테스트 케이스: 실행 결과가 성공인 경우
파란색	통합 테스트/테스트 : 하위에 실행 결과가 실패인 테스트 케이스가 존재하고 오류인 테스트 케이스는 존재하지 않음 테스트 케이스: 실행 결과가 실패인 경우
붉은색	통합 테스트/테스트: 하위에 실행 결과가 오류인 테스트 케이스가 존재함 테스트 케이스: 실행 결과가 오류인 경우
주황색	통합 테스트/테스트: 하위에 모든 테스트 케이스를 실행하지 않았고 실행 보장 안 됨 테스트 케이스가 존재함 테스트 케이스: 실행 보장 안 됨



검정 색	통합 테스트/테스트: 하위에 모든 테스트 케이스를 실행하지 않음 테스트 케이스: 실행하지 않음
---------	---

## 통합 테스트 컨텍스트 메뉴



컨텍스트 메뉴	설명
테스트 생성	테스트 생성 마법사 열기
하위 통합 테스트 생성	하위 통합 테스트 생성
복사	테스트 및 테스트 케이스 복사
붙여넣기	복사한 테스트 및 테스트 케이스 붙여넣기
복제	테스트 및 테스트 케이스 복제
여러 번 복제	테스트 및 테스트 케이스를 입력한 수만큼 복제
하위 통합 테스트로 붙여넣기	복사한 통합 테스트를 선택한 통합 테스트의 하위 통합 테스트로 붙여넣기
삭제	테스트 및 테스트 케이스 삭제
이름 바꾸기	테스트 이름 변경
테스트 선택	선택된 모든 테스트의 실행 체크박스 상태를 체크로 변경
테스트 선택 해제	선택된 모든 테스트의 실행 체크박스 상태를 체크 해제로 변경
결함 위치 찾기	오류 또는 실패한 테스트 케이스로부터 결함 위치 확인

## 통합 테스트 생성 및 이름 변경

툴바 메뉴에서 통합 테스트를 생성할 수 있습니다. 통합 테스트 이름은 생성 시 자동으로 부여되며, 오른쪽 클릭 후 [이름 바꾸기]를 선택하여 변경할 수 있습니다.

- 생성

이름	결과	커버리지
> <b>INTEGRATION_0</b>	(98 / 0 / 0) 98	0.0% (0/46)

- 이름 바꾸기

## 하위 통합 테스트

통합 테스트에 하위 통합 테스트를 추가할 수 있습니다. [하위 통합 테스트 생성] 컨텍스트 메뉴를 이용해 추가할 수도 있고, 이미 만들어진 통합 테스트를 [하위 통합 테스트 붙여넣기] 컨텍스트 메뉴를 이용하여 붙여넣을 수도 있습니다.

이름	결과	커버리지
> <b>INTEGRATION_0</b>	(28 / 0 / 0) 28	0.0% (0/10)
> test adler32_combine(unsigned long, unsigned lon	(11 / 0 / 0) 11	
> <b>INTEGRATION_1</b>	(17 / 0 / 0) 17	
> test inflateEnd(struct z_stream_s *)_0_copy_0	(1 / 0 / 0) 1	
> test inflateCopy(struct z_stream_s *, struct z	(1 / 0 / 0) 1	
> test inflateBackInit_(struct z_stream_s *, si	(14 / 0 / 0) 14	

## 테스트 추가 및 실행 순서 변경

통합 테스트에 테스트를 추가하는 방법은 네 가지 방법이 있습니다.

1. 통합 테스트를 선택 후 [테스트 생성] 컨텍스트 메뉴를 이용하여 생성할 수 있습니다.
2. [유닛 테스트 뷰]의 테스트를 끌어서 놓아 추가할 수 있습니다.

3. [통합 테스트 뷰]의 테스트를 선택 후 컨텍스트 메뉴를 이용하여 복사/붙여넣기 할 수 있습니다.
4. [유닛 테스트 뷰]의 테스트를 선택 후 컨텍스트 메뉴를 이용하여 복사/붙여넣기 할 수 있습니다.

테스트 실행 순서를 변경하는 방법은 두 가지 방법이 있습니다.

1. 테스트 또는 테스트 케이스를 선택 후 마우스로 끌어서 순서를 변경할 수 있습니다.
2. 테스트 또는 테스트 케이스를 선택 후 툴바에 있는 [테스트 위로 이동], [테스트 아래로 이동]을 선택하여 변경할 수 있습니다.

유닛 테스트 통합 테스트 실행 ▶ 함수 호출 커버리지

(28 / 0 / 0) 28 41.0% (130/317)

파일, 테스트, 상태, 이슈 입력

이름	결과	커버리지
✓ INTEGRATION_0	(28 / 0 / 0) 28	0.0% (0/10)
> test adler32_combine(unsigned long, unsigned lon	(11 / 0 / 0) 11	
✓ INTEGRATION_1	(17 / 0 / 0) 17	
> test inflateEnd(struct z_stream_s *)_0_copy_0	(1 / 0 / 0) 1	
> test inflateCopy(struct z_stream_s *, struct z	(1 / 0 / 0) 1	
> test inflateBackInit_(struct z_stream_s *, si	(14 / 0 / 0) 14	
> test inflateBackEnd(struct z_stream_s *)_0_co	(1 / 0 / 0) 1	

## 통합 테스트 커버리지 결과 보기

통합 테스트를 선택하면 커버리지 항목에서 결과를 확인할 수 있습니다.

유닛 테스트 통합 테스트 실행 ▶ 구문 커버리지

(28 / 0 / 0) 28 41.3% (1616/3906)

파일, 테스트, 상태, 이슈 입력

이름	결과	커버리지
✓ INTEGRATION_0	(28 / 0 / 0) 28	39.0% (32/82)
> test adler32_combine(unsigned long, unsigned lon	(11 / 0 / 0) 11	
✓ INTEGRATION_1	(17 / 0 / 0) 17	
> test inflateEnd(struct z_stream_s *)_0_copy_0	(1 / 0 / 0) 1	
> test inflateCopy(struct z_stream_s *, struct z	(1 / 0 / 0) 1	
> test inflateBackInit_(struct z_stream_s *, si	(14 / 0 / 0) 14	
> test inflateBackEnd(struct z_stream_s *)_0_co	(1 / 0 / 0) 1	

## 17.5. 커버리지 뷰

테스트 수행 후 해당 테스트 대상의 함수가 얼마만큼을 커버했는지 백분율로 보여줍니다.

커버리지 오류

'libc' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	_tr_align(struct internal_state *)	71.42% (15/21)	75.00% (3/4)	50.00% (1/2)	100.00% (1/1)	Y
2	_tr_flush_bits(struct internal_state *)	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
3	_tr_flush_block(struct internal_state *, char *, unsigned lo...	57.14% (24/42)	68.75% (11/16)	50.00% (5/10)	80.00% (8/10)	Y
4	_tr_init(struct internal_state *)	100.00% (10/10)	N/A	N/A	100.00% (2/2)	Y
5	_tr_stored_block(struct internal_state *, char *, unsigned l...	100.00% (11/11)	100.00% (2/2)	100.00% (1/1)	100.00% (1/1)	Y
6	_tr_tally(struct internal_state *, unsigned int, unsigned int)	22.22% (2/9)	0.00% (0/4)	0.00% (0/2)	N/A	Y
7	adler32(unsigned long, const unsigned char *, unsigned lo...	98.05% (101/103)	87.50% (21/24)	75.00% (9/12)	N/A	Y
8	adler32_combine(unsigned long, unsigned long, signed lo...	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
9	adler32_combine64(unsigned long, unsigned long, signed lo...	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
10	adler32_combine_(unsigned long, unsigned long, signed lo...	100.00% (21/21)	100.00% (10/10)	100.00% (5/5)	N/A	Y
11	bi_flush(struct internal_state *)	100.00% (9/9)	100.00% (4/4)	100.00% (2/2)	N/A	Y
12	bi_reverse(unsigned int, signed int)	100.00% (5/5)	100.00% (2/2)	100.00% (1/1)	N/A	Y
합계		43.88% (1962/...	30.08% (723/2...	18.53% (265/1...	44.75% (175/3...	100.00% ...

타깃 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

## CT 2023.12에서 제공하는 커버리지 종류

커버리지	설명
구문	테스트에 의해 수행되는 소스코드 구문의 백분율
분기	테스트에 의해 수행된 분기의 백분율 100% 분기 커버리지는 100%의 결정 커버리지와 100% 구문 커버리지를 포함
MC/DC	조건 커버리지와 결정 커버리지를 복합적으로 고려한 커버리지 (MC/DC 뷰에서 자세히 설명)
함수 호출	전체 함수 중 테스트에 의해 호출되는 함수의 백분율
함수	전체 함수 중 한 번이라도 호출된 함수의 백분율

## [커버리지 뷰] 레이블 아이콘

아이콘	설명
	변경된 함수
	Asm 코드가 포함된 함수

## [커버리지 뷰] 툴바 메뉴

툴바 아이콘	설명
	테스트별 커버리지 보기
	전체 커버리지 보기
	전체 커버리지 보기(외부 커버리지 포함)

## 커버리지 보기

[커버리지 뷰]나 [소스 코드 편집기] 영역에서 테스트의 커버리지를 확인할 수 있습니다.

### [커버리지 뷰]에서 보기

[커버리지 뷰]에서는 [유닛 테스트 뷰]와 [통합 테스트 뷰]에서 선택한 테스트 또는 테스트 케이스의 커버리지 정보를 보여줍니다. [전체 커버리지 보기]를 클릭하면, 전체 테스트의 커버리지 정보를 병합해서 볼 수 있습니다. 그리고 [전체 커버리지 보기(외부 커버리지 포함)]를 클릭하면, 외부에서 가져온 커버리지까지 병합해서 보여줍니다.

[커버리지 뷰] 하단에는 트리에서 선택한 항목의 커버리지가 표시됩니다.

합계	43.88% (1962/...	30.08% (723/2...	18.53% (265/1...	44.75% (175/3...	100.00% ...
----	------------------	------------------	------------------	------------------	-------------

[커버리지 뷰]에는 [유닛 테스트 뷰] 또는 [통합 테스트 뷰]에서 선택한 항목에 대하여 각 함수에 대한 커버리지가 표시됩니다.

커버리지 오류

'zlib' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	_tr_align(struct internal_state *)	71.42% (15/21)	75.00% (3/4)	50.00% (1/2)	100.00% (1/1)	Y
2	_tr_flush_bits(struct internal_state *)	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
3	_tr_flush_block(struct internal_state *, char *, unsigned lo...	57.14% (24/42)	68.75% (11/16)	50.00% (5/10)	80.00% (8/10)	Y
4	_tr_init(struct internal_state *)	100.00% (10/10)	N/A	N/A	100.00% (2/2)	Y
5	_tr_stored_block(struct internal_state *, char *, unsigned l...	100.00% (11/11)	100.00% (2/2)	100.00% (1/1)	100.00% (1/1)	Y
6	_tr_tally(struct internal_state *, unsigned int, unsigned int)	22.22% (2/9)	0.00% (0/4)	0.00% (0/2)	N/A	Y
7	adler32(unsigned long, const unsigned char *, unsigned...	98.05% (101/103)	87.50% (21/24)	75.00% (9/12)	N/A	Y
8	adler32_combine(unsigned long, unsigned long, signed ...	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
9	adler32_combine64(unsigned long, unsigned long, sign...	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
10	adler32_combine_(unsigned long, unsigned long, signed...	100.00% (21/21)	100.00% (10/10)	100.00% (5/5)	N/A	Y
11	bi_flush(struct internal_state *)	100.00% (9/9)	100.00% (4/4)	100.00% (2/2)	N/A	Y
12	bi_reverse(unsigned int, signed int)	100.00% (5/5)	100.00% (2/2)	100.00% (1/1)	N/A	Y
합계		43.88% (1962/...	30.08% (723/2...	18.53% (265/1...	44.75% (175/3...	100.00% ...

타깃 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

[커버리지 뷰] 테이블에서 컨텍스트 메뉴를 통하여 각 함수의 제어 흐름 그래프, 함수 호출 그래프, MC/DC를 확인할 수 있습니다.


커버리지 오류

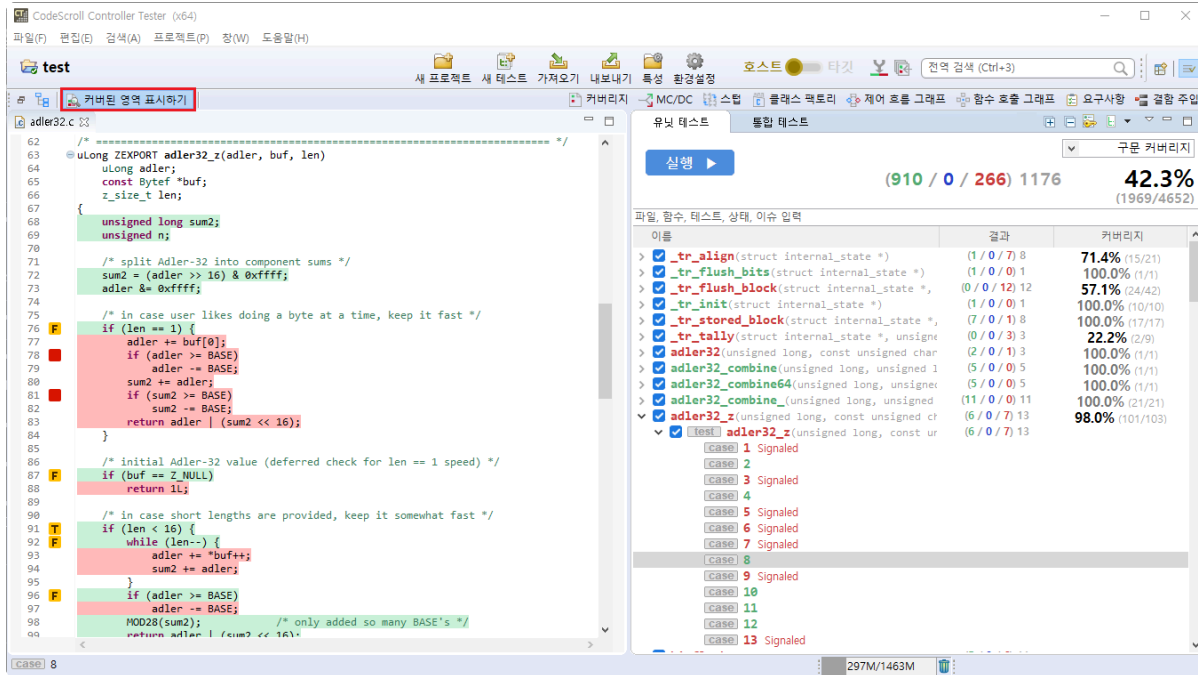
'zlib' 프로젝트의 호스트 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	_tr_align(struct internal_state *)	72.72% (32/...	70.00% (7/...	4...		Y
2	_tr_flush_block(struct internal_state *, char *, unsign...	57.14% (24/42)	68.75% (11/16)	5...		Y
3	_tr_init(struct internal_state *)	100.00% (11/...	N/A	N/A		Y
4	_tr_stored_block(struct internal_state *, char *, unsig...	100.00% (11/...	100.00% (2/2)	1...		Y
5	_tr_tally(struct internal_state *, unsigned int, unsign...	22.22% (2/9)	0.00% (0/4)	0.00% (0/2)	N/A	Y
6	adler32(unsigned long, const unsigned char *, unsi...	97.08% (100/...	87.50% (21/24)	75.00% (9/12)	N/A	Y
7	adler32_combine(unsigned long, unsigned long, si...	100.00% (18/...	100.00% (8/8)	100.00% (4/4)	N/A	Y
8	adler32_combine_symbols(unsigned long, unsigned...	100.00% (18/...	87.50% (7/8)	75.00% (3/4)	N/A	Y
9	bi_flush(struct internal_state *)	100.00% (9/9)	100.00% (4/4)	100.00% (2/2)	N/A	Y
10	bi_reverse(unsigned int, signed int)	100.00% (5/5)	100.00% (2/2)	100.00% (1/1)	N/A	Y
11	bi_windup(struct internal_state *)	100.00% (7/7)	100.00% (4/4)	100.00% (2/2)	N/A	Y
12	build_bi_tree(struct internal_state *)	100.00% (11/...	75.00% (3/4)	50.00% (1/2)	100.00% (3/3)	Y
합계		41.37% (1616...	27.21% (563/...	18.02% (221/...	41.00% (130/...	100.00%...

타깃 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

## 소스 코드 편집기에서 보기

메인 툴바의 [  커버된 영역 표시하기 ] 아이콘을 활성화 시킨 후 [유닛 테스트 뷰] 또는 [통합 테스트 뷰]에서 원하는 함수/테스트/테스트 케이스를 클릭하면 해당 소스 코드의 어느 부분이 커버되었는지 색상 정보로 표시해 줍니다. 커버된 코드는 초록색으로, 커버되지 않은 코드는 붉은색으로, 일부만 커버된 코드는 노란색으로 표시됩니다.



[소스 코드 편집기] 왼쪽 세로 컬럼에서 분기문의 참/거짓 여부가 마커로 표시됩니다.

마커	설명
	분기가 true만 커버된 경우
	분기가 false만 커버된 경우
	분기가 true/false 모두 커버된 경우
	분기가 커버되지 않은 경우
	한 줄에 여러 분기가 있고 일부 분기만 커버된 경우

## 17.6. MC/DC 뷰

MC/DC 뷰에서는 변경 조건/결정 커버리지(Modified Condition/Decision Coverage, MC/DC)의 정보를 보여줍니다.

### MC/DC 커버리지

MC/DC란 각 개별 조건식이 다른 개별 조건식에 영향을 받지 않고 전체 조건식의 결과에 독립적으로 영향을 주도록 함으로써 조건/결정 커버리지를 향상시킨 것으로 결정 커버리지, 조건/결정 커버리지 보다 강력합니다.

다른 상태(condition)들의 변동이 없고 자신의 상태가 변경 되었을 때 결과값에 영향을 미치는 경우 해당 상태는 MC/DC를 만족한다 할 수 있고 이때 해당 진리표를 만드는 조건은 다음과 같습니다.

- 해당 디시전(결정문)의 상태는 모든 가능한 결과(참, 거짓)를 적어도 한 번씩 만족해야 한다.
- 디시전에 속한 모든 개별 조건식은 모든 가능한 결과(참, 거짓)를 적어도 한 번씩 만족해야 한다.
- 디시전에 속한 각각의 개별 조건식은 다른 개별 조건식에 영향을 받지 않고 자신이 속한 디시전의 결과값에 독립적으로 영향을 준다.

### MC/DC 뷰 테이블 설명

- 목표 커버리지를 만족하는 조합 목록 보기

아래 그림은 목표 커버리지를 달성하기 위해 커버해야 하는 조합을 쉽게 확인할 수 있는 뷰입니다.

좌측에는 선택한 함수에 대한 Decision 목록이 있고, 그 하위에는 목표 커버리지를 만족하는 조합 목록이 나타납니다.

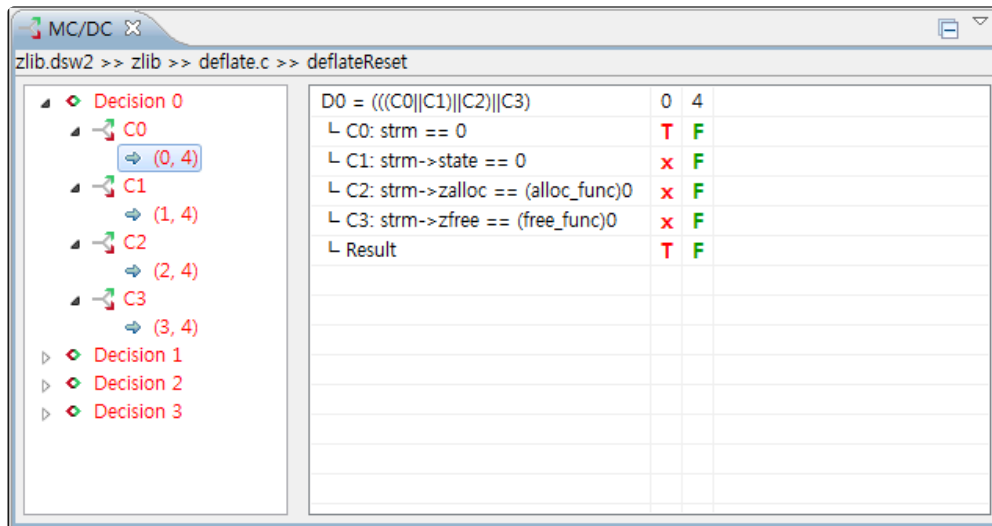
우측에서는 선택한 Decision 또는 조합의 진리표와 커버 여부를 보여줍니다. 커버된 조합은 녹색, 커버되지 않은 조합은 빨간색으로 표시됩니다.

환경 설정 페이지에서 목표 커버리지와 화면 보기 모드를 변경할 수 있습니다. 그리고 진리표를 전체 선택하여 'Ctrl + C' 또는 컨텍스트 메뉴로 클립보드 복사 기능을 사용할 수 있습니다.

	0	1	2	3	4
D0 = (((C0  C1)  C2)  C3)					
└ C0: strm == 0	T	F	F	F	F
└ C1: strm->state == 0	x	T	F	F	F
└ C2: strm->zalloc == (alloc_func)0	x	x	T	F	F
└ C3: strm->zfree == (free_func)0	x	x	x	T	F
└ Result	T	T	T	T	F

- 조건 별 커버리지를 만족하는 조합 목록 보기

아래 그림은 하나의 조건을 만족하는 한 쌍(Pair)의 조합을 확인할 수 있는 뷰입니다. 좌측 화면에는 각 조건 별 커버리지를 만족하는 조합 목록을 보여줍니다. 우측 화면에는 선택한 조합에 대한 진리표와 커버 여부 정보를 보여줍니다.



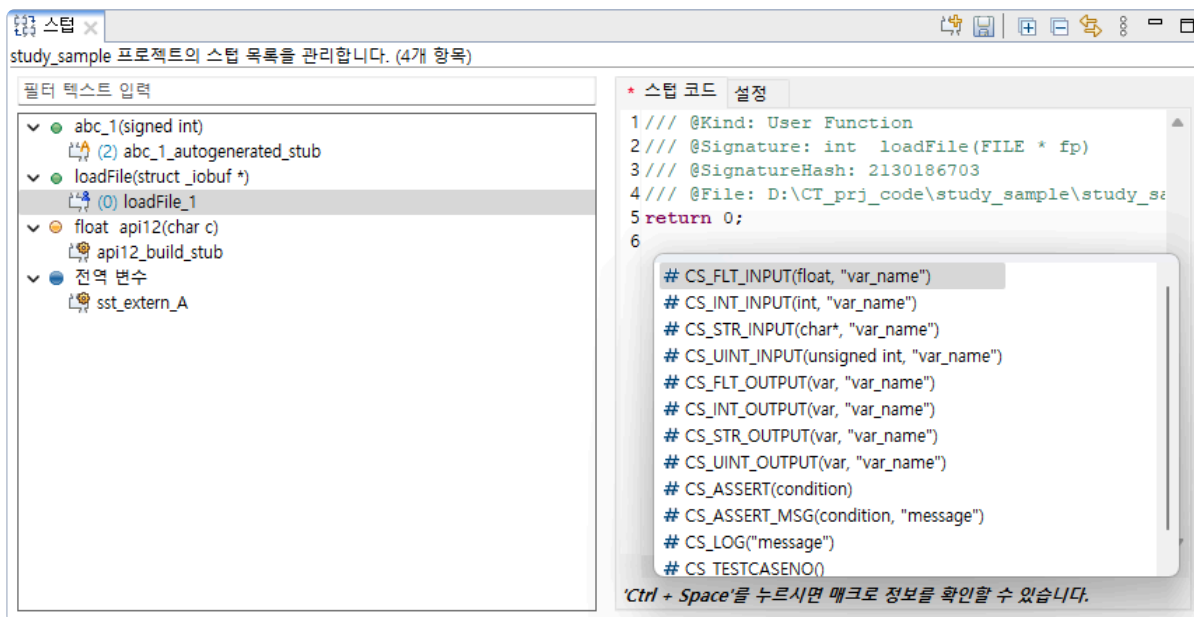


## 17.7. 스텝 뷰

CT 2023.12에서는 테스트 수행 시 대상 함수에서 사용하는 라이브러리가 없거나 아직 개발되지 않은 경우 혹은 선언되어 있지 않은 변수를 사용하는 경우, 이를 대신하는 스텝을 자동으로 생성합니다. 이때 만들어진 스텝은 테스트 수행 시 생성되며 함수 내부는 구현되어 있지 않습니다.

CT 2023.12가 자동으로 생성한 스텝 외에 사용자가 스텝을 직접 추가할 수 있습니다. 동일한 함수에 대하여 여러 개의 스텝을 생성할 수 있으며 생성된 스텝은 각각의 테스트에 연결할 수 있습니다.

스텝 내부에서 사용자 매크로를 사용하여 각 스텝의 동작을 제어할 수 있습니다. 사용자가 입력한 매크로에서 사용하는 값은 연결된 테스트의 테스트 케이스 편집 탭에서 제어할 수 있습니다.



## 함수 타입

타입	설명
함수	정의가 없는 함수
함수	정의가 있는 함수
변수	전역 변수

## 스텝 타입

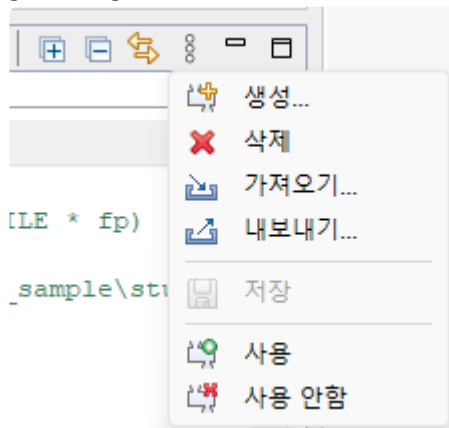
타입	설명
사용자 스텝	사용자 정의 스텝
빌드 스텝	자동 생성된 스텝
구 버전 스텝	2.3 이하 버전의 스텝
고립화 스텝	테스트 대상 함수를 분리하기 위한 스텝

## 메뉴

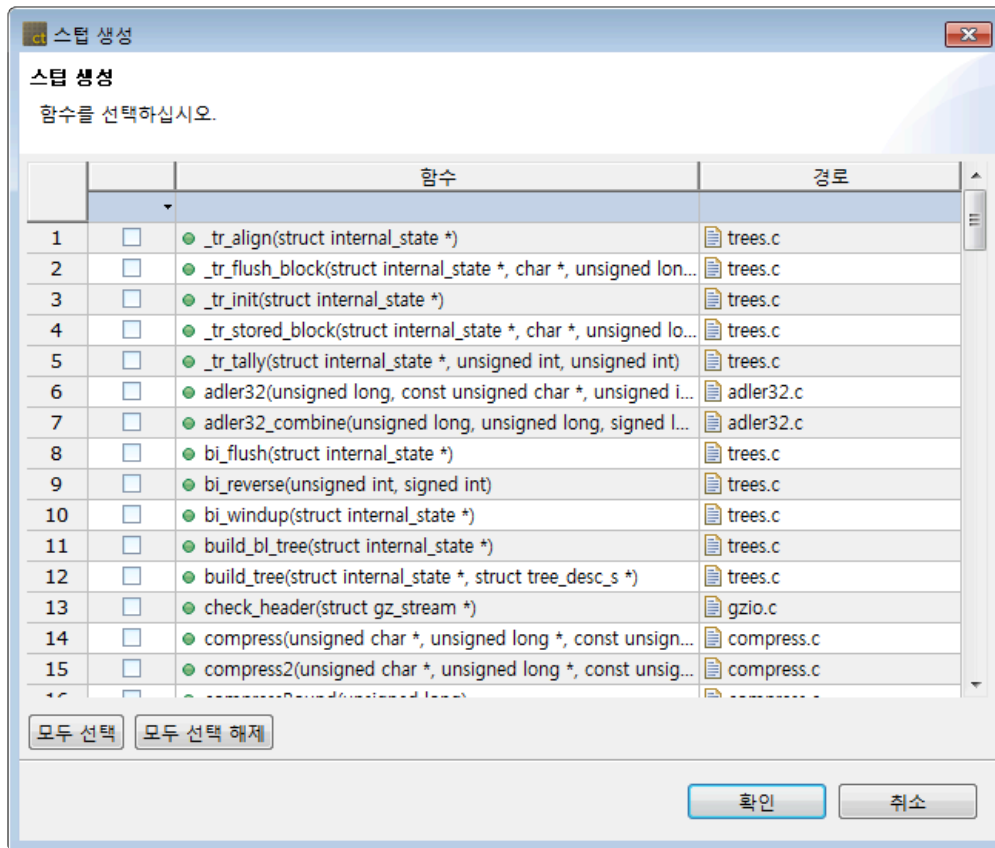
메뉴	설명
 생성	스텝 생성
 삭제	스텝 삭제
 가져오기	내보낸 스텝 파일 가져오기
 내보내기	스텝 파일 내보내기
 저장	변경된 스텝 정보 저장하기
 사용	스텝 활성화
 사용 안함	스텝 비활성화

## 스텝 생성

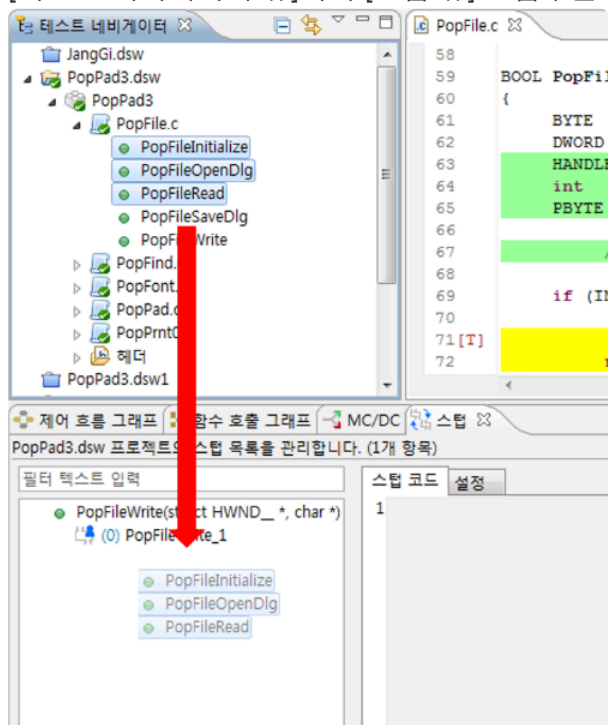
- [스텝 뷰]에서 생성하기
  1. [스텝 뷰]의 풀다운 메뉴를 클릭하여 [생성] 메뉴를 클릭합니다.



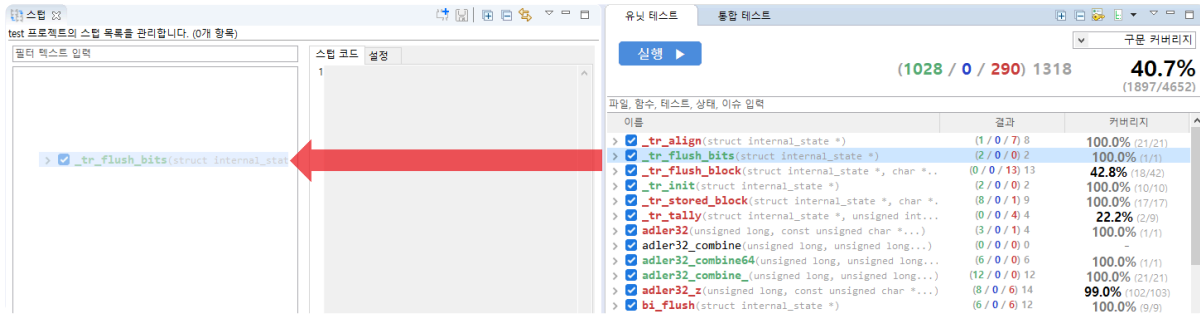
2. 스텝으로 생성할 함수를 체크한 후 [확인] 버튼을 선택합니다.



- [테스트 네비게이터 뷰]에서 [스텝 뷰]로 함수를 끌어 놓아 생성하기

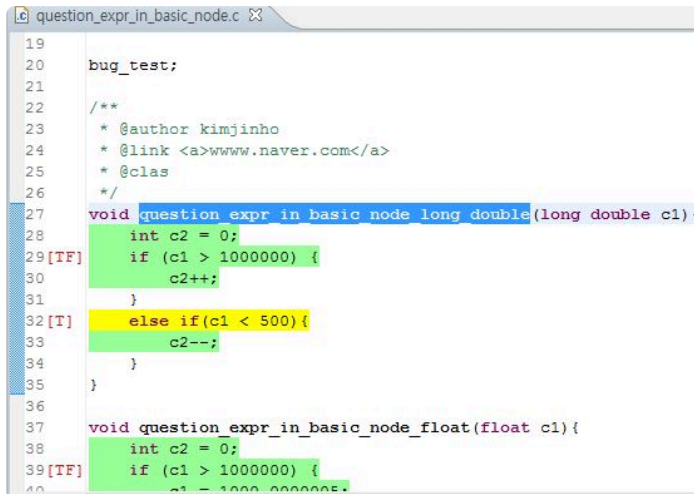


- [유닛 테스트 뷰]에서 [스텝 뷰]로 함수 노드를 끌어 놓아 생성하기

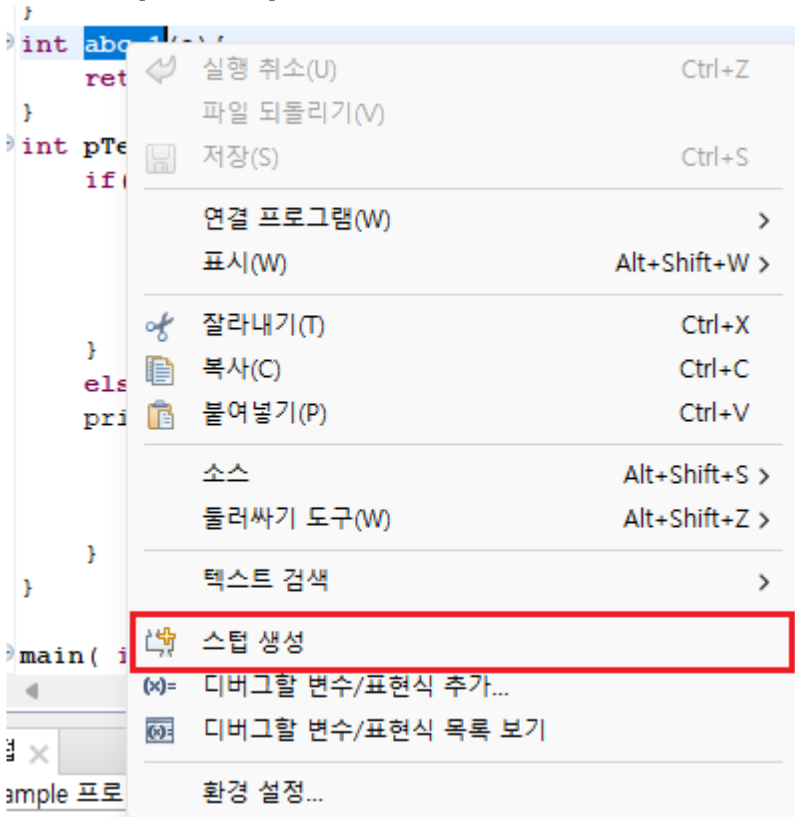


- 소스 코드 편집기에서 생성하기

1. 스텝으로 생성할 함수를 선택합니다.

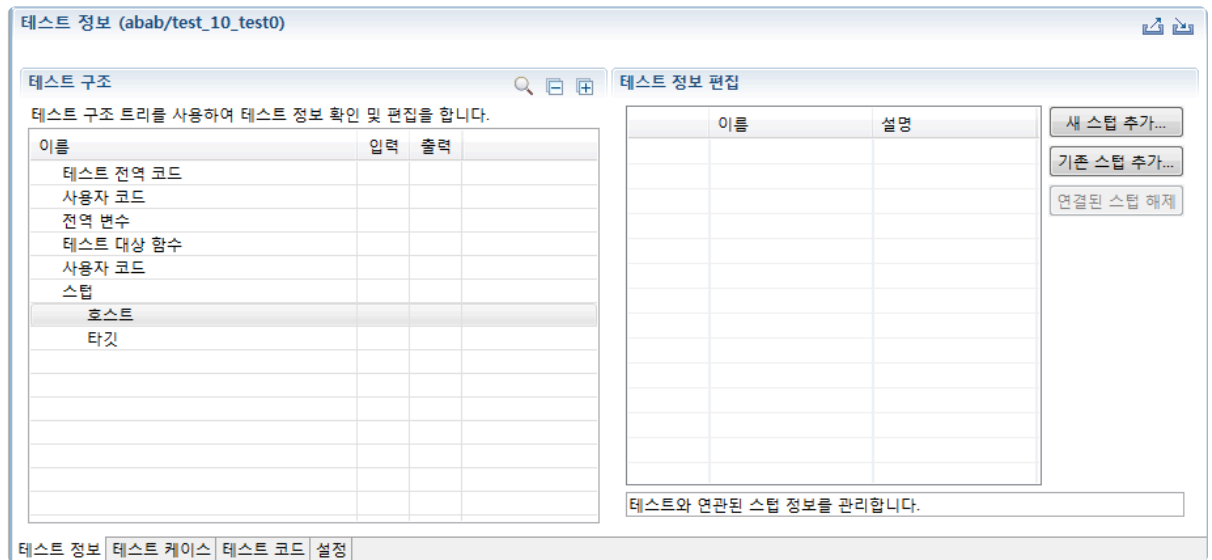


2. 우클릭 후 [스텝 생성] 컨텍스트 메뉴를 선택합니다.

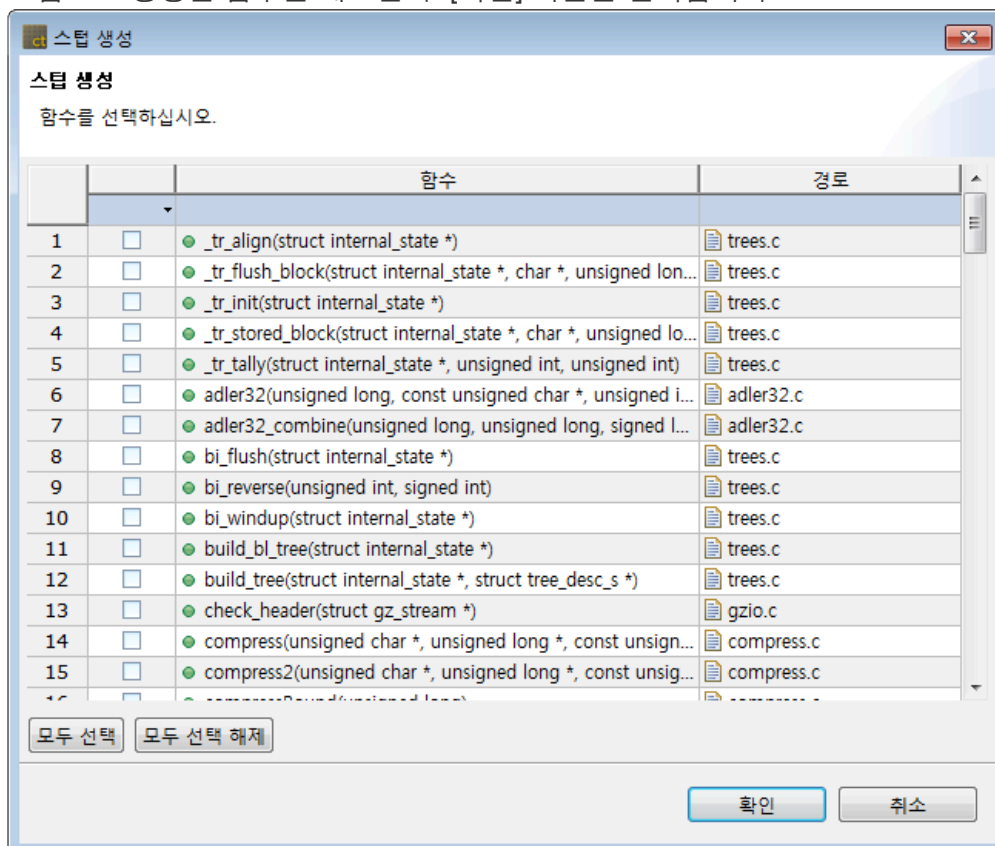


- 테스트 에디터에서 생성 후 연결하기

1. 테스트 에디터의 테스트 구조에서 스텝 또는 하위 노드를 선택합니다.
2. 우측 화면에 있는 [새 스텝 추가...] 버튼을 선택합니다.

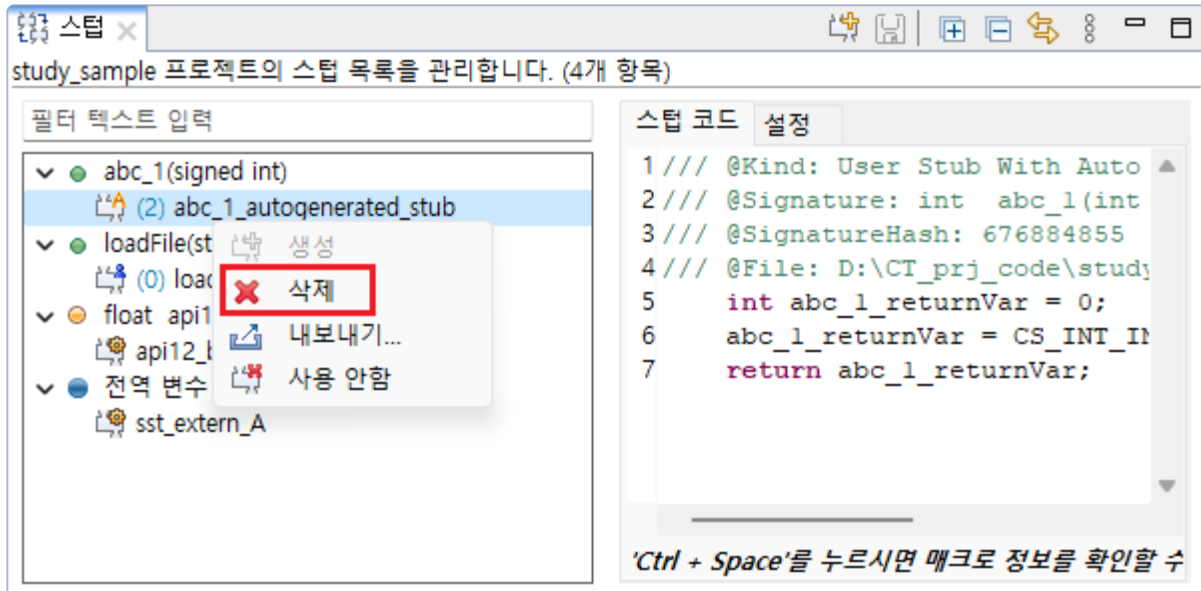


3. 스텝으로 생성할 함수를 체크한 후 [확인] 버튼을 선택합니다.



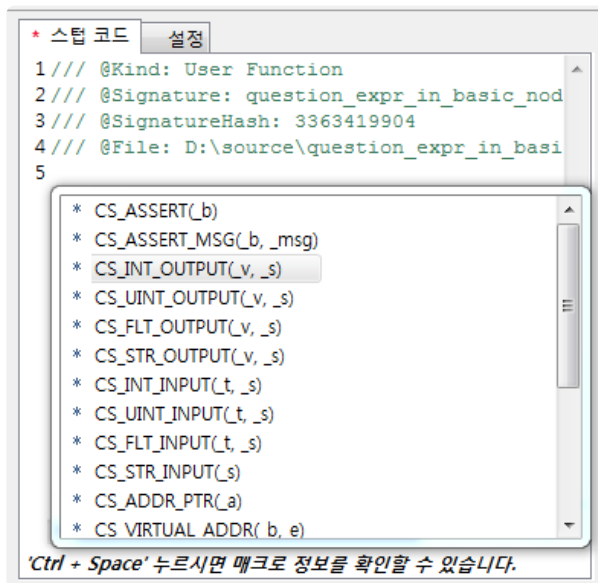
## 스텝 삭제

- 불필요한 스텝을 선택하고 우클릭하여 [삭제] 메뉴를 선택합니다.



## 스텝 편집

- 스텝 코드 화면에서 'Ctrl + Space' 누르면 매크로 정보를 확인할 수 있습니다. 스텝 코드를 편집 후 'Ctrl + S'를 누르거나, [저장] 톨바 버튼을 누르면 저장됩니다.



- 스텝 설정 화면에서는 설명을 편집할 수 있습니다.

스텝 코드	설정
스텝 키:	1986616262142690049
연관 파일:	E:\#02.OpenSource#wzlib-1.2.8(ALM)#wzlib-1.2.8#adler32.c
연관 테스트:	adler32(unsigned long, const unsigned char *, unsigned int)_0-호스트 adler32(unsigned long, const unsigned char *, unsigned int)_0-타깃
설명:	스텝 설명
저장	

## 스텝과 테스트 연결하기

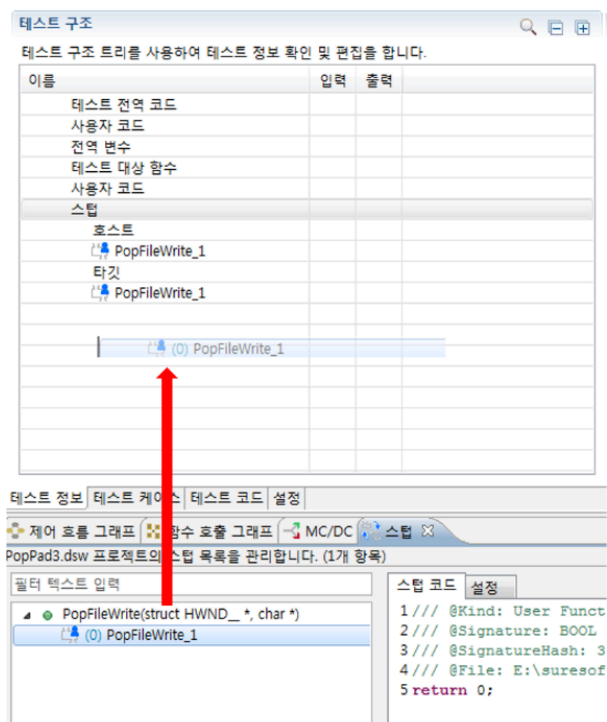
각 스텝은 호스트/타깃 테스트에 연결 가능합니다.

테스트 구조 편집기의 “스텝”을 선택 하면 호스트와 타깃에 속한 모든 스텝을 편집할 수 있습니다.

스텝
호스트
타깃

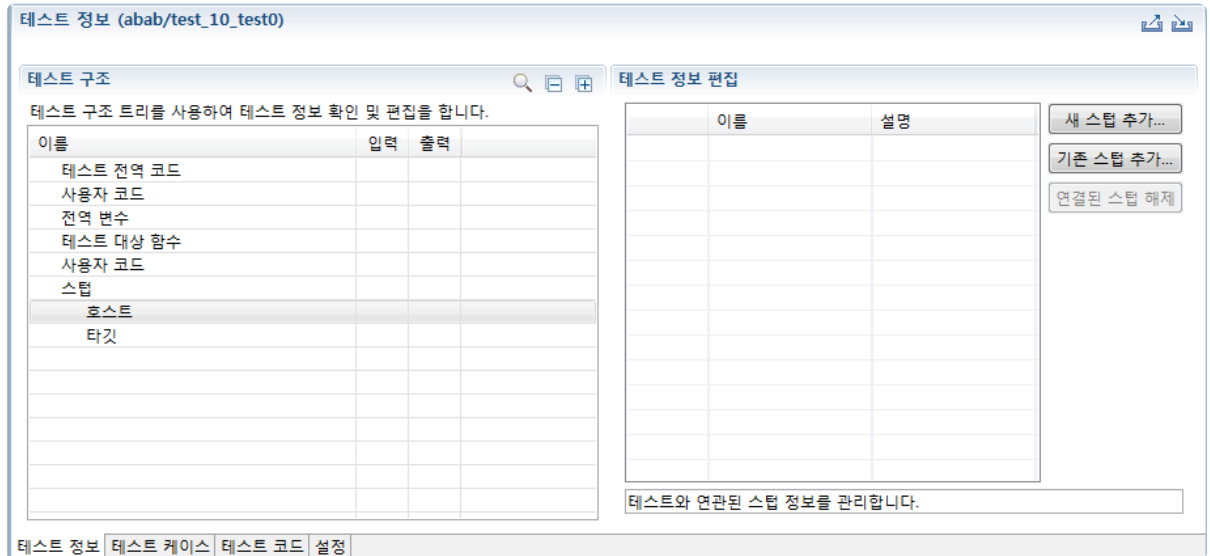
“호스트” 또는 “타깃”을 선택하면 선택된 시험 환경의 스텝만 편집할 수 있습니다.

- 스텝 뷰의 스텝을 테스트 에디터에 끌어서 놓기로 연결하기

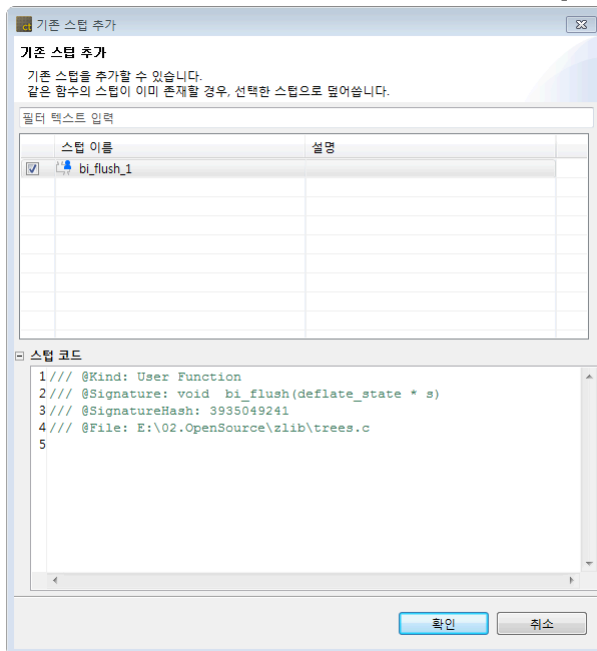


✿ 끌어서 놓기로 연결 시 타깃과 호스트 전부에 연결이 됩니다.

- 테스트 에디터에서 [기존 스텝 추가...] 버튼으로 연결하기
  1. 테스트 에디터의 테스트 구조에서 스텝을 선택합니다.
  2. 우측 화면에 있는 [기존 스텝 추가...] 버튼을 선택합니다.



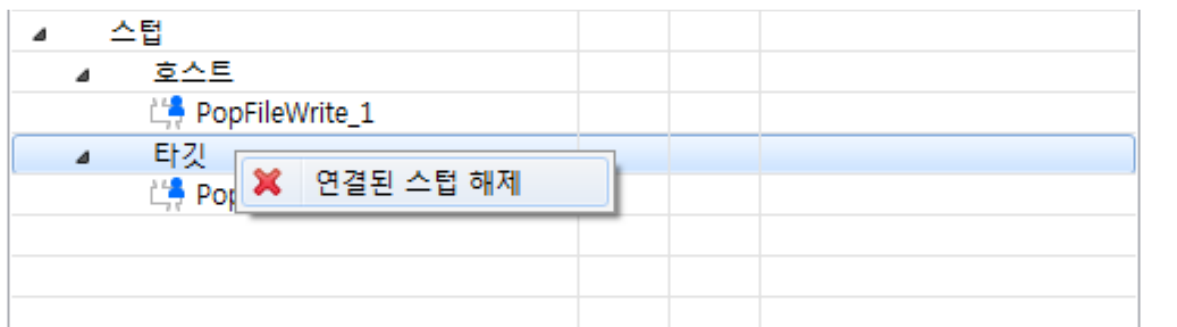
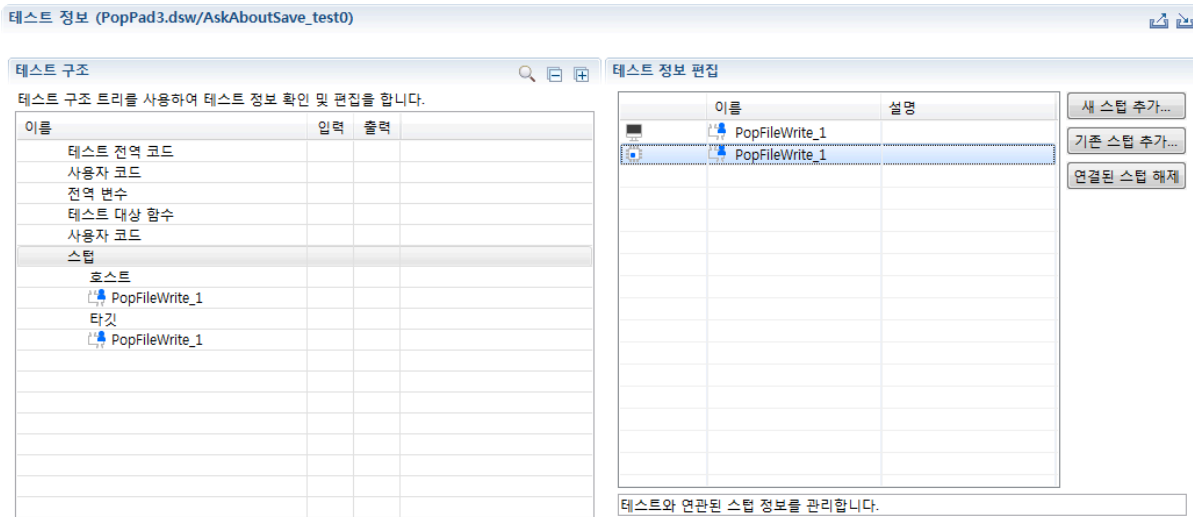
3. 기존 스텝 목록 중 연결할 스텝을 체크한 후 [확인] 버튼을 선택합니다.



## 스텝과 테스트 연결 해제하기

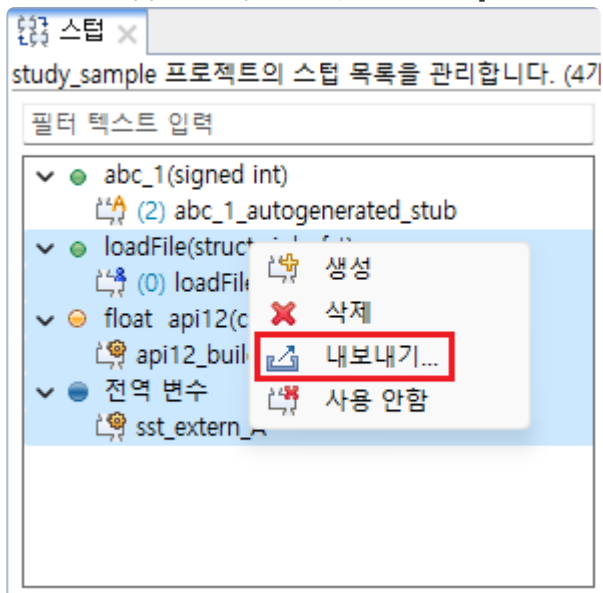
테스트 편집기에서 연결되어 있는 스텝을 선택한 후 [연결 스텝 해제] 버튼을 선택을 하거나, 테스트 구조 트리에서 스텝을 선택한 후 [Delete] 키 또는 우클릭하여 연결 해제를 할 수 있습니다.



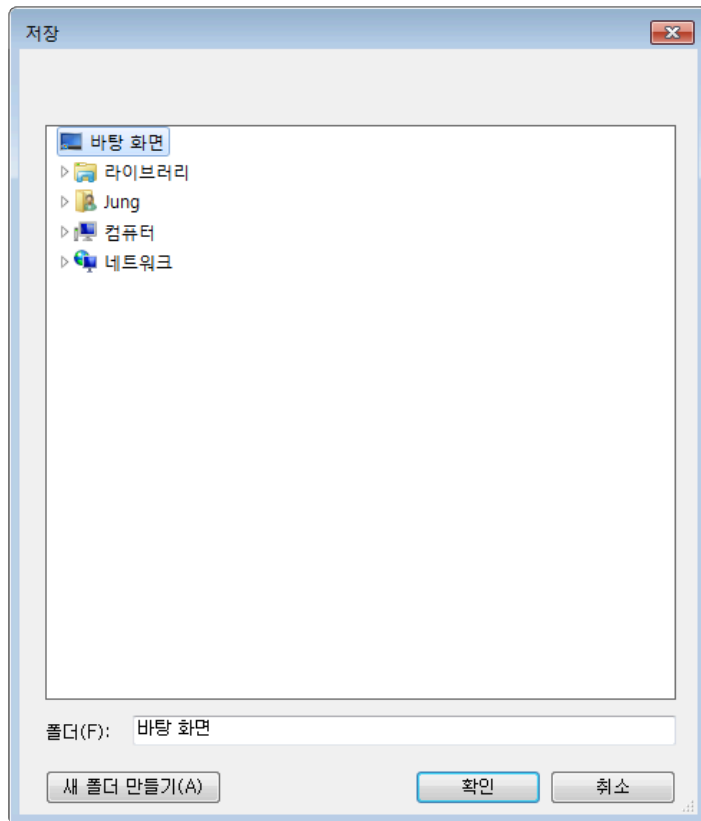


## 스텝 내보내기

1. 내보낼 스텝을 선택한 후 우클릭으로 [내보내기...] 메뉴를 선택합니다.

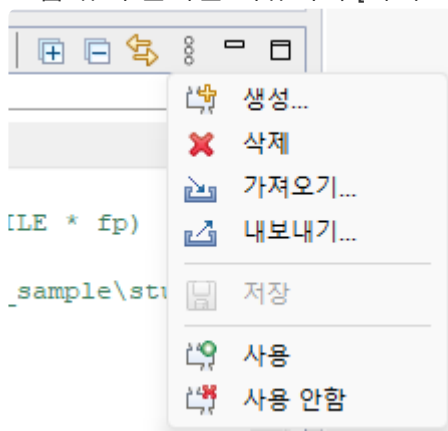


2. 디렉터리 다이얼로그에서 스텝 파일의 저장 위치를 지정한 후 [확인] 버튼을 선택합니다.

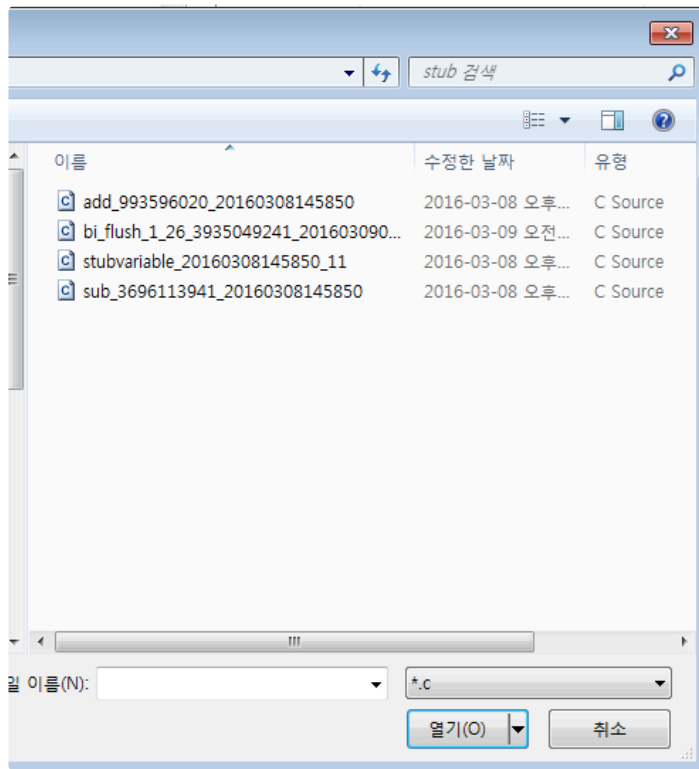


## 스텝 가져오기

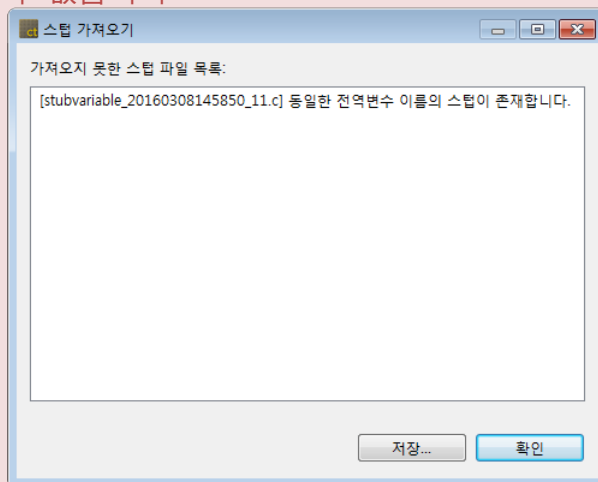
1. 스텝 뷰의 풀다운 메뉴에서 [가져오기...] 메뉴를 선택합니다.



2. 가져올 스텝 파일을 선택한 후 [열기] 버튼을 선택합니다.



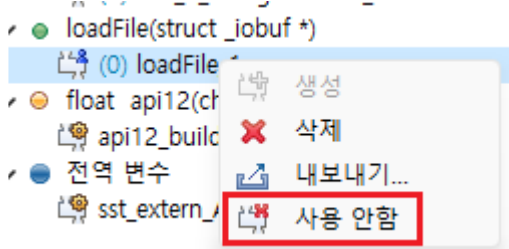
! 가져오려는 스텝과 동일한 구 버전 스텝 또는 빌드 스텝이 프로젝트에 존재할 경우, 가져올 수 없습니다.



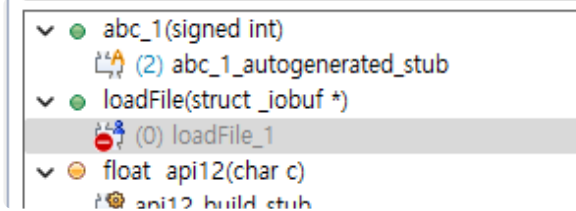
## 스텝 사용, 사용 안함

생성된 스텝 사용 여부를 설정할 수 있습니다.

1. 스텝을 선택한 후 우클릭으로 [사용 안함] 메뉴를 선택합니다.

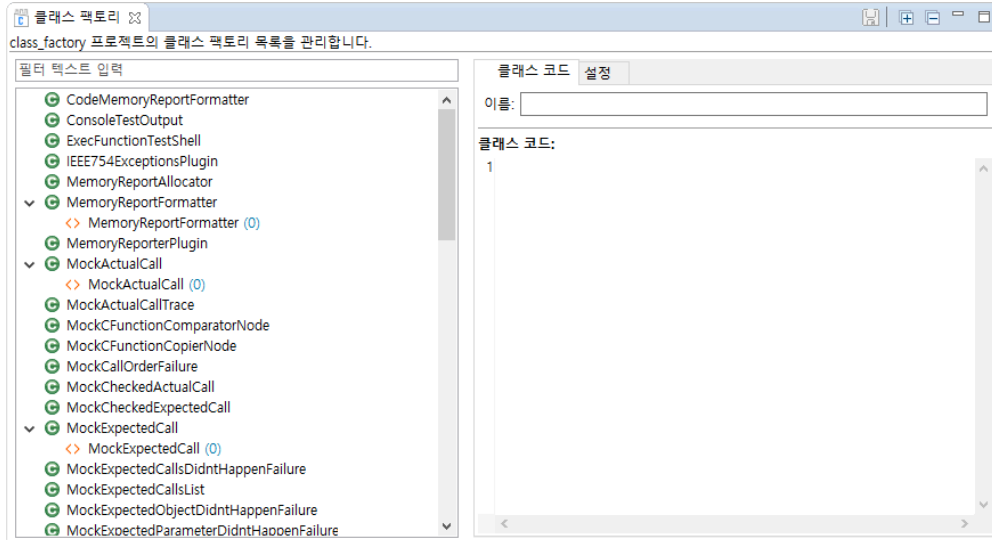


2. 아래 그림과 같이 스텝 이미지 데코레이터가 추가되고 스텝 이름의 색이 회색으로 변경됩니다.



## 17.8. 클래스 팩토리 뷰

C++ 소스 코드를 테스트할 때, 클래스 객체를 만들기 위해 클래스 코드를 사용할 수 있습니다. 추상 클래스는 객체를 생성할 수 없기 때문에 CT 2023.12는 추상 클래스의 객체를 생성할 수 있도록 자동으로 클래스 코드를 만들어 줍니다.



- [클래스 팩토리 뷰]의 왼쪽 영역에서 프로젝트에 포함된 클래스 목록과 클래스 코드를 트리 구조로 확인할 수 있습니다.
- 오른쪽 영역의 [클래스 코드] 탭에서 클래스 코드와 이름을 수정하고, [설정] 탭에서 클래스 코드가 연결된 테스트를 확인할 수 있습니다.

### [클래스 팩토리 뷰]의 톨바 메뉴

항목	설명
저장	클래스 코드의 변경 사항을 저장
모두 펼치기	클래스 팩토리 뷰의 클래스 트리 모두 펼치기
모두 접기	클래스 팩토리 뷰의 클래스 트리 모두 접기

### [클래스 팩토리 뷰]의 폴다운 메뉴







메뉴 이름	설명
클래스 재설정	클래스 코드 대상 클래스가 바뀌었을 경우 클래스 재설정


✿ [클래스 재설정]에 대한 자세한 설명은 User Guide 2023.12의 [\[소스 코드 변경 후 테스트 재\]](#)

[설정](#)을 참고하시기 바랍니다.


## [클래스 팩토리 뷰]의 아이콘

아이콘	설명
 클래스	프로젝트에 속한 클래스
 추상 클래스	프로젝트에 속한 추상 클래스
 모의 객체	테스트에서 사용할 모의 객체
 클래스 코드	테스트에서 사용할 클래스 객체를 생성하는 코드

## 클래스의 컨텍스트 메뉴

항목	설명
 생성	선택한 클래스의 클래스 코드를 생성

## 클래스 코드의 컨텍스트 메뉴

항목	설명
 삭제	선택한 클래스 코드를 삭제

## 클래스 팩토리 사용하기

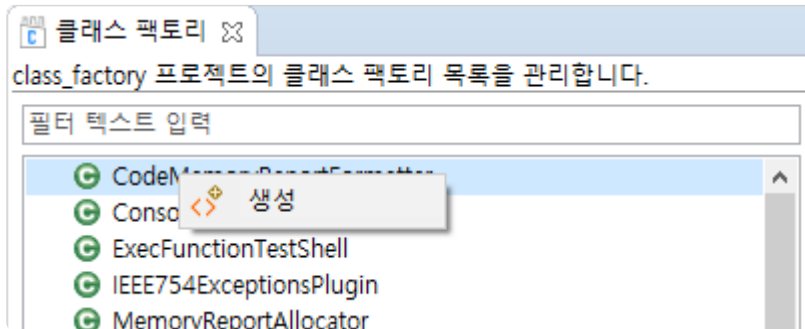
다음 두 가지 방식으로 클래스 팩토리를 사용할 수 있습니다.

- [클래스 코드 사용하기](#)
- [모의 객체 사용하기](#)

## 17.8.1. 클래스 코드 사용하기

### 클래스 코드 생성

프로젝트를 분석하면 소스 코드에 포함된 모든 클래스의 목록이 클래스 팩토리 뷰에 나타납니다. 추상 클래스는 CT 2023.12가 자동으로 클래스 코드를 생성합니다. 클래스의 컨텍스트 메뉴에서 직접 클래스 코드를 생성할 수도 있습니다.

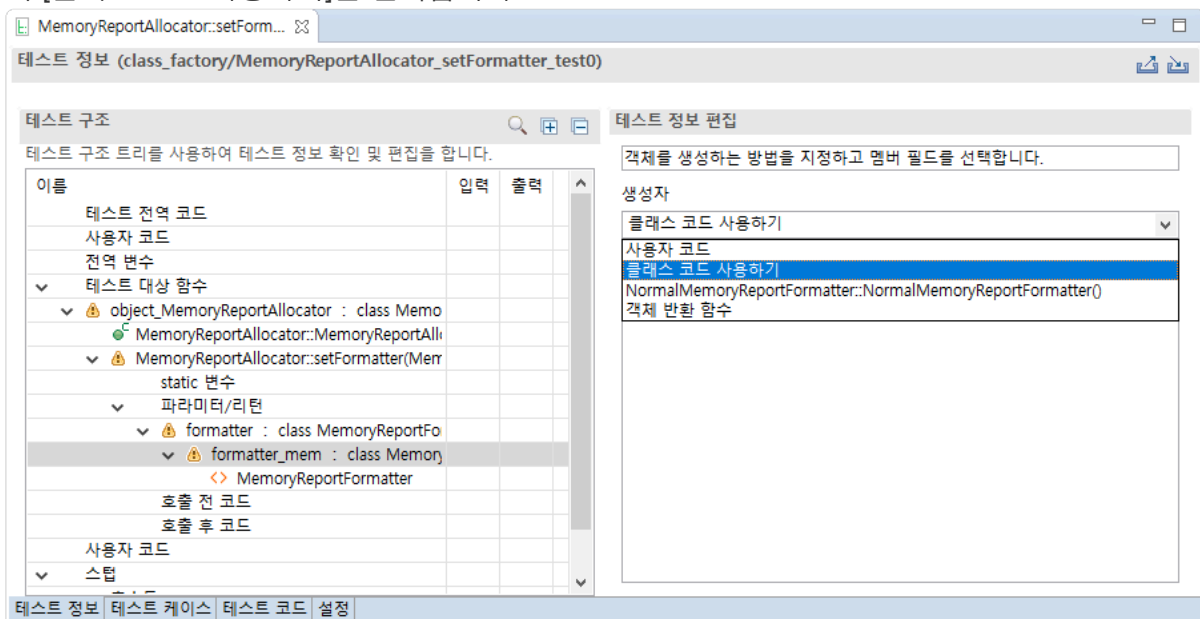


### 클래스 코드 적용

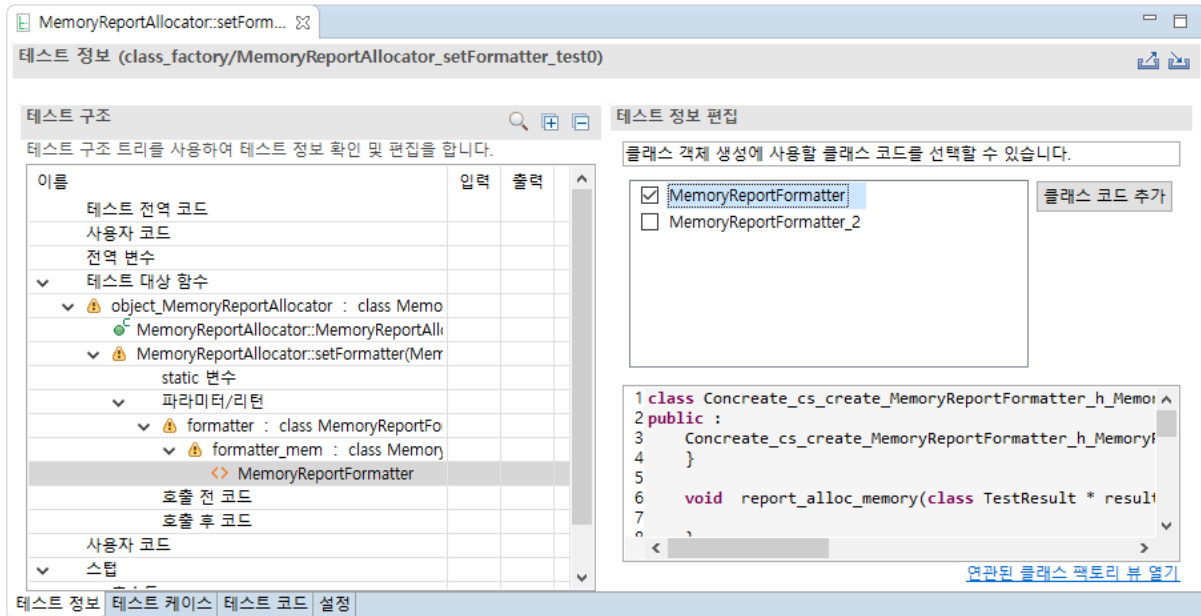
클래스 코드를 테스트에 적용하는 방법은 세 가지입니다.

#### 테스트 편집기에서 적용하기

1. 클래스 코드를 적용할 테스트를 더블 클릭하여 테스트 편집기를 엽니다. 테스트 정보 탭에서 테스트 구조 트리를 펼쳐 클래스 코드를 적용할 객체를 선택하고 오른쪽 테스트 정보 편집 영역의 생성자에서 [클래스 코드 사용하기]를 선택합니다.



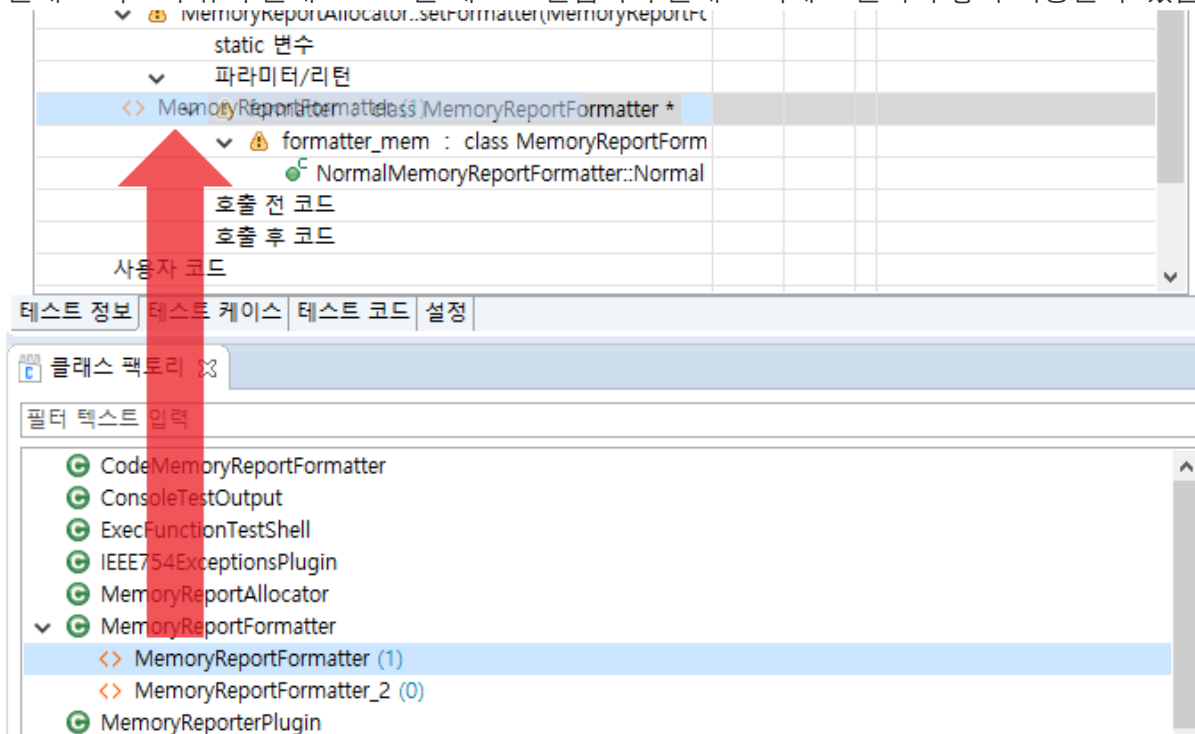
2. 테스트 구조에서 클래스 코드를 선택합니다. 테스트 정보 편집 영역에서 적용할 클래스 코드에 체크하고 저장합니다. 해당 클래스 코드를 선택하여 코드를 확인할 수 있습니다.



- [연관된 클래스 팩토리 뷰 열기]를 선택하여 클래스 코드를 클래스 팩토리 뷰에서 열 수 있습니다.
- [클래스 코드 추가] 버튼을 클릭하여 새 클래스 코드를 추가할 수 있습니다.

## 끌어서 놓기로 적용

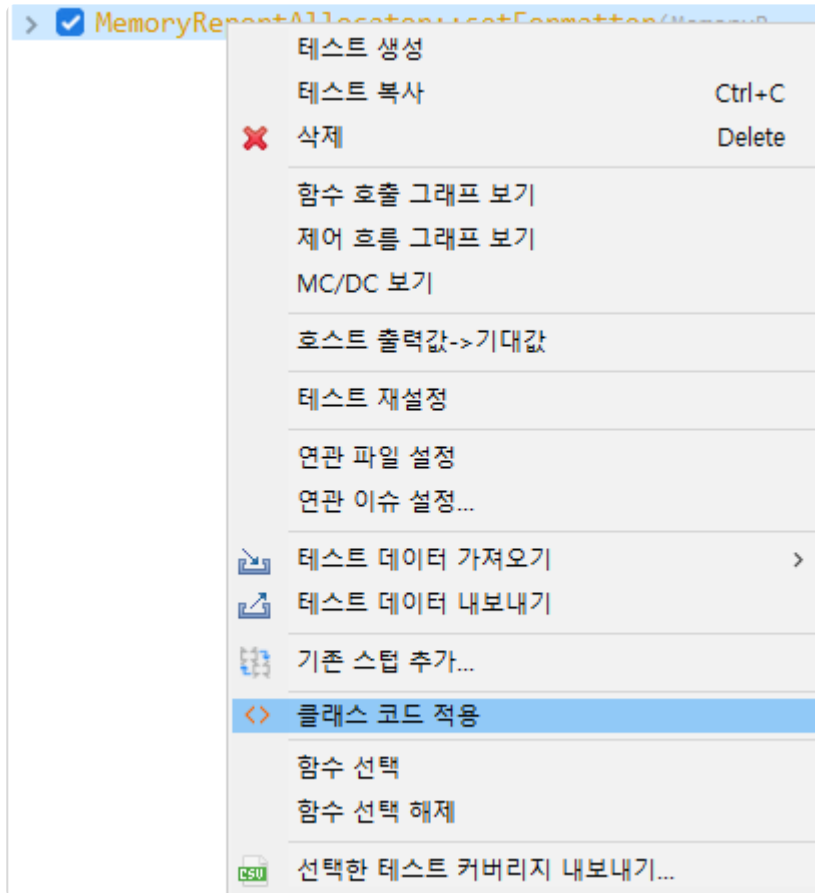
클래스 팩토리 뷰의 클래스 코드를 테스트 편집기의 클래스 객체로 끌어서 놓아 적용할 수 있습니다.



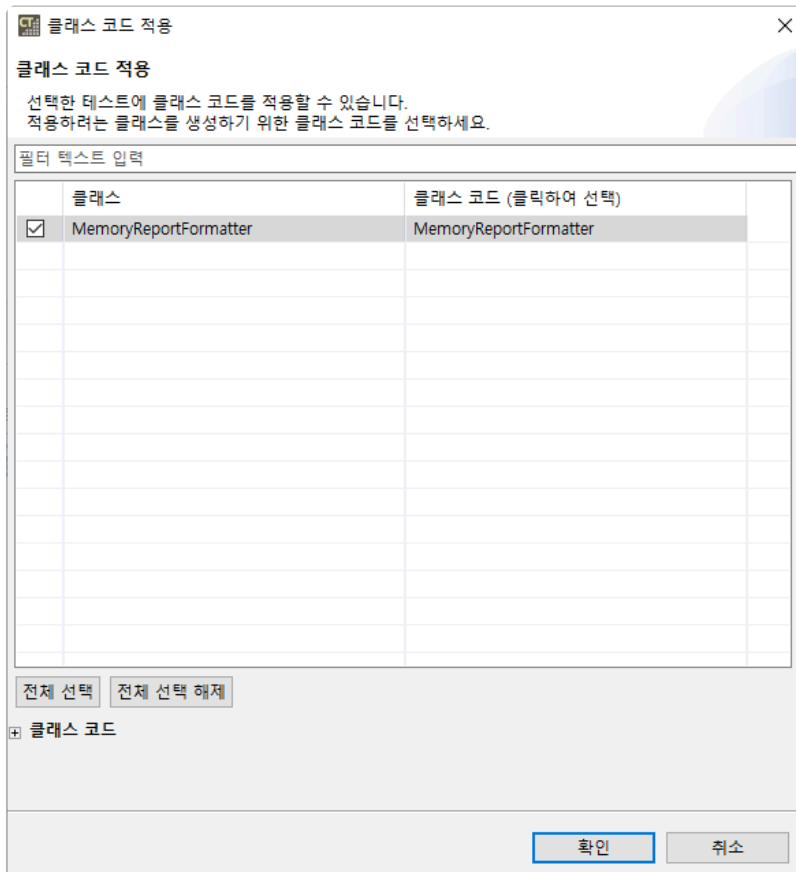
## 테스트 뷰의 컨텍스트 메뉴로 적용하기

1. 테스트 뷰에서 함수 또는 테스트의 컨텍스트 메뉴에서 [클래스 코드 적용]을 선택합니다.





2. [클래스 코드 적용] 창에서 선택한 테스트에 적용할 클래스 코드를 선택하고 [확인] 버튼을 클릭합니다. 아래쪽 클래스 코드를 펼치면 선택한 클래스 코드를 확인할 수 있습니다.



✿ [클래스 코드 적용] 기능을 사용하면 클래스 코드를 적용할 수 있는 모든 위치에 동일한 클래스 코드가 적용됩니다.

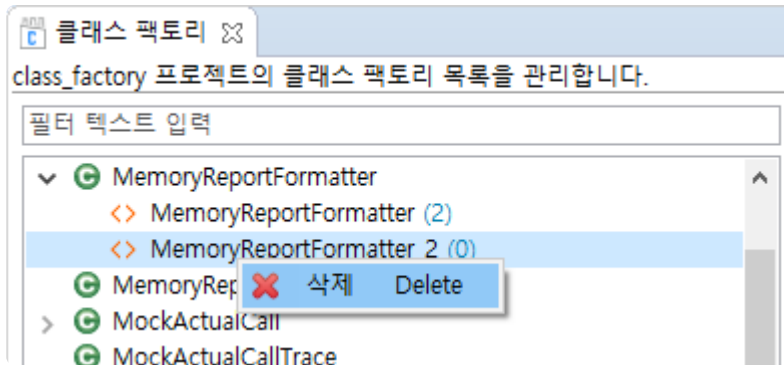
## 클래스 코드 수정

클래스 팩토리 뷰의 [클래스 코드] 탭에서 클래스 코드를 수정할 수 있습니다. 클래스 코드를 수정하면 해당 코드가 연결된 모든 테스트에 반영됩니다.

! 클래스 코드에는 입/출력 사용자 매크로를 사용할 수 없습니다.

## 클래스 코드 삭제

클래스 코드는 클래스 팩토리 뷰에서 컨텍스트 메뉴로 삭제할 수 있습니다.



! 테스트와 연결된 클래스 코드가 있으면 삭제할 수 없습니다. 삭제하기 위해서는 먼저 클래스 코드의 테스트 연결을 모두 해제해야 합니다.

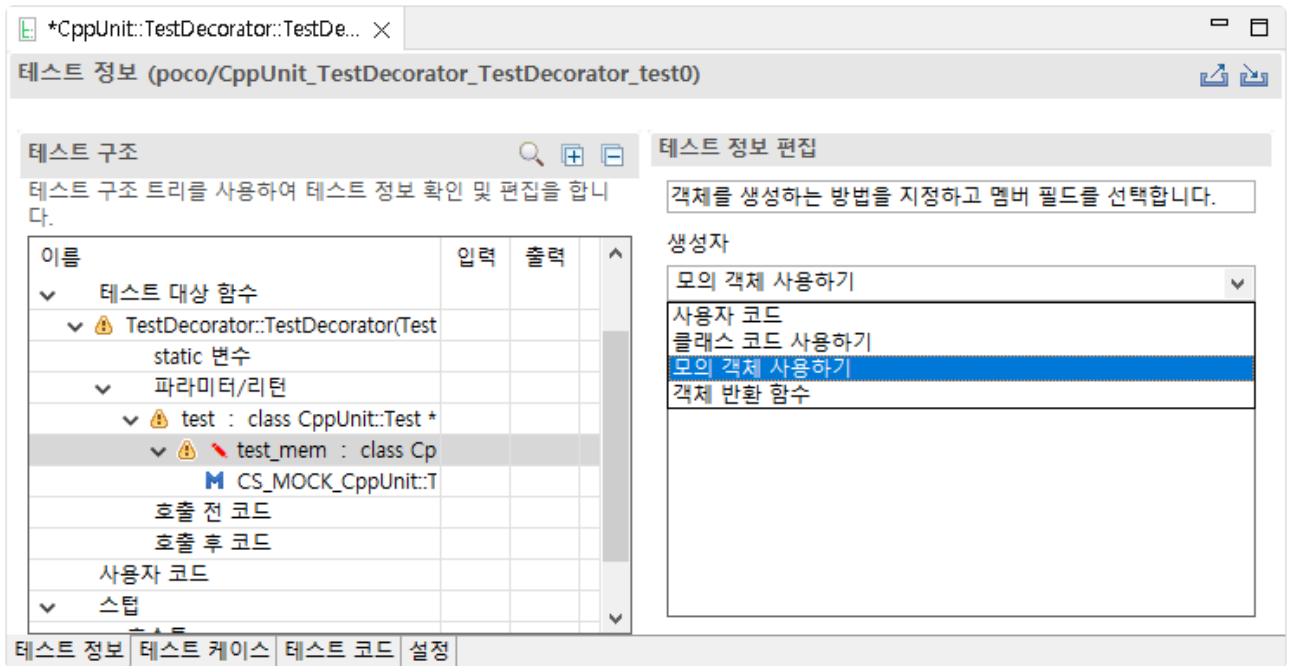
## 17.8.2. 모의 객체 사용하기

모의 객체를 사용할 수 있는 툴체인은 다음과 같습니다.

- GCC 6.0 이상
- Visual Studio 2015 이상

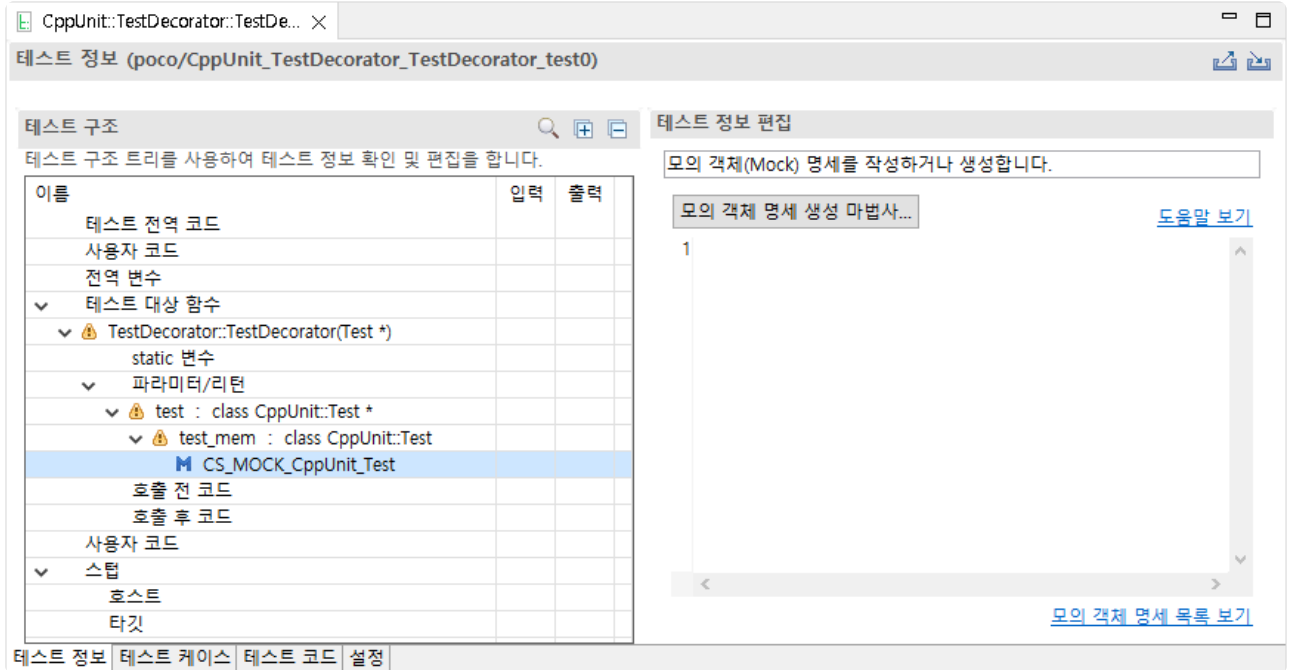
### 모의 객체 생성 및 적용

1. 모의 객체를 생성할 테스트를 더블 클릭하여 [테스트 편집기]를 엽니다.
2. [테스트 정보 탭]에서 테스트 구조 트리를 펼쳐 모의 객체를 생성할 객체를 선택하고 오른쪽 테스트 정보 편집 영역의 생성자에서 [모의 객체 사용하기]를 선택합니다.



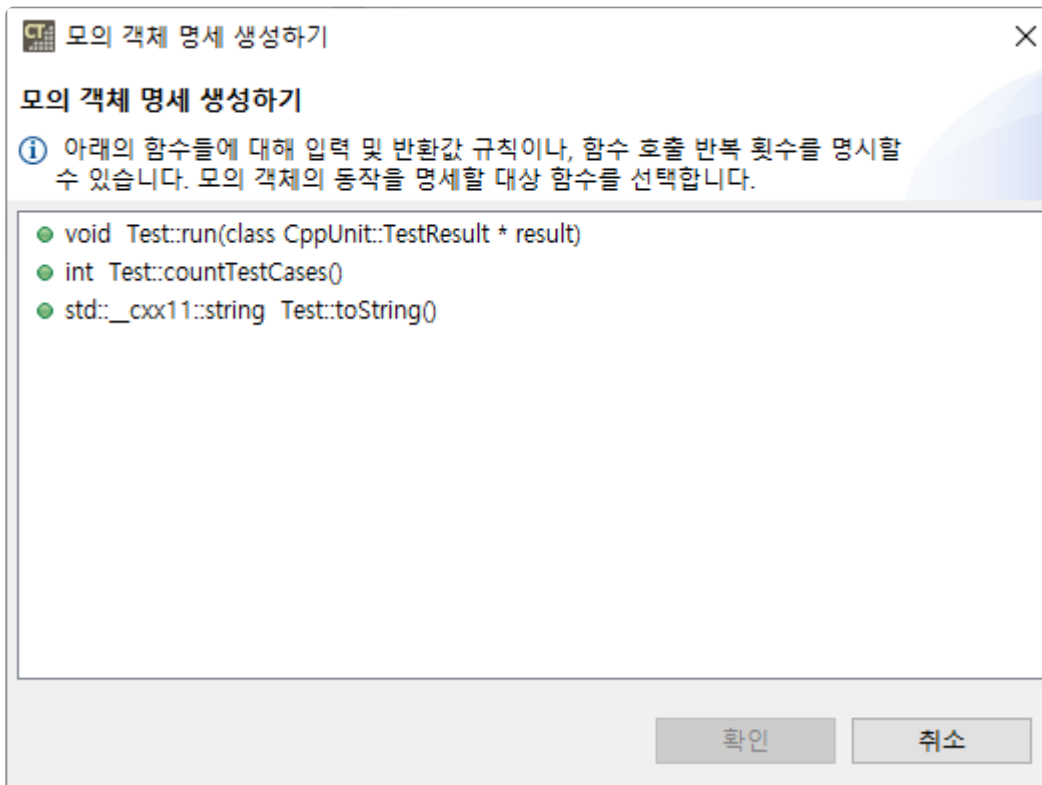
- [모의 객체 사용하기]를 선택하면 [클래스 팩토리 뷰]에 모의 객체의 클래스 코드가 생성됩니다.

3. [테스트 정보 탭]에서 생성된 모의 객체를 클릭합니다.



- [도움말 보기]: 모의 객체의 명세에 관한 설명을 볼 수 있습니다.

4. 오른쪽 테스트 정보 편집 영역에서 [모의 객체 명세 생성 마법사...] 버튼을 클릭하면 나타나는 마법사에서 모의 객체 명세를 추가할 수 있습니다.



자세한 사항은 User Guides의 [C++ 테스트에서 모의 객체 사용하기](#) 페이지를 참고하시기 바랍니다.

## 모의 객체 결과 확인

1. [테스트 뷰]에서 실행 버튼을 클릭합니다.
2. 명세를 만족하지 못하는 경우 테스트가 실패하고 테스트 케이스 옆에 [모의 객체 실패] 메시지가 표시됩니다.

- ▼ ☒ **TestDecorator::countTestCases()**
  - ▼ ☒ **test** **CppUnit::TestDecorator::countTestCases()\_0**
    - case** **1** **[모의 객체 실패]**

3. [테스트 편집기] > [테스트 케이스 탭]에서 모의 객체 실패 정보를 확인할 수 있습니다.

- [▶ 모의 객체 실패]를 펼쳐서 모의 객체가 실패한 원인을 확인할 수 있습니다.

▼ 모의 객체 실패

이름	타입	함수
object TestDecorator_test	CppUnit::Test	countTestCases()

Actual function call count doesn't match EXPECT\_CALL  
(csmock\_object\_TestDecorator\_\_test\_mem, countTestCases)...

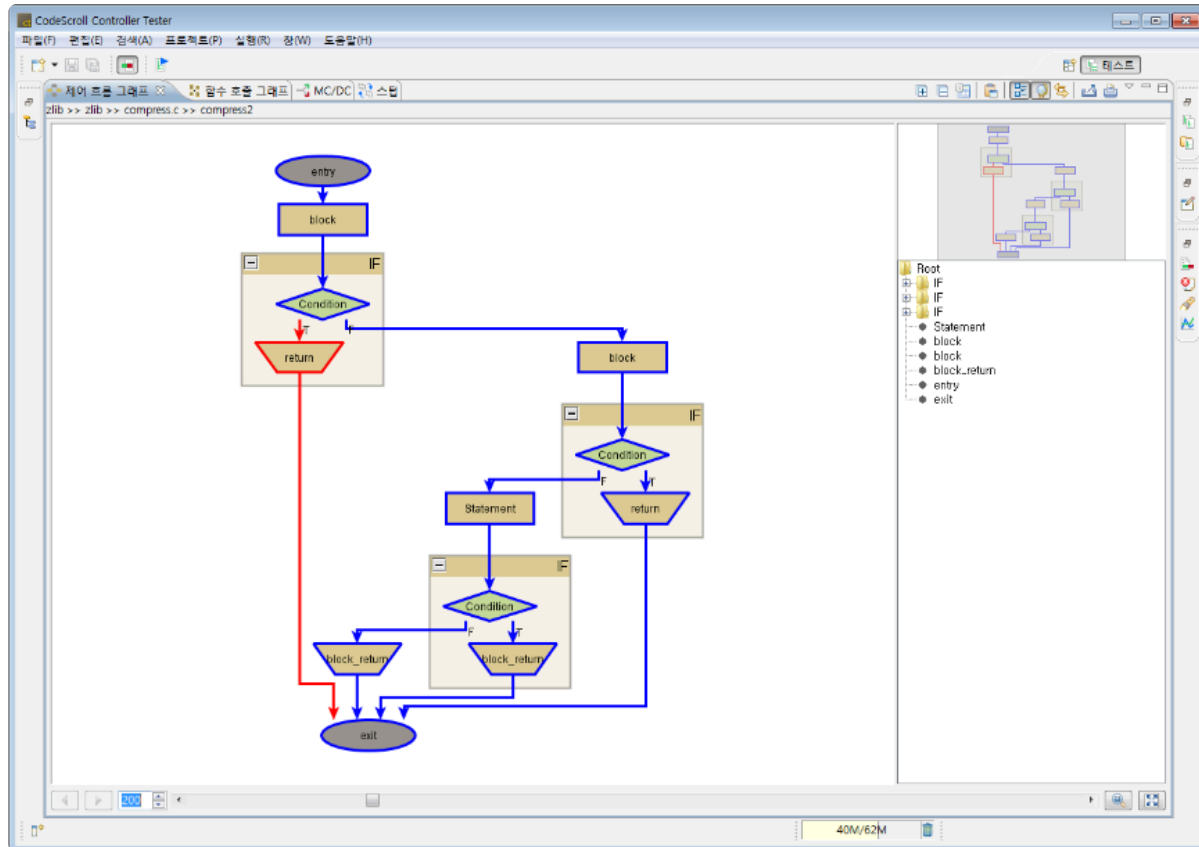
Expected: to be called 3 times

Actual: called once - unsatisfied and active

## 17.9. 제어 흐름 그래프 뷰

제어 흐름 그래프 뷰는 선택된 함수의 제어 흐름 정보를 그래프로 보여줍니다.

메인 툴바의 [커버 여부 보여주기] 버튼을 클릭하면 커버된 영역은 파란색, 커버되지 않은 영역은 붉은 색으로 표시 됩니다.



### 툴바 메뉴

메뉴	설명
모두 펼치기	모든 그룹 노드를 펼쳐 보임
모두 접기	모든 그룹 노드를 접어 보임
범례 표시	범례(현재 보이는 그래프 상의 노드 종류)를 표시
시스템 클립보드에 복사	현재 화면에 보이는 그래프를 클립보드에 복사
아웃라인 표시	그래프를 트리 형태로 표시
오버뷰 표시	그래프의 오버뷰 표시
편집기와 연결	그래프에서 선택된 항목을 원클릭으로 편집기에 표시
뷰 내용 내보내기	뷰의 내용을 리포트로 내보내기
뷰 내용 인쇄	뷰의 내용을 인쇄

## 풀다운 메뉴

메뉴	설명
 모두 펼치기	모든 그룹 노드를 펼쳐 보임
 모두 접기	모든 그룹 노드를 접어 보임
함수 호출 그래프 보기	현재 뷰에 선택된 함수를 중심으로 함수 호출 그래프 보기
 범례 표시	범례(현재 보이는 그래프 상의 노드 종류)를 표시
 아웃라인 표시	그래프를 트리 형태로 표시
 오버뷰 표시	그래프의 오버뷰 표시
 편집기와 연결	그래프에서 선택된 항목을 원클릭으로 편집기에 표시
그래프 포맷으로 저장	현재 화면에 보이는 그래프에 대한 그래프 모델 파일 생성 네 가지 포맷 지원: <ul style="list-style-type: none"> <li>• Graph Modeling Language XML (*.xgml)</li> <li>• Graph Modeling Language(*.gml)</li> <li>• yWorks Binary Graph Format(.ygf)</li> <li>• Trivial Graph Format(*.tgf)</li> </ul>
이미지 포맷으로 저장	현재 화면에 보이는 그래프를 이미지 포맷 파일(jpg, gif)로 저장
 시스템 클립보드에 복사	현재 화면에 보이는 그래프를 클립보드에 복사
환경 설정	환경 설정 열기

## 노드 팝업 메뉴

메뉴	설명
함수 호출 그래프 보기	현재 뷰에 선택된 함수를 중심으로 함수 호출 그래프 보기

## 히스토리 기능



뷰 왼쪽 하단의 화살표 버튼(뒤로 가기, 앞으로 가기)으로 현재 그래프에서 선택했던 노드를 다시 볼 수 있습니다.



## 확대/축소 기능

뷰 하단에 숫자를 입력하거나 슬라이더를 조정해서 확대/축소 비율을 변경할 수 있습니다.



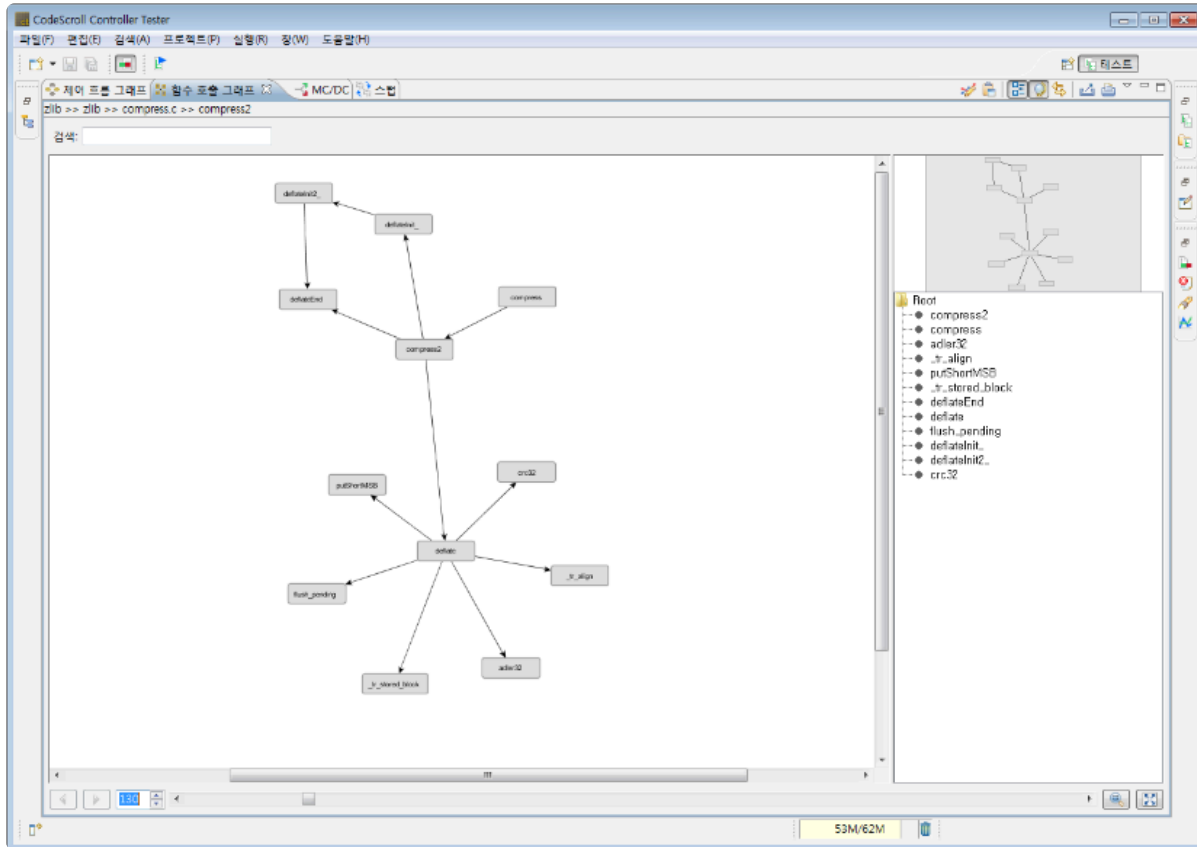
뷰 오른쪽 하단의  [확대/축소 비율 초기화] 버튼으로 확대/축소 비율을 초기화할 수 있고,  [뷰 크기에 맞추기] 버튼으로 뷰의 크기에 맞게 확대/축소 비율을 변경할 수 있습니다.

## 17.10. 함수 호출 그래프 뷰

함수 호출 그래프 뷰는 선택된 함수를 중심으로 함수의 호출 정보를 그래프로 보여줍니다.

(예: 함수 'A'에서 함수 'B'를 호출했을 때, 노드 'A'에서 노드 'B'로의 에지로 표현하며, 여러 번 호출하더라도 하나의 에지로 표현)

노드(함수)를 선택하면 해당 노드를 중심으로 함수 호출 정보를 보여주게 됩니다.







### 툴바 메뉴

메뉴	설명
레이아웃 편집	그래프의 노드 위치를 수정할 수 있게 변경(그래프가 갱신되면 자동으로 편집 불가능)
시스템 클립보드에 복사	현재 화면에 보이는 그래프를 클립보드에 복사
아웃라인 표시	그래프를 트리 형태로 표시
오버뷰 표시	그래프의 오버뷰 표시
편집기와 연결	그래프에서 선택된 항목을 원클릭으로 편집기에 표시
뷰 내용 내보내기	뷰의 내용을 리포트로 내보내기
뷰 내용 인쇄	뷰의 내용을 인쇄



## 폴다운 메뉴

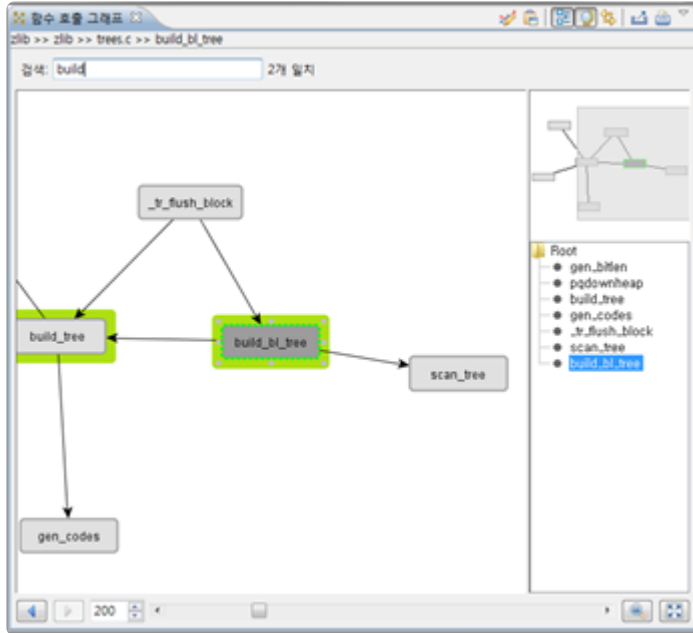
메뉴	설명
제어 흐름 그래프 보기	현재 뷰에 선택된 함수에 대한 제어 흐름 그래프 보기
 아아웃라인 표시	그래프를 트리 형태로 표시
 오버뷰 표시	그래프의 오버뷰 표시
 편집기와 연결	그래프에서 선택된 항목을 원클릭으로 편집기에 표시
그래프 포맷으로 저장	현재 화면에 보이는 그래프에 대한 그래프 모델 파일 생성 네 가지 포맷 지원: <ul style="list-style-type: none"> <li>• Graph Modeling Language XML (*.xgml)</li> <li>• Graph Modeling Language(*.gml)</li> <li>• yWorks Binary Graph Format(.ygf)</li> <li>• Trivial Graph Format(*.tgf)</li> </ul>
이미지 포맷으로 저장	현재 화면에 보이는 그래프를 이미지 포맷 파일(jpg, gif)로 저장
 시스템 클립보드에 복사	현재 화면에 보이는 그래프를 클립보드에 복사
환경 설정	환경 설정 열기

## 노드 팝업 메뉴

메뉴	설명
제어 흐름 그래프 보기	현재 뷰에 선택된 함수에 대한 제어 흐름 그래프 보기
펼치기	선택한 함수 이후의 호출 관계를 한 단계 더 표시
접기	선택한 함수 이후의 호출 관계를 숨김
시작 노드로 설정	2개 함수의 호출 경로를 보기 위한 시작 함수 지정
종료 노드로 설정	2개 함수의 호출 경로를 보기 위한 종료 함수 지정, 현 그래프 상에서 가능한 모든 경로를 표시

## 검색 기능

뷰 상단에 검색어를 입력하면, 입력된 검색어를 포함하는 이름을 가진 함수들이 강조되어 보여집니다.



## 히스토리 기능

뷰 왼쪽 하단의 [화살표(뒤로 가기, 앞으로 가기)] 버튼으로 현재 그래프에서 선택했던 노드를 다시 볼 수 있습니다.



## 확대/축소 기능

뷰 하단에 숫자를 입력하거나 슬라이더를 조정해서 확대/축소 비율을 변경할 수 있습니다.



뷰 오른쪽 하단의 [확대/축소 비율 초기화] 버튼으로 확대/축소 비율을 초기화할 수 있고, [뷰 크기에 맞추기] 버튼으로 뷰의 크기에 맞게 확대/축소 비율을 변경할 수 있습니다.

## 펼치기/접기 기능

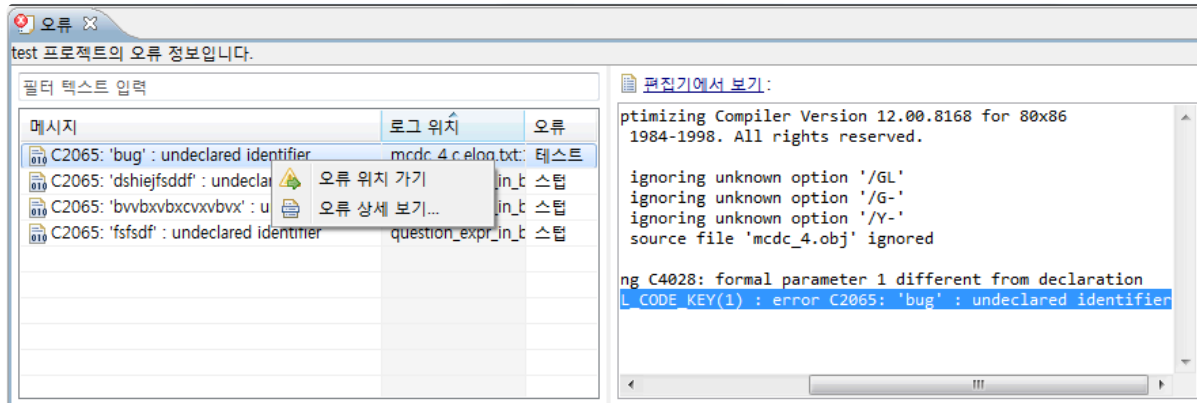
노드를 오른쪽 클릭하면 나타나는 메뉴를 통해 해당 노드에서 나가는 에지를 접거나 펼칠 수 있습니다. [접기]를 선택하면 선택한 노드에 해당하는 함수가 호출하는 모든 함수의 호출 관계를 숨깁니다. 함수 호출 정보가 없는 노드의 접기 메뉴는 비활성화됩니다. [펼치기]를 선택하면 선택된 노드에 해당하는 함수가 호출하는 함수들을 한 단계 더 보여줍니다. 마찬가지로 더 이상의 호출 관계가 없다면 펼치기 메뉴는 비활성화됩니다.

## 노드 사이 경로 강조 기능

시작 노드와 종료 노드를 선택하면, 두 노드 사이에 존재하는 모든 경로를 강조해서 보여줌으로써 복잡한 함수 호출 관계에서 관심 있는 부분을 쉽게 볼 수 있습니다.

## 17.11. 오류 뷰

프로젝트의 분석 오류 정보를 상세하게 확인할 수 있습니다.



### 로그 타입

타입	설명
빌드	빌드 중 발생한 오류
구문 분석	구문 분석 중 발생한 오류
링크	링킹 중 발생한 오류
전처리	전처리 중 발생한 오류

### 오류 타입

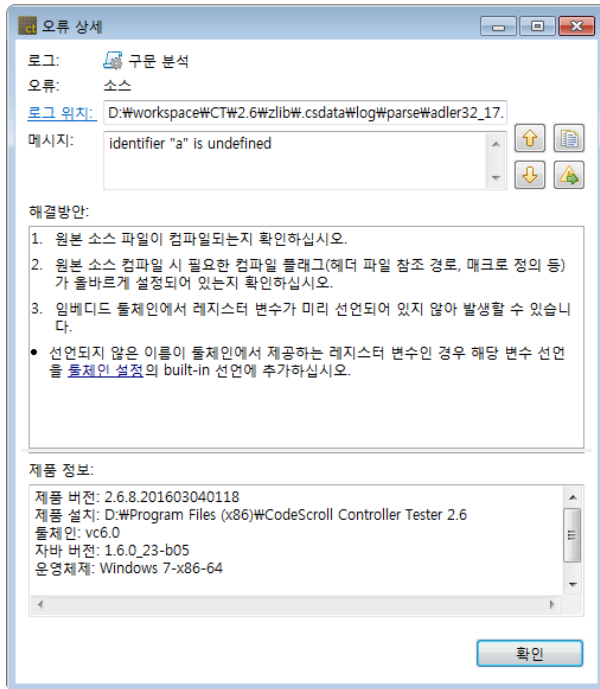
타입	설명
소스	소스 파일에서 발생한 오류
스텝	스텝 파일에서 발생한 오류
테스트	테스트 에디터에서 발생한 오류

### 오류 위치 가기

컨텍스트 메뉴에서 [오류 위치 가기]를 선택하면, 오류가 발생한 위치로 이동하여 쉽게 오류를 수정할 수 있습니다.

### 오류 상세 보기

오류 정보를 더블클릭하거나 컨텍스트 메뉴에서 [오류 상세 보기...]를 선택하면, 오류에 대한 상세 정보를 확인할 수 있습니다.



## 버튼 설명

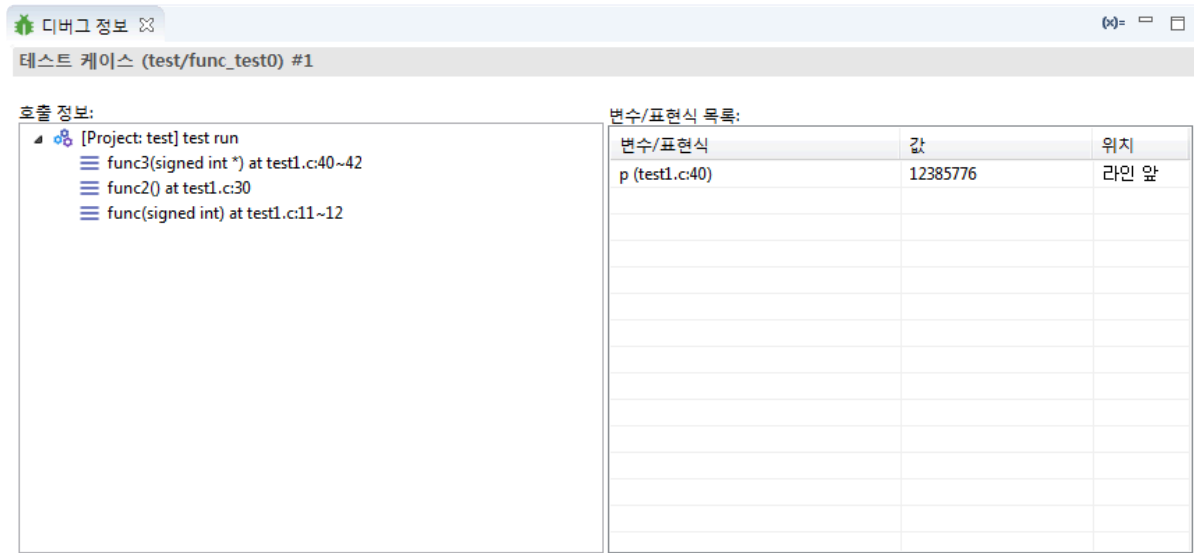
버튼	설명
이전	이전 오류 정보 보기
다음	다음 오류 정보 보기
복사	오류 내용 복사하기
오류 위치 가기	오류 위치 가기

## 17.12. 디버그 정보 뷰

디버그 정보 뷰는 테스트 케이스에서 발생한 오류의 원인을 파악하는 데 도움이 되는 정보를 보여줍니다.

- 오류가 발생한 테스트 케이스의 함수 호출 트레이스
- 디버그 정보 확인을 위해 추가한 변수/표현식의 실제 값

유닛 테스트 뷰에서 [디버그 정보 확인]을 수행한 테스트 케이스를 선택하면 디버그 정보 뷰에 디버그 정보가 표시됩니다.



- 호출 정보는 오류가 발생한 테스트 케이스를 실행했을 때의 함수 호출 트레이스를 표시합니다. 오류가 발생한 위치가 가장 위에 표시됩니다.
- 변수/표현식 목록은 디버그 정보 확인을 위해 추가한 변수/표현식의 실제 값을 보여줍니다.



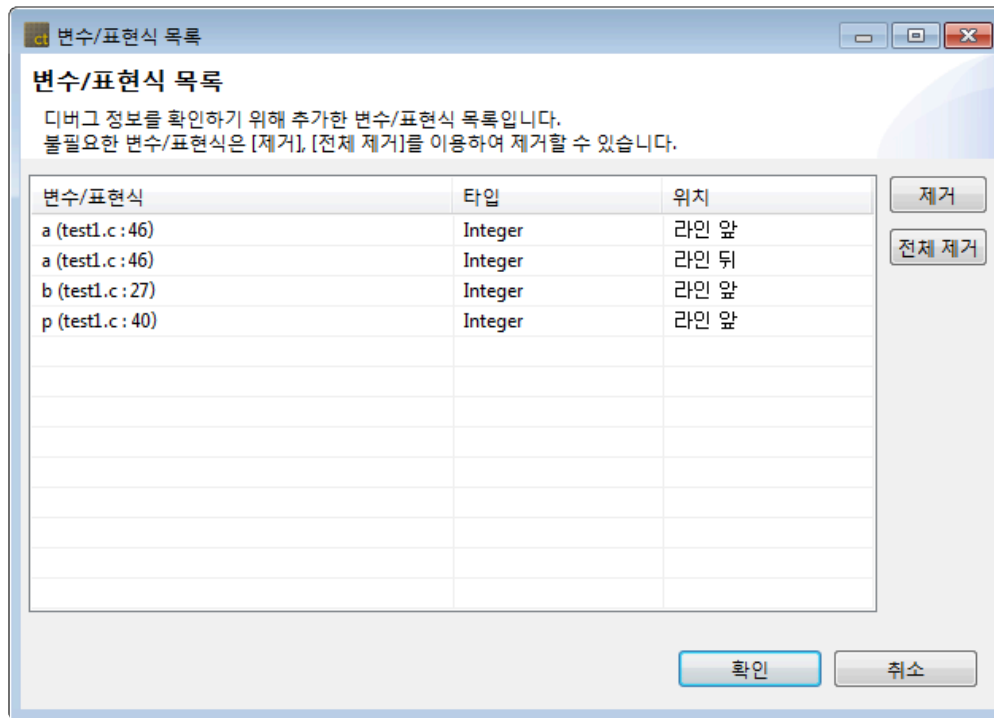
추가한 변수/표현식의 실제 값이 표시되지 않는 경우:

- 변수/표현식을 새로 추가한 뒤에 테스트 케이스의 [디버그 정보 확인]을 실행하지 않았을 때
- 테스트 케이스 실행 시 변수/표현식을 추가한 라인이 실행되지 않았을 때 (조건에 따라 수행되지 않았거나, 실행 전에 오류가 발생함)

### 디버그 정보 뷰 툴바 메뉴

아이콘	설명
	디버그할 변수/표현식 목록 보기

디버그 변수/표현식 목록에서는 변수/표현식을 확인하고 제거할 수 있습니다.

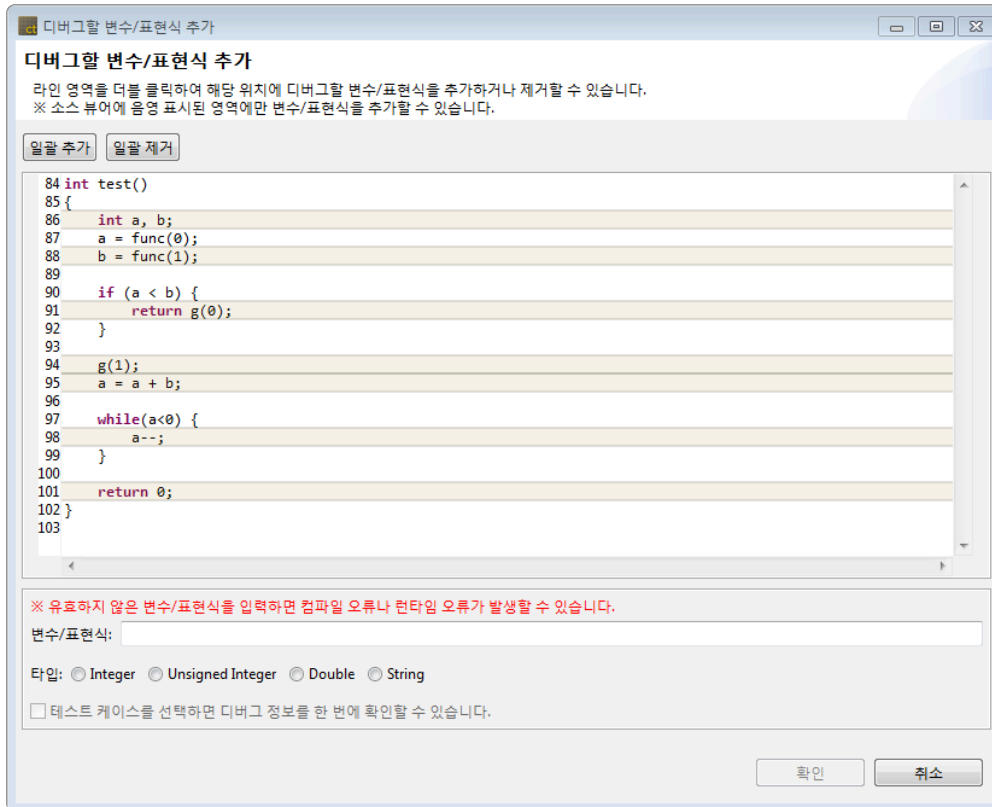


## 디버그할 변수/표현식 추가하기

디버그할 변수/표현식은 다음과 같은 방법으로 추가할 수 있습니다.

1. 소스 코드 편집기에서 라인 영역을 더블 클릭
2. 소스 코드 편집기에서 라인 영역 컨텍스트 메뉴에서 [디버그할 변수/표현식 추가...] 선택
3. 소스 코드 편집기에서 변수 또는 표현식을 블록 지정한 뒤에 컨텍스트 메뉴에서 [디버그할 변수/표현식 추가...] 선택

위의 동작을 수행하면 [디버그할 변수/표현식 추가] 창이 뜹니다.



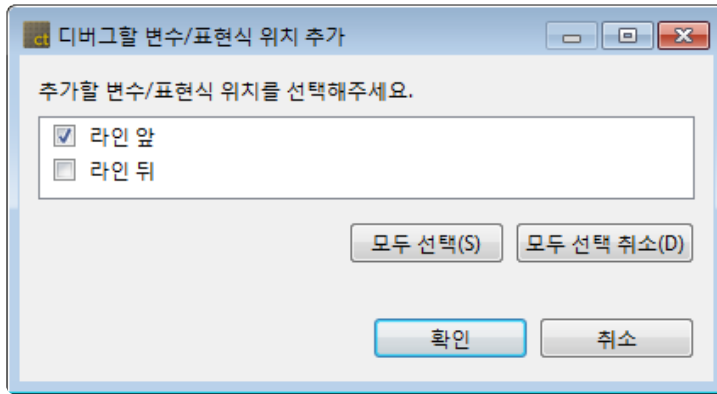
라인 영역을 더블 클릭하여 해당 위치에 디버그할 변수/표현식을 추가하거나 제거할 수 있습니다.

마커	설명
	해당 라인 앞에 변수/표현식 추가됨
	해당 라인 뒤에 변수/표현식 추가됨

음영 표시 영역에만 마커를 추가할 수 있습니다.

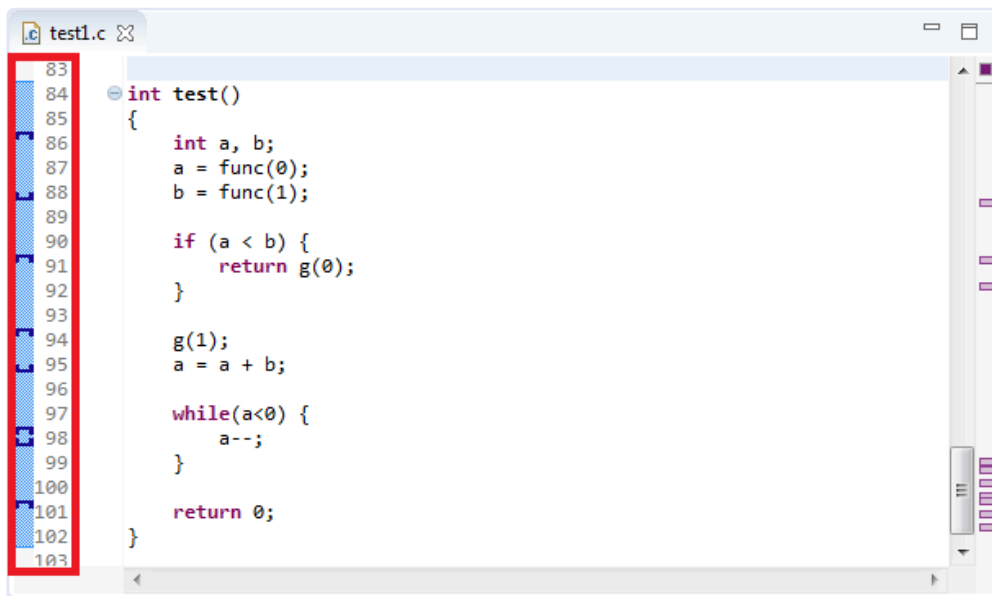


음영 표시된 영역에서 라인 앞 또는 라인 뒤에 디버그할 변수/표현식을 추가할 수 있는 경우, 추가할 위치를 묻는 대화창이 뜹니다.

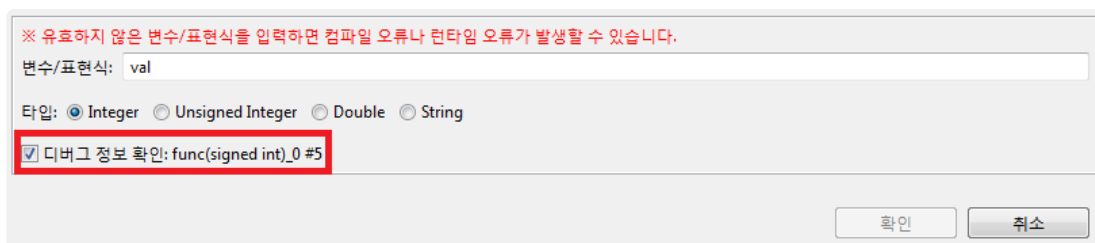


소스 뷰어 위쪽에 있는 [일괄 추가], [일괄 제거] 버튼을 통해 대상 함수에 추가할 수 있는 모든 위치에 변수/표현식을 추가 또는 제거할 수 있습니다.

변수/표현식과 타입을 지정하고 [확인] 버튼을 누르면 소스 코드 편집기에 마커가 추가된 것을 확인할 수 있습니다.



[유닛 테스트 뷰]에서 테스트 케이스가 선택되어 있다면, 디버그할 변수/표현식을 추가하면서 바로 [디버그 정보 확인]을 수행할 수 있습니다.



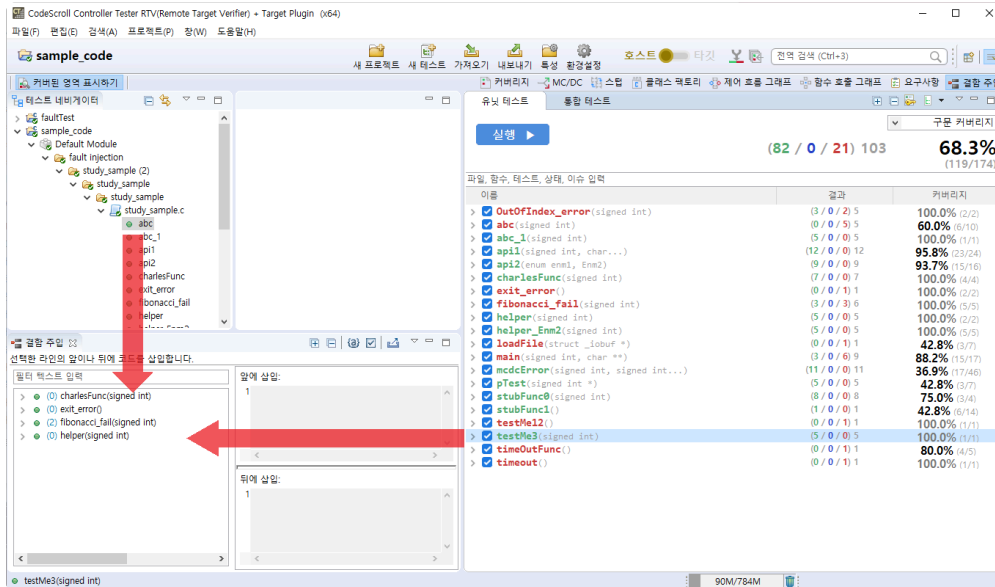
유효하지 않은 변수/표현식을 입력하면 [디버그 정보 확인]을 수행할 때 컴파일 오류나 런타임 오류가 발생할 수 있습니다.



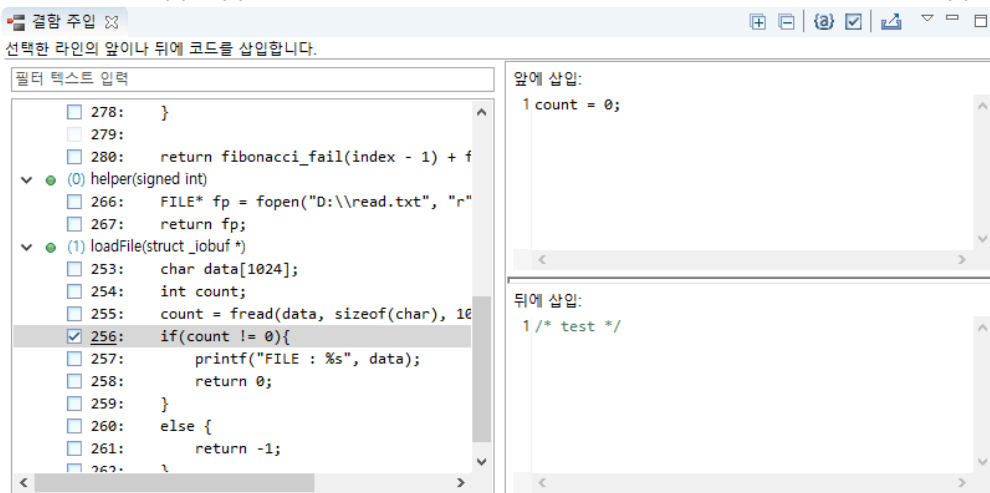
## 17.13. 결함 주입 뷰



[결함 주입 뷰]는 테스트 대상 함수의 특정 영역에 결함을 주입하여 테스트 할 수 있도록 하는 뷰입니다.

1. [테스트 네비게이터]나 [유닛 테스트 뷰]에서 결함을 주입할 함수를 [결함 주입 뷰]로 끌어서 놓습니다.



2. 뷰의 왼쪽 트리 구조에서 결함을 삽입할 라인을 선택한 후, 우측 편집 창에서 라인 앞/뒤에 삽입할 사용자 코드를 입력합니다. 사용자 코드가 작성된 라인의 번호에는 밑줄이 생깁니다.



- 결함을 주입할 라인의 체크박스를 체크하면 테스트 실행 시, 작성한 사용자 코드가 적용됩니다. 소스 코드 편집기 영역에서 해당 라인에  (라인 앞),  (라인 뒤) 마커가 표시됩니다. 마커에 마우스를 올리면 결함 주입 정보 및 작성한 코드가 툴팁으로 표시됩니다.

```

251
252  int loadFile(FILE* fp){
253      char data[1024];
254      int count;
255      count = fread(data, sizeof(char), 10
256      if(count != 0){
257          printf("FILE : %s", data);
258          return 0;
259      }
260      else {
261          return -1;
262      }
263  }
264

```

257 [결함 주입]  
258 뒤에 삽입:  
259 /\* test \*/

! 사용자 코드에는 한 줄 주석(//) 및 입/출력 사용자 매크로를 입력할 수 없습니다.

## 결함 주입 뷰의 툴바 메뉴

툴바 아이콘	설명
모두 펼치기	결함 주입 뷰의 함수 트리 모두 펼치기
모두 접기	결함 주입 뷰의 함수 트리 모두 접기
비어있지 않은 결함 주입만 보기	결함 주입 라인들 중 결함 주입 코드가 작성된 라인만 보기
<input checked="" type="checkbox"/> 활성화된 결함 주입만 보기	결함 주입 라인들 중 활성화된 라인만 보기
결함 주입 가져오기	결함 주입 정보 가져오기(.xls 파일)
결함 주입 내보내기	결함 주입 정보 내보내기(.xls 파일)

## 결함 주입 뷰의 컨텍스트 메뉴




함수 노드 및 라인은 다중 선택하여 컨텍스트 메뉴를 띄우는 것이 가능합니다.




### 함수 노드의 컨텍스트 메뉴

제거 Delete

메뉴 이름	설명
제거	사용자가 삽입한 함수를 제거

## 라인의 컨텍스트 메뉴

	복사	Ctrl+C
	붙여넣기	Ctrl+V
	초기화	
<input checked="" type="checkbox"/>	활성화	
<input type="checkbox"/>	비활성화	

메뉴 이름	설명
 복사	사용자가 라인 앞/뒤에 삽입한 정보를 복사
 붙여넣기	사용자가 라인 앞/뒤에 삽입한 정보를 붙여넣기
 초기화	선택된 라인의 체크박스 체크 해제 및 작성한 코드 삭제
<input checked="" type="checkbox"/> 활성화	선택된 라인의 체크박스를 체크하여 결함 주입 활성화
<input type="checkbox"/> 비활성화	선택된 라인의 체크박스를 체크 해제하여 결함 주입 비활성화

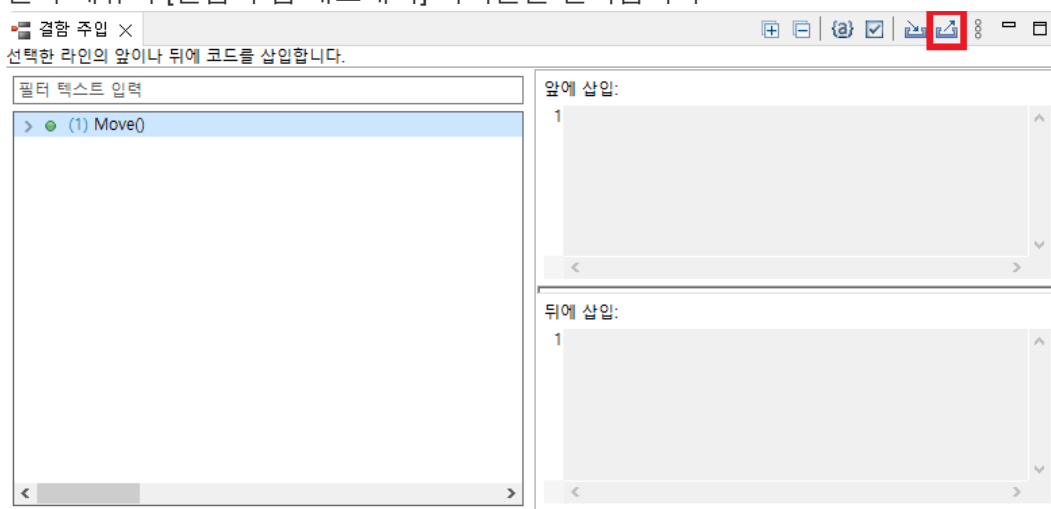
## 풀다운 메뉴

메뉴 이름	설명
 결함 주입 재설정	<a href="#">결함 주입 재설정</a> 참고

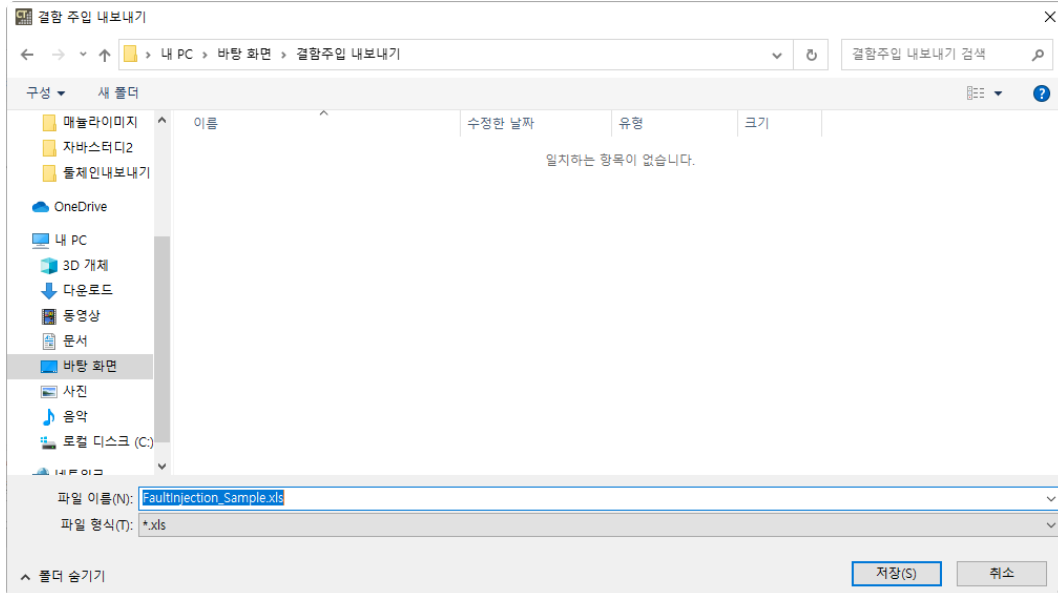
\* 소스 코드 변경 시 결함 주입 정보가 정상적으로 반영되지 않을 수 있습니다. 변경 시 결함 주입을 재설정해야 합니다.

## 결함 주입 내보내기

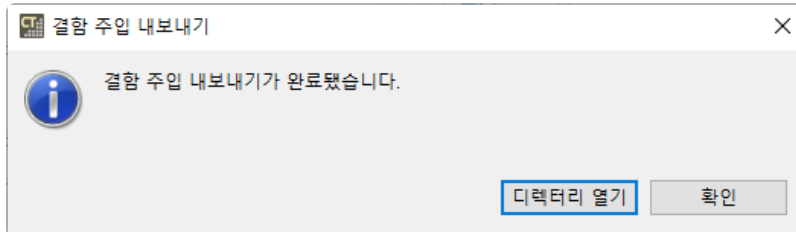
1. 툴바 메뉴의 [결함 주입 내보내기] 아이콘을 선택합니다.



2. 디렉터리 다이얼로그에서 결함 주입 파일(.xls)의 저장 위치를 지정한 후 [저장] 버튼을 선택합니다.

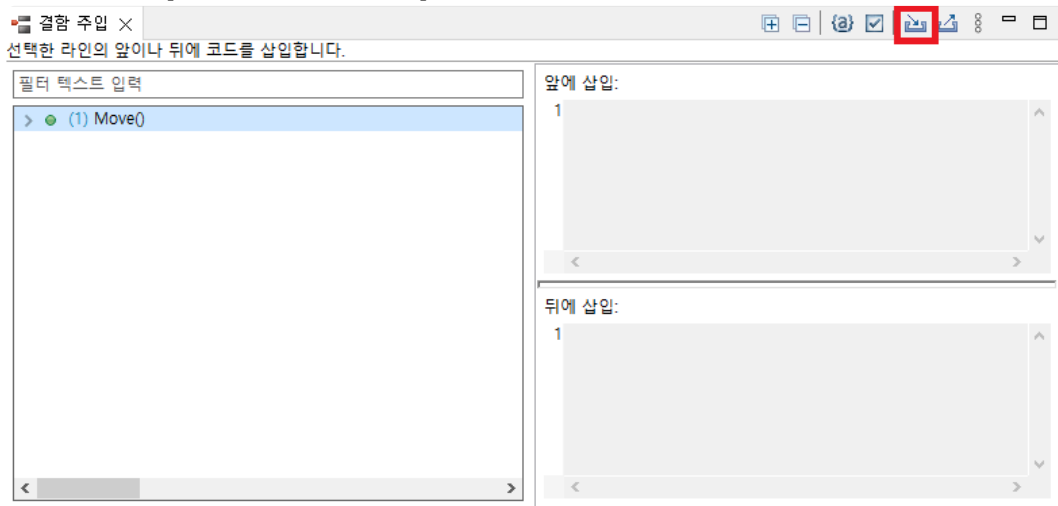


3. 내보내기에 성공하면 해당 경로 위치로 바로 이동할 수 있습니다.

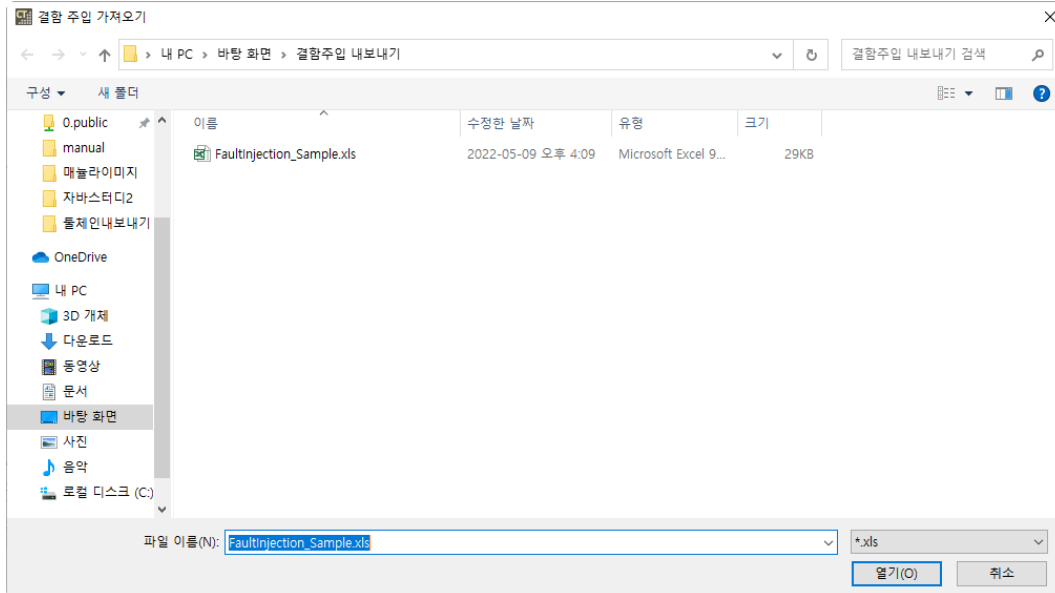


## 결함 주입 가져오기

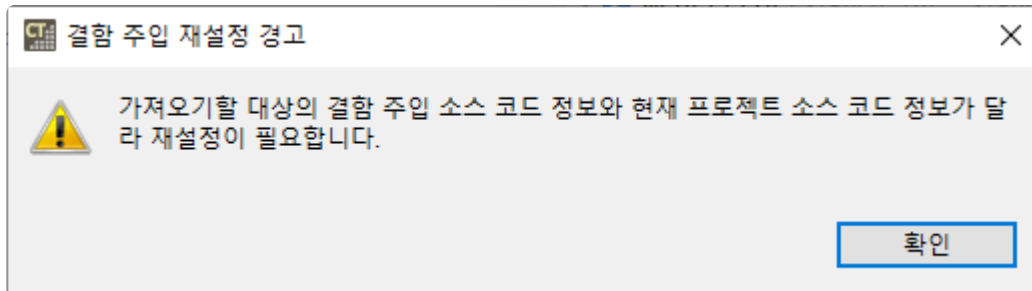
1. 툴바 메뉴의 [결함 주입 가져오기] 아이콘을 선택합니다.



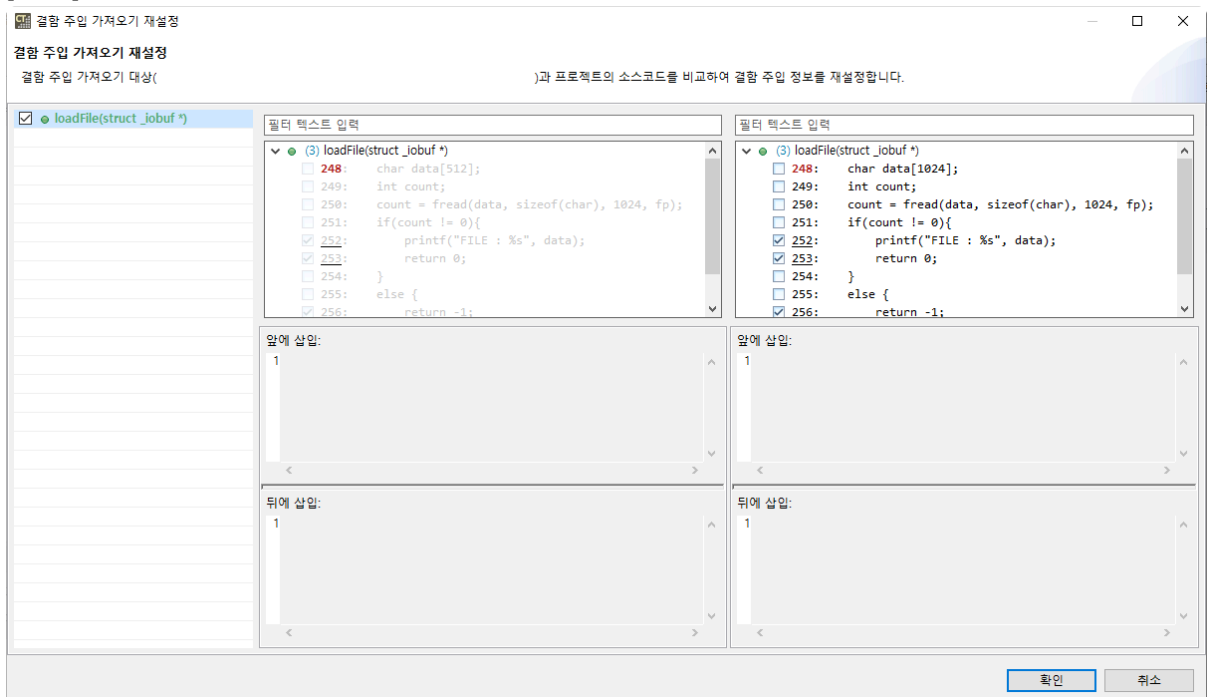
2. 가져올 결함 주입 파일을 선택합니다.



3. 가져오려는 결함 주입 정보의 소스 코드가 CT 2023.12 프로젝트의 소스 코드와 다를 경우 결함 주입 가져오기 재설정 기능을 사용하여 변경된 소스 코드 형상에 맞춰 결함 주입을 재사용 할 수 있습니다.
- a. 결함 주입 가져오기를 수행할 때 소스 코드 형상이 다르면 아래와 같은 대화상자가 뜹니다.



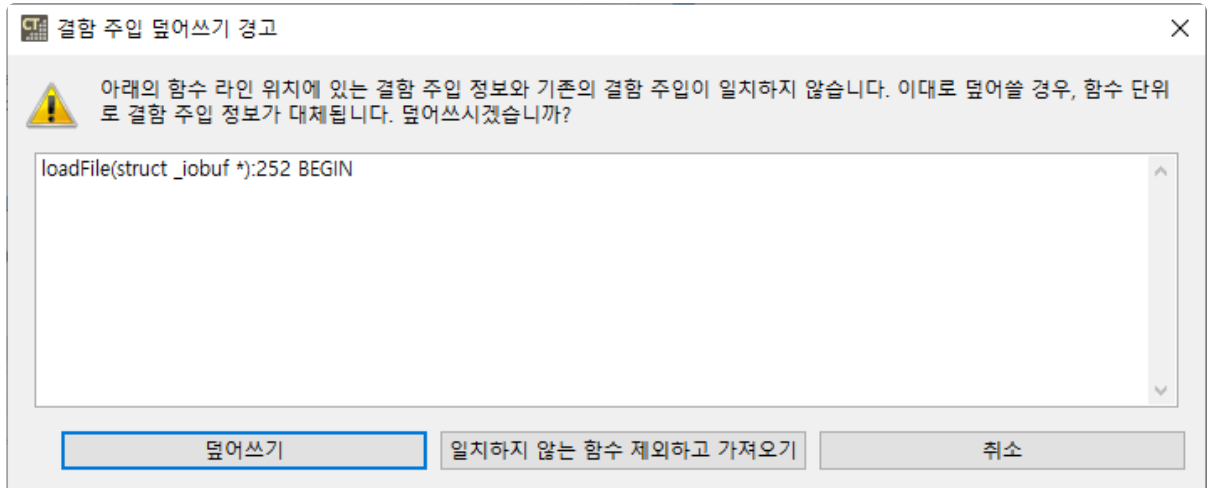
- b. [확인] 버튼을 누르면 결함 주입 가져오기 재설정을 할 수 있습니다.



- 왼쪽 영역에는 가져오려는 결함 주입 정보의 소스 코드 형상이 나타납니다.
  - 오른쪽 영역에는 현재 프로젝트의 소스 코드 형상이 나타납니다.
- c. [결함 주입 가져오기 재설정] 대화상자에서 [확인] 버튼을 누르면 재설정된 결함 주입 정보를 가져올 수 있습니다.

\* 가져오려는 결함 주입 정보의 파일 및 함수 정보가 프로젝트와 다른 경우, 함수 재설정이 먼저 수행됩니다.

4. 프로젝트로 가져오려는 결함 주입 정보와 동일한 위치에 이미 작성된 결함 주입이 있는 경우, 기존 결함 주입에 덮어쓸지 여부를 결정해야 합니다.
  - a. 결함 주입 가져오기를 수행할 때 기존 위치에 결함 주입 정보가 있으면 아래와 같은 대화상자가 뜹니다.



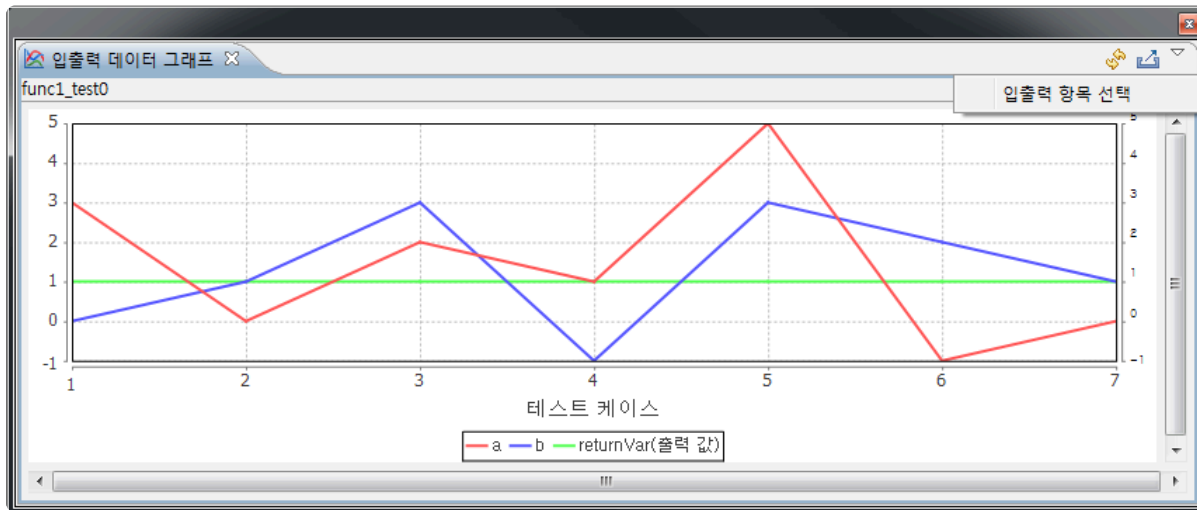
- [결함 주입 덮어쓰기 경고] 대화상자에서는 기존 정보와 일치하지 않는 결함 주입의 함수, 라인, 위치에 대한 내용을 보여줍니다.
- [덮어쓰기]를 선택하면 가져오려는 결함 주입 정보로 기존 결함 주입 정보를 덮어쓰게 됩니다.
- [일치하지 않는 함수 제외하고 가져오기]를 선택하면 일치하지 않는 함수를 제외한 함수들의 결함 주입 정보만 가져옵니다.
- [취소] 버튼을 누르면 결함 주입 가져오기가 취소됩니다.

\* Controller Tester 3.7 이전에 내보낸 결함 주입 파일에는 활성화 여부가 포함되어 있지 않기 때문에, 결함 주입 코드가 있는 라인만 활성화 상태로 가져옵니다.

## 17.14. 입출력 데이터 그래프 뷰

CT 2023.12에서는 입력값/기대값/출력값 데이터를 그래프 형식으로 제공합니다.

그래프의 가로축은 테스트 케이스 번호, 세로축은 테스트 케이스 데이터를 의미합니다.



툴바 아이콘	설명
	새로 고치기
	그래프 저장

[그래프 저장]을 클릭하면 그래프 저장 경로를 입력할 수 있는 알림 창을 출력합니다. 그래프가 저장될 경로, 파일 이름 및 파일 형식을 입력하고 [확인] 버튼을 클릭하면 그래프 저장이 완료됩니다.

### 그래프 설정

풀다운 메뉴에서 [입출력 항목 선택]을 선택하면 그래프 설정 창을 출력합니다.

- 체크박스에 선택된 항목에 대해서 [입력값], [출력값], [기대값]이 입출력 데이터 그래프에 출력이 됩니다. 단, 기대값을 ~, &, |, ! 등으로 지정 시 입출력 데이터 그래프 뷰에 기대값이 반영되지 않습니다.
- [한 화면에 출력할 테스트 케이스 수]에는 사용자가 한 화면에 출력할 테스트 케이스 개수를 지정할 수 있습니다.

입출력 항목 선택

입력값	기대 값	출력 값
<input checked="" type="checkbox"/> int a	returnVar <input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> int b		
<input type="checkbox"/> int c		

한 화면에 출력할 테스트 케이스 수: 20

확인

취소



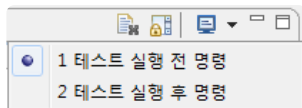
## 17.15. 콘솔 뷰

[프로젝트] -> [특성] -> [테스트] -> [외부 명령 실행]에 [테스트 실행 전 명령], [테스트 실행 후 명령]에 명령어 또는 배치 파일을 입력하고, 유닛 테스트를 실행하면 콘솔 뷰에 테스트 실행 전, 후 외부 명령 수행 결과가 나타납니다.

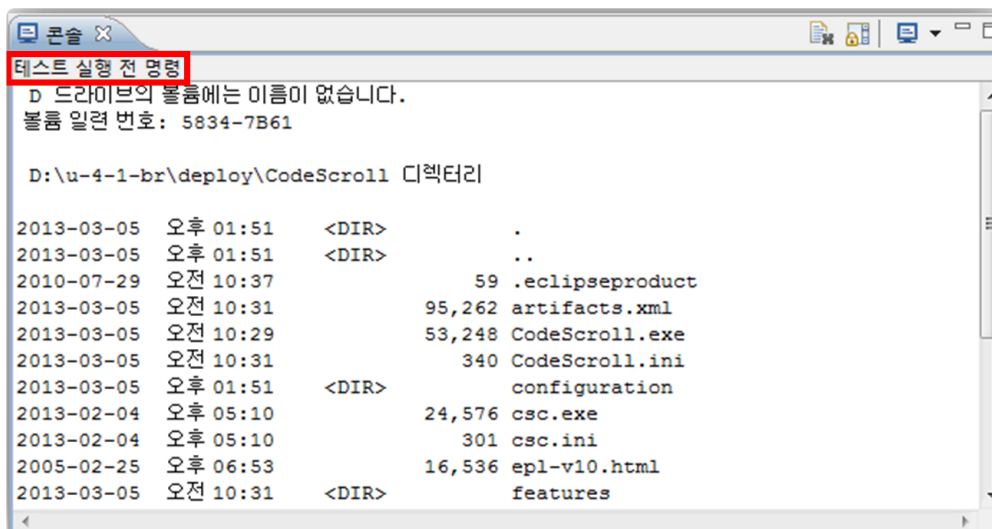
### 콘솔 뷰 툴바 아이콘

툴바 아이콘	설명
 콘솔 지우기	콘솔 내용 지우기
 화면 이동 잠금	콘솔 화면 이동 잠금
 선택된 콘솔 표시	선택된 콘솔 표시

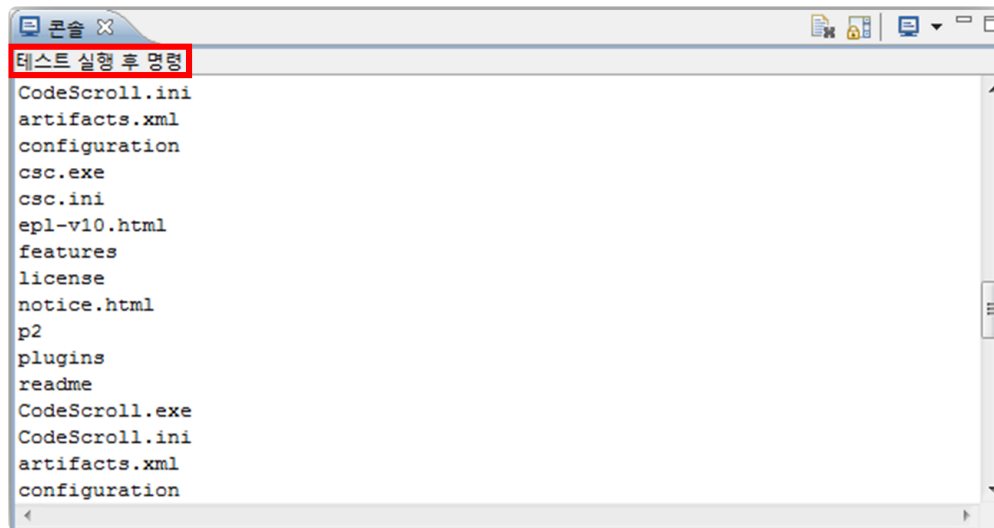
[선택된 콘솔 표시]를 선택하면 [테스트 실행 전 명령]과 [테스트 실행 후 명령] 2개의 메뉴가 나타납니다.



[테스트 실행 전 명령]을 선택하면 테스트 실행 전 외부 명령 수행 결과가 콘솔 뷰에 출력됩니다.

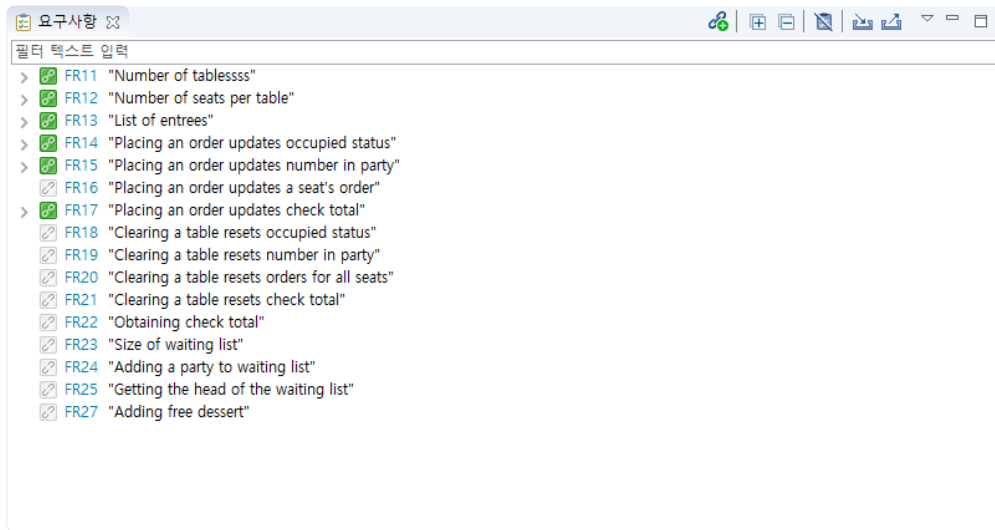


[테스트 실행 후 명령]을 선택하면 테스트 실행 후 외부 명령 수행 결과가 콘솔 뷰에 출력됩니다.



## 17.16. 요구사항 뷰

CSV 파일 또는 요구사항 관리 도구로 작성된 요구사항을 CT 2023.12로 가져올 수 있습니다. 가져온 요구사항을 테스트와 연결하고 연결 정보를 내보낼 수 있습니다.



### 요구사항 상태

상태	설명
연결되지 않음	연결된 테스트가 없는 요구사항
연결됨	연결된 테스트가 있는 요구사항
검토 필요	검토가 필요한 요구사항

### 요구사항 뷰 툴바 메뉴

아이콘	설명
요구사항과 테스트 연결하기	[요구사항과 테스트 연결하기] 대화상자 열기
모두 펼치기	요구사항 트리 모두 펼치기
모두 접기	요구사항 트리 모두 접기
연결되지 않은 요구사항 숨기기	연결되지 않은 요구사항 숨기기
가져오기	CSV 파일 또는 요구사항 관리 도구로 작성된 요구사항 가져오기
내보내기	요구사항 테스트 커버리지 보고서 또는 요구사항 관리 도구로 요구사항 내보내기

자세한 사항은 [내보내기](#), [가져오기](#) 페이지를 참고하시길 바랍니다.

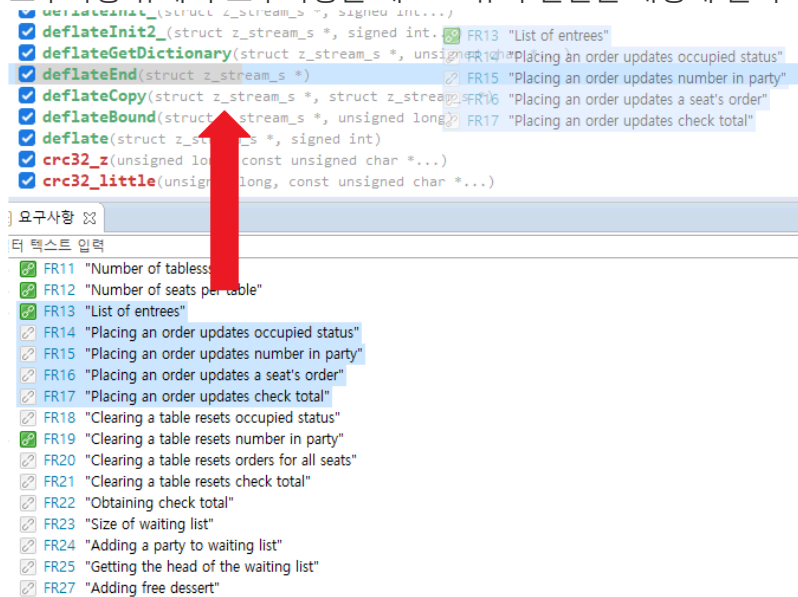
## 요구사항 뷰 풀다운 메뉴

아이콘	설명
 요구사항 추적성 정보 내보내기 (csv)...	요구사항과 테스트 간의 연결 정보 내보내기
 요구사항 추적성 정보 내보내기 [모든 테스트 케이스 포함] (csv)...	모든 테스트 케이스를 포함하여 요구사항과 테스트 간의 연결 정보 내보내기
 요구사항 추적성 정보 가져오기 (csv)...	요구사항과 테스트 간의 연결 정보 가져오기
V-SPICE 요구사항 추적성 정보 내보내기...	요구사항과 테스트 간의 연결 정보를 V-SPICE의 xml 파일 형식으로 내보내기

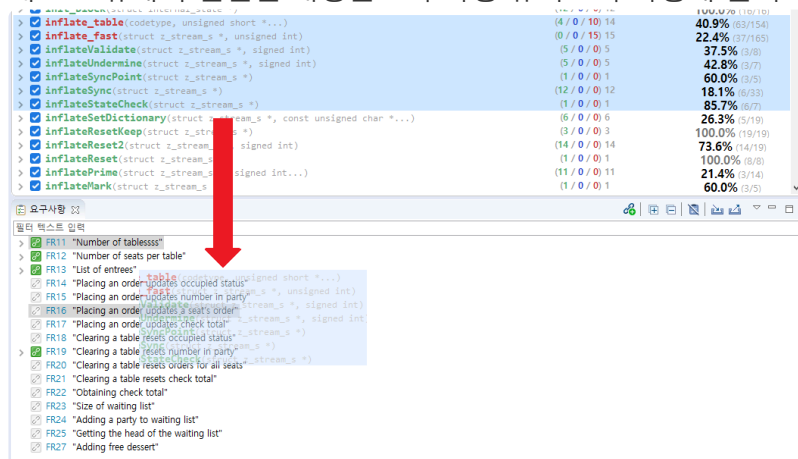
## 테스트와 요구사항 연결하기

요구사항은 테스트/테스트 케이스/통합 테스트에 연결 가능합니다.

- 요구사항 뷰에서 요구사항을 테스트 뷰의 연결할 대상에 끌어서 놓기로 연결하기



- 테스트 뷰에서 연결할 대상을 요구사항 뷰의 요구사항에 끌어서 놓기로 연결하기

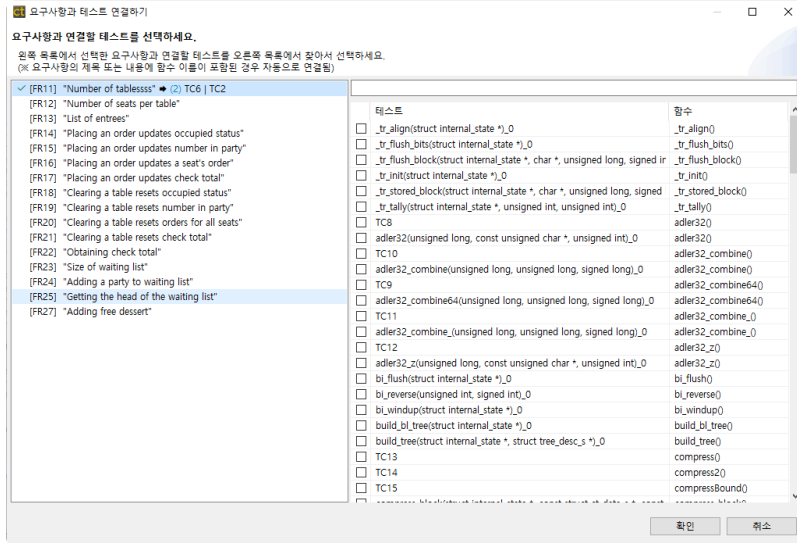


- 요구사항과 테스트 연결하기 대화상자를 통해 연결하기

- 툴바 메뉴의 [요구사항과 테스트 연결하기]를 선택합니다.

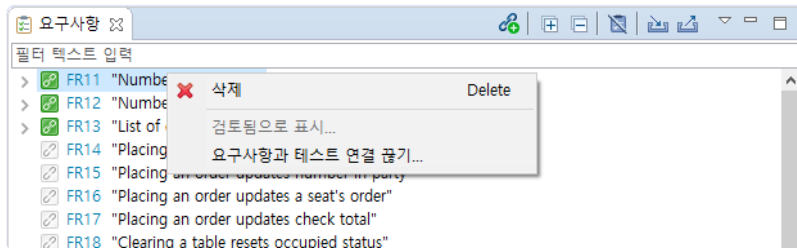


2. 요구사항과 테스트 연결하기 대화상자를 통해 요구사항과 테스트를 연결할 수 있습니다.



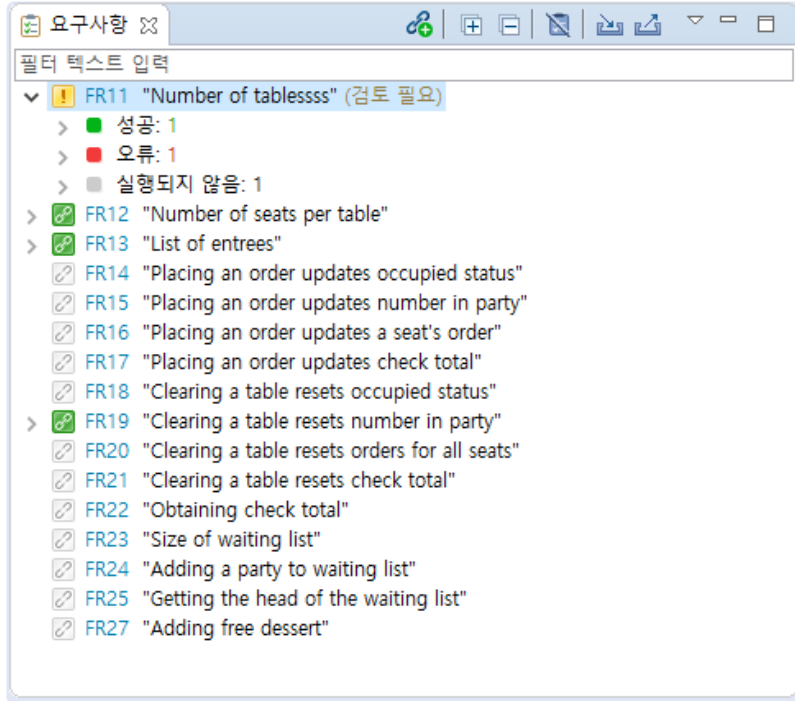
## 요구사항과 테스트 연결 끊기

요구사항 뷰에서 컨텍스트 메뉴로 요구사항과 테스트의 연결을 끊을 수 있습니다.

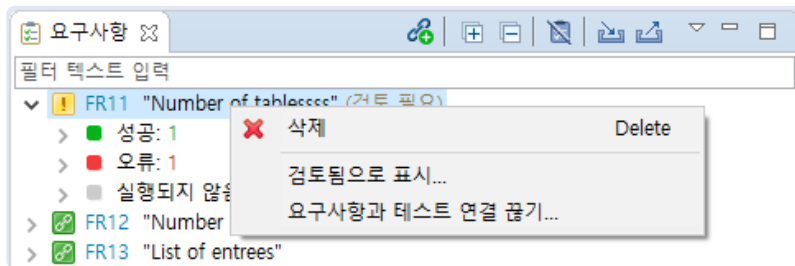


## 요구사항과 연결된 테스트 검토

1. 요구사항을 다시 가져왔을 때, 제목 또는 설명이 변경된 경우 '검토 필요' 상태로 표시됩니다.

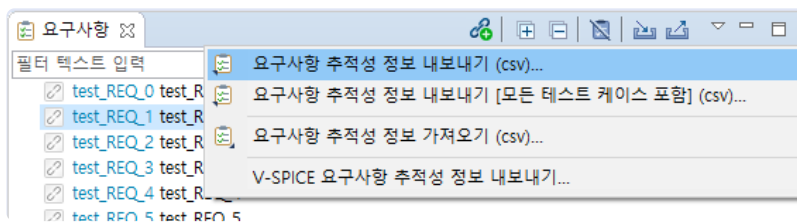


2. 검토가 필요한 요구사항과 연결된 테스트를 검토한 후에 컨텍스트 메뉴로 검토됨으로 표시할 수 있습니다.



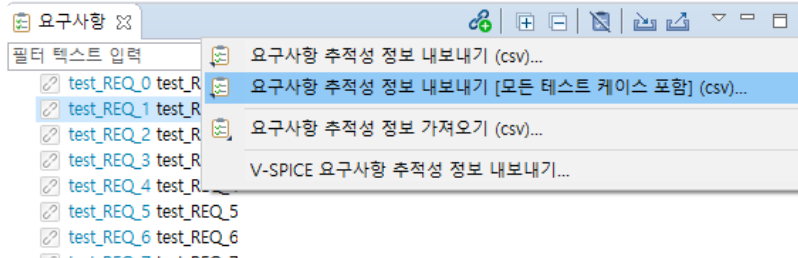
## 요구사항 추적성 정보 내보내기

요구사항과 테스트 간의 연결 정보를 CSV 파일로 내보낼 수 있습니다.



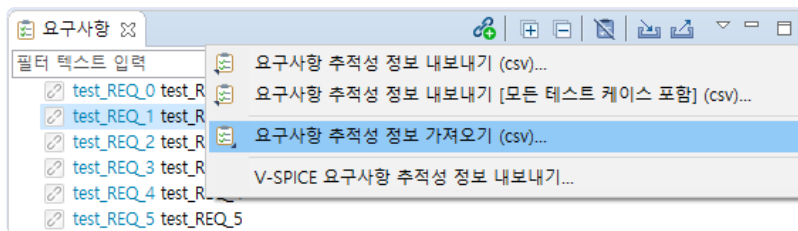
## 요구사항 추적성 정보 내보내기 [모든 테스트 케이스 포함]

모든 테스트 케이스를 포함한 요구사항과 테스트 간의 연결 정보를 CSV 파일로 내보낼 수 있습니다.



## 요구사항 추적성 정보 가져오기

[요구사항 추적성 정보 내보내기]한 CSV 파일에 추적성 정보를 입력한 후 [요구사항 추적성 정보 가져오기] 메뉴를 선택하여 요구사항 추적성 정보를 가져올 수 있습니다.



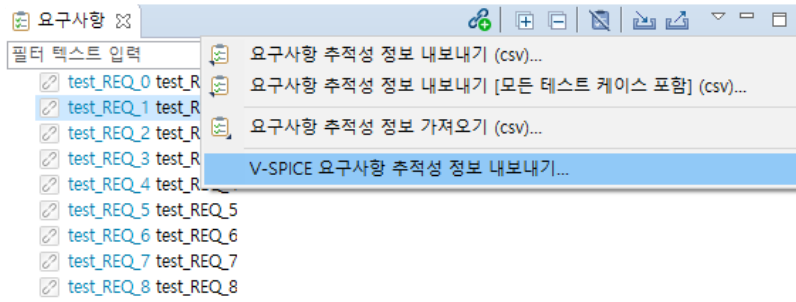
- 요구사항 추적성 정보 파일 설명

열	설명
TEST_TYPE	테스트: 0 통합 테스트: 1 테스트 케이스: 2
TEST_NAME	통합 테스트: 이름 테스트 또는 테스트 케이스: 테스트의 고유한 이름
TESTCASE_NO	테스트 케이스인 경우 : 번호 테스트 케이스 외의 경우: 0
REQ_KEY	테스트와 연결된 또는 연결하려는 요구사항의 키
TEST_ID	통합 테스트: 이름 테스트: 테스트 뷰에 표시되는 이름 테스트 케이스: 테스트 뷰에서 번호 옆에 표시되는 이름

✿ 테스트의 고유한 이름은 테스트 뷰의 툴바 메뉴에서 '고유한 테스트 이름으로 보기'를 선택하거나, 검색하여 찾을 수 있습니다.

## V-SPICE 요구사항 추적성 정보 내보내기

요구사항과 테스트 간의 연결 정보를 V-SPICE 형식에 맞는 xml 파일로 내보낼 수 있습니다.



✿ 내보내기한 xml 파일은 모든 테스트 정보를 포함합니다.

## 요구사항 관리 도구에서 열기

요구사항 관리 도구에서 가져온 요구사항이나 요구사항 관리 도구로 내보낸 테스트는 [브라우저로 열기] 컨텍스트 메뉴를 통해 요구사항 관리 도구에서 확인할 수 있습니다.





## 18. 분석 퍼스펙티브

---

분석 퍼스펙티브는 테스트 결과로부터 확인할 수 있는 분석 정보들에 대한 UI를 제공합니다.

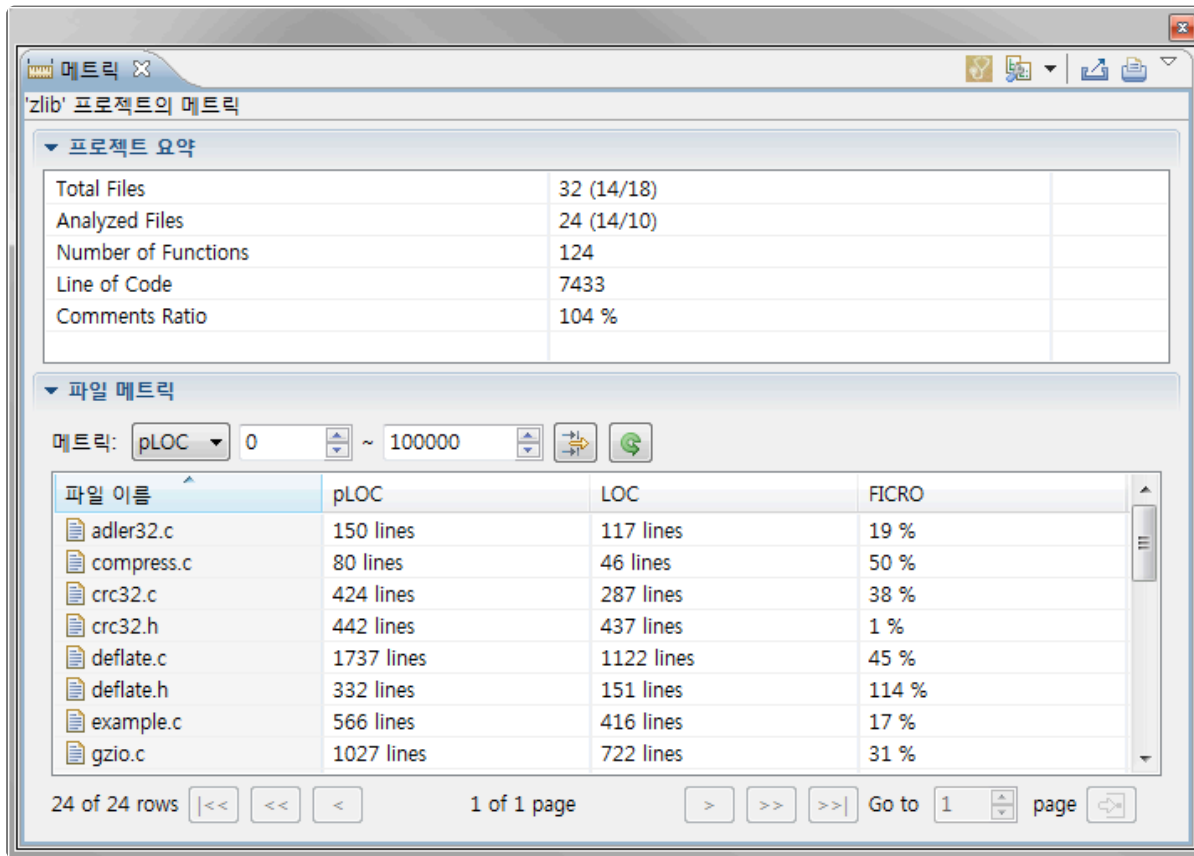
분석 퍼스펙티브를 구성하는 요소는 다음과 같습니다.

- [메트릭 뷰](#)
- [메트릭 차트 뷰](#)
- [메트릭 탭 차트 뷰](#)
- [메트릭 바 차트 뷰](#)
- [메트릭 진단 차트 뷰](#)
- [미사용 함수 뷰](#)
- [소스-헤더 연관 뷰](#)
- [전역 변수 연관 뷰](#)
- [함수 호출 계층 뷰](#)
- [소스 코드 편집기](#)

기본으로 포함되지 않는 뷰는 상단 메뉴의 [창] > [뷰 표시] > [기타...] 에서 열 수 있습니다.

## 18.1. 메트릭 뷰

메트릭 뷰는 프로젝트의 메트릭을 측정한 결과를 보여줍니다. 프로젝트 전체 메트릭 요약 정보와, 프로젝트에 속한 각 항목들(모듈, 파일, 클래스, 함수)의 메트릭 정보를 보여줍니다.



### 아이콘 설명

아이콘	설명
	모듈(프로젝트를 기능별로 분할한 논리적인 단위)
	파일(프로젝트에 속한 소스 파일과 헤더 파일)
	클래스
	함수

### 툴바 메뉴

메뉴	설명
진단 이미지 표시	메트릭 값의 좌측에 해당 값에 대한 진단 이미지 표시
모드 변경	보기 모드를 변경 (  모듈,  파일,  클래스,  함수)
뷰 내용 내보내기	뷰의 내용을 리포트로 내보내기



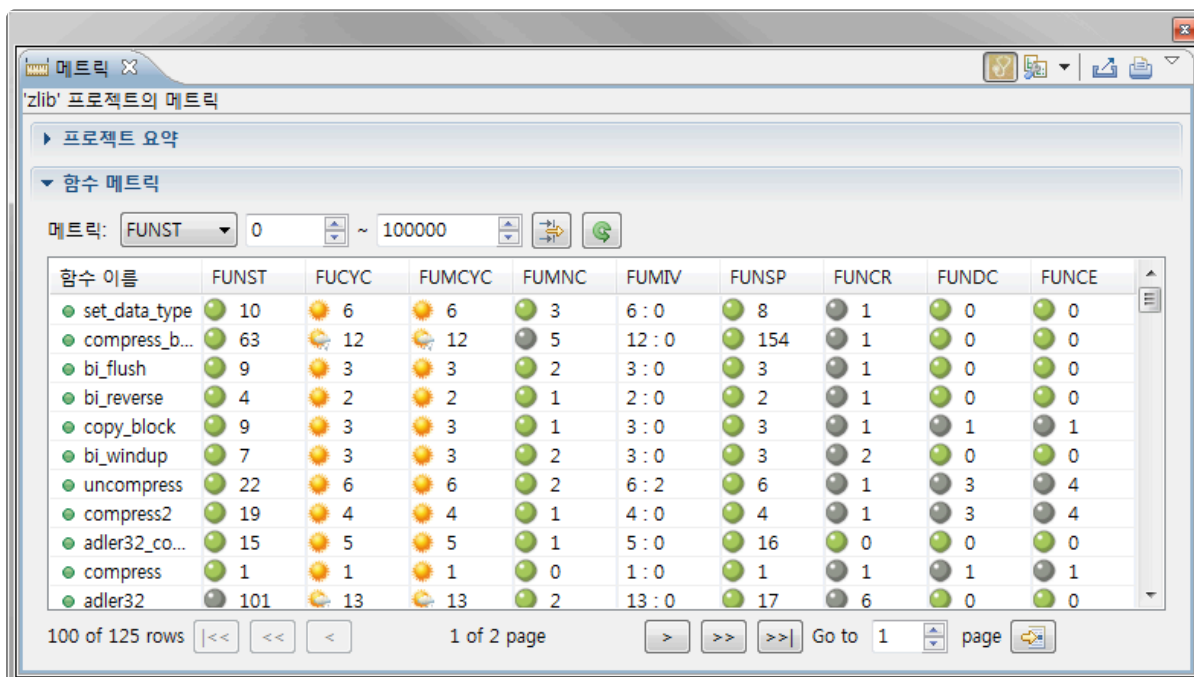
뷰 내용 인쇄

뷰의 내용을 인쇄

## 폴다운 메뉴

메뉴	설명
정렬	선택한 항목을 기준으로 정렬(테이블의 열 이름을 클릭해도 동일하게 정렬됨)
열	열의 순서와 너비를 변경(테이블의 열 이름 부분을 끌어서 놓기로 순서 변경 가능)
진단 정보 구성	진단 환경 설정 페이지를 띄움(각각의 메트릭에 대한 진단 단계와 범위 및 이미지 설정 가능)
메트릭 숨김/표시 설정	메트릭 표시 여부 설정
메트릭 뷰 옵션	메트릭 뷰의 '페이지 당 줄 수'를 설정

## 필터링



필터링 기능으로 특정 메트릭에 대하여 원하는 범위에 있는 항목들만 볼 수 있습니다.

메트릭을 선택하고 범위 값을 넣은 뒤에 우측의 [필터링] 버튼을 클릭하면, 입력한 정보에 해당하는 값들만 보여집니다.

필터링 이전 상태로 돌아가고 싶다면 [필터링] 버튼 우측의 [필터링 기준 초기화] 버튼을 클릭합니다.

## 페이징

메트릭 X

'zlib' 프로젝트의 메트릭

▶ 프로젝트 요약

▼ 함수 메트릭

메트릭: FUNST 0 ~ 100000

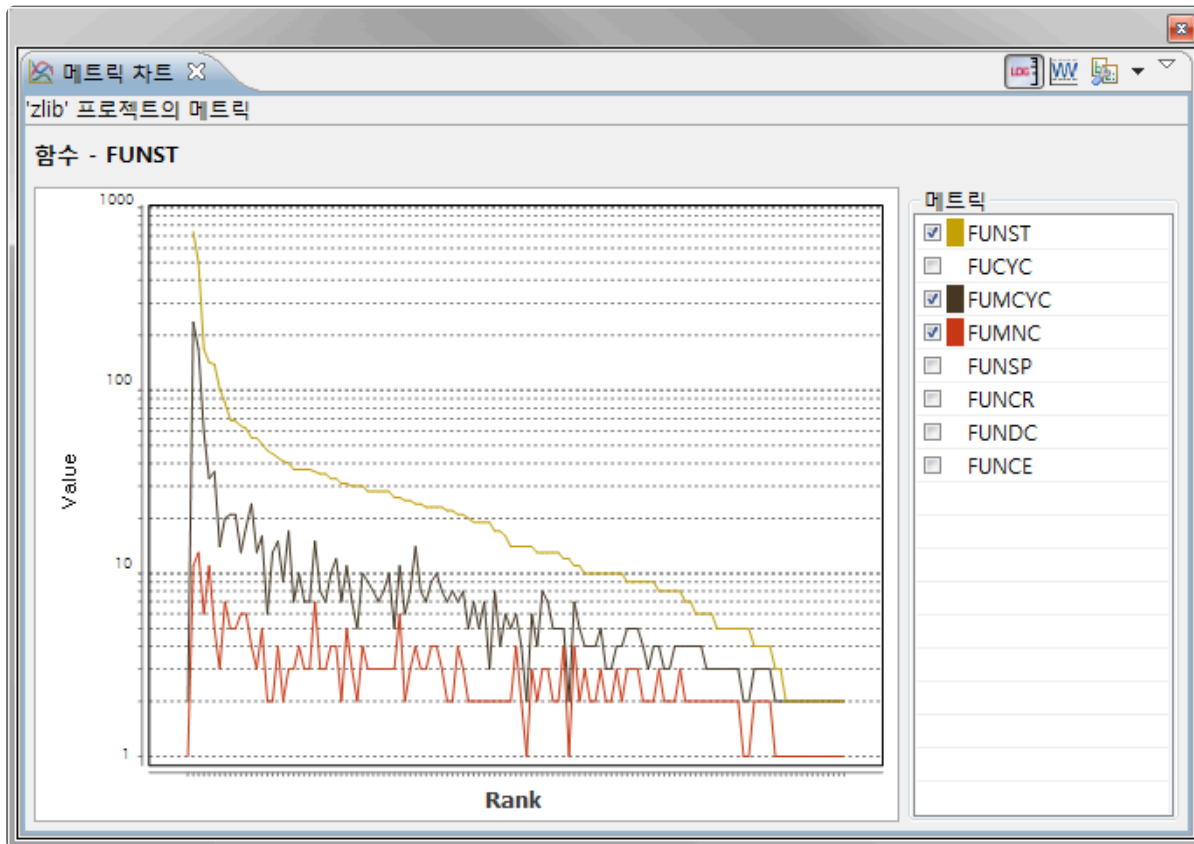
함수 이름	FUNST	FUCYC	FUMCYC	FUMNC	FUMIV	FUNSP	FUNCR	FUNDC	FUNCE
set_data_type	10	6	6	3	6 : 0	8	1	0	0
compress_b...	63	12	12	5	12 : 0	154	1	0	0
bi_flush	9	3	3	2	3 : 0	3	1	0	0
bi_reverse	4	2	2	1	2 : 0	2	1	0	0
copy_block	9	3	3	1	3 : 0	3	1	1	1
bi_windup	7	3	3	2	3 : 0	3	2	0	0
uncompress	22	6	6	2	6 : 2	6	1	3	4
compress2	19	4	4	1	4 : 0	4	1	3	4
adler32_co...	15	5	5	1	5 : 0	16	0	0	0
compress	1	1	1	0	1 : 0	1	1	1	1
adler32	101	13	13	2	13 : 0	17	6	0	0

100 of 125 rows 1 of 2 page

페이징 기능으로 많은 항목들을 편리하게 볼 수 있습니다.

## 18.2. 메트릭 차트 뷰

메트릭 차트 뷰는 메트릭 결과를 꺾은선 차트로 보여줍니다. x 축은 각각의 항목들이고, y 축은 항목의 메트릭 값입니다. 프로젝트에서 서로 다른 메트릭 사이의 값의 분포를 비교할 수 있습니다.



### 툴바 메뉴

메뉴	설명
로그 스케일 표시	차트의 y축을 로그 단위로 변경
표준화된 메트릭 표시	표준화된 메트릭 값을 표시
모드 변경	보기 모드를 변경 (모듈, 파일, 클래스, 함수)

### 풀다운 메뉴

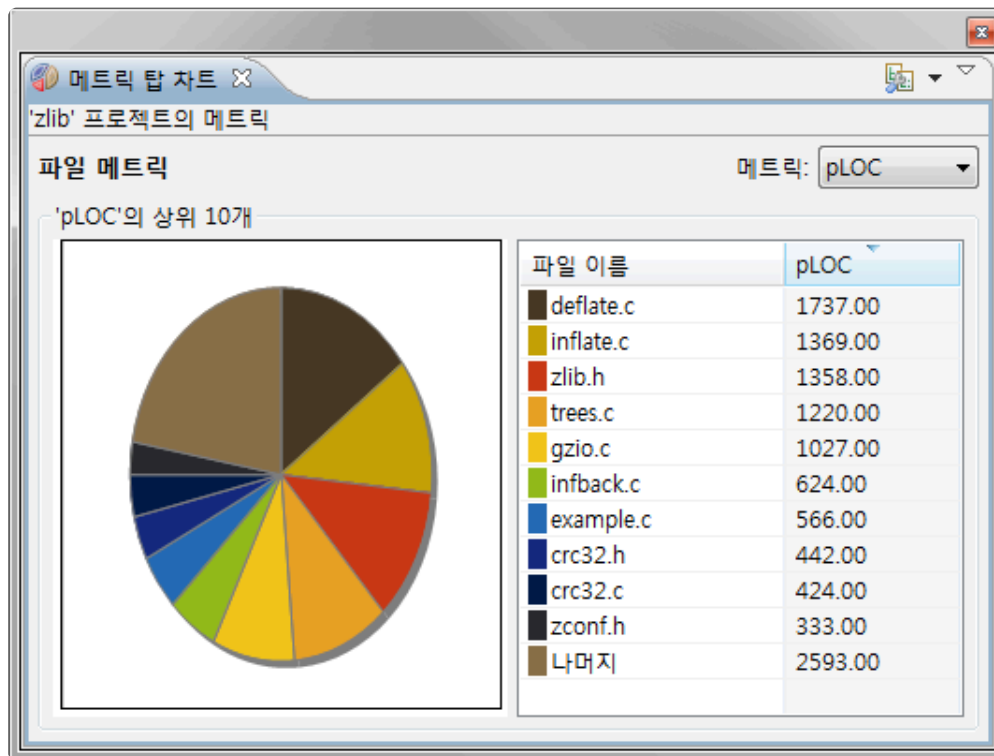
메뉴	설명
정렬	선택한 항목을 기준으로 정렬(테이블에서 메트릭을 선택해도 동일하게 정렬됨)
메트릭 숨김/표시 설정	메트릭 표시 여부 설정
메트릭 뷰 옵션	메트릭 뷰와 연동 기능에 대한 설정

## 메트릭 선택

우측의 테이블에서 메트릭 이름 좌측의 체크 박스를 체크하면 체크된 메트릭이 차트에 보여집니다. 체크된 메트릭 중에서 메트릭 이름 부분을 클릭해서 선택하면, 해당 메트릭 값을 기준으로 항목들이 정렬됩니다.

## 18.3. 메트릭 탭 차트 뷰

메트릭 탭 차트 뷰는 메트릭 값의 상위 항목들을 파이 차트로 보여줍니다. 전체에서 상위 항목들이 차지하는 비중을 볼 수 있습니다.



메뉴	설명
모드 변경	보기 모드를 변경 (  모듈,  파일,  클래스,  함수 )

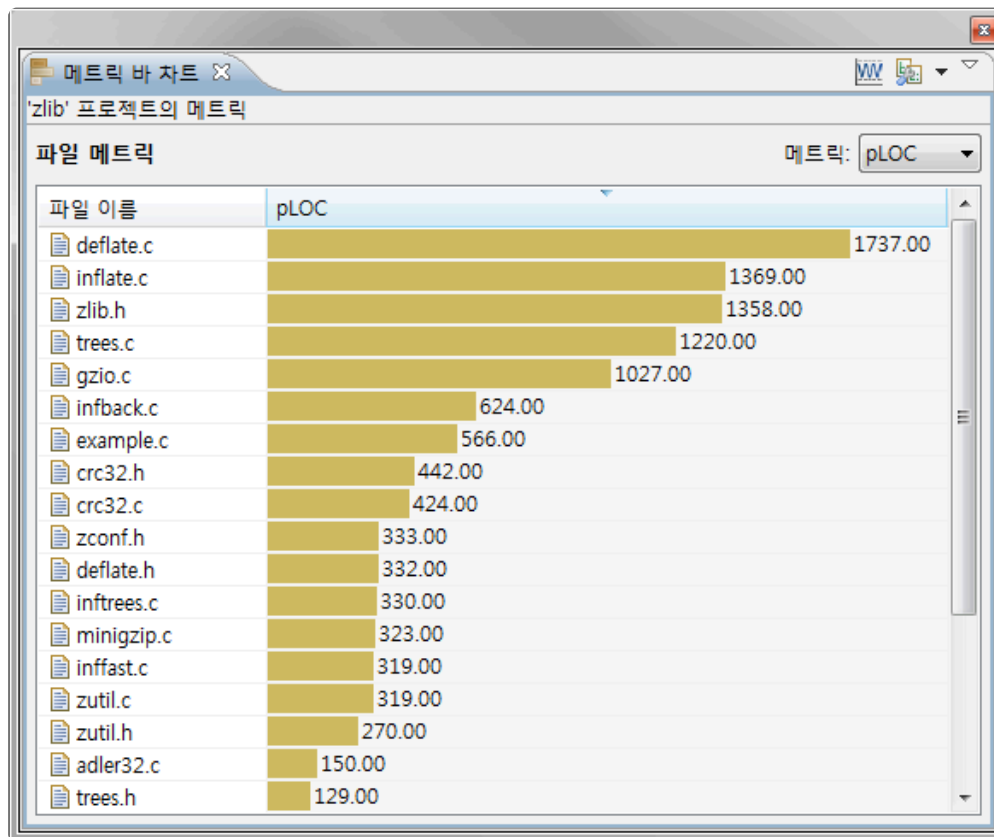
### 폴다운 메뉴

메뉴	설명
메트릭 숨김/표시 설정	메트릭을 숨길지 또는 표시할지 설정
메트릭 뷰 옵션	표시할 최상위 항목 개수와 나머지 항목 표시 여부 설정

파이 차트의 선택과 테이블의 선택된 항목이 서로 연동됩니다.

## 18.4. 메트릭 바 차트 뷰

메트릭 바 차트 뷰는 메트릭 값을 바 차트로 보여줍니다. 각 항목들의 값의 크기를 바 차트로 확인할 수 있습니다.



### 아이콘 설명

아이콘	설명
	모듈(프로젝트를 기능별로 분할한 논리적인 단위)
	파일(프로젝트에 속한 소스 파일과 헤더 파일)
	클래스
	함수

### 툴바 메뉴

메뉴	설명
표준화된 메트릭 표시	표준화된 메트릭 값을 표시
모드 변경	보기 모드를 변경 (  모듈,  파일,  클래스,  함수 )

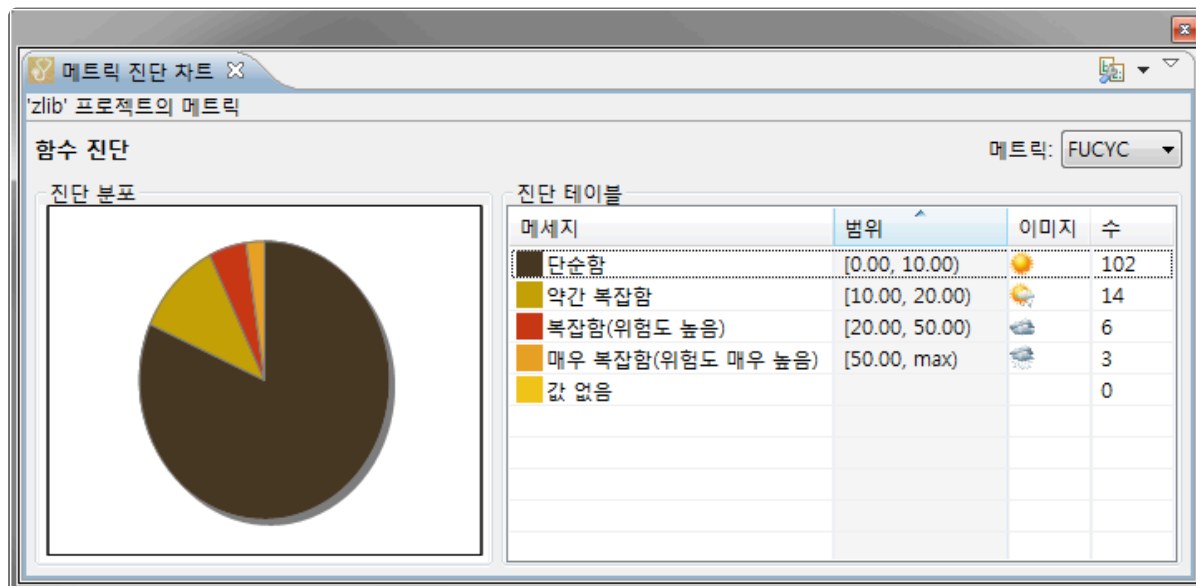


## 폴다운 메뉴

메뉴	설명
메트릭 숨김/표시 설정	메트릭 표시여부 설정
메트릭 뷰 옵션	표시할 항목의 개수 설정

## 18.5. 메트릭 진단 차트 뷰

메트릭 진단 차트 뷰는 진단 범위에 속하는 항목의 수를 파이 차트로 보여줍니다. 프로젝트 내에서 각 진단 범위에 항목들의 수를 확인할 수 있습니다.



### 툴바 메뉴

메뉴	설명
모드 변경	보기 모드를 변경 (📦 모듈, 📄 파일, 🟢 클래스, 🟢 함수)

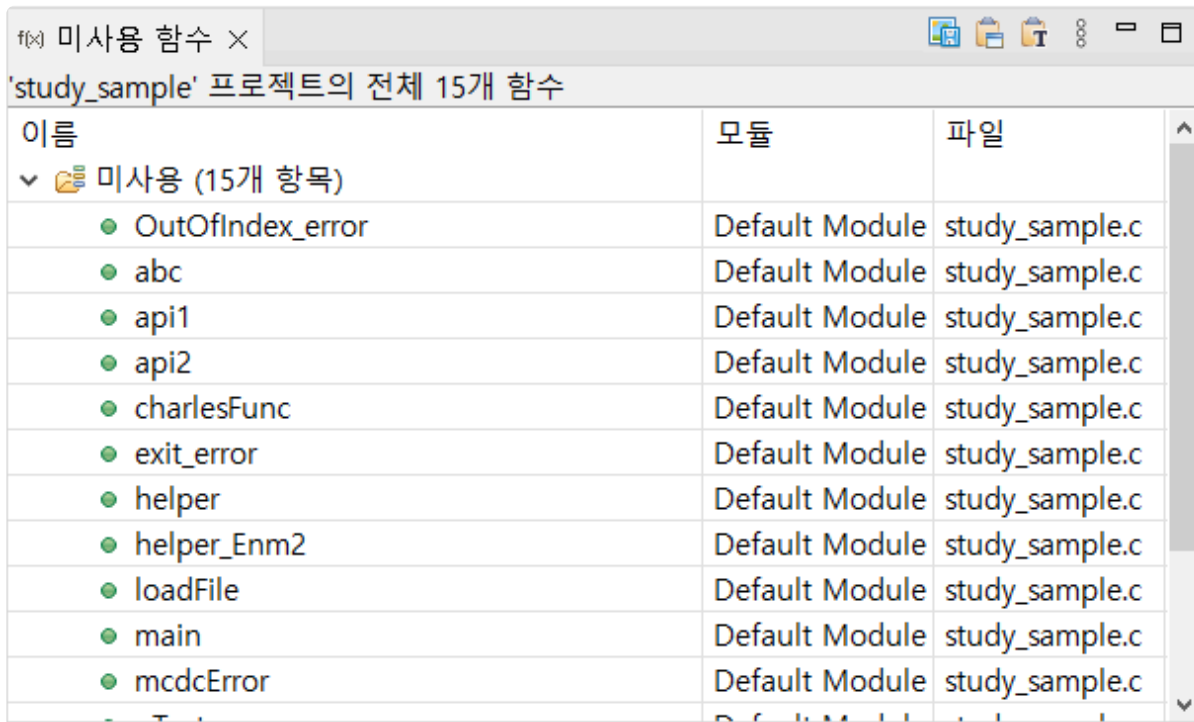
### 풀다운 메뉴

메뉴	설명
진단 정보 구성	진단 환경 설정 페이지를 띄움(각각의 메트릭에 대한 진단 단계와 범위 및 이미지 설정 가능)
메트릭 숨김/표시 설정	메트릭 표시 여부 설정
메트릭 뷰 옵션	진단 값이 없는 항목 표시 여부 설정

파이 차트의 선택과 테이블의 선택된 항목이 서로 연동됩니다.

## 18.6. 미사용 함수 뷰

미사용 함수 뷰는 선택된 프로젝트에 속한 함수들을 정해진 기준으로 분류하여 보여줍니다. 현재 미사용 함수들을 보여주고 있으며, 분류하는 기준은 추가될 수 있습니다.



메뉴	설명
 이미지로 저장	미사용 함수 뷰를 이미지로 저장
 시스템 클립보드에 복사	[미사용 함수 뷰] 이미지를 시스템 클립보드에 복사
 텍스트로 복사	[미사용 함수 뷰]의 콘텐츠를 클립보드에 텍스트로 복사

### 아이콘 설명

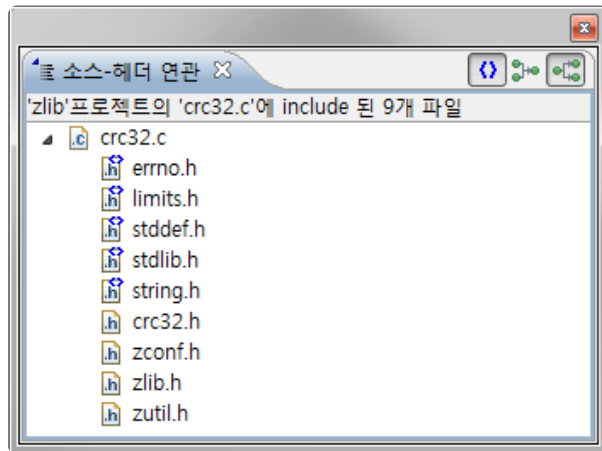
아이콘	설명
	분류 기준
	함수 그룹
	함수

### 폴다운 메뉴

메뉴	설명
함수 뷰 옵션	그룹별로 보여줄 함수의 수 설정

## 18.7. 소스-헤더 연관 뷰

소스-헤더 연관 뷰는 소스 파일과 헤더 파일 사이의 <include> 와 #include 관계를 보여줍니다.



### 아이콘 설명

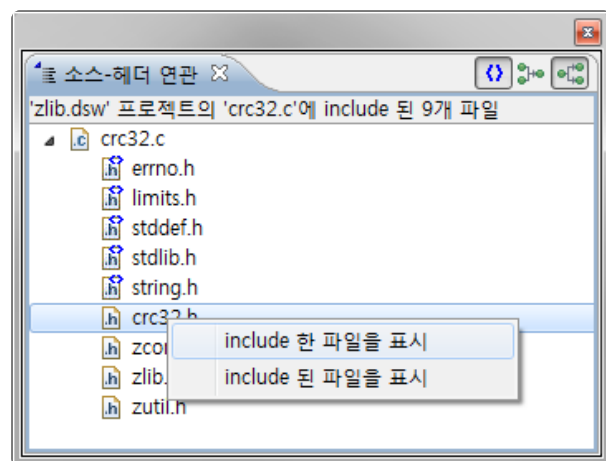
아이콘	설명
	소스 파일
	헤더 파일
	시스템 헤더 파일

### 툴바 메뉴

메뉴	설명
시스템 헤더 표시	시스템 헤더를 표시
include한 파일 표시	선택한 파일을 include한 파일을 표시
include된 파일 표시	선택한 파일에 include된 파일을 표시

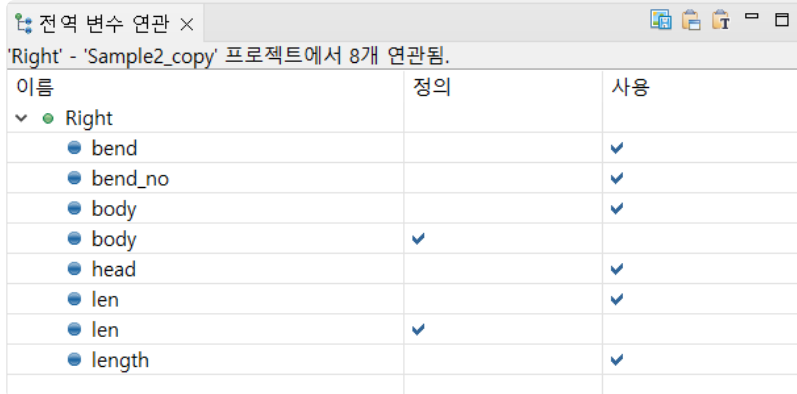
include 한 파일을 표시, include 된 파일을 표시

뷰에서 파일을 오른쪽 클릭하면 나타나는 메뉴를 통해, 선택한 파일을 include 한 파일을 표시하거나, 선택한 파일이 include 된 파일을 표시할 수 있습니다.






## 18.8. 전역 변수 연관 뷰

전역 변수 연관 뷰는 전역 변수가 어떤 함수에서 정의되고 사용되었는지를 보여줍니다. 함수가 선택되면 해당 함수에서 정의되거나 사용된 전역 변수를 보여주고, 전역 변수가 선택되면 해당 전역 변수가 정의되거나 사용된 함수를 보여줍니다.



이름	정의	사용
Right		
bend		✓
bend_no		✓
body		✓
body	✓	
head		✓
len		✓
len	✓	
length		✓

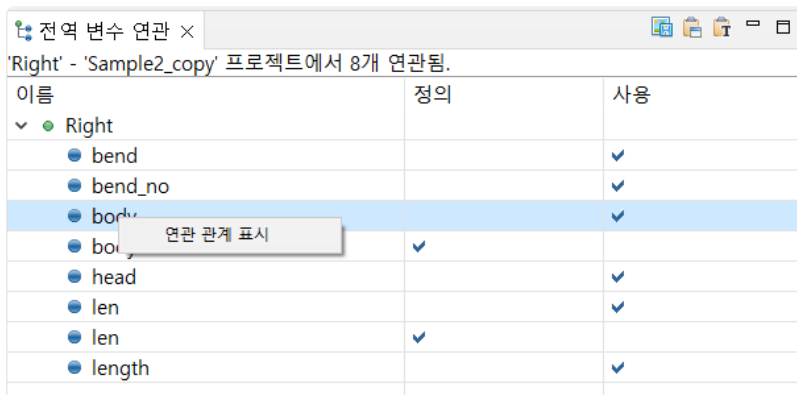
메뉴	설명
 이미지로 저장	전역 변수 연관 뷰를 이미지로 저장
 시스템 클립보드에 복사	[전역 변수 연관 뷰] 이미지를 시스템 클립보드에 복사
 텍스트로 복사	[전역 변수 연관 뷰]의 콘텐츠를 클립보드에 텍스트로 복사

### 아이콘 설명

아이콘	설명
	함수
	전역 변수

### 연관 관계 표시

뷰에서 연관된 항목을 오른쪽 클릭하면 나타나는 메뉴를 통해, 선택한 항목을 중심으로 다시 연관 관계를 볼 수 있습니다.

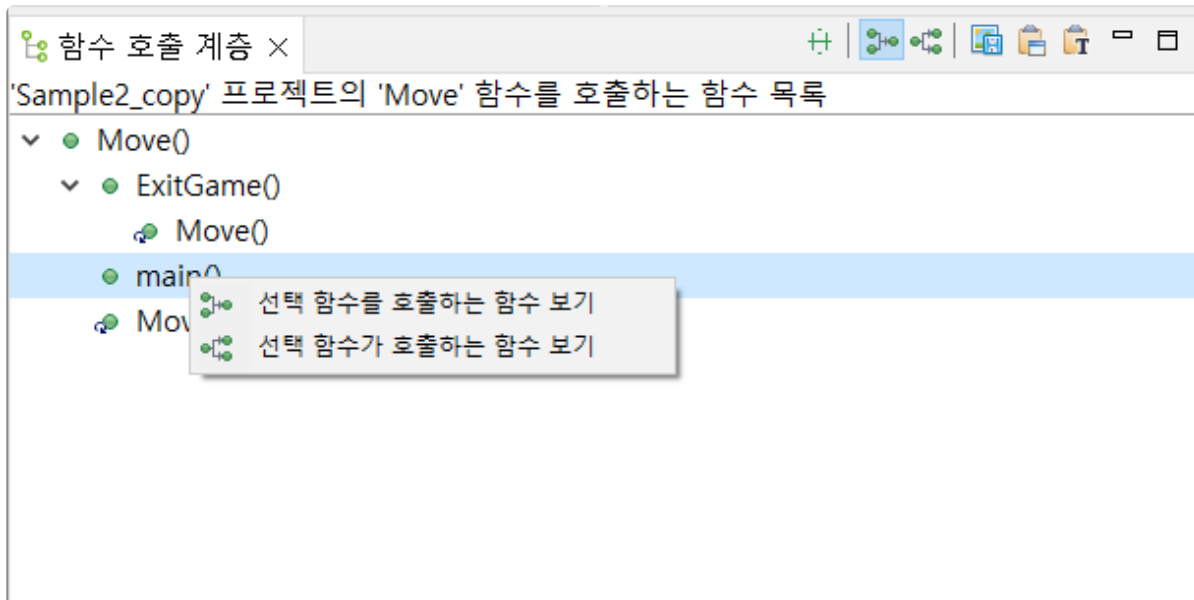
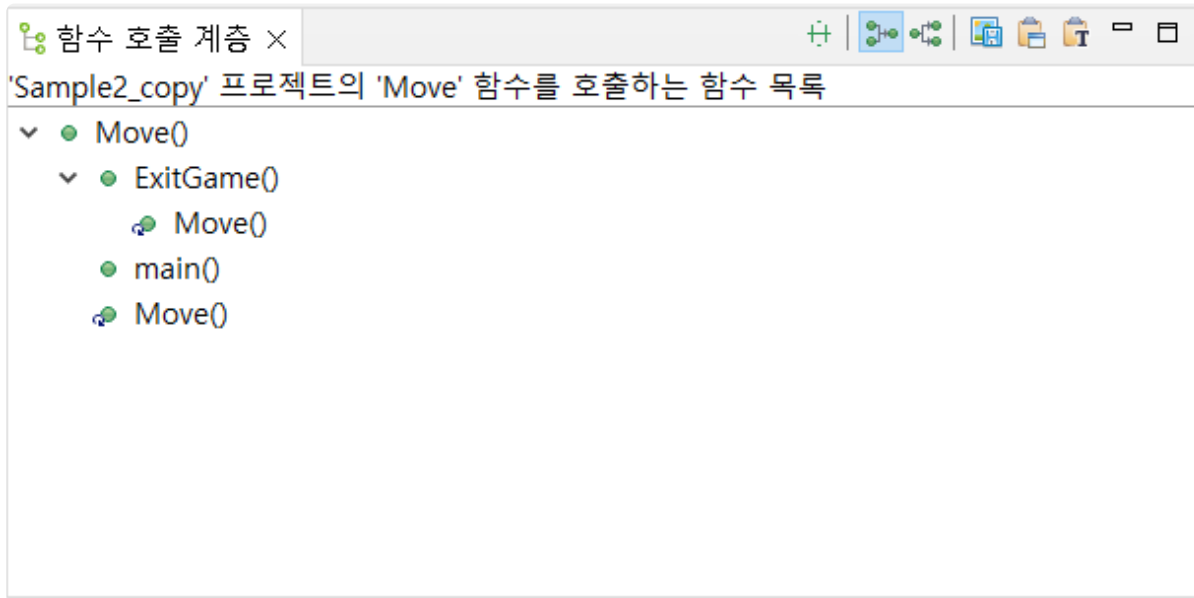


이름	정의	사용
Right		
bend		✓
bend_no		✓
body		✓
body	✓	
head		✓
len		✓
len	✓	
length		✓

## 18.9. 함수 호출 계층 뷰

[함수 호출 계층 뷰]는 선택된 함수가 호출하는 또는 선택된 함수를 호출하는 함수 정보를 계층 구조로 보여줍니다.

계층 구조에 표시되는 함수를 선택하고 우 클릭하였을 때 나타나는 컨텍스트 메뉴를 이용하여 선택한 함수의 호출 정보로 변경할 수 있으며, 더블 클릭을 하면 함수가 정의된 [소스 코드 편집기]가 열리고 선택한 함수의 위치로 이동합니다. (단, 시스템 함수인 경우에는 포함되지 않습니다.)



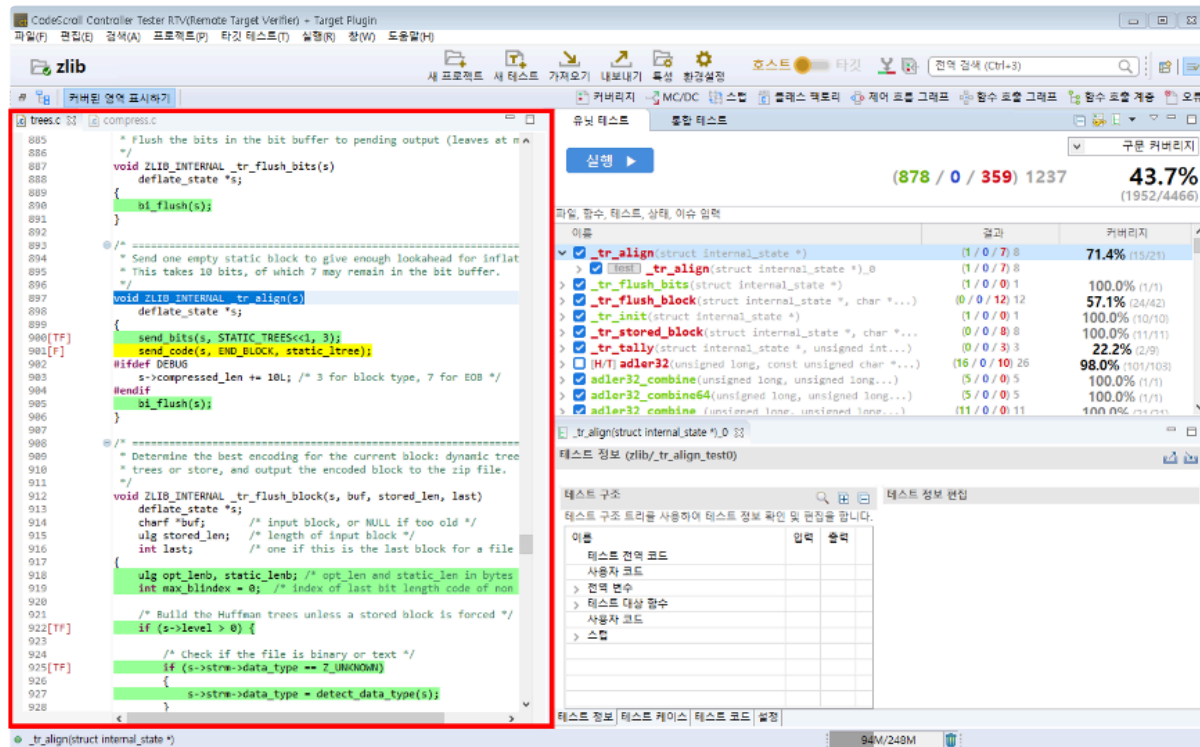
메뉴	설명
 시스템 함수 보기	시스템 함수 호출 정보를 계층구조에 표시여부 설정
 선택한 함수를 호출하는 함수 보기	선택한 함수를 호출하는 함수들을 계층 구조로 표시
 선택한 함수가 호출하는 함수 보기	선택한 함수가 호출하는 함수들을 계층 구조로 표시
 이미지로 저장	[함수 호출 계층 뷰]를 이미지로 저장

 시스템 클립보드에 복사	[함수 호출 계층 뷰] 이미지를 시스템 클립보드에 복사
 텍스트로 복사	[함수 호출 계층 뷰]의 콘텐츠를 클립보드에 텍스트로 복사



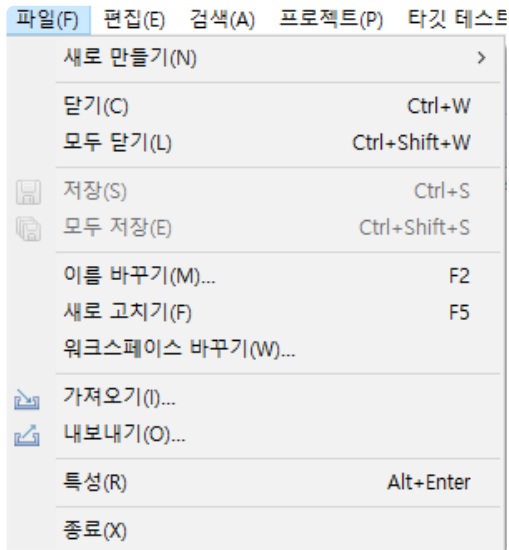
# 18.10. 소스 코드 편집기

소스 코드 편집기는 기본으로 화면 왼쪽에 위치합니다.



## 19. 파일 메뉴

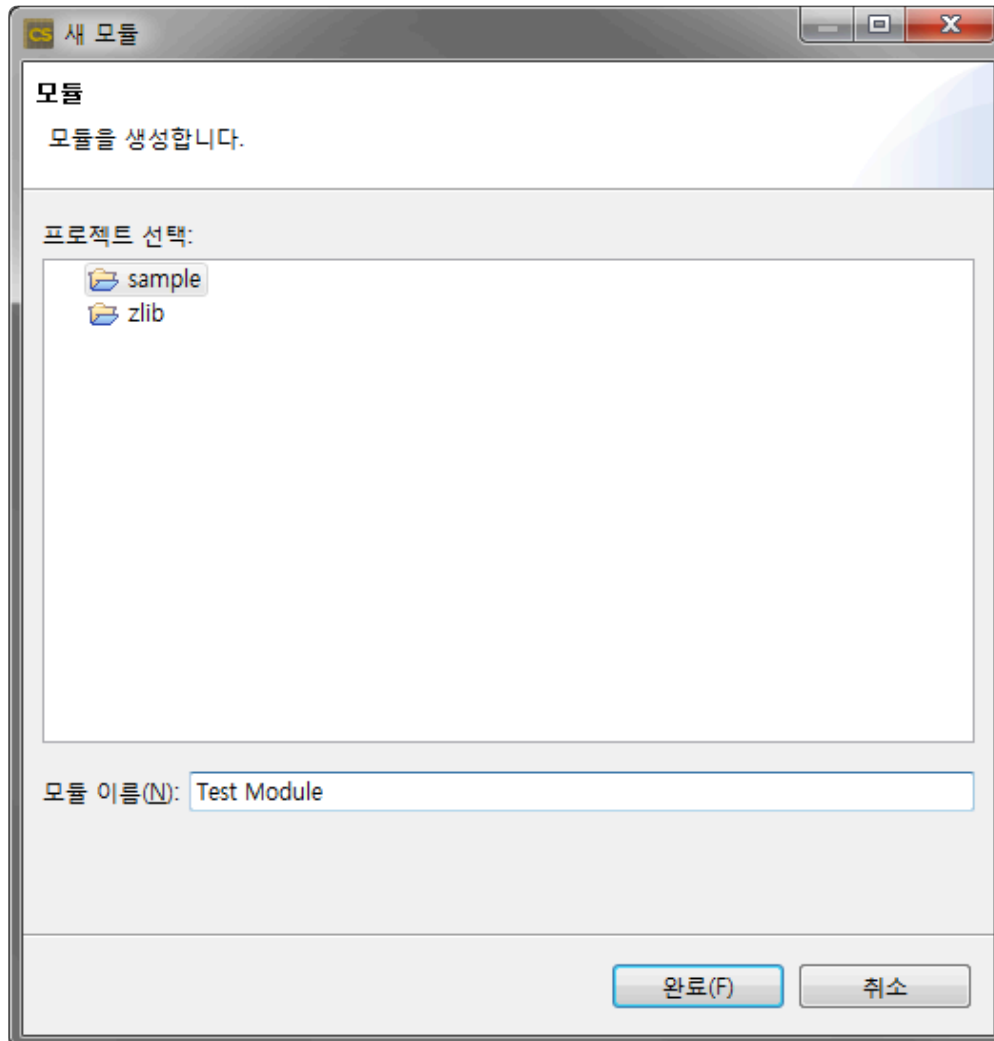
파일 메뉴에서는 새 프로젝트 또는 모듈 생성, 편집기 관련 작업, 새로 고치기, 워크스페이스 바꾸기, 가져오기와 내보내기, 선택된 항목의 특성 보기, 도구 종료 등을 수행할 수 있습니다.



- 새로 만들기:
  - [프로젝트 생성](#)
  - [모듈 생성](#)
- 닫기/모두 닫기/저장/모두 저장: 편집기 관련 메뉴입니다.
- 이름 바꾸기: 프로젝트 또는 모듈의 이름을 변경할 수 있습니다.
- [워크스페이스 바꾸기](#)
- [가져오기](#)
- [내보내기](#)
- 종료: 도구를 종료합니다.

## 19.1. 모듈 생성

프로젝트에 새 모듈을 생성할 수 있습니다. 모듈 이름을 제외한 다른 정보는 대상 프로젝트에 이미 존재하는 모듈과 같게 생성됩니다.



1. [새로 만들기] 메뉴에서 [기타...]를 클릭한 뒤에, [기타] 분류의 [모듈]을 선택하고 [다음]을 클릭합니다.
2. 새 모듈을 생성할 프로젝트를 선택합니다.
3. 모듈 이름을 입력합니다. 이미 존재하는 모듈의 이름은 사용할 수 없습니다.
4. [완료]를 클릭하면 모듈이 생성됩니다.

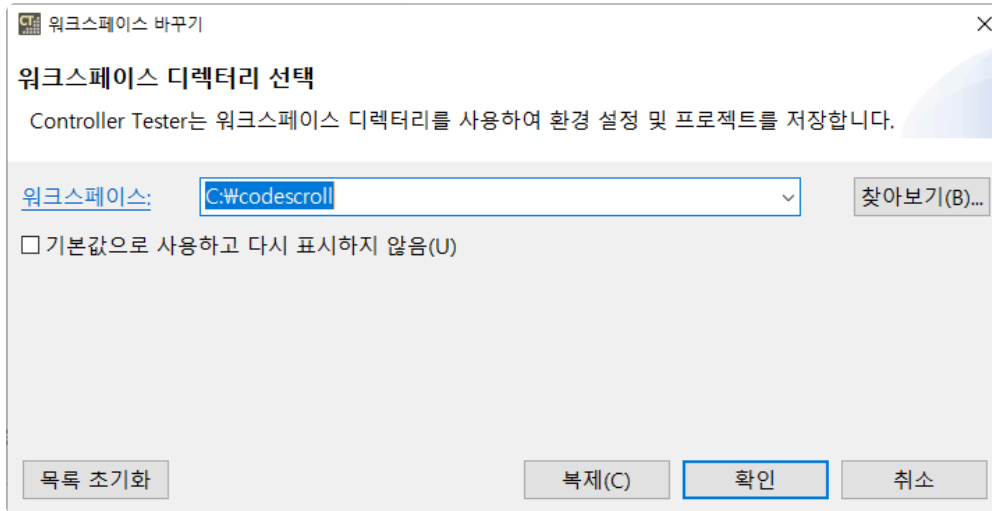


테스트 네비게이터 뷰에서 프로젝트를 오른쪽 클릭하면 나타나는 [새로 만들기] 메뉴를 통해서도 모듈을 생성할 수 있습니다.

## 19.2. 워크스페이스 바꾸기

### 워크스페이스 바꾸기

워크스페이스 디렉터리를 변경할 수 있습니다. 워크스페이스를 변경하면 변경된 워크스페이스로 도구가 다시 시작됩니다.



[찾아보기]를 클릭하여 변경할 워크스페이스 디렉터리를 선택하거나, 수동으로 입력할 수 있습니다. [v]를 클릭하면 나타나는 리스트에서 이전에 선택했던 워크스페이스 디렉터리를 선택할 수 있습니다.

[기본값으로 사용하고 다시 표시하지 않음]을 선택하면 다음에 도구를 실행할 때 워크스페이스 디렉터리 선택을 다시 묻지 않습니다.

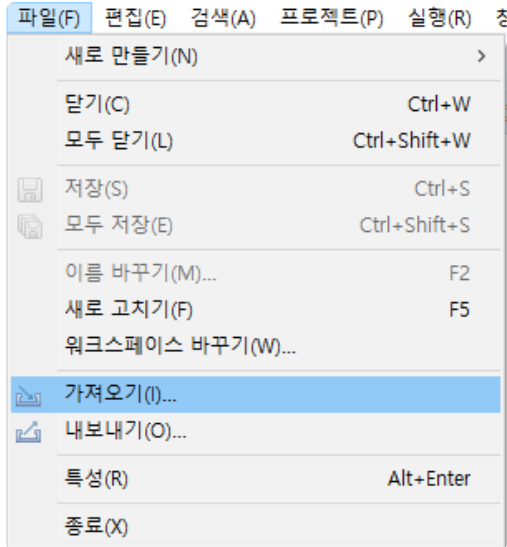
[복제]를 클릭하면 선택한 워크스페이스를 다른 디렉터리로 복제합니다.

[목록 초기화]를 클릭하여 워크스페이스 목록을 삭제할 수 있습니다.

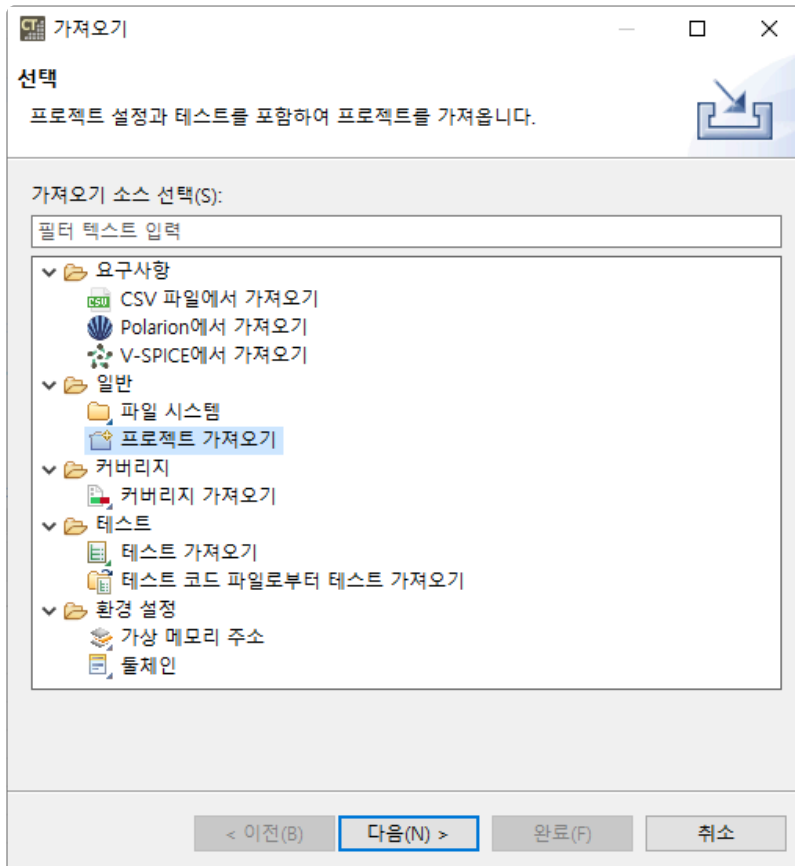
## 19.3. 가져오기

CT 2023.12는 이전 버전 또는 다른 PC에서 작업한 프로젝트 및 데이터를 가져오는 기능을 제공합니다.

1. 메인 메뉴에서 [파일] > [가져오기]를 선택하거나 대시보드에서 [가져오기]를 선택하면 가져오기 마법사가 열립니다.



2. 가져오려는 항목을 선택한 후 [다음] 버튼을 클릭합니다.



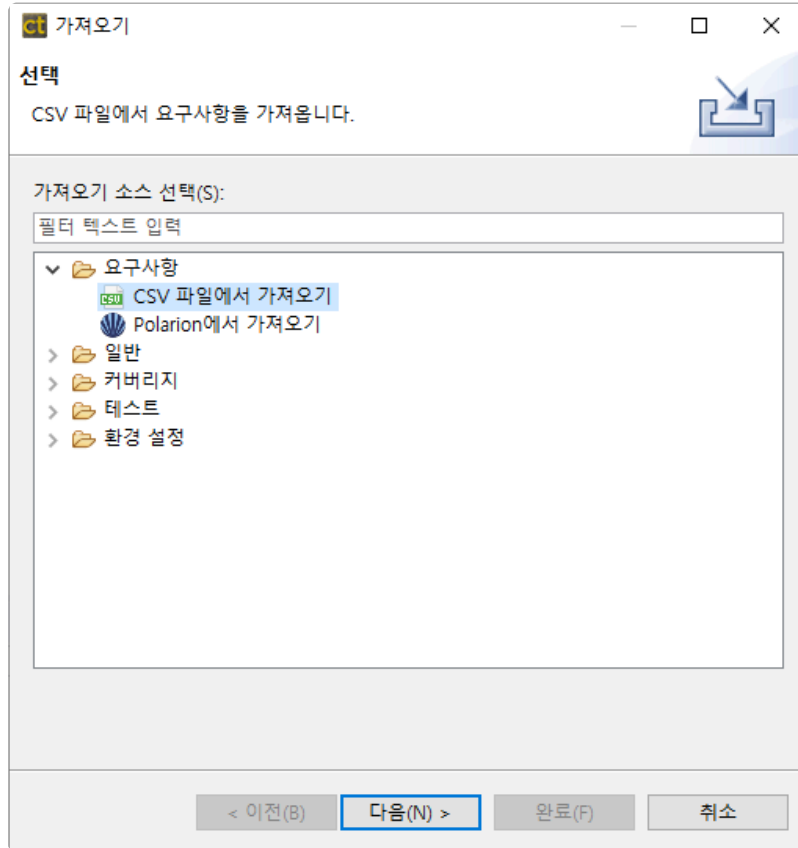
상위 항목	하위 항목	설명
-------	-------	----

요구사항	<a href="#">CSV 파일에서 가져오기</a> <a href="#">Polarion에서 가져오기</a> <a href="#">V-SPICE에서 가져오기</a>	CSV 파일이나 요구사항 관리 도구에서 요구사항을 가져옵니다.
일반	<a href="#">파일 시스템</a> <a href="#">프로젝트 가져오기</a>	파일이나 프로젝트를 가져옵니다.
커버리지	<a href="#">커버리지 가져오기</a>	csd 형식의 커버리지 정보를 가져옵니다.
테스트	<a href="#">테스트 가져오기</a> <a href="#">테스트 코드 파일로부터 테스트 가져오기</a>	내보낸 테스트 또는 테스트 코드에서 테스트를 가져옵니다.
환경설정	<a href="#">가상 메모리 주소</a> <a href="#">틀체인</a>	내보낸 틀체인 또는 가상 메모리 주소를 가져옵니다.

## 19.3.1. 요구사항 - CSV 파일에서 가져오기

CSV 파일로 작성된 요구사항 문서를 CT 2023.12로 가져올 수 있습니다.

1. 가져오기 마법사에서 [요구사항] -> [CSV 파일에서 가져오기]를 선택하고 [다음] 버튼을 클릭합니다.



2. 요구사항을 가져오기 위한 정보를 입력하고 [완료] 버튼을 클릭합니다.

- a. 파일 선택
  - 요구사항이 저장된 CSV 파일의 경로를 입력합니다.
- b. 설정
  - 값 구분자 : CSV 파일에서 각각의 값을 구분하는 구분자를 입력합니다. 기본 값은 ,(콤마)입니다.
  - 인코딩 : CSV 파일의 인코딩을 입력합니다. 기본 값은 US-ASCII 입니다.
  - 형식 : CSV 파일의 형식을 입력합니다. 기본 값은 EXCEL 입니다.
  - **CSV 파일에 헤더가 포함** : CSV 파일에 헤더가 포함되어있는 경우 체크합니다.
- c. 속성 할당
  - 키 : 요구사항의 키에 해당하는 열을 선택합니다.
  - 제목 : 요구사항의 제목에 해당하는 열을 선택합니다.
  - 설명 : 요구사항의 설명에 해당하는 열을 선택합니다.

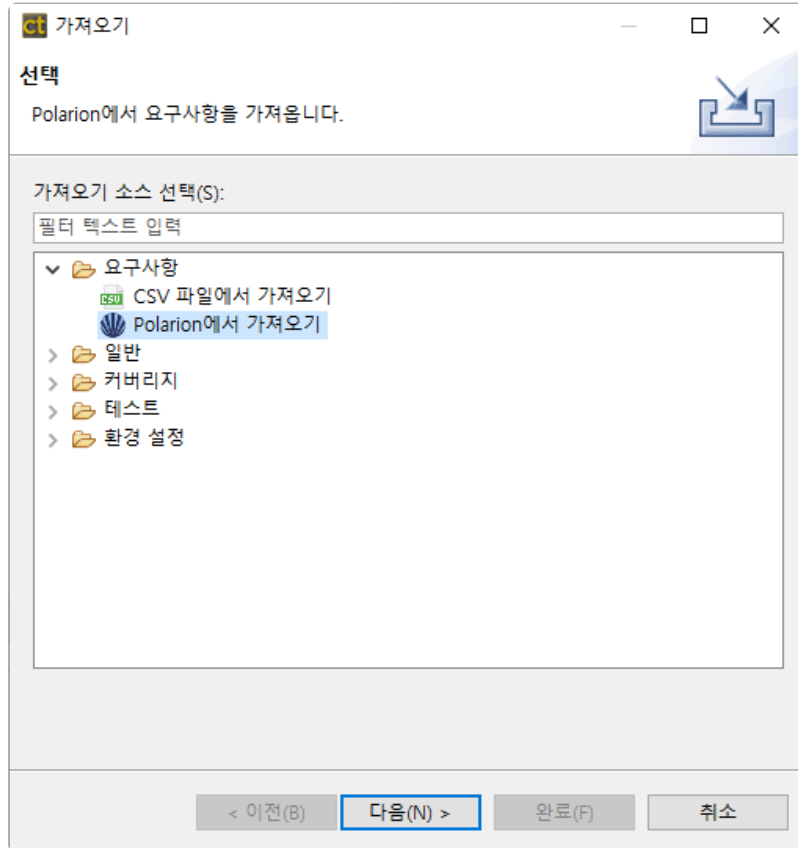
\* 이전에 가져오기를 할 때 입력한 정보가 자동으로 입력됩니다.




## 19.3.2. 요구사항 – Polarion에서 가져오기

Polarion에서 작성한 요구사항을 CT 2023.12로 가져올 수 있습니다.

1. 내 보내기 마법사에서 [Polarion에서 가져오기]를 선택하고 [다음] 버튼을 클릭합니다.



2. Polarion 서버에 접속하기 위한 정보를 입력한 후 [다음] 버튼을 클릭합니다.  버튼을 클릭하면 입력한 비밀번호를 확인할 수 있습니다.

**Polarion에서 요구사항 가져오기**

Polarion 서버에 접속하기 위한 정보를 입력하세요.

서버:

포트:

사용자:

암호:

< 이전(B)    **다음(N) >**    완료(F)    취소

- 서버와 포트에는 Polarion의 데모 서버 정보가 입력되어 있습니다.
- 로컬 서버를 이용할 경우 서버에 `http://ip_주소`를 입력하고 포트에 로컬 서버의 포트를 입력합니다.

3. Polarion에서 가져올 요구사항 문서의 프로젝트, 문서, 타입 정보를 입력하면 해당 요구사항 목록이 뜹니다. 요구사항을 확인하고 [완료] 버튼을 클릭하여 요구사항을 가져옵니다.

**Polarion에서 요구사항 가져오기**

요구사항을 가져올 대상에 대한 정보를 선택하세요.

Project:

Document:

Type:

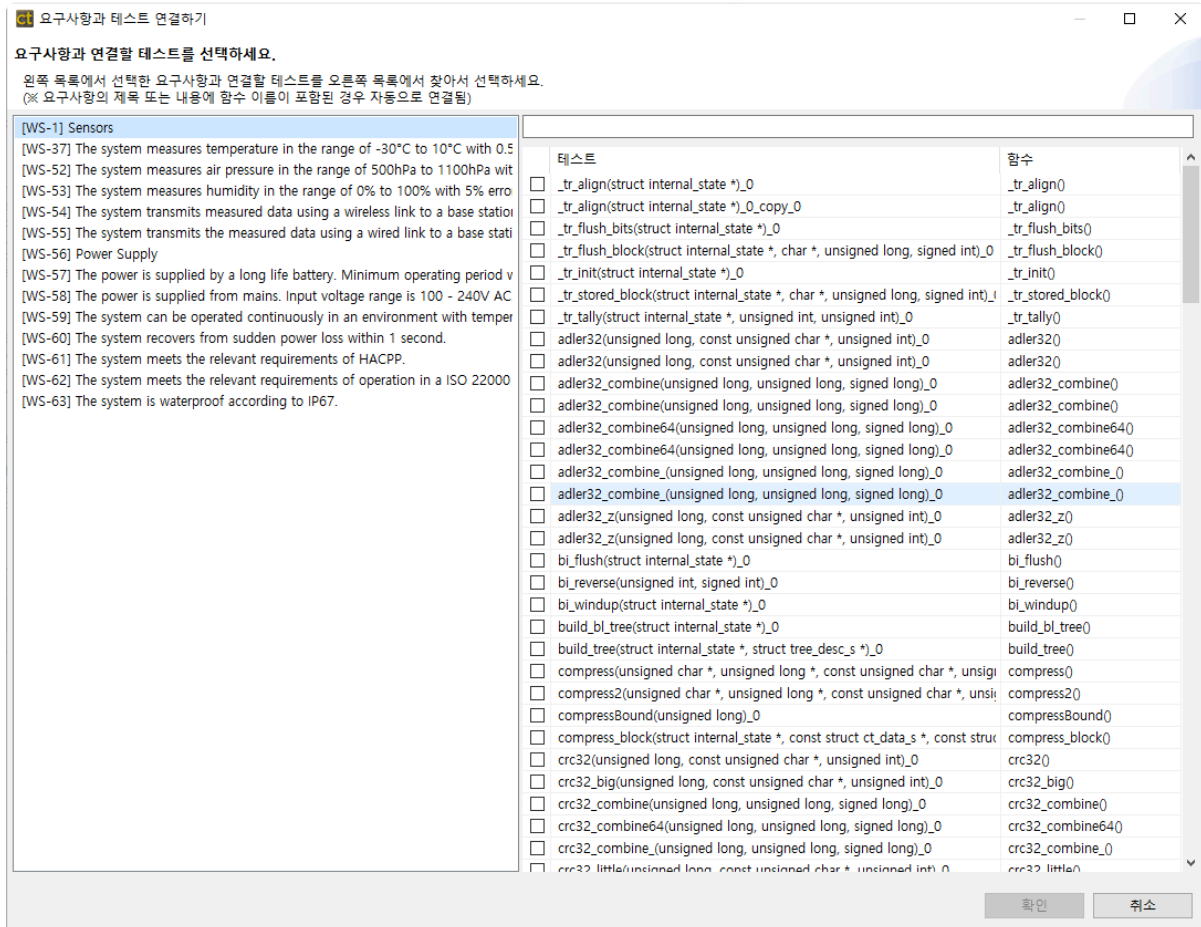
ID	Title	Description
WS-1	Sensors	Sensors
WS-56	Power Supply	Power Supply
WS-37	The system measures temp	The system measures temperature in the
WS-52	The system measures air pr	The system measures air pressure in the
WS-53	The system measures humi	The system measures humidity in the ra
WS-54	The system transmits measi	The system transmits measured data usi
WS-55	The system transmits the r	The system transmits the measured data
WS-57	The power is supplied by a	The power is supplied by a long life bat
WS-58	The power is supplied from	The power is supplied from mains. Input
WS-59	The system can be operater	The system can be operated continuous

<    >

< 이전(B)    다음(N) >    **완료(F)**    취소

- 문서를 선택하지 않으면 특정 문서에 포함되지 않은 요구사항도 가져올 수 있습니다.

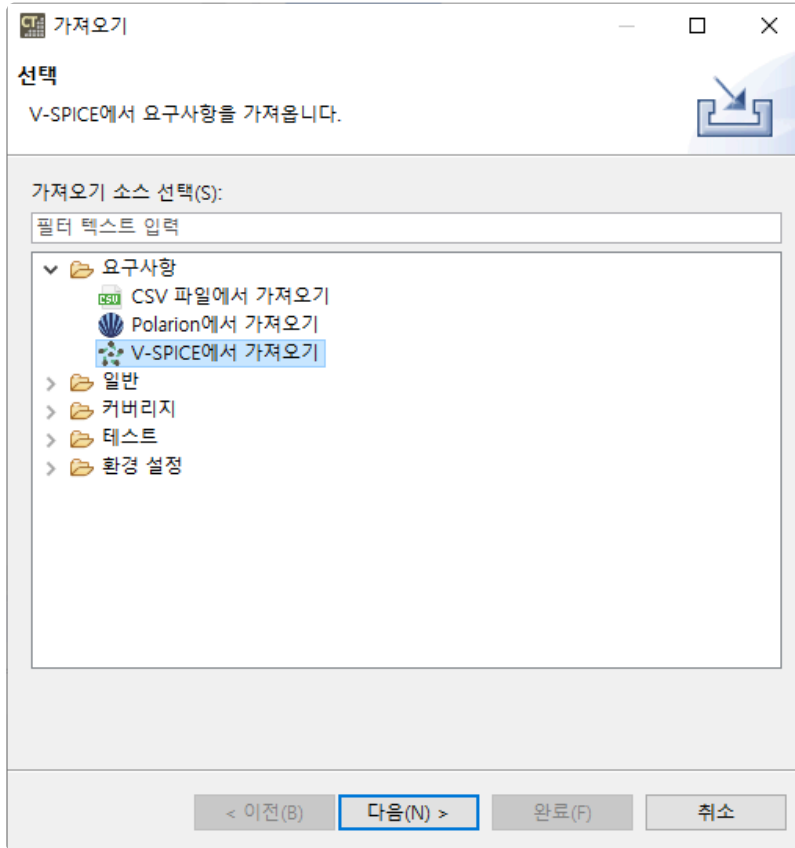
4. 가져온 요구사항과 테스트를 연결할 수 있는 [요구사항과 테스트 연결하기] 창이 뜹니다. 요구사항의 제목이나 설명에 테스트와 연관된 함수 이름이 포함되어 있으면 자동으로 연결합니다. 요구사항과 테스트를 연결한 후 [확인] 버튼을 클릭하여 요구사항과 테스트를 연결합니다. 요구사항과 테스트를 연결하지 않고 [요구사항과 테스트 연결하기]를 종료하려면 [취소] 버튼을 클릭합니다.




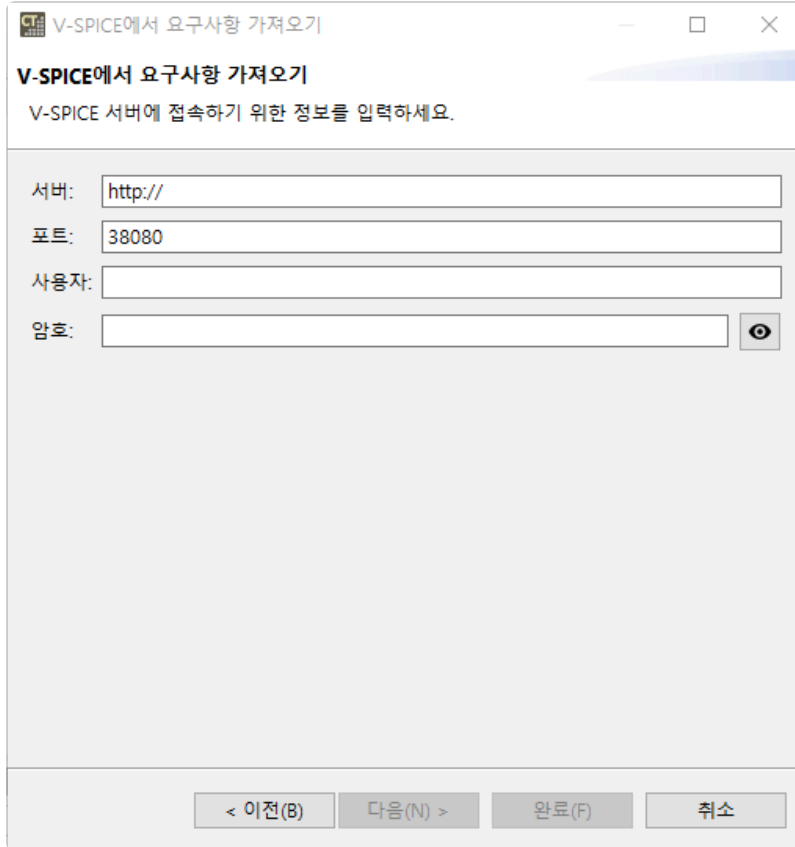
## 19.3.3. 요구사항 – V-SPICE에서 가져오기

V-SPICE에서 작성한 요구사항을 CT 2023.12로 가져올 수 있습니다.

1. 내보내기 마법사에서 [V-SPICE에서 가져오기]를 선택하고 [다음] 버튼을 클릭합니다.



2. V-SPICE 서버에 접속하기 위한 정보를 입력한 후 [다음] 버튼을 클릭합니다.  버튼을 클릭하면 입력한 비밀번호를 확인할 수 있습니다.




**V-SPICE에서 요구사항 가져오기**  
V-SPICE 서버에 접속하기 위한 정보를 입력하세요.

서버:

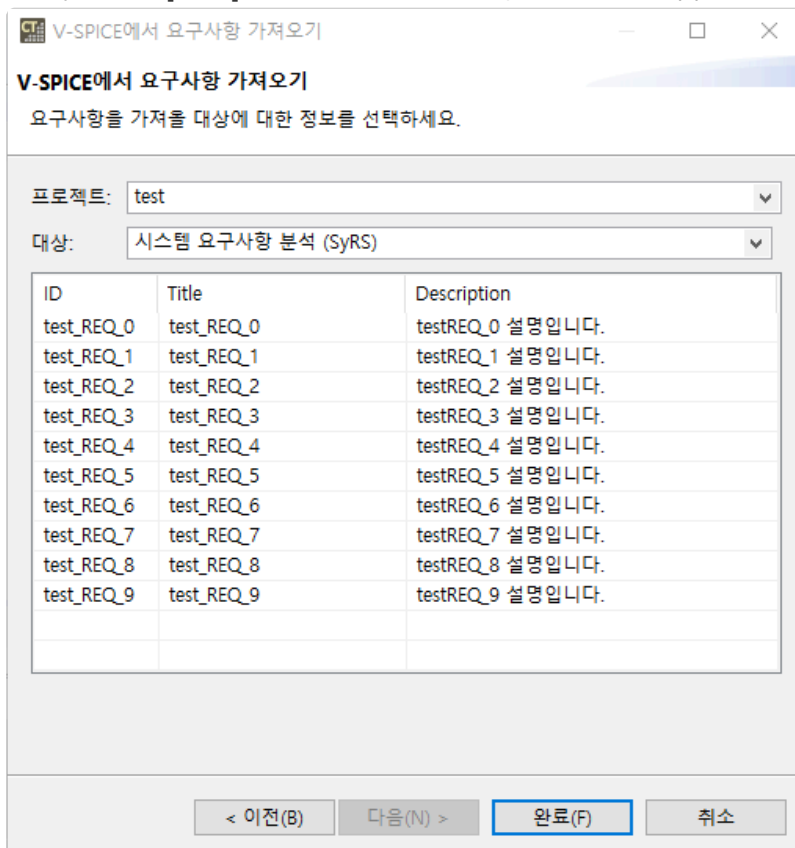
포트:

사용자:

암호:  

< 이전(B)    다음(N) >    완료(F)    취소

3. V-SPICE에서 가져올 요구사항 문서의 프로젝트를 선택하면 해당 요구사항 목록이 뜹니다. 요구사항을 확인하고 [완료] 버튼을 클릭하여 요구사항을 가져옵니다.



**V-SPICE에서 요구사항 가져오기**  
요구사항을 가져올 대상에 대한 정보를 선택하세요.

프로젝트:

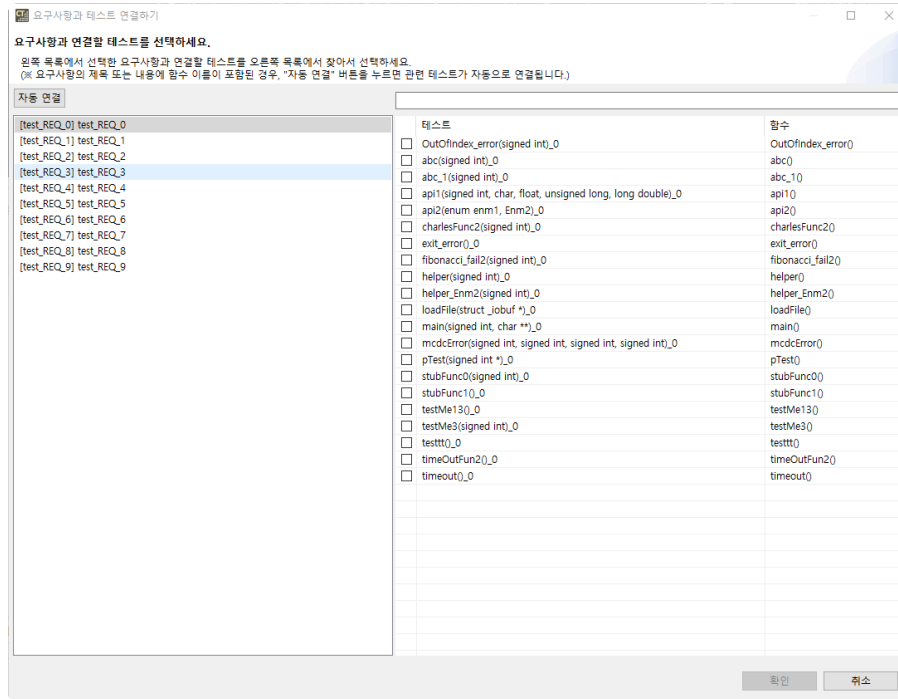
대상:

ID	Title	Description
test_REQ_0	test_REQ_0	testREQ_0 설명입니다.
test_REQ_1	test_REQ_1	testREQ_1 설명입니다.
test_REQ_2	test_REQ_2	testREQ_2 설명입니다.
test_REQ_3	test_REQ_3	testREQ_3 설명입니다.
test_REQ_4	test_REQ_4	testREQ_4 설명입니다.
test_REQ_5	test_REQ_5	testREQ_5 설명입니다.
test_REQ_6	test_REQ_6	testREQ_6 설명입니다.
test_REQ_7	test_REQ_7	testREQ_7 설명입니다.
test_REQ_8	test_REQ_8	testREQ_8 설명입니다.
test_REQ_9	test_REQ_9	testREQ_9 설명입니다.

< 이전(B)    다음(N) >    **완료(F)**    취소

4. 가져온 요구사항과 테스트를 연결할 수 있는 [요구사항과 테스트 연결하기] 창이 뜹니다. 요구사항의 제목이나 설명에 테스트와 연관된 함수 이름이 포함되어 있으면 자동으로 연결합니다. 요구사항과 테스트를 연결한 후 [확인] 버튼을 클릭하여 요구사항과 테스트를 연결합니다. 요구사항과 테스트를 연

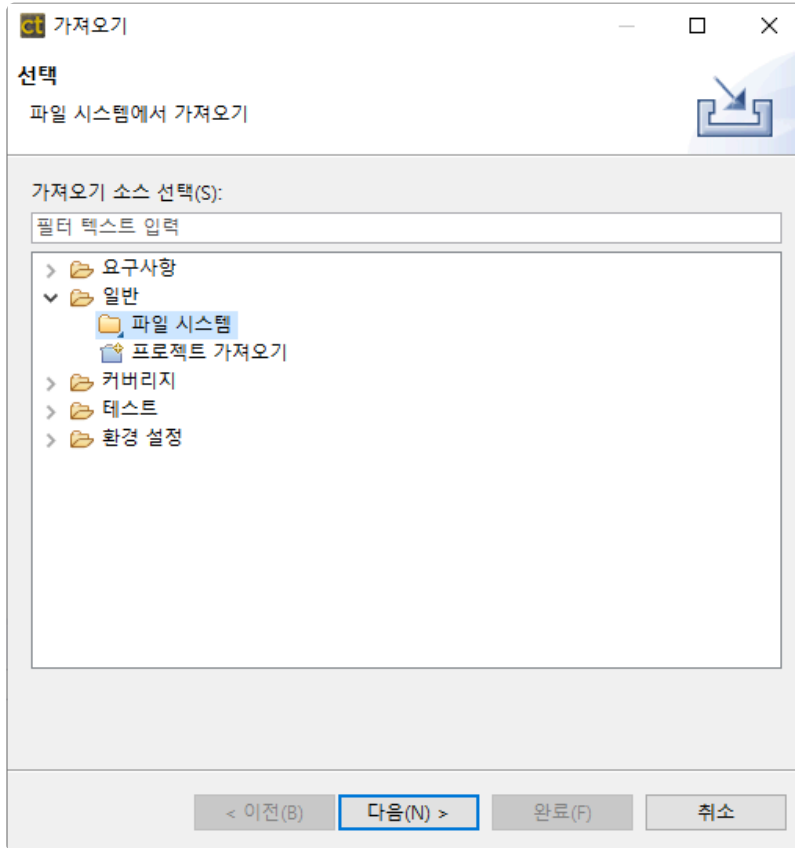
결하지 않고 [요구사항과 테스트 연결하기]를 종료하려면 [취소] 버튼을 클릭합니다.



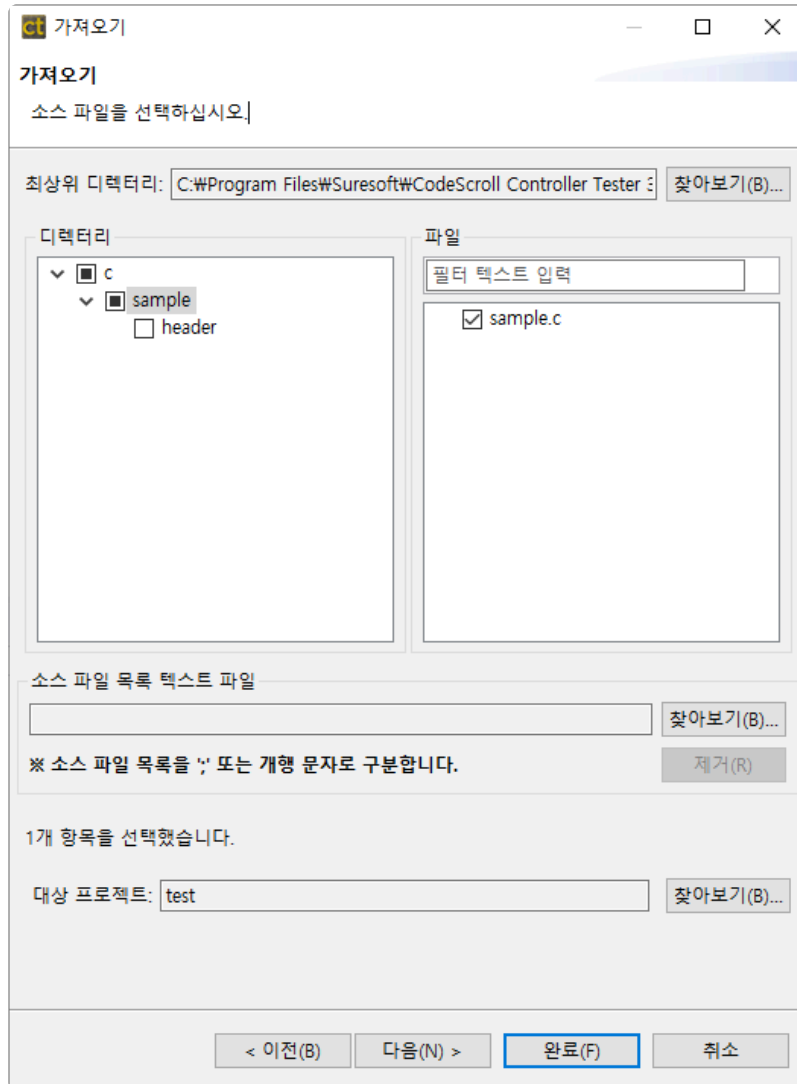
## 19.3.4. 일반 - 파일 시스템

파일 시스템 가져오기 기능을 이용하여, 프로젝트에 소스 파일을 추가할 수 있습니다.

1. 가져오기 마법사에서 [일반] -> [파일 시스템]을 선택한 후 [다음] 버튼을 클릭합니다.



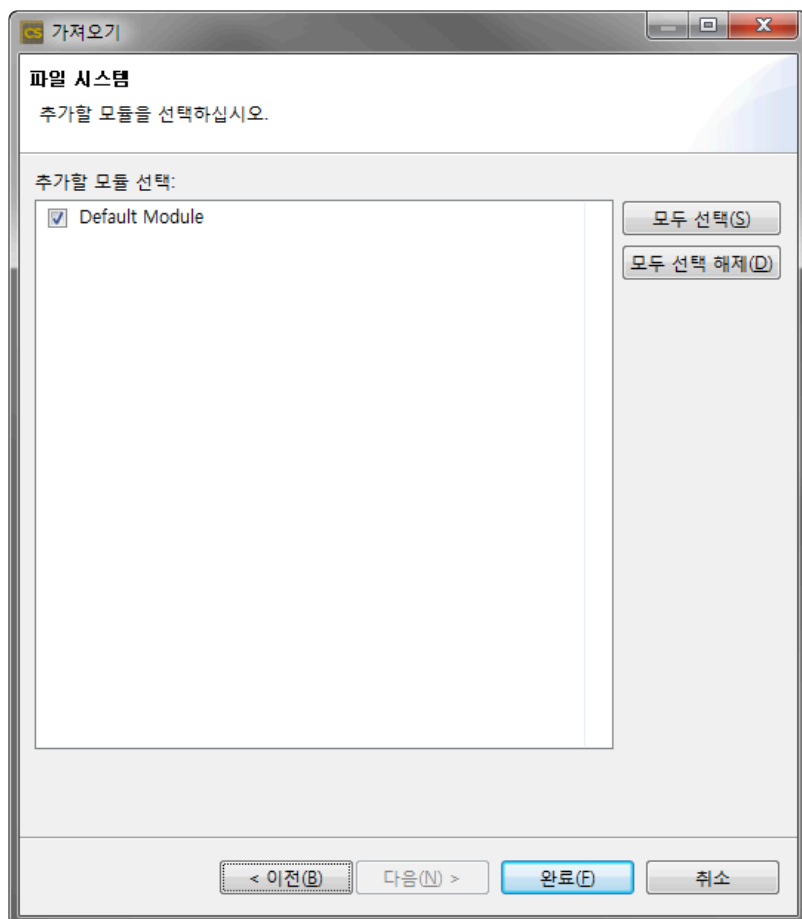
2. 가져올 파일의 정보를 입력한 후 [다음] 버튼을 클릭합니다.



항목	설명
최상위 디렉터리	가져올 소스 파일의 디렉터리를 선택합니다.
디렉터리, 파일	가져올 소스 파일을 선택합니다.
소스 파일 목록 텍스트 파일	소스 파일의 목록이 적힌 txt 파일을 선택합니다.(선택 사항).
대상 프로젝트	가져온 소스 파일을 추가할 프로젝트를 선택합니다.

3. 프로젝트에 포함된 모듈 목록에서 소스 파일을 추가할 모듈을 선택하고 [완료] 버튼을 클릭합니다.

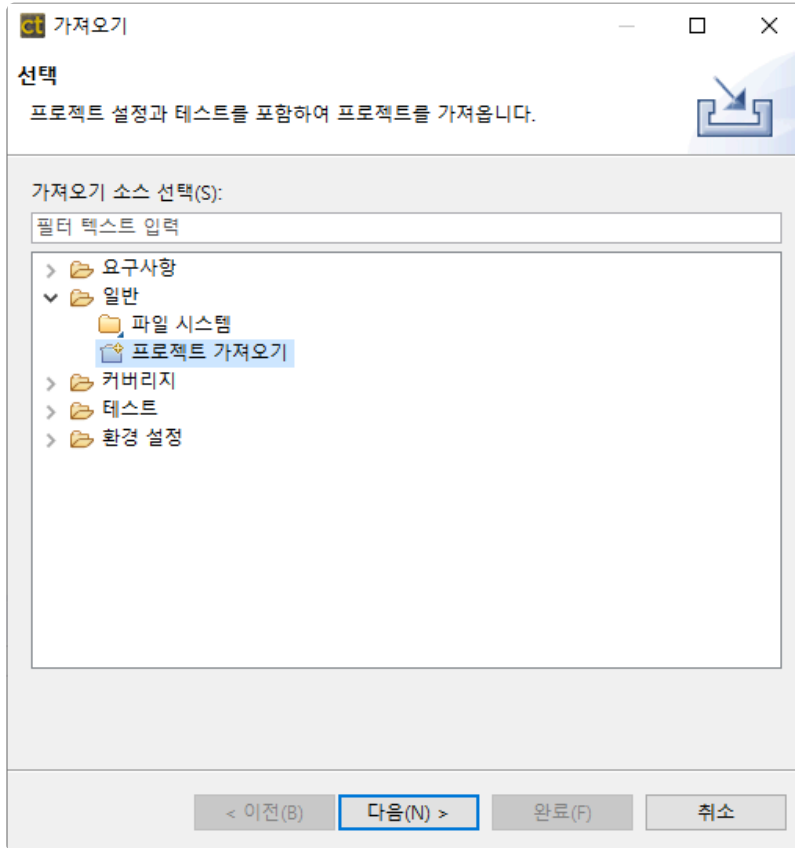




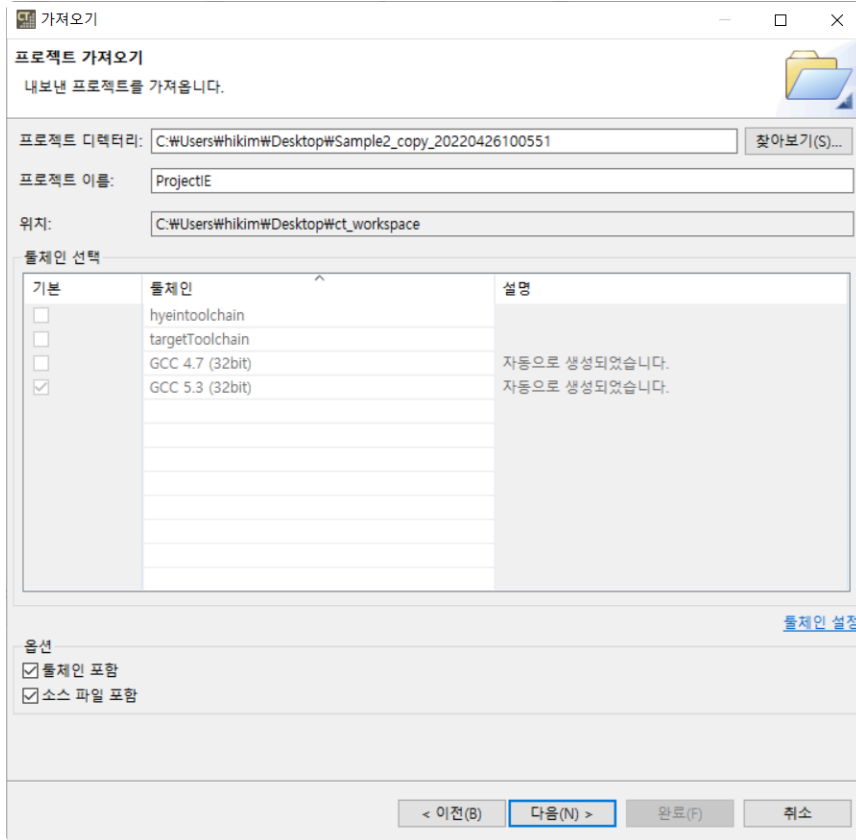
## 19.3.5. 일반 - 프로젝트 가져오기

프로젝트 가져오기 기능을 이용하여, 다른 PC에서 내보낸 프로젝트를 가져올 수 있습니다.

1. 가져오기 마법사에서 [일반] > [프로젝트 가져오기]를 선택한 후 [다음] 버튼을 클릭합니다.

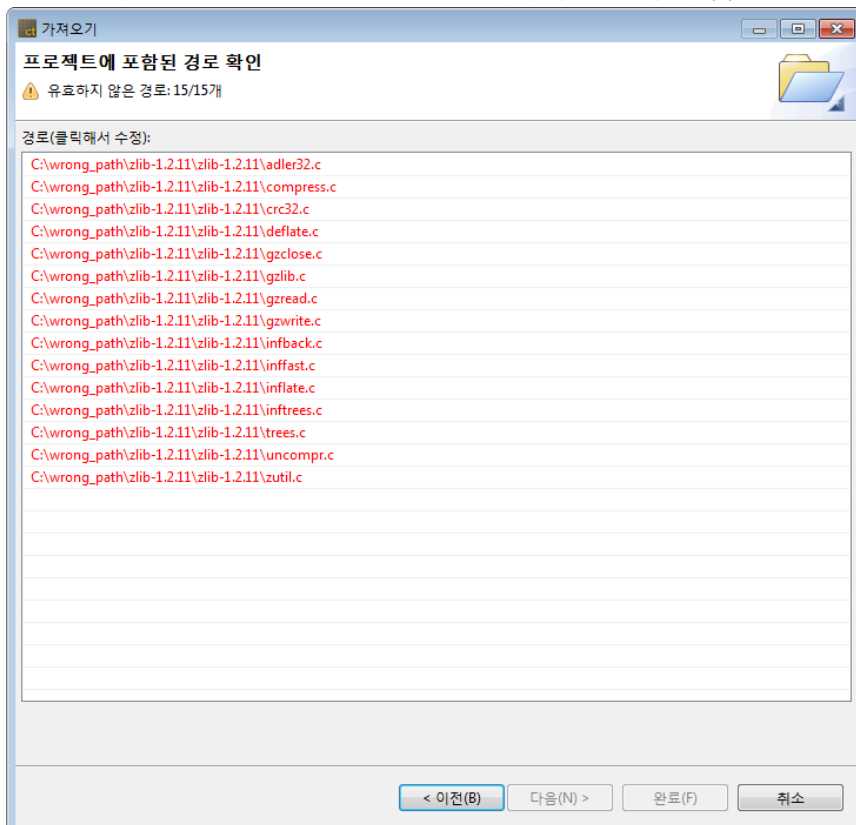


2. 내보낸 프로젝트 디렉터리를 입력하고 톨체인을 선택한 후 [다음] 버튼을 클릭합니다. 내보낸 프로젝트에 톨체인 및 소스 코드 정보가 있는 경우 디렉터리를 선택하면 톨체인 포함, 소스파일 포함이 자동으로 선택됩니다.

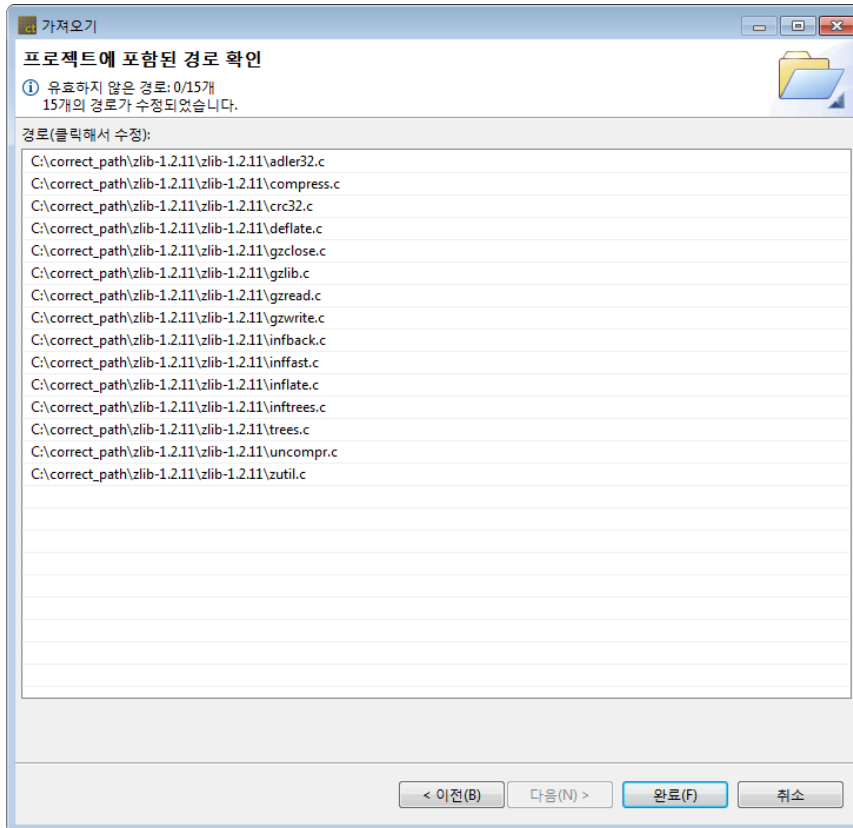


\* 적절한 툴체인이 없는 상태에서 툴체인 포함 옵션을 사용하지 않으면, 툴체인 내보내기/가져오기 기능을 사용하여 툴체인을 가져온 뒤에 프로젝트 가져오기를 수행해야 합니다.

3. 가져올 프로젝트에 포함된 경로 정보를 확인할 수 있습니다. 유효하지 않은 경로는 빨간색으로 표시되며, 마우스로 클릭하거나 F2키를 눌러서 수정할 수 있습니다.



4. 경로 하나를 수정하면 상대 경로로 찾을 수 있는 모든 경로들이 자동으로 수정됩니다. 마법사 위쪽에 수정된 경로의 수가 표시됩니다. 수정을 마치면 [완료] 버튼을 클릭합니다.



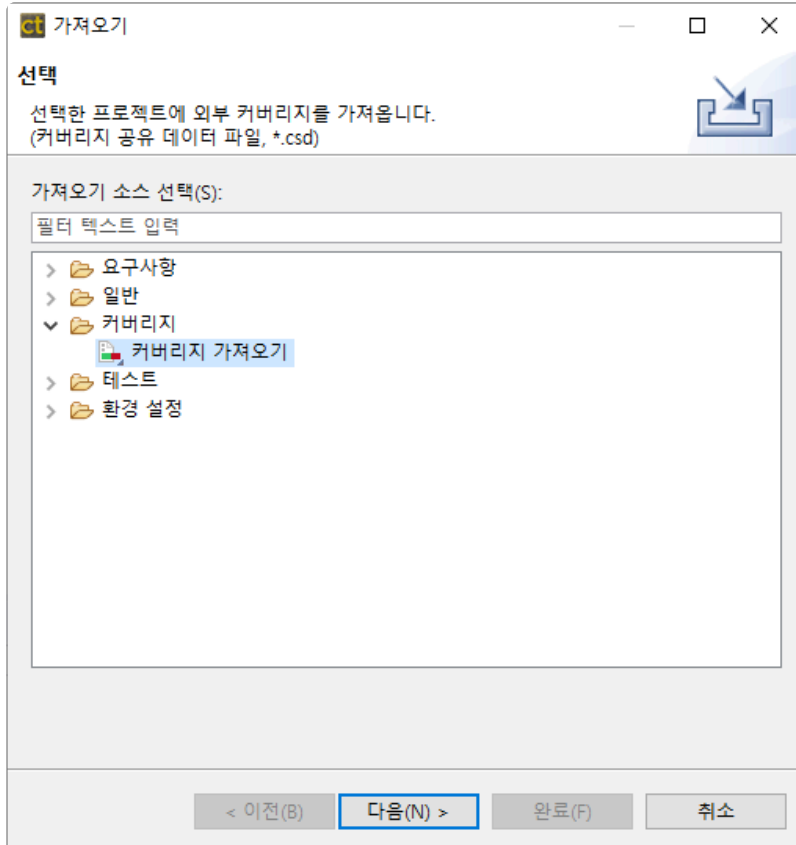
\* Windows 운영체제의 절대 경로 형식이 아닌 경우에는 경로가 유효한지 여부를 검사하지 않습니다.

## 19.3.6. 커버리지 - 커버리지 가져오기

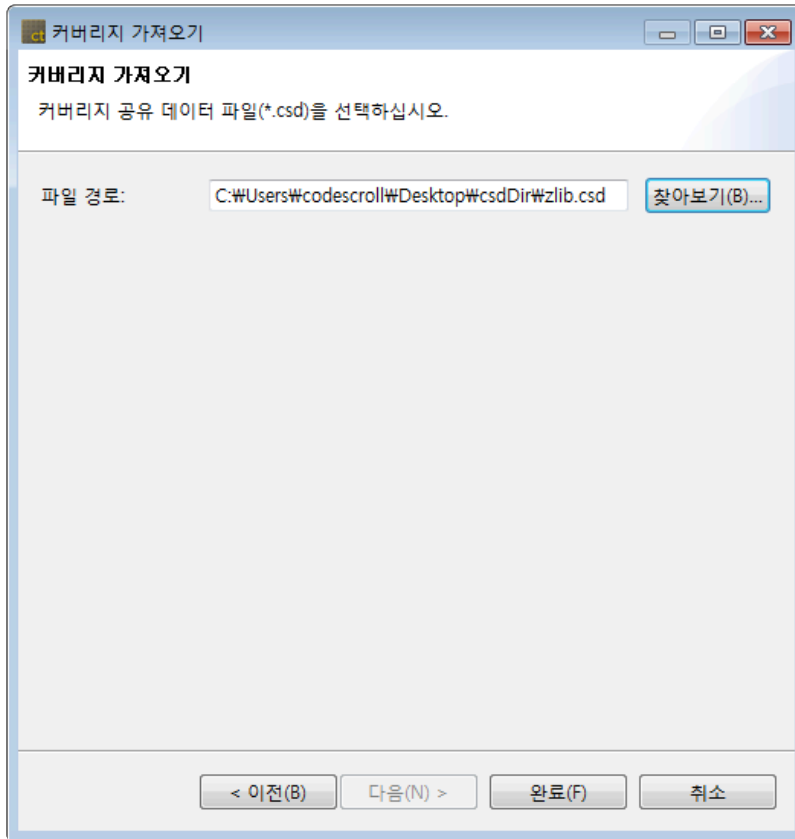


가져올 커버리지 파일의 버전, 커버리지 종류, 삼항연산자 설정이 CT 2023.12의 설정과 같을 경우 커버리지를 가져올 수 있습니다. 자세한 사항은 User Guides의 [커버리지 가져오기 가이드](#) 페이지를 참고하시기 바랍니다.

1. [가져오기] 마법사에서 [커버리지] > [커버리지 가져오기]를 선택하고 다음을 클릭합니다.



2. 가져올 커버리지 파일 경로를 입력 후 [완료] 버튼을 클릭합니다.



가져온 커버리지 정보는 [커버리지 뷰]의 툴바 메뉴에서 [전체 커버리지 보기(외부 커버리지 포함)]를 선택하면 볼 수 있습니다.

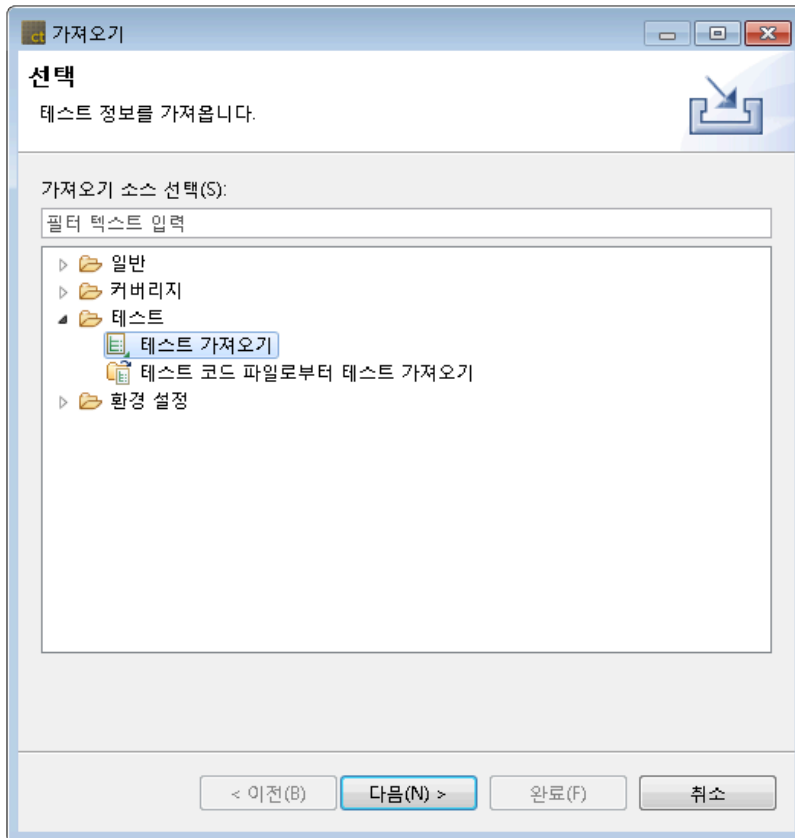
zlib 프로젝트의 호스트/타깃 커버리지 정보를 보여줍니다.

	대상 함수	구문	분기	MC/DC	함수 호출	함수
1	_tr_align(struct internal_state *)	71.42% (15/21)	75.00% (3/4)	50.00% (1/2)	100.00% (1/1)	Y
2	_tr_flush_bits(struct internal_state *)	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
3	_tr_flush_block(struct internal_state *, char *, unsigned lon...	57.14% (24/42)	68.75% (11/16)	50.00% (5/10)	80.00% (8/10)	Y
4	_tr_init(struct internal_state *)	100.00% (10/10)	N/A	N/A	100.00% (2/2)	Y
5	_tr_stored_block(struct internal_state *, char *, unsigned lo...	100.00% (11/11)	100.00% (2/2)	100.00% (1/1)	100.00% (1/1)	Y
6	_tr_tally(struct internal_state *, unsigned int, unsigned int)	22.22% (2/9)	0.00% (0/4)	0.00% (0/2)	N/A	Y
7	adler32(unsigned long, const unsigned char *, unsigned i...	99.02% (102/103)	91.66% (22/24)	83.33% (10/12)	N/A	Y
8	adler32_combine(unsigned long, unsigned long, signed l...	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
9	adler32_combine64(unsigned long, unsigned long, signe...	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
10	adler32_combine_(unsigned long, unsigned long, signed l...	100.00% (21/21)	90.00% (9/10)	80.00% (4/5)	N/A	Y
11	bi_flush(struct internal_state *)	100.00% (9/9)	100.00% (4/4)	100.00% (2/2)	N/A	Y
12	bi_reverse(unsigned int, signed int)	100.00% (5/5)	100.00% (2/2)	100.00% (1/1)	N/A	Y
13	bi_windup(struct internal_state *)	100.00% (7/7)	100.00% (4/4)	100.00% (2/2)	N/A	Y
14	build_bl_tree(struct internal_state *)	100.00% (11/11)	75.00% (3/4)	50.00% (1/2)	100.00% (3/3)	Y
15	build_tree(struct internal_state *, struct tree_desc_s *)	100.00% (42/42)	87.50% (14/16)	75.00% (6/8)	100.00% (5/5)	Y
16	compress(unsigned char *, unsigned long *, const unsig...	100.00% (1/1)	N/A	N/A	100.00% (1/1)	Y
17	compress2(unsigned char *, unsigned long *, const unsig...	95.23% (20/21)	87.50% (7/8)	75.00% (3/4)	100.00% (4/4)	Y
18	compressBound(unsigned long)	100.00% (1/1)	N/A	N/A	N/A	Y
19	compress_block(struct internal_state *, const struct st...	75.21% (30/70)	70.16% (7/14)	16.66% (3/13)	N/A	Y
합계		42.92% (1919/...	28.79% (692/2...	17.76% (254/1...	45.26% (177/3...	100.00% ...

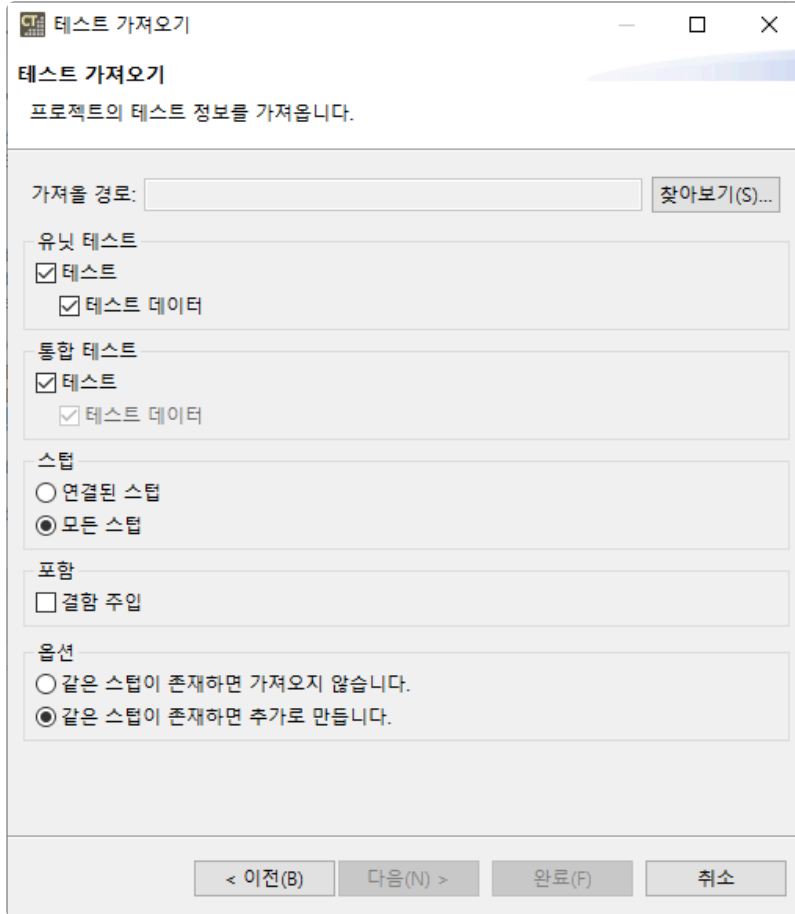
타깃 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

## 19.3.7. 테스트 - 테스트 가져오기

1. 가져오기 마법사에서 [테스트] > [테스트 가져오기]를 선택하고 [다음] 버튼을 클릭합니다.

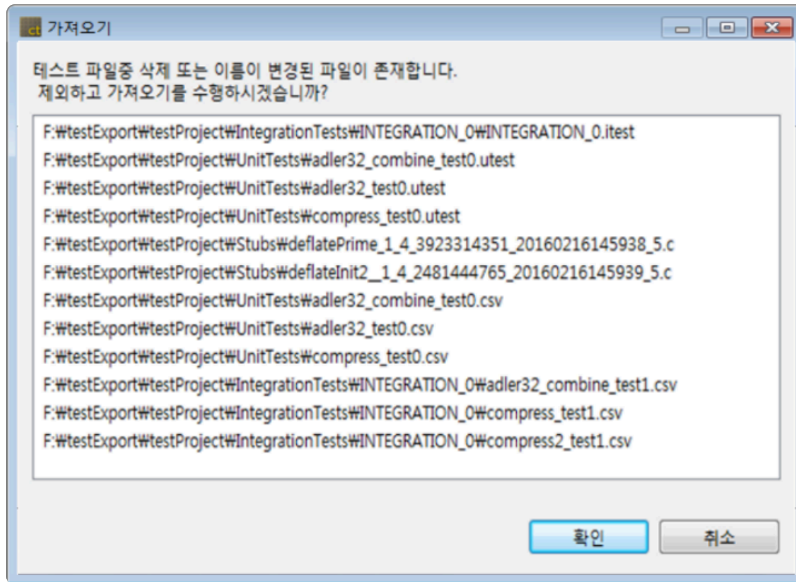


2. 가져올 테스트에 대한 정보를 입력 후 [완료] 버튼을 선택합니다.

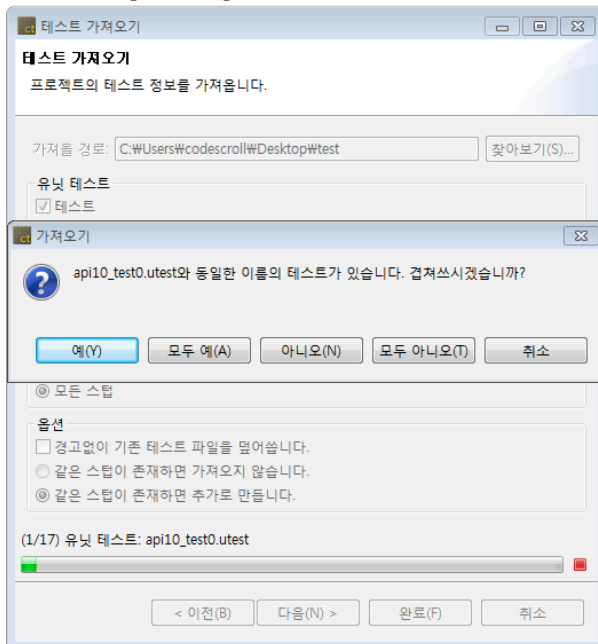


- a. [가져올 경로]에 테스트를 가져올 경로를 입력합니다.
  - b. [유닛 테스트] 그룹에서 [테스트]와 [테스트 데이터] 가져오기 여부를 각각 체크합니다.
  - c. [통합 테스트] 그룹에서 [테스트] 가져오기 여부를 체크합니다.
    - 통합 테스트를 가져올 때는 테스트 데이터와 세트로 가져옵니다.
  - d. [스텝] 그룹에서 [연결된 스텝]과 [모든 스텝] 중 가져올 스텝을 선택합니다.
    - 연결된 스텝: 가져올 테스트와 연결되어 있는 스텝만 가져옵니다.
    - 모든 스텝: 가져올 경로에 있는 모든 스텝을 가져옵니다.
  - e. [포함] 그룹에서 결함 주입을 포함하여 가져올지 선택합니다.
  - f. [옵션] 그룹에서 기존 스텝이 존재하는 경우 가져올 방식을 선택합니다.
3. 내보낸 테스트 정보 중에 가져오기를 수행할 때 파일 이름이 변경되었거나, 파일이 제거된 항목을 제외하고 가져올 지 선택할 수 있는 알림 창이 나타난다. 이 알림 창은 Controller Tester 2.6.14 이후의 버전에서 한 번에 내보낸 정보일 경우에만 나타나며, 다른 메뉴 및 이전 버전에서 내보낸 정보일 경우에는 알림이 나타나지 않습니다.





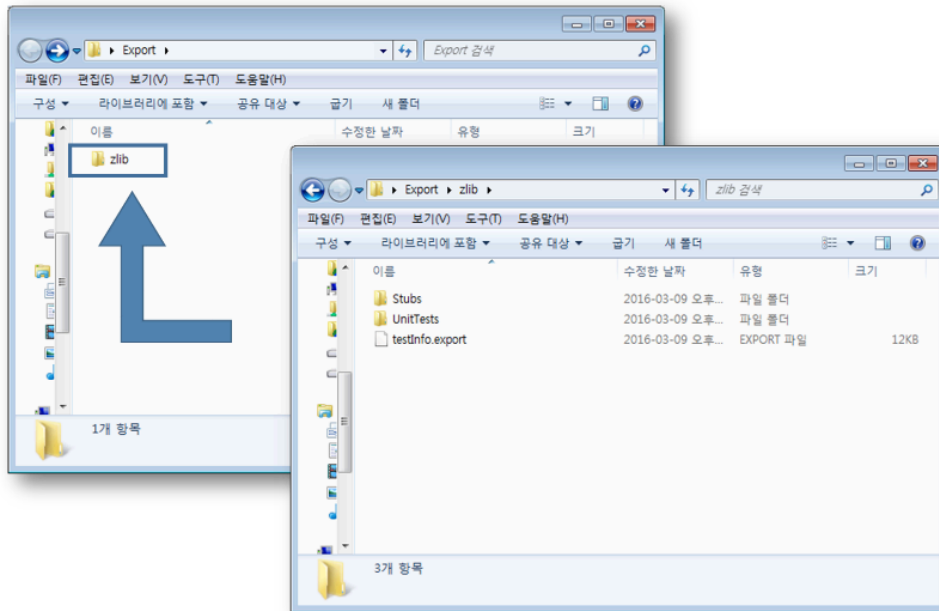
4. 프로젝트에 동일한 테스트가 존재하는 경우 테스트를 덮어 쓸지 여부를 확인합니다. [예]를 선택하면 덮어쓰고 [아니오]를 선택하면 가져오기를 수행하지 않습니다.



5. 가져오기 완료 후 선택한 프로젝트에 적용됩니다.

## Controller Tester 2.6.14 이하 버전에서 내보낸 정보 가져오기

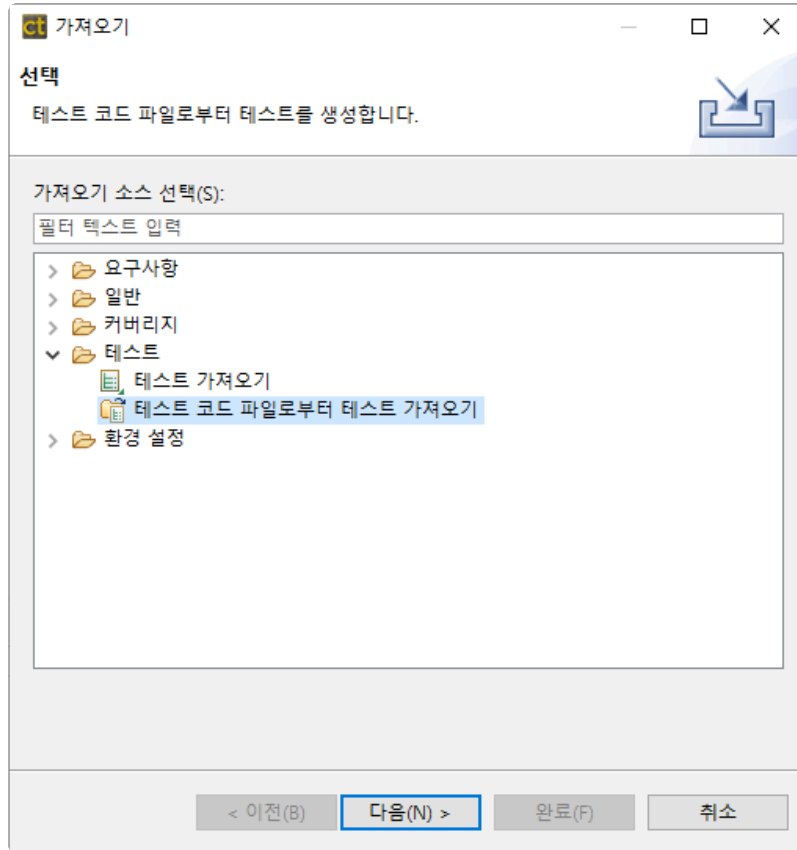
1. [가져올 경로]에 테스트를 가져올 경로를 입력합니다.



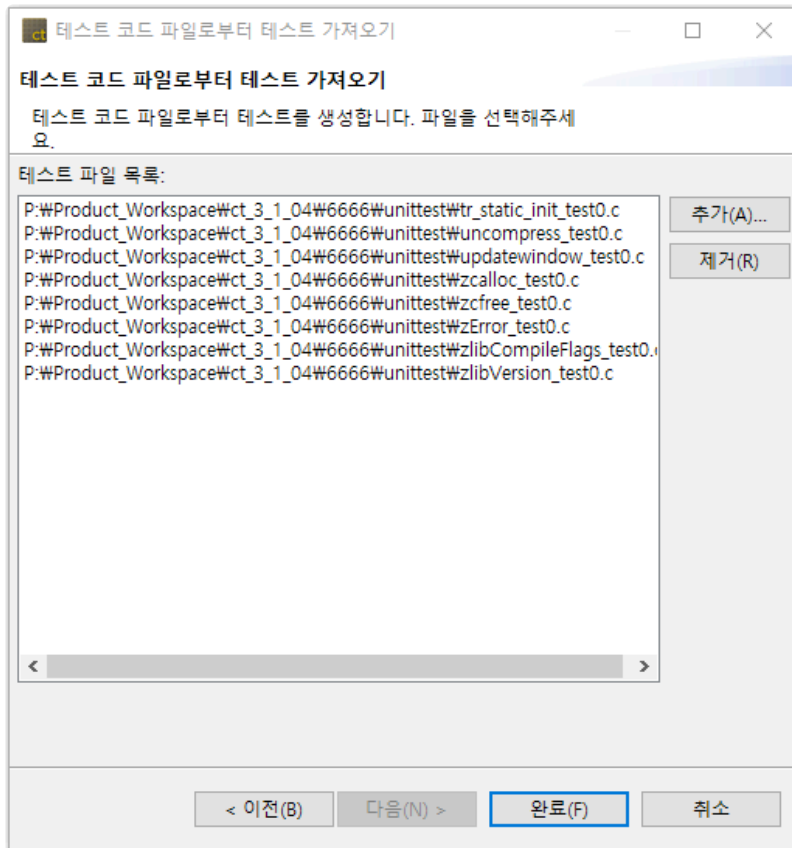
- 가져올 테스트 경로에 [testinfo.export]파일이 존재한다면 파일이 포함된 디렉터리를 선택합니다.
- Controller Tester 2.6.14 이하의 버전에서 각각 내보낸 정보를 가져오기 할 경우에는 내보낸 정보가 있는 상위의 디렉터리를 선택합니다.
2. [유닛 테스트] 그룹에서 [테스트]와 [테스트 데이터] 가져오기 여부를 각각 체크합니다.
  3. [통합 테스트] 그룹에서 [테스트] 가져오기 여부를 체크합니다.
    - 테스트와 데이터는 항상 같은 디렉터리에 있어야 하며, 같은 디렉터리에 없으면 데이터를 가져오지 않습니다.
  4. [스텝] 그룹에서 [연결된 스텝]과 [모든 스텝] 중 내보낼 스텝을 선택합니다.
    - Controller Tester 2.3버전에서 생성한 스텝 또는 Controller Tester 2.6으로 마이그레이션된 스텝의 경우 프로젝트에 동일한 스텝이 존재하는 경우에 가져오기를 수행할 수 없습니다.
    - Controller Tester 2.6에서 생성한 스텝의 경우 동일한 스텝이 존재하는 경우 선택적으로 가져오기를 수행하지 않을 수도 있고, 새로운 스텝으로 추가 생성 할 수 있습니다.
    - **Controller Tester 2.3**버전에는 테스트에 스텝을 연결하는 기능이 존재하지 않아, **Controller Tester 2.6** 이후의 버전으로 마이그레이션 수행 시 테스트에 직접 연결해야 합니다.
  5. [옵션] 그룹에서 기존 테스트 파일이 존재 시 덮어쓸지 여부, 같은 스텝 존재 시 처리 여부를 선택합니다.

## 19.3.8. 테스트 - 테스트 코드 파일로부터 테스트 가져오기

1. 가져오기 마법사에서 [테스트] -> [테스트 코드 파일로부터 테스트 가져오기]를 선택하고 [다음] 버튼을 클릭합니다.



2. [추가] 버튼을 클릭하여 가져올 테스트를 선택합니다.

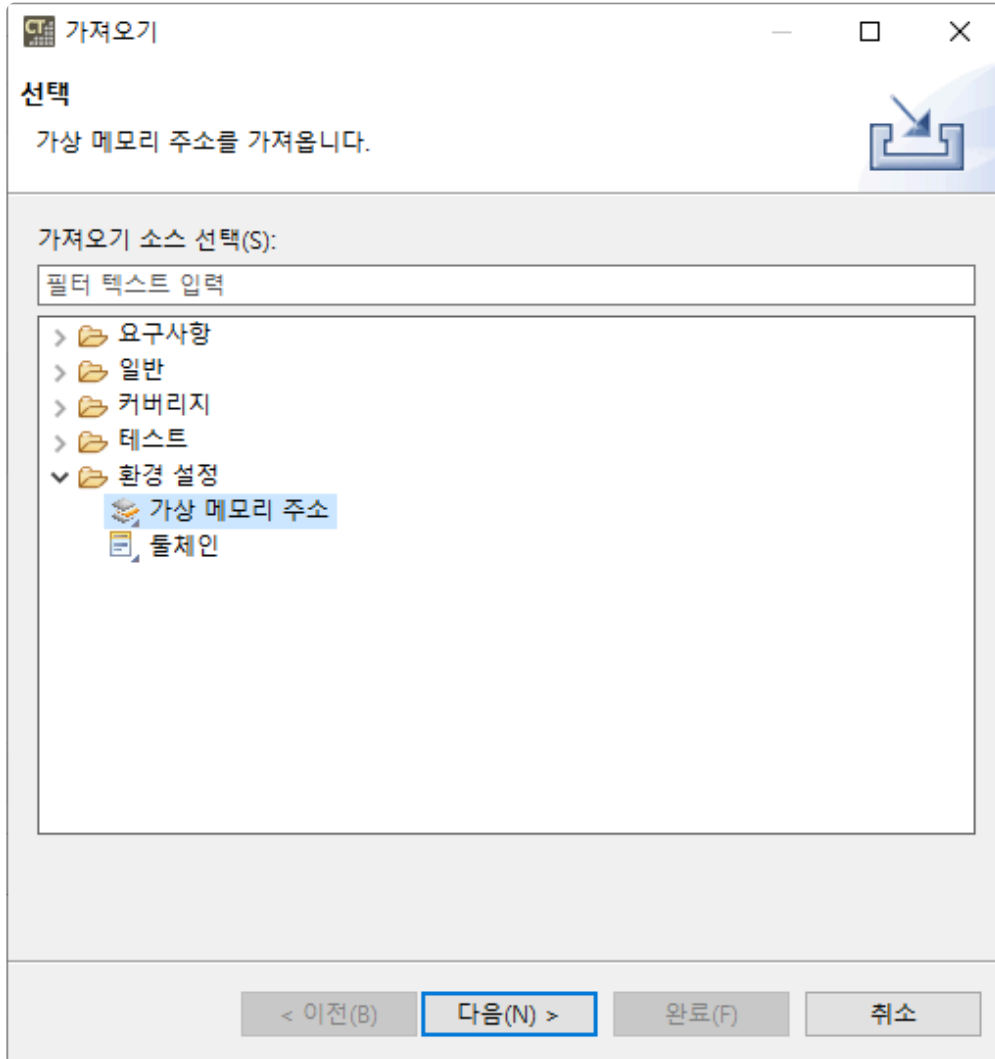


3. [완료] 버튼을 클릭하여 선택한 테스트를 선택한 프로젝트에 적용합니다.

## 19.3.9. 환경 설정 – 가상 메모리 주소

가져오기 기능을 이용하여 내보낸 가상 메모리 주소를 가져올 수 있습니다.

1. 가져오기 마법사에서 [환경 설정] > [가상 메모리 주소]를 선택한 후 [다음] 버튼을 클릭합니다.



2. 가져올 파일을(\*.csvm) 선택하면, 선택한 파일에 포함된 가상 메모리 주소 목록이 나타납니다. 가상 메모리 주소 목록에서 가져올 가상 메모리 주소를 선택합니다.

**가상 메모리 주소 가져오기**

가져올 파일과 가상 메모리 주소를 선택하십시오. 같은 이름의 가상 메모리 주소가 존재할 경우, 설정 이름 뒤에 '(1)'과 같이 숫자가 추가됩니다.

가져올 파일 선택:  [찾아보기\(S\)...](#)

가져온 가상 메모리 주소 목록:

	설정 이름	설명	상태
<input type="checkbox"/>	virtual (2)		이름 중복, 범위 중복
<input type="checkbox"/>	vm (6)		이름 중복
<input checked="" type="checkbox"/>	vm (7)		-

[전체 선택](#) [전체 선택 해제](#)

가상 메모리 주소 범위 목록:


시작	종료

< 이전(B)   다음(N) >   완료(F)   취소

- 상태
  - 이름 중복: 같은 이름의 가상 메모리 주소가 있을 경우, 가져오기 상태가 “이름 중복”이 됩니다. “이름 중복” 상태인 가상 메모리 주소를 가져올 경우 표시된 설정 이름으로 추가됩니다.
  - 범위 중복: 같은 범위의 가상 메모리 주소가 있을 경우, 가져오기 상태가 “범위 중복”이 됩니다. “범위 중복” 상태인 가상 메모리 주소를 가져올 경우 동일한 범위의 가상 메모리 주소가 추가됩니다.

3. [완료] 버튼을 클릭합니다. 링크를 클릭하여 가져온 가상 메모리 주소를 [환경 설정] 페이지에서 확인할 수 있습니다.

**가상 메모리 주소 가져오기**

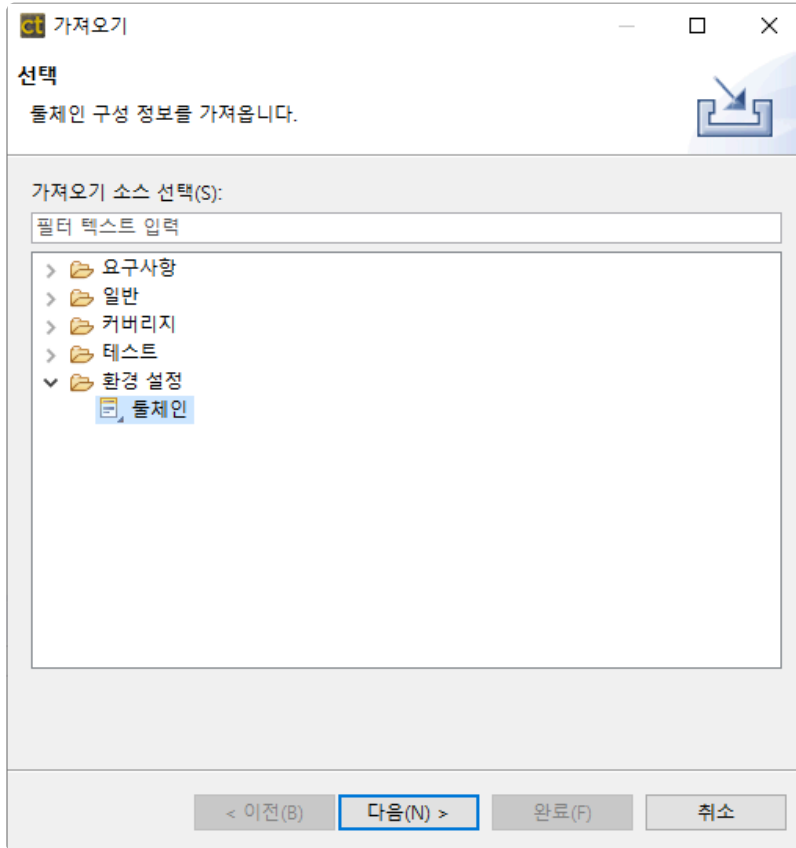
 가상 메모리 주소 가져오기가 성공하였습니다.  
가져온 가상 메모리 주소는 [가상 메모리 주소 환경 설정 페이지](#)에서 확인할 수 있습니다.

[확인](#)

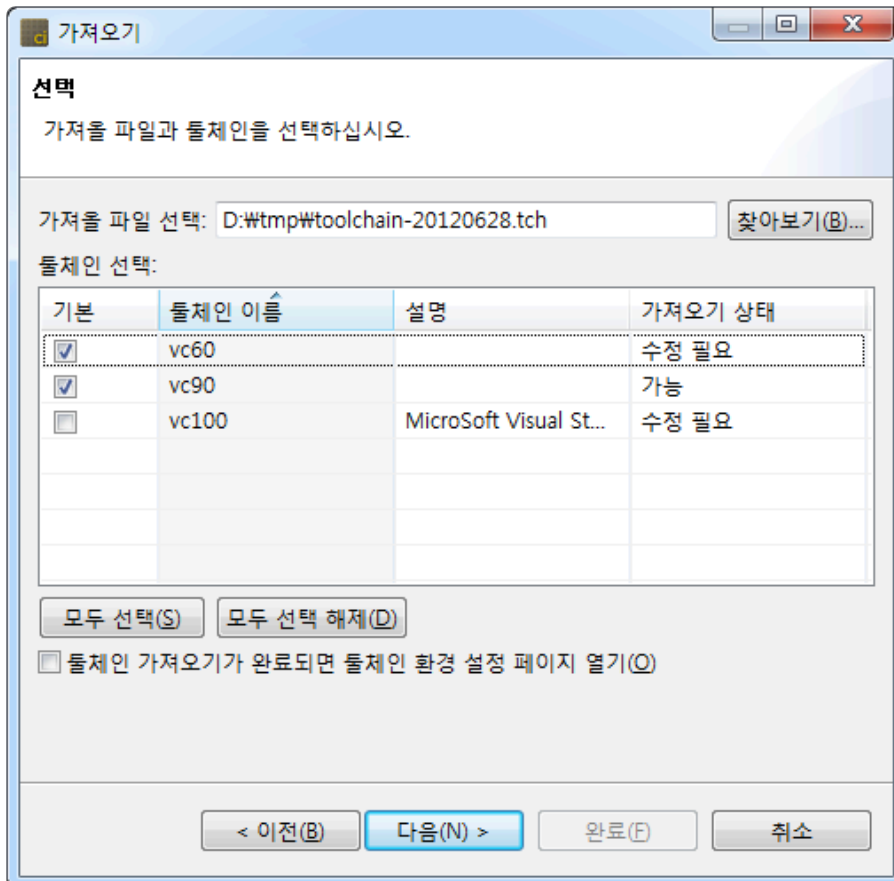
## 19.3.10. 환경 설정 - 툴체인

가져오기 기능을 이용하여 내보낸 툴체인 정보를 가져올 수 있습니다.

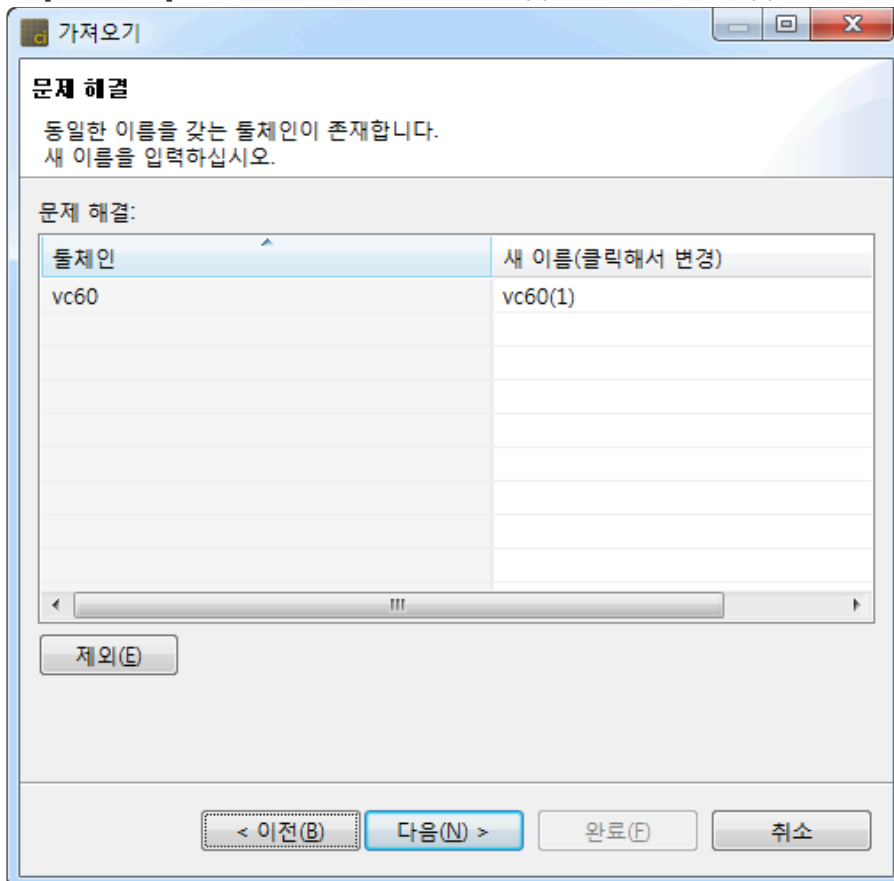
1. 가져오기 마법사에서 [환경 설정] > [툴체인]을 선택한 후 [다음] 버튼을 클릭합니다.



2. 가져올 파일을 (\*.tch) 선택하면, 선택한 파일에 포함된 툴체인 목록이 나타납니다. 툴체인 목록에서 가져올 툴체인을 선택합니다.



3. 같은 이름의 툴체인이 있을 경우, 가져오기 상태가 “수정 필요”가 됩니다. [다음] 버튼을 클릭하여 나오는 [문제해결] 창에서 가져올 툴체인의 이름을 변경할 수 있습니다.



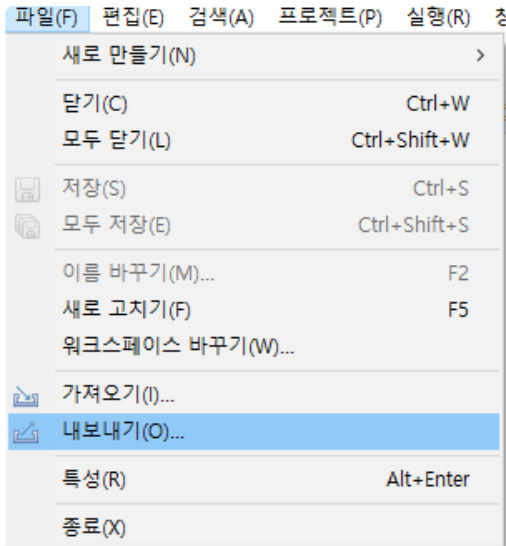
4. [완료] 버튼을 클릭합니다.



## 19.4. 내보내기

CT 2023.12의 프로젝트, 테스트, 튜체인 등을 내보낼 수 있습니다.

1. 메인 메뉴에서 [파일] > [내보내기]를 선택하거나 대시보드에서 [내보내기]를 선택하면 내보내기 마법사가 열립니다.



2. 내보내려는 항목을 선택한 후 [다음] 버튼을 클릭합니다.



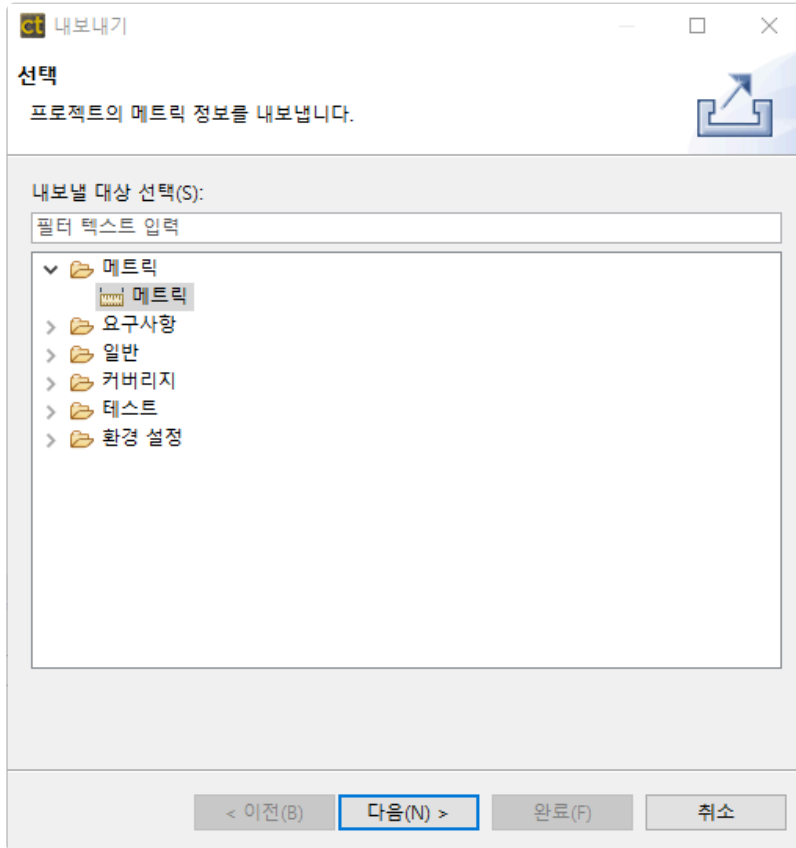
상위 항목	하위 항목	설명
-------	-------	----

메트릭	<a href="#">메트릭 내보내기</a>	프로젝트의 메트릭 측정 결과를 보고서 형식으로 내보냅니다.
요구사항	<a href="#">Polarion에 요구사항과 연결된 테스트 결과를 내보내기</a> <a href="#">V-SPICE에 테스트 결과를 내보내기</a> <a href="#">요구사항 테스트 커버리지 보고서</a>	요구사항과 연결된 테스트의 결과를 요구사항 관리 도구로 보내거나 xls 파일로 내보냅니다.
일반	<a href="#">프로젝트 내보내기</a>	CT 2023.12의 프로젝트를 내보냅니다.
커버리지	<a href="#">커버리지 내보내기</a>	커버리지 결과를 csd 파일로 내보냅니다.
테스트	<a href="#">테스트 내보내기</a> <a href="#">테스트 보고서 생성</a>	테스트를 내보내거나 테스트 보고서를 생성합니다.
환경 설정	<a href="#">가상 메모리 주소</a> <a href="#">툴체인</a>	가상 메모리 주소 또는 툴체인 정보를 내보냅니다.

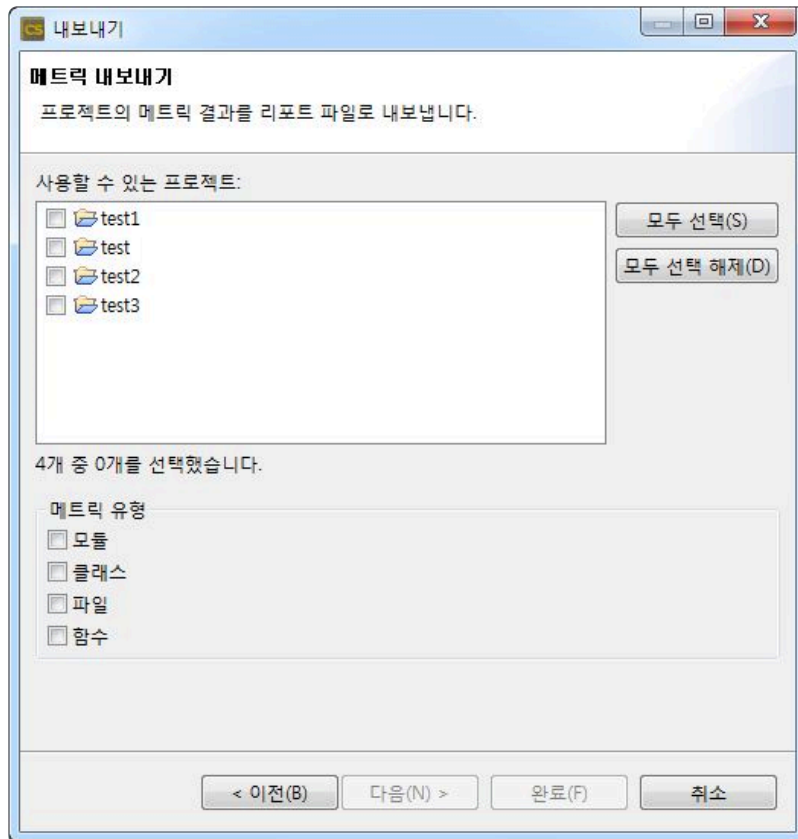
## 19.4.1. 메트릭 - 메트릭 내보내기

프로젝트의 메트릭 측정 결과를 보고서 형식으로 내보냅니다.

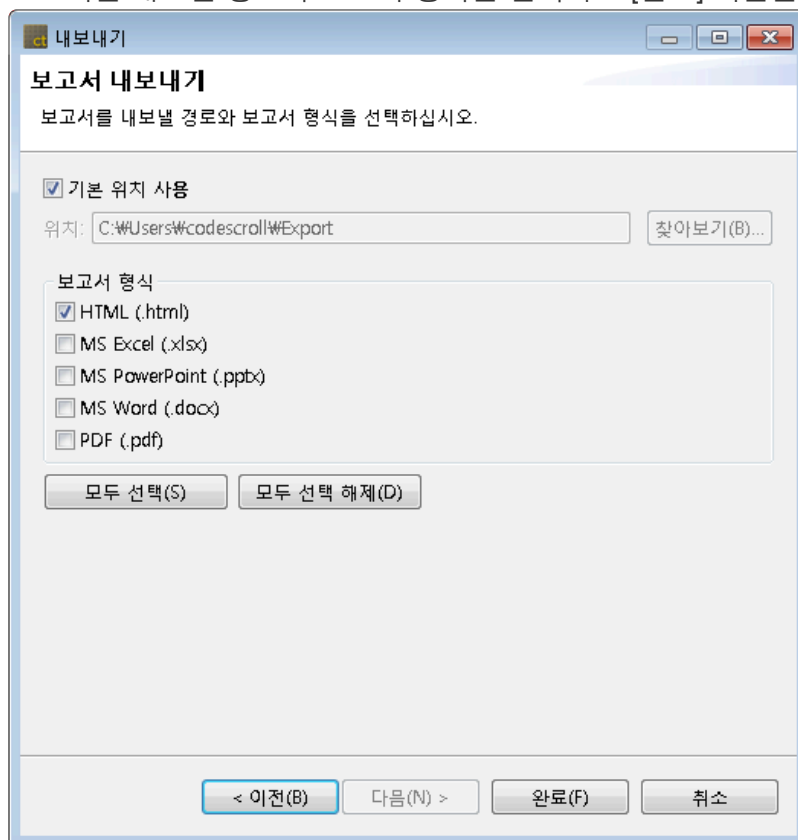
1. 내보내기 마법사에서 [메트릭] -> [메트릭]을 클릭한 후 [다음] 버튼을 클릭합니다.



2. 메트릭 정보를 내보낼 프로젝트와 메트릭 유형을 선택하고 [다음] 버튼을 클릭합니다.



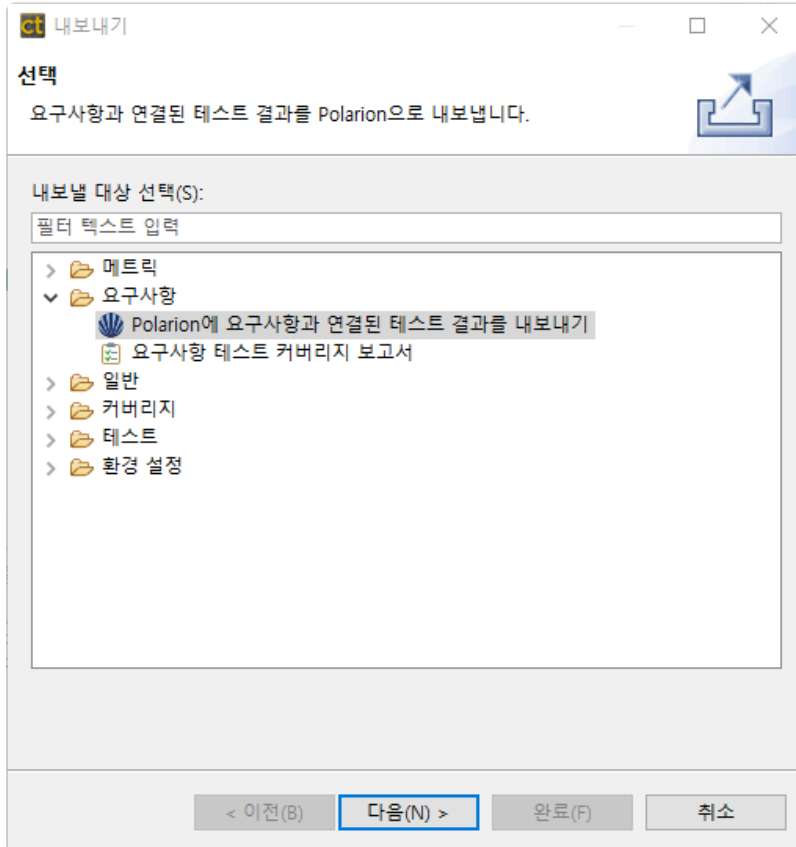
3. 보고서를 내보낼 경로와 보고서 형식을 선택하고 [완료] 버튼을 클릭합니다.




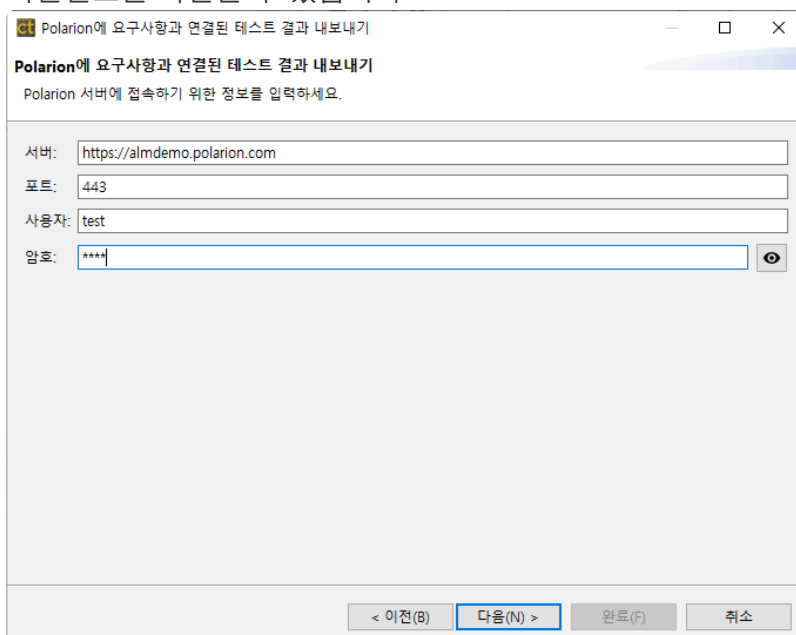
## 19.4.2. 요구사항 - Polarion에 요구사항과 연결된 테스트 결과를 내보내기

요구사항과 연결된 테스트 결과를 Polarion으로 내보낼 수 있습니다.

1. 내보내기 마법사에서 [Polarion에 요구사항과 연결된 테스트 결과를 내보내기]를 선택하고 [다음] 버튼을 클릭합니다.



2. Polarion 서버 정보와 계정 정보를 입력한 후 [다음] 버튼을 클릭합니다.  버튼을 클릭하면 입력한 비밀번호를 확인할 수 있습니다.



- 서버와 포트에는 Polarion의 데모 서버 정보가 입력되어 있습니다.
- 로컬 서버를 이용할 경우 서버에 http://ip\_주소를 입력하고 포트에 로컬 서버의 포트를 입력합니다.

3. 테스트 결과를 내보낼 대상의 정보를 입력하고 [완료] 버튼을 누릅니다.

**Polarion에 요구사항과 연결된 테스트 결과 내보내기**

내보낼 테스트 결과가 없습니다.

Project: TEST(TEST)

Document: Testing/Test Specification 2 Test Cases

Type: Test Case

ID	Title	성공	실패	오류	실행되지...	합계
TEST-152	User name must be longer than 3 cha	1	0	1	0	2
TEST-151	User name must contain at least one i	1	0	0	0	1
TEST-147	Password must be validated when the	1	0	0	0	1
TEST-157	User may not be told if the password	1	0	0	0	1
TEST-146	User name may not contain spaces	1	0	0	0	1

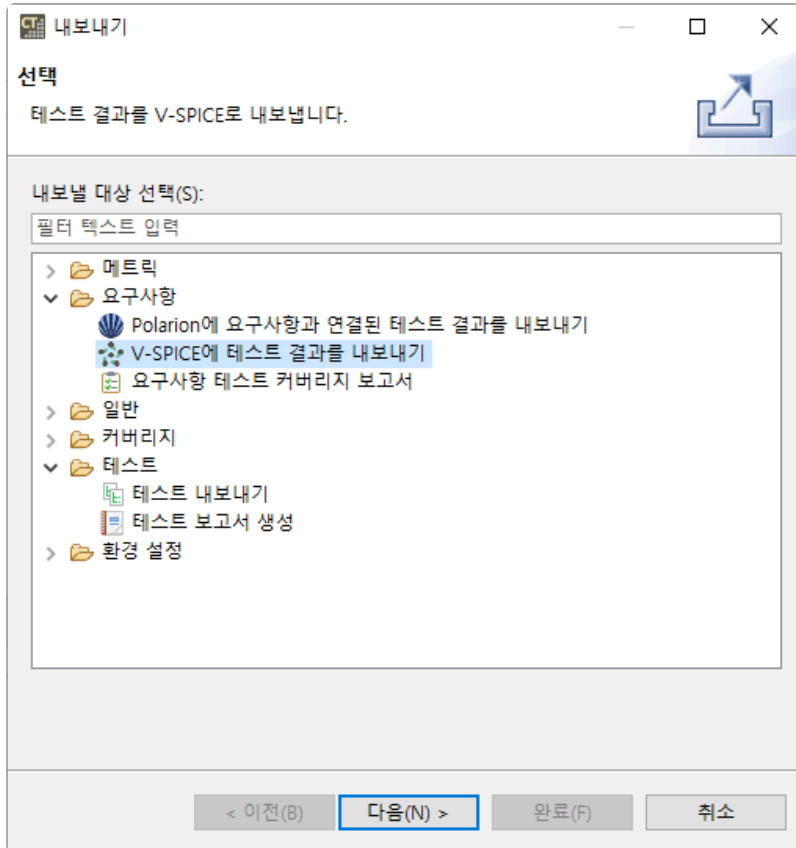
< 이전(B)   다음(N) >   **완료(F)**   취소


- **Project** : 테스트 결과를 내보낼 프로젝트를 선택합니다. 요구사항을 가져온 프로젝트가 기본으로 선택됩니다.
- **Document** : 테스트 결과를 내보낼 문서와 문서 내부의 목차를 선택합니다. 문서를 선택하지 않아도 테스트 결과를 내보낼 수 있습니다.
- **Type** : 테스트 결과를 내보낼 work item의 타입을 선택합니다.

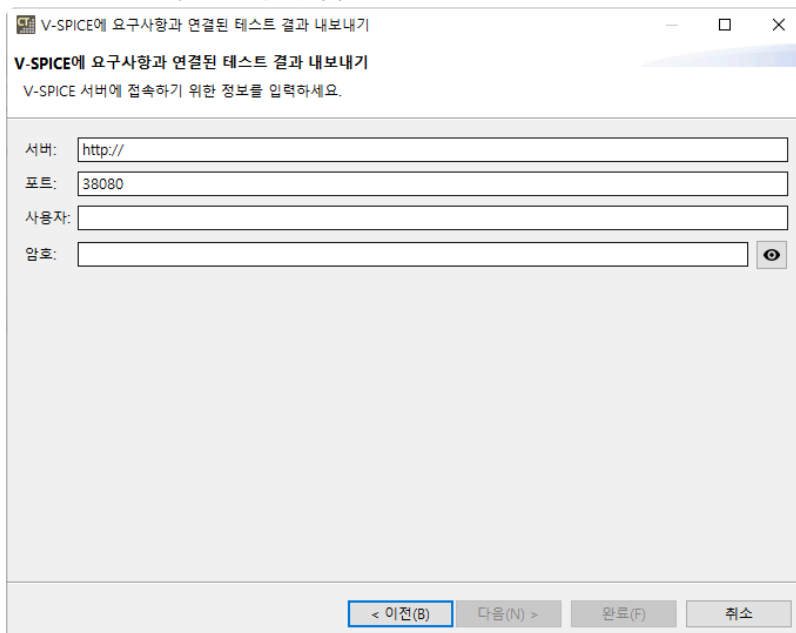
## 19.4.3. 요구사항 – V-SPICE에 테스트 결과를 내보내기

V-SPICE에 테스트 결과를 내보낼 수 있습니다.

1. 내보내기 마법사에서 [V-SPICE에 테스트 결과를 내보내기]를 선택하고 [다음] 버튼을 클릭합니다.



2. V-SPICE 서버 정보와 계정 정보를 입력한 후 [다음] 버튼을 클릭합니다.  버튼을 클릭하면 입력한 비밀번호를 확인할 수 있습니다.



3. 테스트 결과를 내보낼 대상의 정보를 입력하고 [완료] 버튼을 누릅니다.

**V-SPICE에 테스트 결과 내보내기**

테스트 결과를 내보낼 대상을 선택하세요.  
(※아래 테이블에는 선택한 대상의 요구사항과 연결된 테스트 결과만 표시됩니다.)

프로젝트: test ▼

ID	Title	성공	실패	오류	실행되지...	합계
test_REQ_1	test_REQ_1	2	0	1	0	3
test_REQ_3	test_REQ_3	2	0	1	0	3
test_REQ_5	test_REQ_5	2	0	2	0	4
test_REQ_7	test_REQ_7	2	0	0	0	2

< 이전(B)
다음(N) >
완료(F)
취소

- 프로젝트: 테스트 결과를 내보낼 프로젝트를 선택합니다. 요구사항을 가져온 프로젝트가 기본으로 선택됩니다.

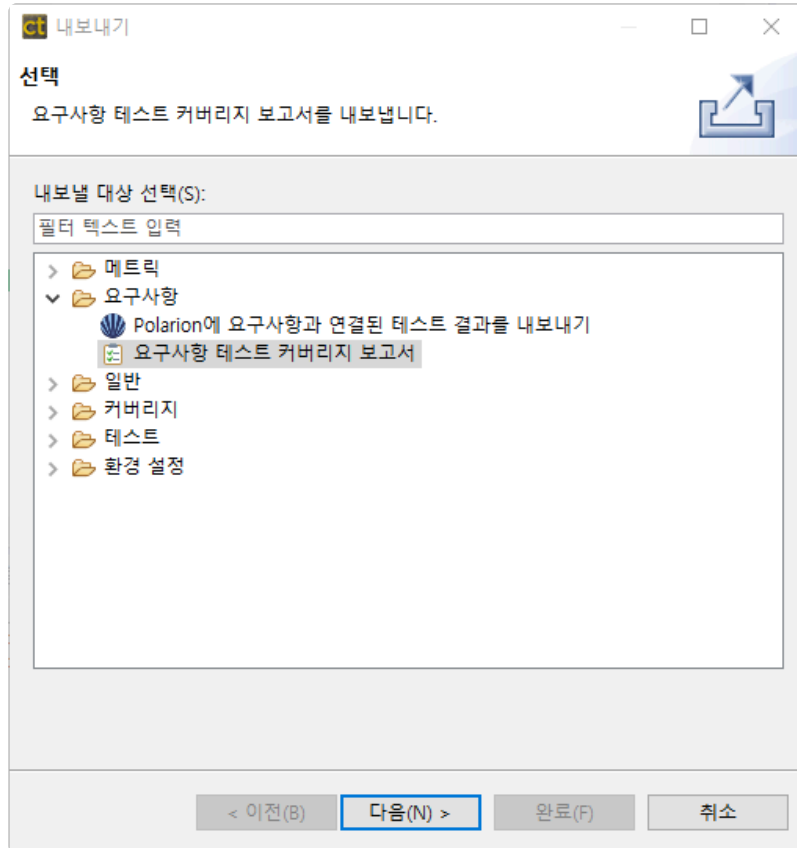
✿ 선택된 V-SPICE 프로젝트의 요구사항과 연결된 테스트가 없으면 요구사항 목록이 표시되지 않습니다. 요구사항 목록이 표시되지 않아도 모든 테스트 정보는 내보내집니다.



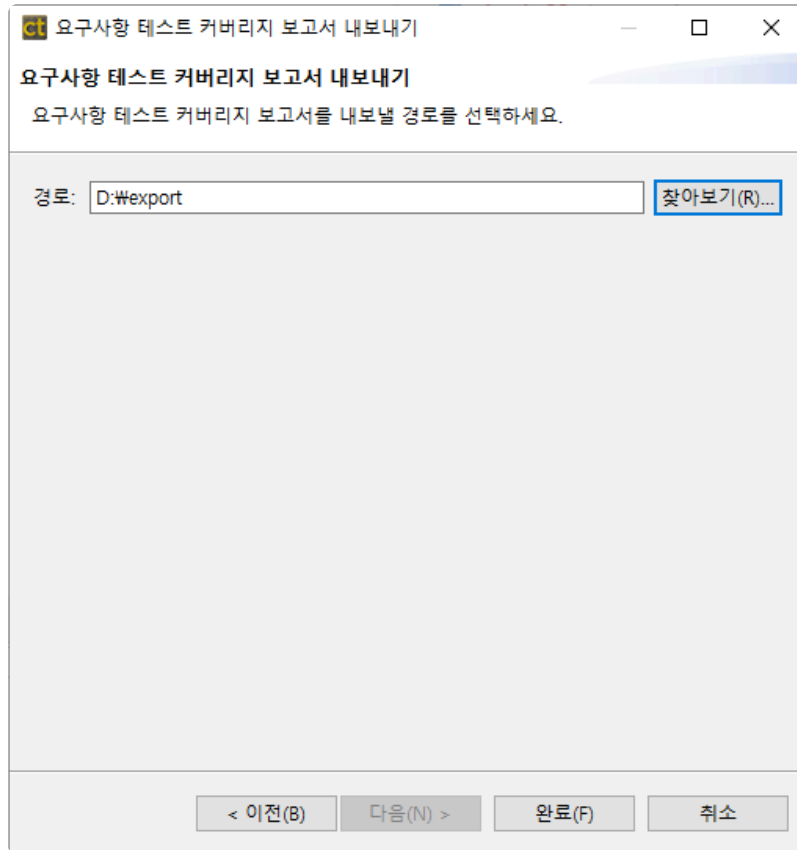
## 19.4.4. 요구사항 - 요구사항 테스트 커버리지 보고서

요구사항 테스트 커버리지를 Excel 형식의 보고서로 만들 수 있습니다.

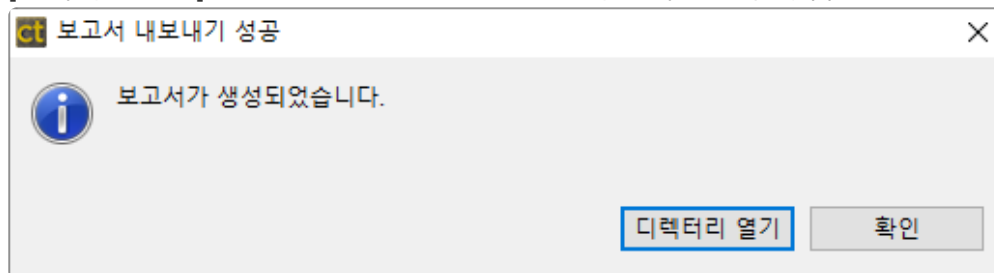
1. 내보내기 마법사에서 [요구사항 테스트 커버리지 보고서]를 선택하고 [다음] 버튼을 클릭합니다.



2. 요구사항 테스트 커버리지 보고서를 내보낼 경로를 선택하고 [완료] 버튼을 클릭합니다.



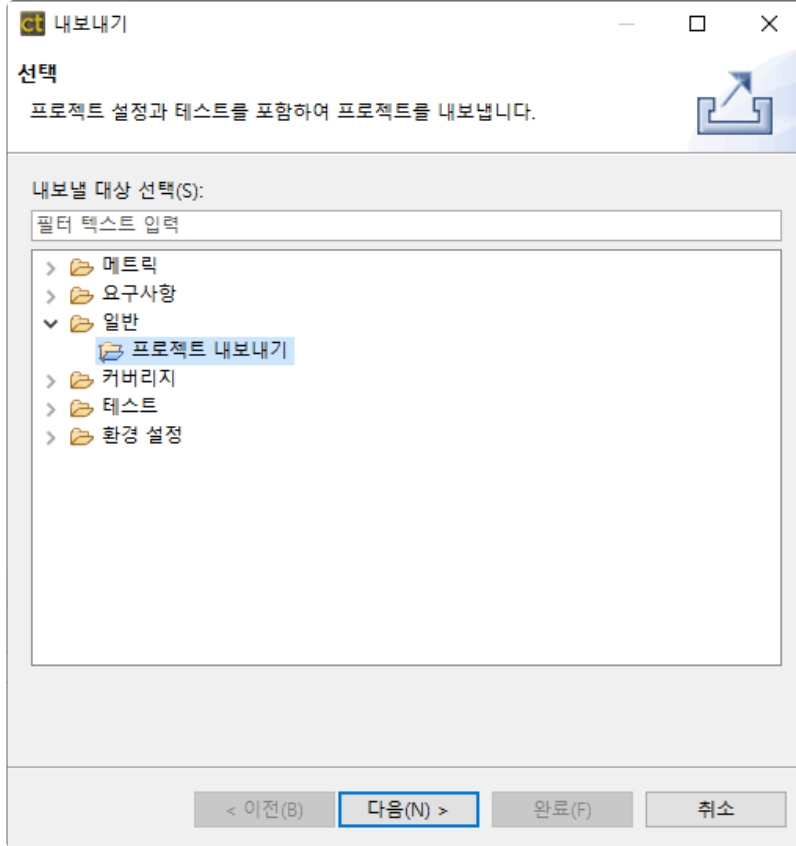
3. [디렉터리 열기] 버튼을 클릭하여 내보낸 보고서를 확인할 수 있습니다.



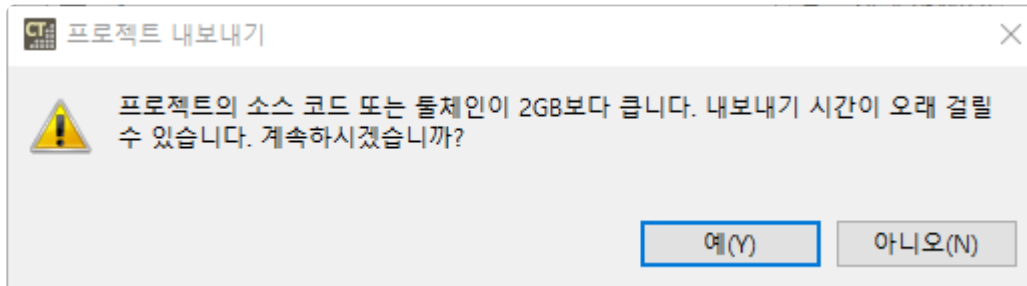
## 19.4.5. 일반 - 프로젝트 내보내기

프로젝트 설정과 테스트를 포함하여 프로젝트를 내보낼 수 있습니다. 내보낸 프로젝트는 [프로젝트 가져오기]로 가져올 수 있습니다.

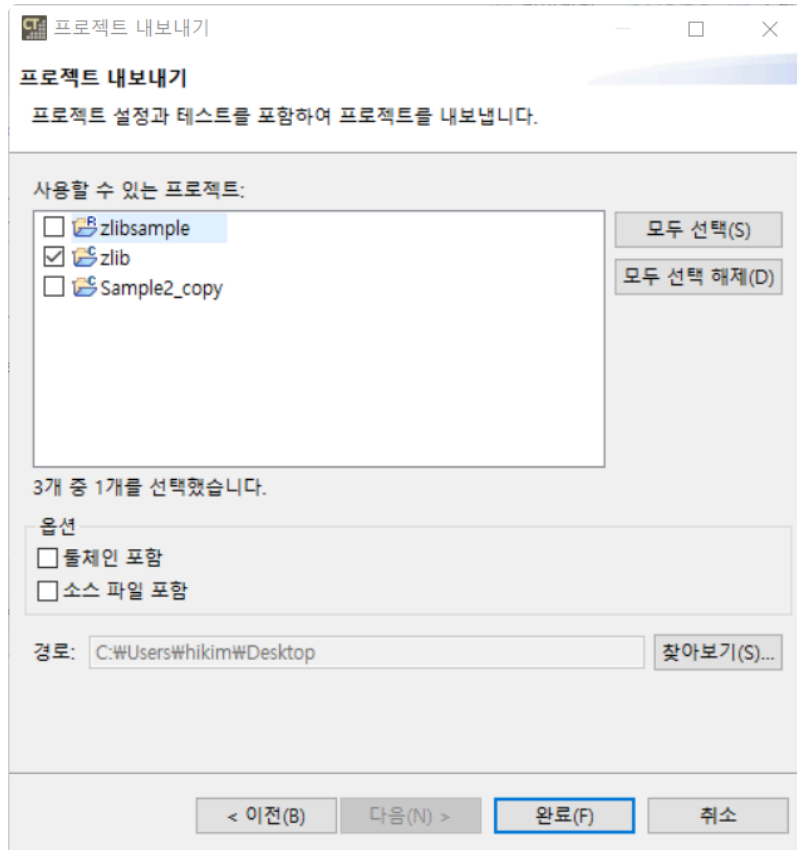
1. 내보내기 마법사에서 [일반] > [프로젝트 내보내기]를 선택하고 [다음] 버튼을 클릭합니다.



2. 툴체인이나 소스 파일을 포함하여 내보내기 시 2GB가 넘어가면, 시간이 오래 걸릴 수 있다는 경고가 발생합니다. 프로젝트 소스 파일 경로에 불필요한 파일들이 존재한다면 해당 파일을 삭제하는 방법으로 내보내는 속도를 향상시킬 수 있습니다. 그대로 진행을 원한다면 [예] 버튼을 클릭합니다.



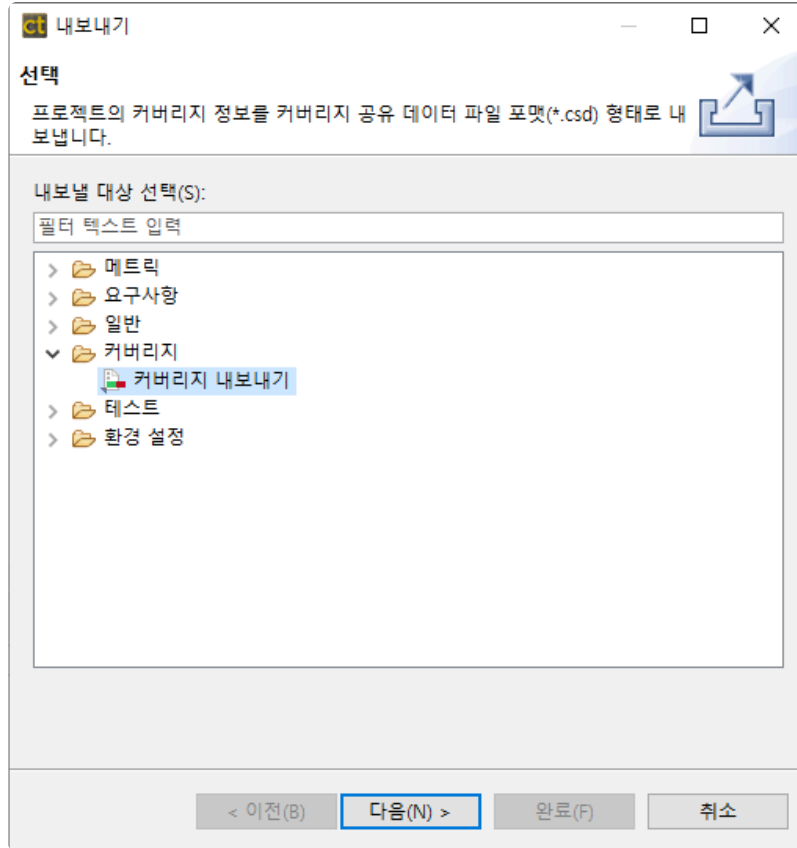
3. 내보내기 대상 프로젝트와 내보낼 경로를 선택한 후, [완료] 버튼을 클릭합니다. 프로젝트를 내보낼 때 툴체인과 소스 파일을 포함해서 내보낼 수 있습니다.



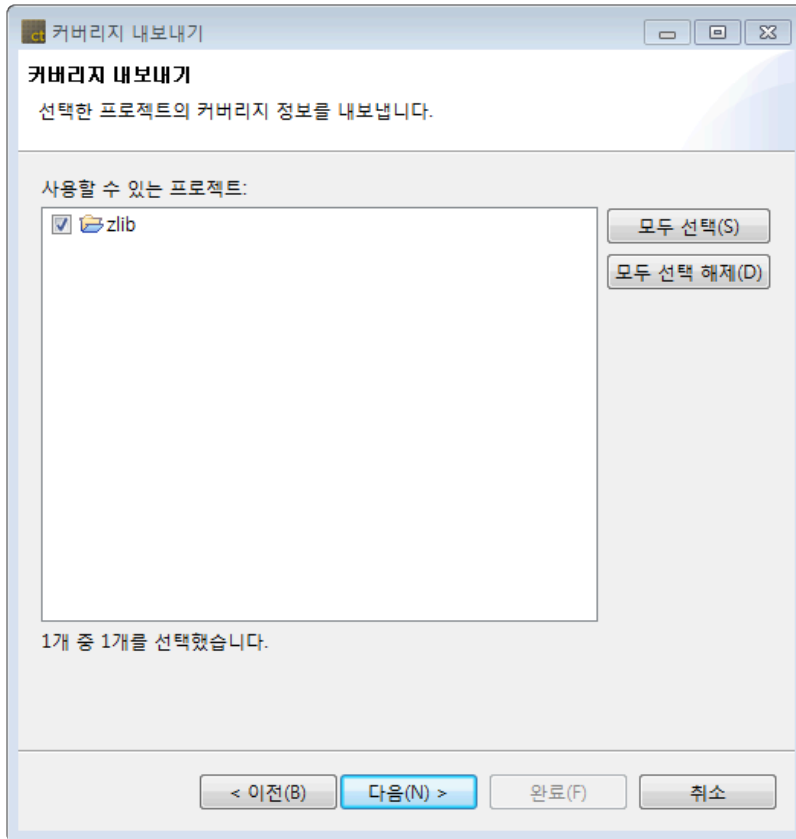
## 19.4.6. 커버리지 - 커버리지 내보내기

프로젝트에서 측정한 커버리지 정보를 커버리지 공유 데이터(\*.csd) 형식의 파일로 내보낼 수 있습니다. 내보낸 커버리지는 [커버리지 가져오기]로 가져올 수 있습니다.

1. 내보내기 마법사에서 [커버리지] -> [커버리지 내보내기]를 선택하고 [다음] 버튼을 클릭합니다.



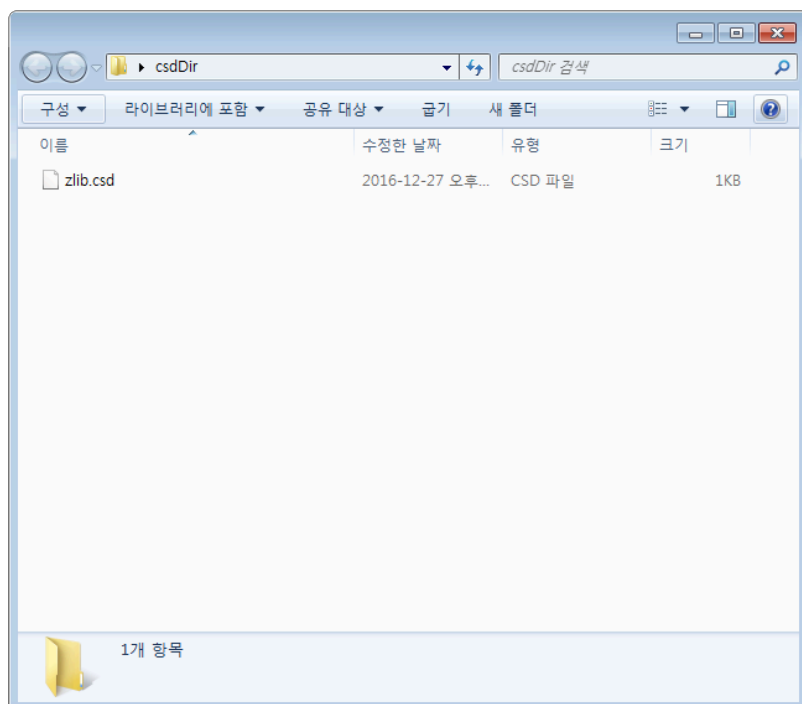
2. 커버리지 정보를 내보낼 프로젝트를 선택하고 [다음] 버튼을 클릭합니다.



3. 내보낼 디렉터리 경로를 입력 후 [완료] 버튼을 클릭합니다.



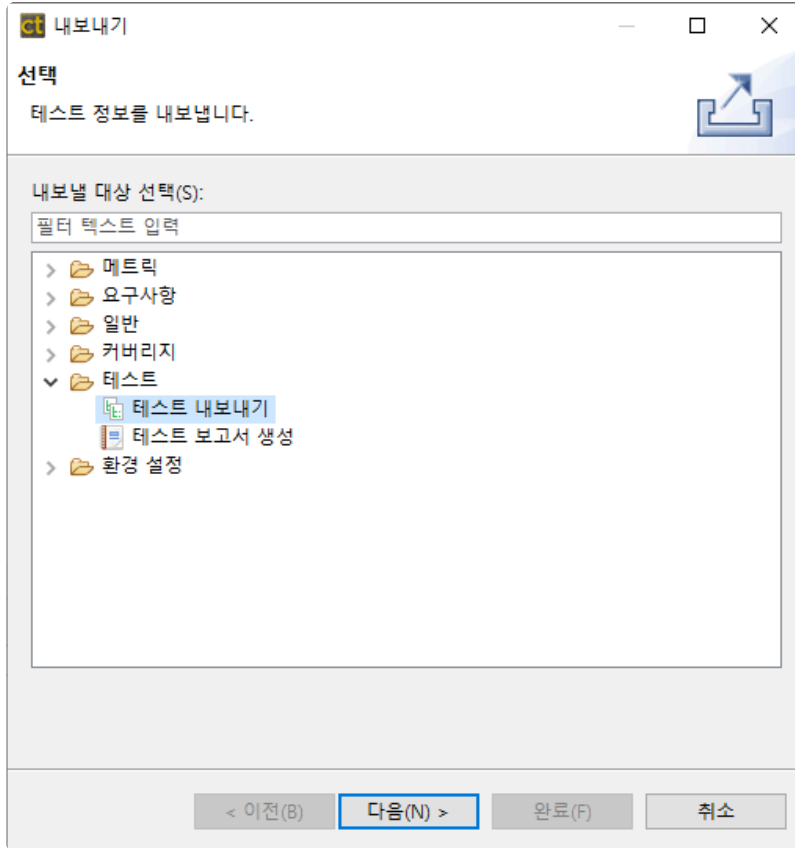
4. 프로젝트 이름과 동일한 이름의 파일이 생성됩니다.



## 19.4.7. 테스트 - 테스트 내보내기

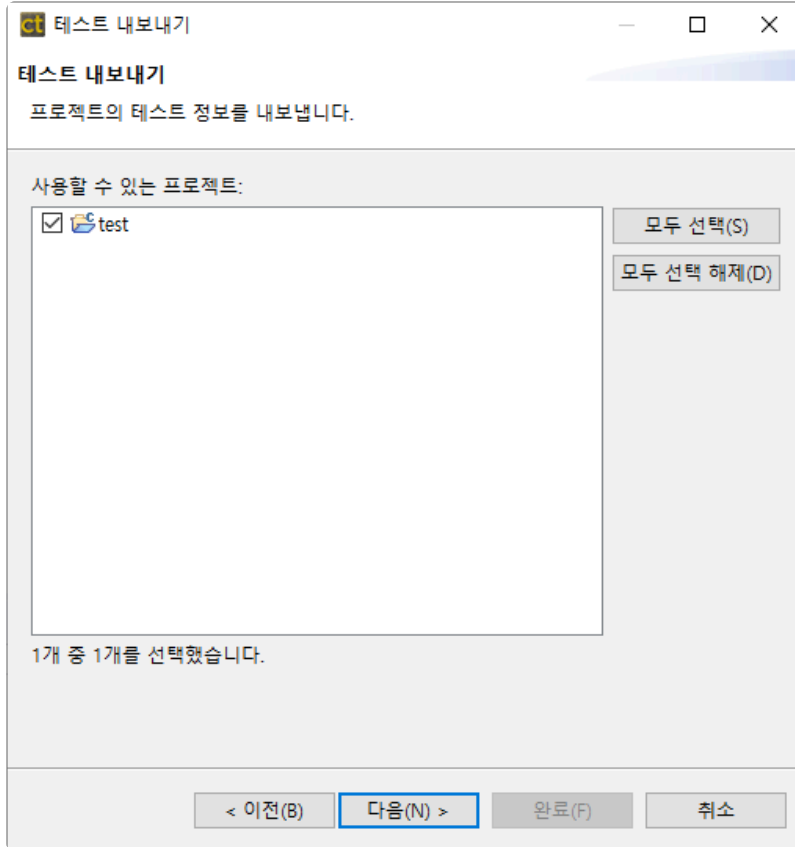
프로젝트에서 생성한 테스트 정보를 한 번에 내보낼 수 있습니다. 프로젝트 분석 전이나 생성된 테스트 정보 (유닛/통합 테스트, 스텝)가 없는 경우는 내보내기를 수행할 수 없습니다.

1. 내보내기 마법사에서 [테스트] > [테스트 내보내기]를 선택하고 [다음] 버튼을 클릭합니다.

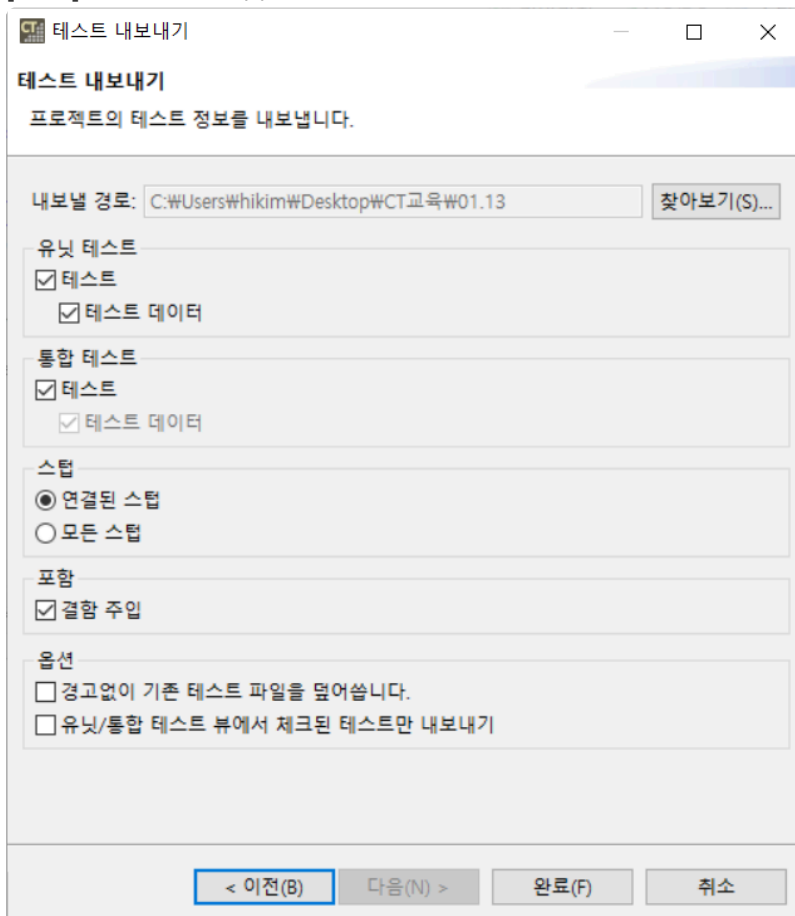


2. 테스트를 내보낼 프로젝트를 선택하고 [다음] 버튼을 클릭합니다.





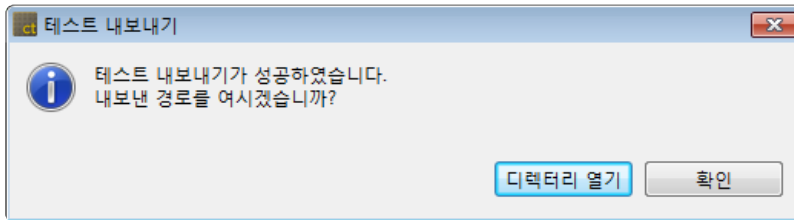
3. 내보낼 경로, 유닛 테스트 항목, 통합 테스트 항목, 스텝 항목, 결함 주입 포함 여부와 옵션을 선택 후 [완료] 버튼을 클릭합니다.



- a. [내보낼 경로]에 테스트를 내보낼 경로를 입력합니다.
- b. [유닛 테스트] 그룹에서 [테스트]와 [테스트 데이터] 내보내기 여부를 각각 체크합니다.

- c. [통합 테스트] 그룹에서 [테스트] 내보내기 여부를 체크합니다.
  - 통합 테스트를 내보낼 때는 테스트 데이터와 세트로 내보내집니다.
- d. [스텝] 그룹에서 [연결된 스텝]과 [모든 스텝] 중 내보낼 스텝을 선택합니다.
  - 연결된 스텝: 내보낼 테스트와 연결되어 있는 스텝. 테스트와 연결이 없는 스텝 혹은 내보낼 테스트가 선택 되어 있지 않은 경우 스텝을 내보내지 않습니다.
  - 모든 스텝: 프로젝트에 생성된 모든 스텝을 내보냅니다.
- e. [포함] 그룹에서 결함 주입을 포함하여 내보낼지 선택합니다.
- f. [옵션] 그룹에서 기존 테스트 파일이 존재 시 덮어쓸지 여부 또는 유닛/통합 테스트 뷰에서 체크된 테스트만 내보내기를 선택합니다.

4. 테스트 내보내기 성공 메시지 창에서 [디렉터리 열기] 버튼을 선택하여 테스트가 저장된 폴더에서 보내진 테스트를 확인하거나 [확인] 버튼을 선택하여 내보내기를 종료합니다.

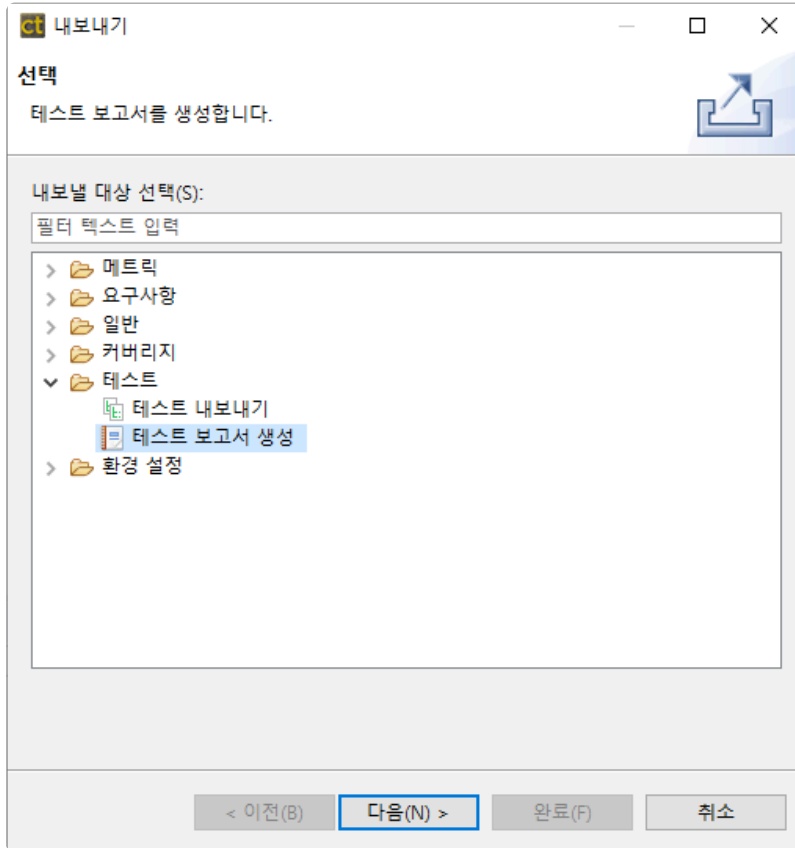


- \* Controller Tester 2.6.14 이후의 버전부터 테스트 내보내기 시, 내보낸 프로젝트 디렉터리에 **testinfo.export** 파일이 생성됩니다.  
(testinfo.export: 내보낸 정보를 기록하고 있는 파일)

## 19.4.8. 테스트 - 테스트 보고서 생성

내보내기 기능을 통하여 테스트의 수행 결과 보고서를 다양한 형태의 문서로 생성할 수 있습니다.

1. 내보내기 마법사에서 [테스트] -> [테스트 보고서 생성]을 클릭합니다.



2. 보고서를 생성할 프로젝트와 보고서 형식을 선택합니다.

**내보내기**

**테스트 결과 내보내기**

프로젝트를 선택해주시요.

사용할 수 있는 프로젝트:

☒ zlib

모두 선택(S)

모두 선택 해제(D)

1개 중 1개를 선택했습니다.

**커버리지**

☒ 유닛 테스트

☒ 통합 테스트

☒ 구문

☒ 함수

☒ 분기

☒ 함수 호출

☒ MC/DC

**선택사항**

☐ 프로젝트 정보 포함 (틀체인, 분석대상 등)

**그 외 보고서**

☒ 파일 커버리지 보고서 (형식: XLS)

☐ 테스트별 보고서 (형식: XLS)

☐ 테스트 스텝 보고서 (형식: XLS)

☐ 외부 커버리지 포함 보고서

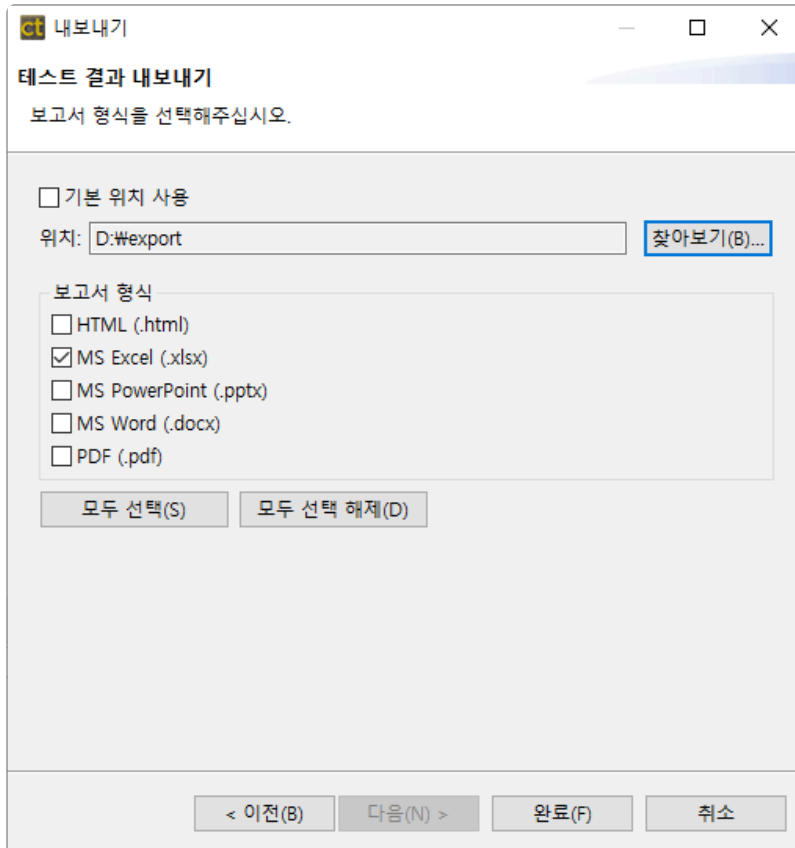
☐ 커버되지 않은 함수 보고서 (형식: XLS)

< 이전(B)   다음(N) >   완료(F)   취소

- [사용할 수 있는 프로젝트]에서 보고서를 생성하고자 하는 프로젝트를 선택합니다.
- [커버리지] 보고서에 포함할 커버리지를 체크합니다.
- [선택사항] 그룹에서 프로젝트 정보(틀체인, 분석 대상 등) 포함 여부를 선택합니다.
- [그 외 보고서] 그룹에서 내보낼 보고서 형식에 각각 체크합니다.

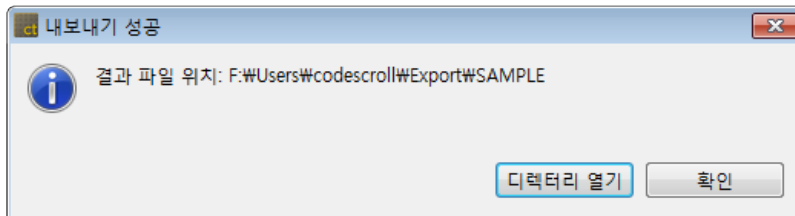
보고서 종류	설명
파일 커버리지 보고서	선택한 프로젝트에서 커버된 함수의 보고서를 파일별로 생성합니다.
테스트별 보고서	선택한 프로젝트에서 테스트 결과와 커버리지 보고서를 테스트별로 생성합니다.
테스트 스텝 보고서	선택한 프로젝트에서 생성된 스텝에 대한 정보를 보고서로 생성합니다.
외부 커버리지 포함 보고서	선택한 프로젝트에서 외부 커버리지를 포함한 보고서를 생성합니다.
커버되지 않은 함수 보고서	선택한 프로젝트에서 커버되지 않은 함수 보고서를 생성합니다.

- 보고서를 내보낼 위치와 보고서 형식을 선택한 후 [완료] 버튼을 선택합니다.



- a. [기본 위치 사용]은 기본값으로 “사용자계정\Export” 디렉터리를 지정합니다.
- b. [보고서 형식]은 보고서의 파일 형식으로 html, pdf, docx, pptx, xlsx를 지원합니다. 보고서의 파일 확장자는 중복 선택이 가능합니다.

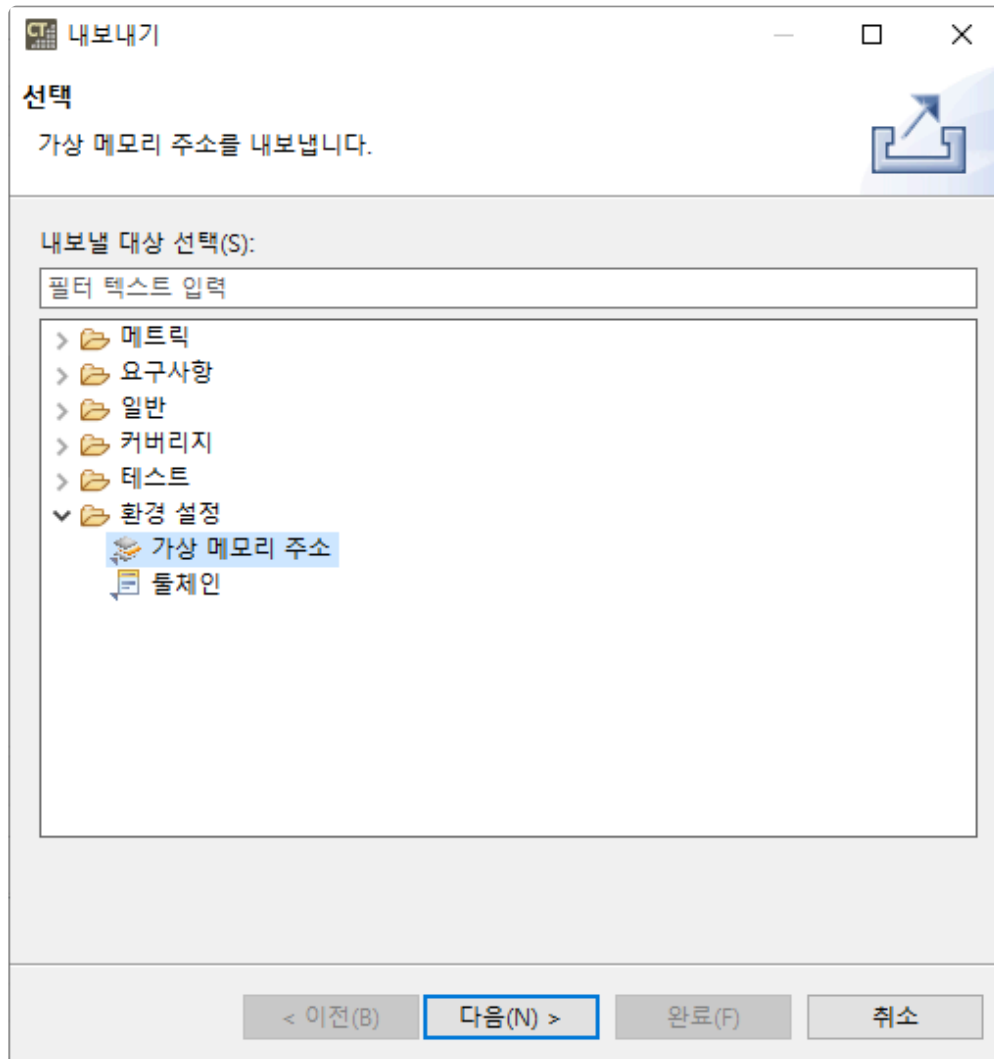
4. 보고서 생성 완료 메시지 창에서 [디렉터리 열기] 버튼을 클릭하여 보고서가 저장된 디렉터리를 열 수 있습니다.



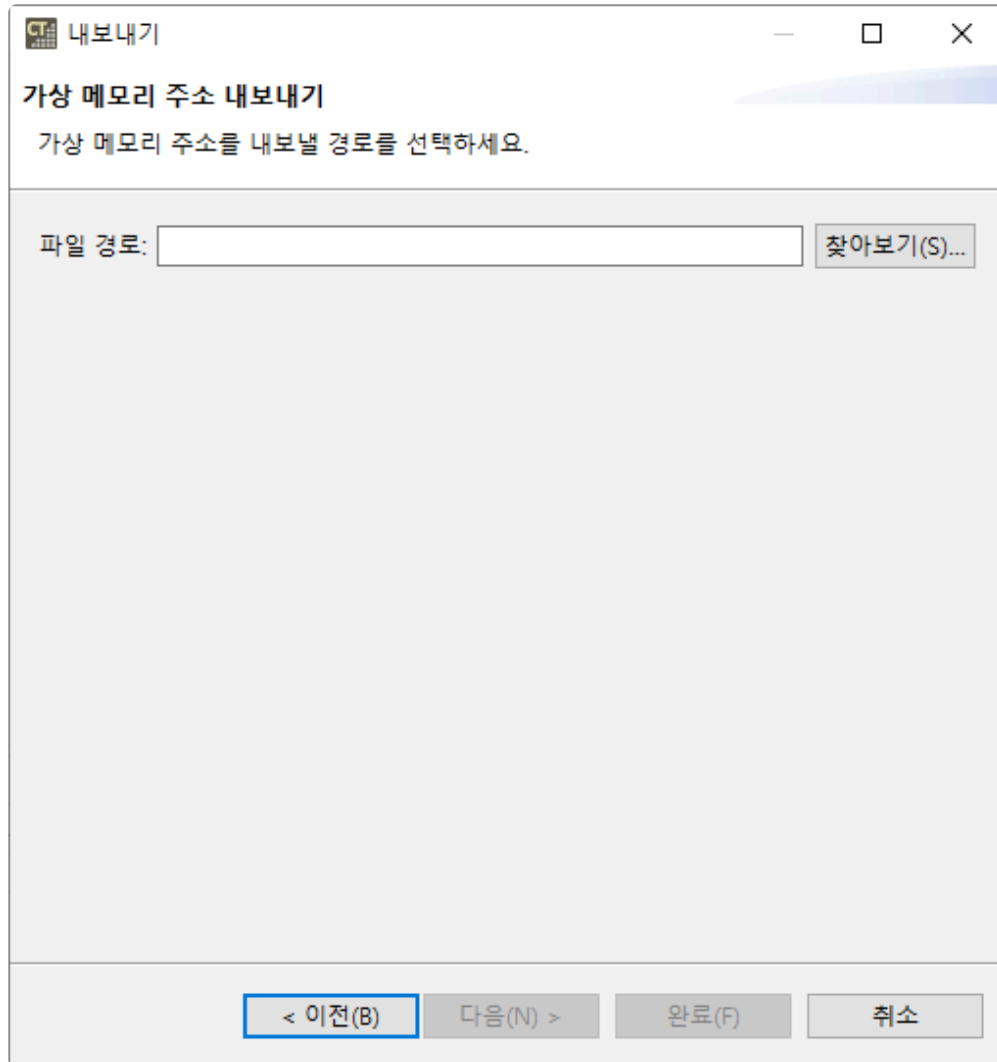
## 19.4.9. 환경 설정 – 가상 메모리 주소

가상 메모리 주소 정보를 내보낼 수 있습니다.

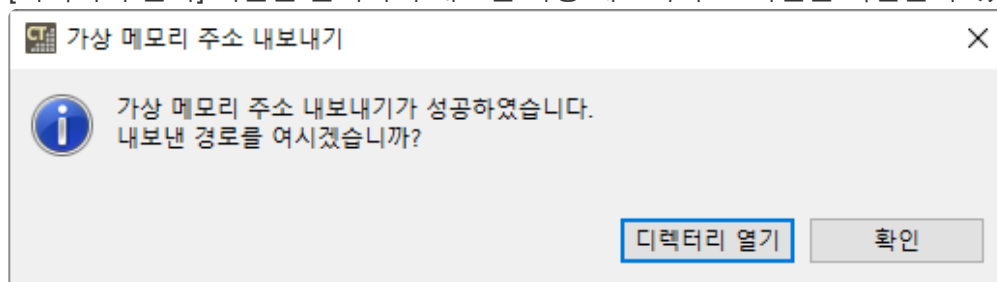
1. 내보내기 마법사에서 [환경 설정] > [가상 메모리 주소]를 클릭한 후 [다음] 버튼을 클릭합니다.



2. 가상 메모리 주소를 내보낼 경로를 선택하고 [완료] 버튼을 클릭합니다.



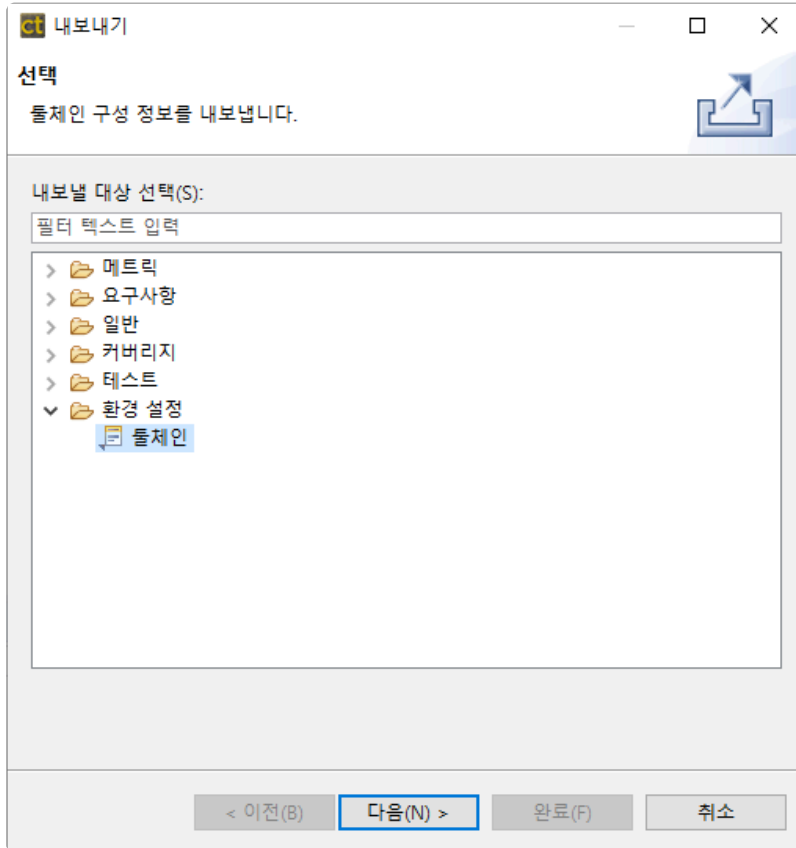
3. [디렉터리 열기] 버튼을 클릭하여 내보낸 가상 메모리 주소 파일을 확인할 수 있습니다.



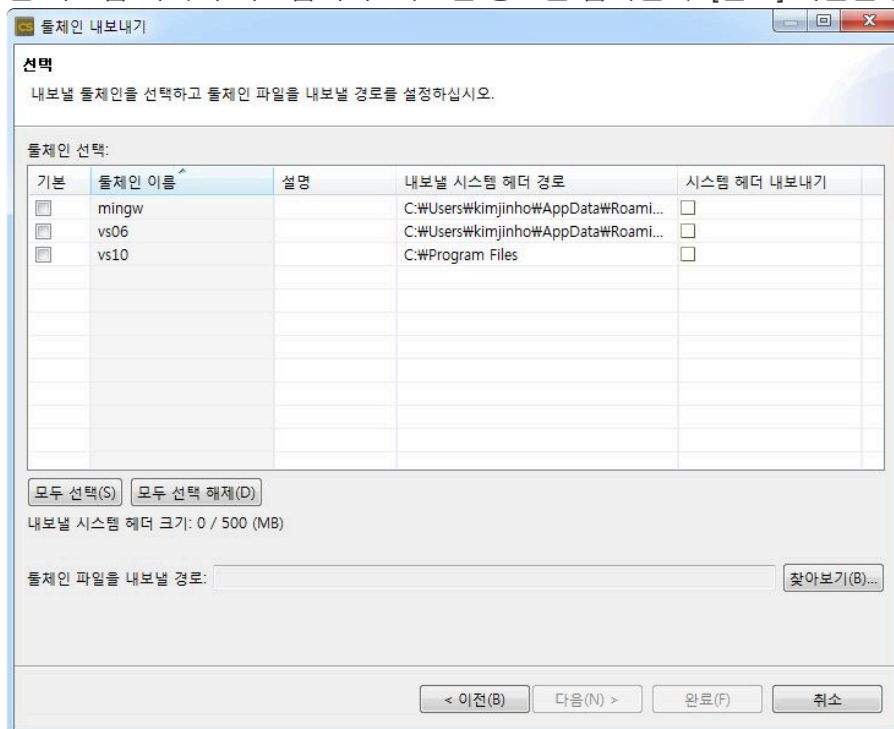
## 19.4.10. 환경 설정 - 툴체인

생성된 툴체인 정보를 내보낼 수 있습니다.

1. 내보내기 마법사에서 [환경 설정] > [툴체인]을 클릭한 후 [다음] 버튼을 클릭합니다.



2. 현재 등록되어 있는 툴체인 목록에서 내보낼 툴체인을 선택합니다. 시스템 헤더를 포함하여 내보내려면 시스템 헤더에 체크합니다. 내보낼 경로를 입력한 후 [완료] 버튼을 클릭합니다.





## 20. 편집 메뉴

편집 메뉴에서는 뷰 또는 편집기에서 선택된 항목에 대한 잘라내기, 복사, 붙여넣기, 삭제, 찾기/바꾸기 기능을 수행하거나, 마지막으로 수행된 액션을 실행 취소하거나 다시 실행할 수 있습니다.

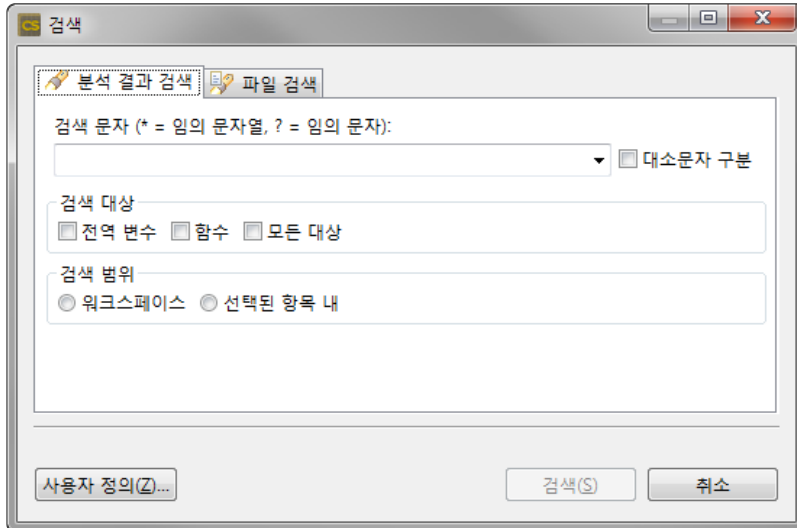


선택된 뷰 또는 편집기에 따라서 메뉴의 구성이 달라질 수 있습니다.

## 21. 검색 메뉴

### 분석 결과 검색

분석 후 생성된 결과를 검색할 수 있습니다.



### 검색 문자

검색할 문자를 입력합니다.

사용 가능한 와일드 카드는 검색 대화 상자에 표시됩니다.

- “\*” 임의의 문자열: 빈 문자열을 포함한 문자의 집합과 일치
- “?” 임의의 문자: 모든 문자에 대해 일치

### 검색 대상

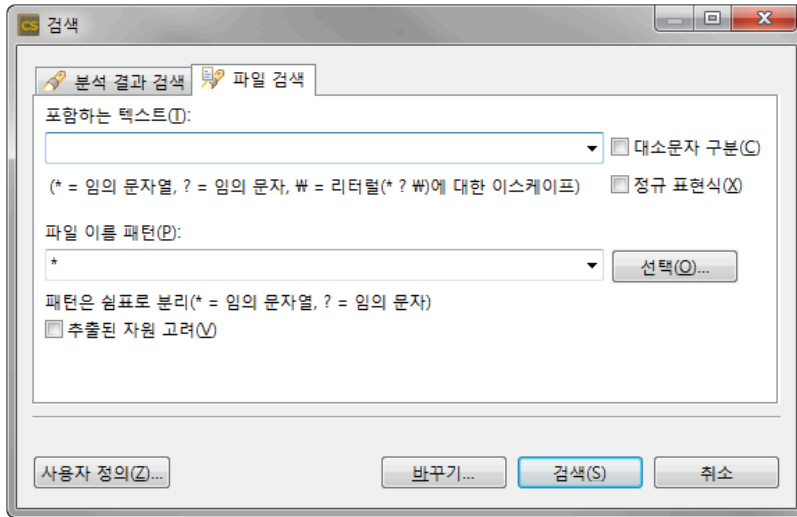
검색 대상을 선택합니다. 전역변수, 함수 또는 모든 대상을 검색할 수 있습니다.

### 검색 범위

검색의 범위를 선택합니다. 워크스페이스 또는 선택된 항목 범위에서 검색할 수 있습니다.

### 파일 검색

파일을 검색할 수 있습니다.



## 포함하는 텍스트

검색할 문자를 입력합니다. 파일을 검색하려면 필드를 비워둬야 합니다.

[▼]를 클릭하면 최근에 검색한 문자를 선택할 수 있습니다.

사용 가능한 와일드 카드는 검색 대화 상자에 표시됩니다.

- “\*” 임의의 문자열: 빈 문자열을 포함한 문자의 집합과 일치
- “?” 임의의 문자: 모든 문자에 대해 일치
- “\*”, “?”, 또는 “\” 문자를 검색하려면 “\\*” 와일드 카드로 이러한 문자를 사용하지 않는 것을 나타내기 위해, 문자 앞에 백슬래시를 입력합니다. (예, “\?” 또는 “\”)

## 파일 이름 패턴

특정 표현을 통해 검색할 파일에 대한 모든 파일 이름 패턴을 입력합니다.

파일 이름 패턴을 사용할 수 있는 와일드 카드는 검색 대화 상자에 표시됩니다.

- “\*” 임의의 문자열: 빈 문자열을 포함한 문자의 집합과 일치
- “?” 임의의 문자: 모든 문자에 대해 일치

## 22. 프로젝트 메뉴

프로젝트 메뉴는 프로젝트에 대한 작업(열기, 닫기, 초기화)을 수행할 수 있습니다.



### 프로젝트 열기

테스트 네비게이터 뷰에서 선택된 닫혀있는 프로젝트를 열 수 있습니다.

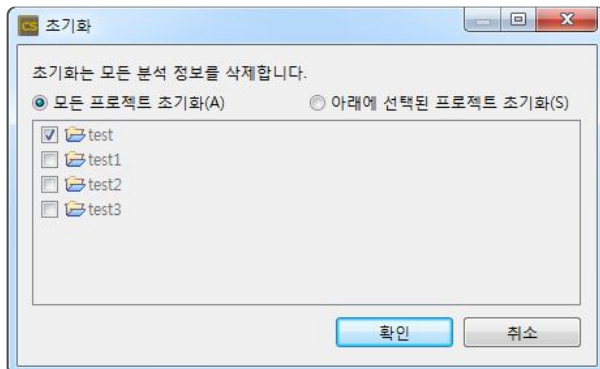
### 프로젝트 닫기

테스트 네비게이터 뷰에서 선택된 열려있는 프로젝트를 닫을 수 있습니다.

### 프로젝트 초기화

워크스페이스에 열려있는 모든 프로젝트 또는 아래의 프로젝트 목록 중에서 선택한 프로젝트를 초기화합니다.

프로젝트를 초기화하면 분석된 모든 결과가 사라집니다.



### CLI 설정파일 만들기

테스트 네비게이터 뷰에서 선택한 프로젝트 정보를 바탕으로 CLI 환경에서 CT 2023.12를 수행하는데 필요한 설정파일을 생성할 수 있습니다. 자세한 내용은 본 문서의 [CLI](#) 페이지를 확인하시기 바랍니다.

### 프로젝트 로그 수집

테스트 네비게이터 뷰에서 선택한 프로젝트의 로그 파일을 지정한 경로에 내보낼 수 있습니다.

## 특성

테스트 네비게이터 뷰에서 선택된 프로젝트에 대한 정보를 보거나 설정을 변경할 수 있습니다.  
자세한 설명은 [특성 페이지](#) 에서 확인하실 수 있습니다.

## 23. 창 메뉴

창 메뉴에서는 활성화된 편집기와 동일한 새 편집기를 열거나, 퍼스펙티브 또는 새 뷰를 열고, 환경 설정을 볼 수 있습니다.



### 새 편집기

선택한 소스코드 편집기 또는 테스트 편집기를 복제합니다.

### 퍼스펙티브 열기

CT 2023.12에 등록된 퍼스펙티브를 선택하여 열 수 있습니다.

### 뷰 표시

CT 2023.12에 등록된 뷰를 선택하여 열 수 있습니다.

### 퍼스펙티브 재설정

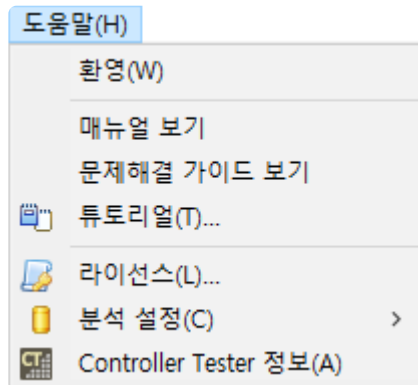
퍼스펙티브를 초기 상태로 재설정할 수 있습니다.

### 환경설정

현재 도구에 적용된 설정을 확인하거나 변경할 수 있습니다.  
자세한 내용은 [환경설정](#) 에서 확인하실 수 있습니다.

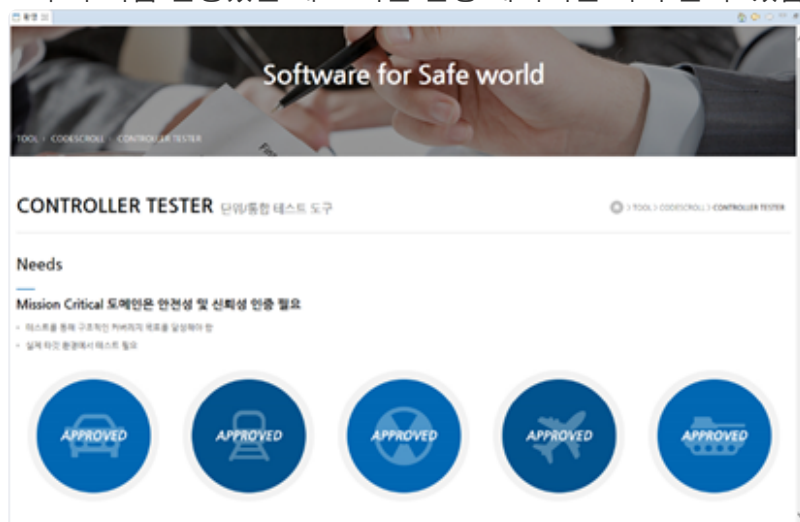
## 24. 도움말 메뉴

도움말 메뉴에서는 매뉴얼, 문제해결 가이드, 튜토리얼, 라이선스 설정, 분석 설정 변경 및 되돌리기 그리고 설치된 제품의 정보를 확인할 수 있습니다.



### 환영

도구가 처음 실행됐을 때 보이는 환영 페이지를 다시 볼 수 있습니다.



### 매뉴얼 보기

CT 2023.12 매뉴얼을 확인할 수 있습니다.

CODESCROLL

CONTROLLER TESTER

User Manual

3.6

Search

Korean

시작하기 전에

개요

설치하기

제거하기

실행

라이선스 설정

▼

플체인(분석기) 설정

플체인 자동 등록

플체인 추가

플체인 편집

플체인 복제

플체인 삭제

플체인 내보내기

플체인 가져오기

▼

프로젝트 만들기

소스 파일 C/C++ 프로젝트

임베디드 C/C++ 프로젝트

비주얼 스튜디오 프로젝트

시작하기 전에

문서의 목적

CodeScroll Controller Tester 제품의 사용법에 관한 정보를 제공합니다.

개요 >

Need more help with this?  
Don't hesitate to contact us here.

Copyright © 2021 Suresofttech — Powered by MariaDB

## 문제해결 가이드 보기

CT 2023.12 문제 해결 가이드를 확인할 수 있습니다.

CODESCROLL

CONTROLLER TESTER

Controller Tester Troubleshooting Guides

3.6

Search

Korean

C++20 항목 포함된 소스 코드로 테스트를 수행하는 경우

테스트 실행 시 발생할 수 있는 여러 통합 테스트 가져오기 후 전역 변수를 찾지 못하는 경우

테스트 실행시 "C2118 : 첨자가 음수입니다" 여러 발생시

Visual Studio 2015 플체인 사용시 SDK 버전 문제

Controller Tester가 설치되지 않는 경우

대용량 제어 흐름 그래프를 이미지 파일로 저장하지 못하는 경우

워크스페이스가 망가진 경우

VPES 입로트 시 발생할 수 있는 문제

메트릭 뷰 관련 오류

지역 변수의 값을 확인하고 테스트 케이스 결과에 반영하는 방법

함수 호출 커버리지와 함수 커버리지의 결과가 맞지 않는 경우

실수형 변수의 기대값, 출력값 관련 문제

Controller Tester 문제 해결 가이드

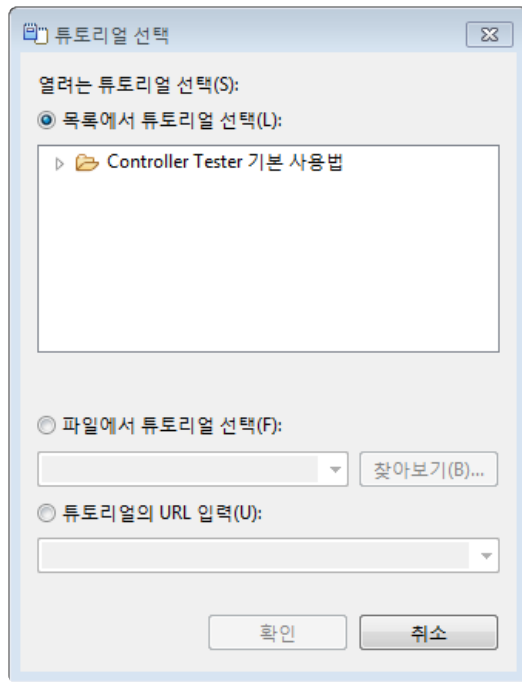
Controller Tester를 사용하며 발생할 수 있는 일반적인 문제에 대한 해결 가이드 문서입니다.

- 가변 배열 멤버에 값을 할당하는 방법
- 메모리가 부족할 때 힙 메모리 늘리는 방법
- 비정상 종료로 인해 프로젝트 DB 파일(\*.csp)이 손상된 경우
- QNX 소프트웨어를 테스트할 때 testrun.exe가 비정상 종료되는 경우
- 커버리지 뷰에 특정 함수가 표시되지 않는 경우
- 프로젝트 분석 실행 후 오류 메시지가 출력되지 않는 경우
- Windows 디스플레이 배율이 100%가 아닐 때, 화면이 잘리는 문제
- Windows.h 파일을 찾지 못하는 문제
- 로그에 INFO [ut.hio]: runTest.testrun exit code(105)가 출력되고 유닛 테스트 실행이 안되는 경우
- "플체인 정보 자동 추출"이 실패하는 경우
- 오류 뷰에서 메시지가 비정상적으로 표시될 경우
- C++20 항목 포함된 소스 코드로 테스트를 수행하는 경우
- 테스트 실행 시 발생할 수 있는 여러
- 통합 테스트 가져오기 후 전역 변수를 찾지 못하는 경우
- 테스트 실행시 "C2118 : 첨자가 음수입니다" 여러 발생시
- Visual Studio 2015 플체인 사용시 SDK 버전 문제

## 튜토리얼

CT 2023.12 사용법에 대한 튜토리얼을 실행할 수 있습니다.



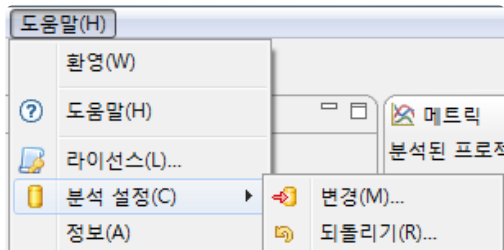


## 분석 설정 변경

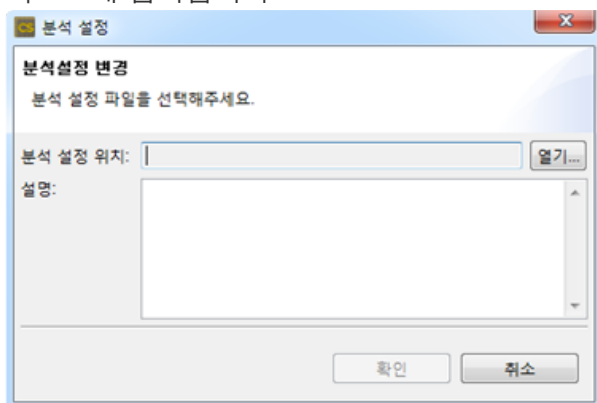
슈어소프트테크(주)에서 제공한 압축 파일을 입력하여 분석 설정 정보를 변경 할 수 있습니다.

### 사용방법

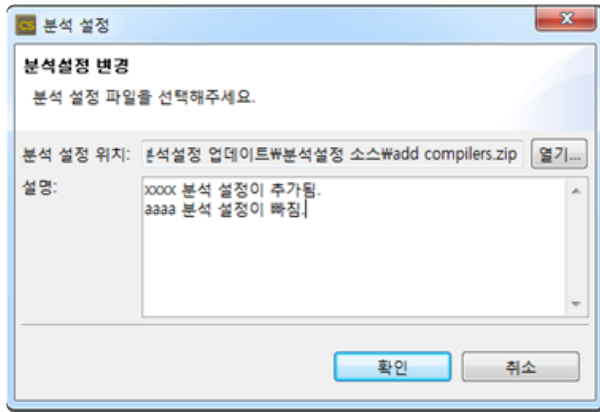
1. [도움말] > [분석 설정] > [변경] 메뉴를 선택합니다.



2. [열기...] 버튼을 클릭하여 압축된 분석 설정 파일을 선택하고 옵션으로 분석 설정 변경 사항을 [설명] 텍스트에 입력합니다.

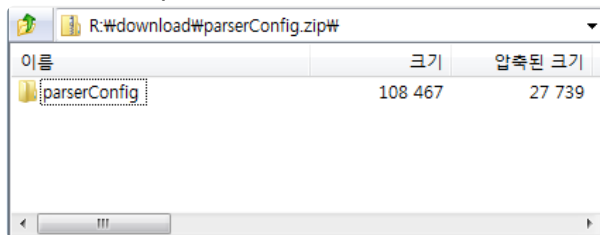


3. [확인] 버튼을 선택합니다.



## 압축된 분석 설정 파일 형식

- 아래 그림과 같이 압축파일을 열면 바로 [parserConfig] 디렉터리가 존재해야 합니다.
- 확장자는 “zip” 입니다.

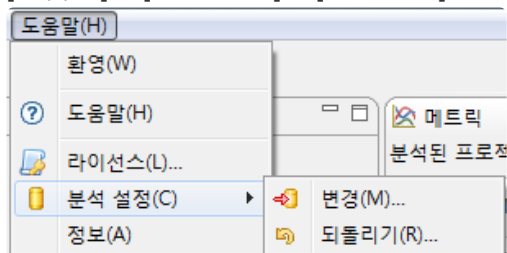


## 분석 설정 되돌리기

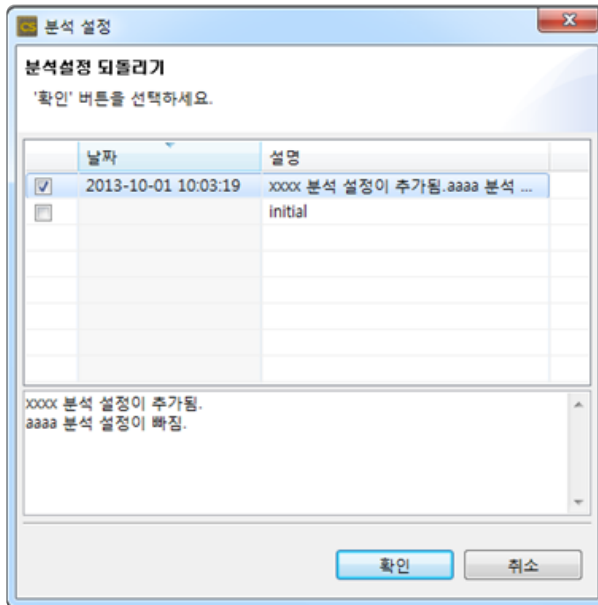
이전 분석 설정 정보 중 하나를 선택하여 되돌릴 수 있습니다.

### 사용 방법

1. [도움말] > [분석 설정] > [되돌리기] 메뉴를 선택합니다.

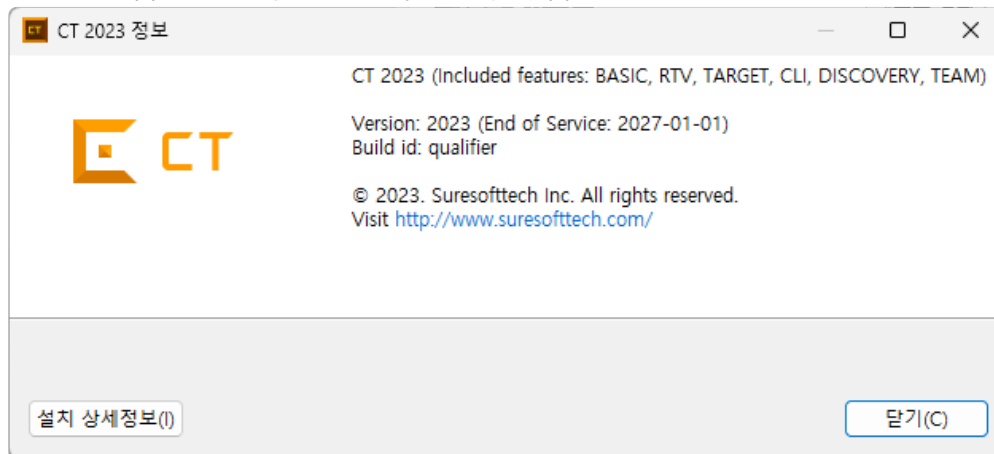


2. 기존 분석 설정 목록 중 되돌릴 분석 설정 정보를 체크하고, [확인] 버튼을 선택합니다.



## 정보

설치된 제품의 정보와 버전을 확인할 수 있습니다.



[설치 상세정보]를 선택하여 포함된 기능과 플러그인, 서비스 종료 날짜를 확인할 수 있습니다. 서비스가 종료되면 CT 2023.12는 사용 가능하지만 유지보수를 받을 수 없습니다.

## 25. 문제 해결

### 설치 시 보안 프로그램과의 충돌

일부 보안프로그램에서 도구 설치 패키지 실행을 중지시킵니다. 설치 과정에서 오류가 발생한다면, 일시적으로 보안프로그램 동작을 중지하고 설치를 다시 진행해주십시오.

### 제품 실행 시 작업공간 지정 후 제품이 구동되지 않는 경우

제품 최초 실행 시 CodeScroll의 전역 데이터가 \APPDATA\ 경로에 저장되어 있습니다. 이 경로에 대한 접근에 제약이 있을 경우 제품이 구동되지 않을 수 있습니다.

이를 해결하기 위한 방법으로 전역 데이터 경로를 설정할 수 있는 기능을 사용하시기 바랍니다.

설정 방법: CodeScroll.ini 파일의 -g default(기본값) 대신에 설정할 전역 데이터 경로를 입력합니다(-g 또는 -global, csc.ini 파일도 동일)

-g 옵션이 주어지지 않으면, 제품 최초 실행 시에 임의의 전역 데이터 경로를 설정하는 다이얼로그가 나타납니다. 이때 취소하면 기본 경로로 설정됩니다.

### Windows Vista/7 에서 설치 및 실행

User Access Control(UAC) 기능이 비활성화 된 상태에서만 실행할 수 있습니다.

UAC를 비활성화하지 않고 사용하려면, Program Files 이하에 설치하지 않도록 설치 과정에서 디렉터리를 변경하십시오.

또한, 설치 및 실행 Windows 계정이 동일해야 하며, 반드시 관리자 계정으로 설치해야 합니다.

### 분석 실패 또는 Compile(전처리) 오류

테스트 수행이 불가능한 소스 코드 상태입니다. 컴파일/링크 플래그와 헤더파일/라이브러리 설정을 확인해 주십시오.

설정 내용 확인 방법은 메뉴의 [환경 설정] -> [툴체인 설정]에서 가능합니다.

만약, 모든 설정 내용이 정상일 경우(실제 개발 환경(IDE 등)에서 문제 없이 Build가 가능할 경우)에도 오류가 발생하면, 슈어소프트테크(주)로 문의하시기 바랍니다.

### 모든 테스트 케이스 수행이 실패한 경우

방화벽에 의해 테스트 엔진 동작이 차단되어서 발생하는 현상입니다. 테스트 수행 시 메모리에 로드 되는 테스트 엔진을 방화벽 설정의 예외 사항으로 추가하여야 정상적인 수행이 가능합니다.

### 툴체인 자동 등록이 실패하는 경우

PC에 설치된 툴체인 정보에 문제가 있어 자동 등록이 실패하는 경우, 툴체인 자동 등록 기능을 끌 수 있습니다. (--disableAutoToolchain 옵션)

- CodeScroll.ini 파일에 --disableAutoToolchain 옵션을 추가

## 스텝 연결 후 실행시 전처리 오류가 나는 경우

스텝을 생성하여 연결한 뒤 테스트를 실행할 때 스텝에서 `catastrophic error: cannot open source file` 오류가 나는 경우가 있습니다.

이 오류는 해당 스텝 파일의 경로가 너무 길어서 발생합니다. Windows에서는 경로의 최대 길이를 260으로 정의하고있기 때문에, 워크스페이스와 프로젝트의 이름을 바꾸어 경로를 줄이시기 바랍니다.

## 기타

이 외에 다른 문제는 [트러블슈팅 가이드](#) 문서를 참고하시기 바랍니다.

## 기술지원

문제 발견 시 아래 기술지원 연락처로 문의하시기 바랍니다.

- [help@suresofttech.com](mailto:help@suresofttech.com)
- +82-31-606-2000

## 26. CT Target Plug-in

---

### 소개

CT Target Plug-in은 기본 CT의 기능을 보완합니다.

CT를 사용하면 자체 호스트 환경에서 테스트를 자동화할 수 있습니다. 즉, 소프트웨어 개발 환경(일반 PC)에서 테스트를 실행합니다. CT Target Plug-in을 사용하면 실제 임베디드 타겟 환경에서 CT의 테스트케이스를 실행할 수 있습니다. 이를 통해 호스트와 타겟 환경에서 테스트 결과가 같은지 쉽게 확인할 수 있고, 타겟 환경에 종속적인 요소 때문에 호스트 환경에서 실행할 수 없는 테스트를 실행할 수 있습니다.

## 26.1. 타깃 환경 설정하기

CT Target Plug-in를 사용하여 타깃 환경에서 테스트를 실행하려면 타깃 환경에 대한 정보를 입력해야 합니다.

CT Target Plug-in은 사용자가 입력한 타깃 환경에 대한 정보를 사용하여 테스트 하네스를 빌드하고 타깃 환경에서 실행한 결과를 자동으로 가져옵니다.

타깃 환경은 프로젝트 특성 페이지 또는 타깃 테스트 프로젝트 생성 마법사에서 설정할 수 있습니다.

### 타깃 환경 설정 화면

타깃 환경 설정은 기본 정보와 상세 설정으로 나뉩니다.

▶ GNU Compilers ▶ gcc ▶ 5.3 ▶ others ▶ **nodebugger**

기본 정보를 입력하면, 입력이 필요한 상세 설정이 표시됩니다.

이름	값
language	c
출제인 이름	GCC 5.3 (32bit)
상태	사용 가능한 출제인입니다.
C 컴파일러	C:\gcc\5.3.0\32bit\bin\gcc.exe
시스템 헤더(C 컴파일러)	c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include\c\gcc\5.3.0\32bit\...
LIB(C 컴파일러)	
C++ 컴파일러	C:\gcc\5.3.0\32bit\bin\g++.exe
시스템 헤더(C++ 컴파일러)	c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include\c++\c\gcc\5.3.0\3...
LIB(C++ 컴파일러)	

이름: 시스템 헤더(C++ 컴파일러) 값:

설명:

```
c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include\c++
c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include\c++\mingw32
c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include\c++\backward
c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include
c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include-fixed
c:\gcc\5.3.0\32bit\bin\lib\gcc\mingw32\5.3.0\include-fixed
```

### 타깃 환경 상세 설정

타깃 환경 상세 설정은 분석, 빌드, 실행 그리고 기타로 나뉩니다.

카테고리	설명
분석	툴체인 정보가 표시되며, 타깃 테스트 프로젝트인 경우에는 분석을 위해 필요한 타깃 컴파일러 관련 설정이 표시됩니다.
빌드	테스트 소프트웨어를 빌드하기 위한 설정이 표시됩니다.
실행	타깃 환경에서 테스트를 실행하고 결과를 가져오기 위한 설정이 표시됩니다.
기타	그 외 설정이 표시됩니다. (프로그램 entry point 등)

각 카테고리별 필수 설정은 붉은색으로 표시됩니다. 카테고리별 필수 설정의 입력 여부에 따라 테스트 실행 버튼을 클릭했을 때의 동작이 아래와 같이 달라집니다.

필수 설정 완료된 카테고리	설명
없음	테스트 실행 불가능
분석	테스트를 실행하면 원본 소스코드에 테스트 코드를 덮어씁니다. 테스트를 수행하기 위해서는 사용자가 수동으로 테스트를 빌드하고 실행해야 합니다.
분석, 빌드	테스트를 실행하면 테스트 코드를 빌드합니다. 테스트를 수행하기 위해서는 사용자가 수동으로 테스트를 타깃에서 실행해야 합니다.
분석, 빌드, 실행	타깃 환경에서 자동으로 테스트를 실행합니다.

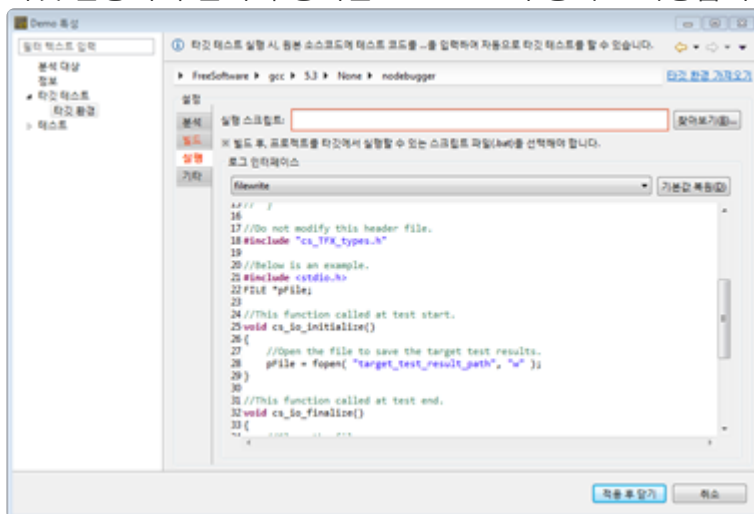
## 타깃 테스트 결과 가져오기

CT Target Plug-in은 타깃 환경에서 테스트를 실행한 결과를 로그 형태로 저장하고 가져옵니다.

디버거를 사용하지 않을 경우에는 타깃 실행 결과를 저장하기 위한 설정(로그 인터페이스)과 가져오기 위한 설정(타깃 로그 수집기 설정 - 환경 설정)이 필요합니다.

## 로그 인터페이스

로그 인터페이스는 타깃 환경에서 테스트를 실행한 결과를 저장하기 위한 설정입니다. 로그 인터페이스는 타깃 환경에서 실제 수행되는 소스 코드의 형태로 작성합니다.



## 로그 인터페이스 구조

함수	설명
<code>viod cs_io_initialize()</code>	전송을 위한 초기 함수
<code>void cs_io_finalize()</code>	전송 종료 함수
<code>void cs_io_flush()</code>	남은 데이터 전송함수
<code>void cs_io_putbyte(codescroll_byte v)</code>	1바이트 데이터 전송함수



## 타깃 로그 자동으로 가져오기

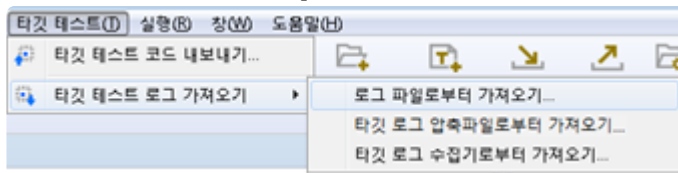
테스트 실행 결과인 타깃 로그는 타깃 로그 수집기를 통해 자동으로 가져올 수 있습니다. 타깃 로그 수집기 설정은 [타깃 로그 수집기](#) 와 [타깃 테스트 환경 설정](#) 의 [타깃 로그 수집기] 내용을 참고하시기 바랍니다.

## 타깃 로그 수동으로 가져오기

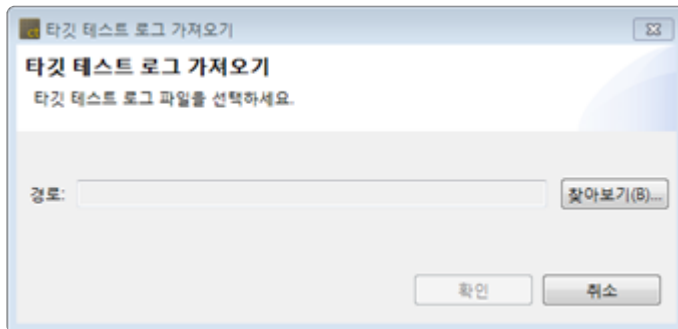
타깃 로그를 자동으로 가져올 수 없을 경우, 타깃 로그를 수동으로 가져올 수 있습니다.

### 로그 파일로부터 가져오기

1. 분석한 프로젝트를 선택한 후 전역 메뉴에 있는 [타깃 테스트] -> [타깃 테스트 로그 가져오기] -> [로그 파일로부터 가져오기...]를 클릭합니다.

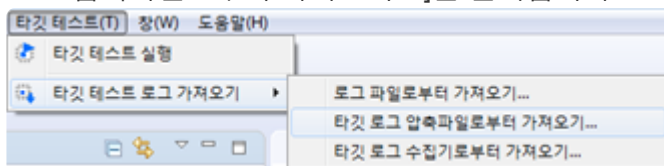


2. 타깃 테스트 로그 파일을 선택한 후 [확인] 버튼을 클릭합니다.

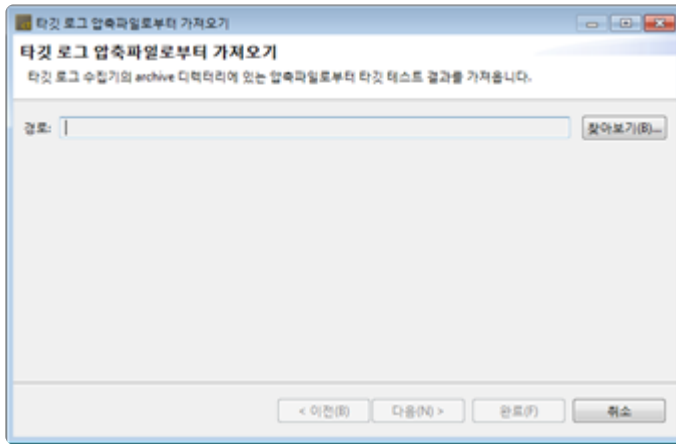


### 타깃 로그 압축파일로부터 가져오기

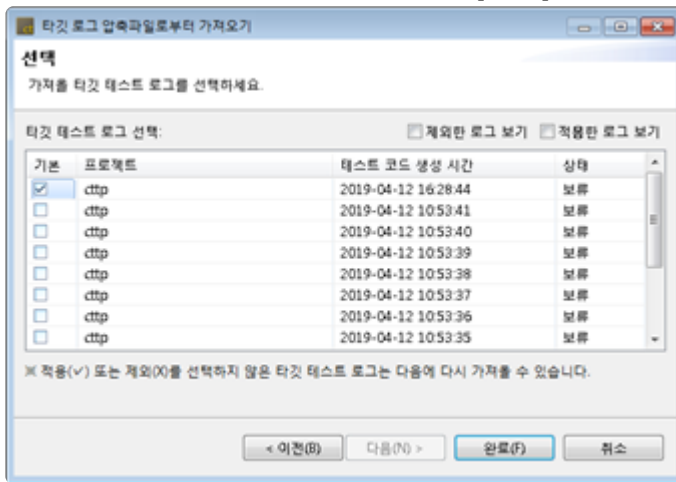
1. 분석한 프로젝트를 선택한 후 전역 메뉴에 있는 [타깃 테스트] -> [타깃 테스트 로그 가져오기] -> [타깃 로그 압축파일로부터 가져오기...]를 클릭합니다.



2. 타깃 로그 압축파일을 선택한 후 [다음] 버튼을 클릭합니다.

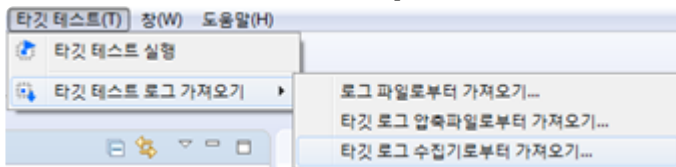


3. 가져올 타겟 테스트 로그를 체크하고 [완료] 버튼을 클릭합니다.

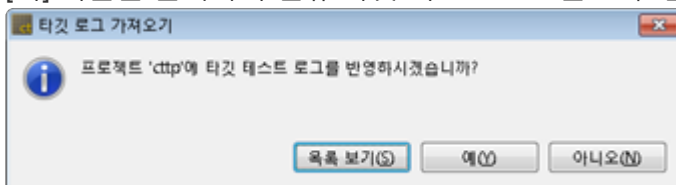


## 타겟 로그 수집기로부터 가져오기

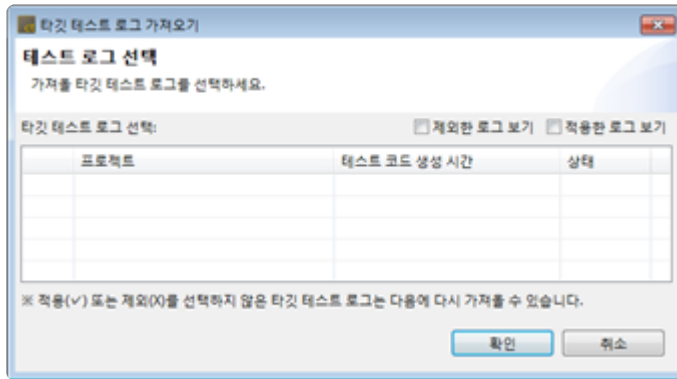
- 타겟 로그 수집기로부터 로그를 가져오기 위해서는 [환경설정] -> [타겟 테스트] -> [타겟 로그 수집기] 에서 '기본 타겟 로그 수집기 사용'을 설정해야 합니다.
- 분석한 프로젝트를 선택한 후 전역 메뉴에 있는 [타겟 테스트] -> [타겟 테스트 로그 가져오기] -> [타겟 로그 수집기로부터 가져오기...]를 클릭합니다.



3. [예] 버튼을 클릭하여 신규 타겟 테스트 로그를 모두 반영하거나, [목록 보기] 버튼을 클릭합니다.



4. [목록 보기] 버튼을 클릭한 경우, 가져올 타겟 테스트 로그를 체크하고 [확인] 버튼을 클릭합니다.



## 결과

가져오기 완료 후 커버리지 뷰에서 타깃 커버리지 정보를 확인할 수 있습니다.

	대상 함수	구분	분기	MC/DC	함수 호출	합수
1	d(signed int, signed int, signed int)	0.00% (0/...)	N/A	N/A	N/A	N
2	m(signed int, signed int)	0.00% (0/...)	N/A	N/A	N/A	N
3	max(signed int, signed int)	66.66% (...)	50.00% (...)	0.00% (0/...)	N/A	Y
합계		50.00% (3/...)	50.00% (1/...)	0.00% (0/...)	N/A	50.00%

타깃 환경에서는 Asm 코드가 포함된 함수의 커버리지를 측정할 수 없습니다.

- \* 타깃 테스트 수행 결과 로그를 가져오기 전, 호스트의 테스트 케이스, 소스 파일, 함수 정보 등 변경 되었을 경우, 타깃 테스트 수행 정보와 호스트의 테스트 정보가 유효하지 않아 [타깃 테스트 로그 가져오기] 가 실패할 수 있습니다.

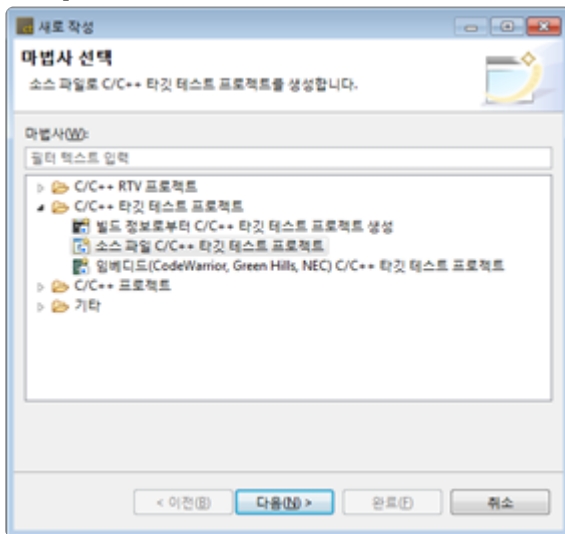
## 26.2. C/C++ 타깃 테스트 프로젝트

C/C++ 타깃 테스트 프로젝트(이하 타깃 테스트 프로젝트)는 타깃 환경 시험을 위한 프로젝트입니다. 타깃 테스트 프로젝트는 호스트 시험을 지원하지 않습니다.

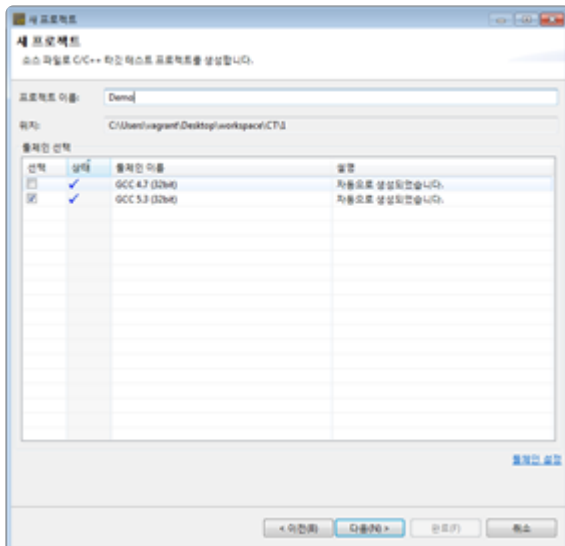
### 타깃 테스트 프로젝트 생성하기

#### 소스 파일 C/C++ 타깃 테스트 프로젝트

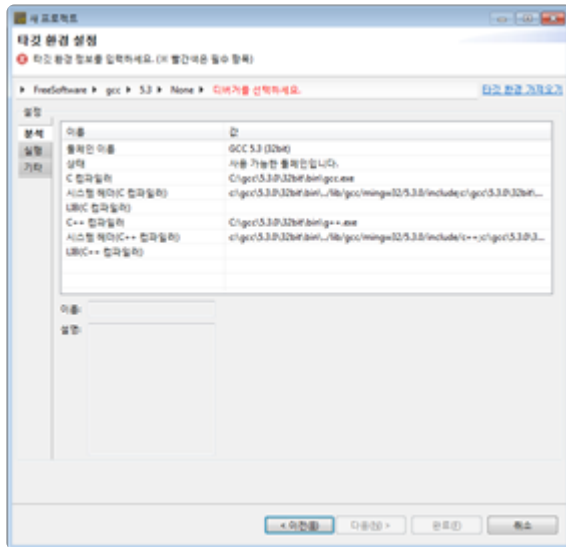
1. 전역 메뉴에 있는 [파일] -> [새로 만들기] -> [기타...]를 선택하고 [소스 파일 C/C++ 타깃 테스트 프로젝트]를 클릭하여 프로젝트를 생성합니다.



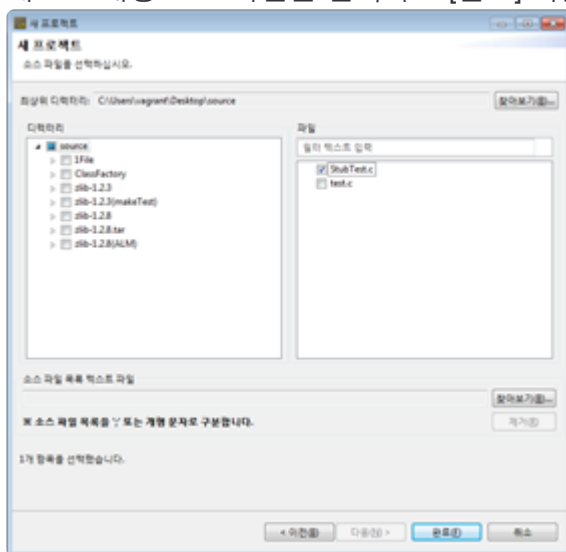
2. 프로젝트 이름을 입력하고 사용할 툴체인을 선택합니다.



3. 타깃 환경의 기본 정보를 선택한 후 각 카테고리별 상세 설정을 입력한 후 [다음] 버튼을 클릭합니다.

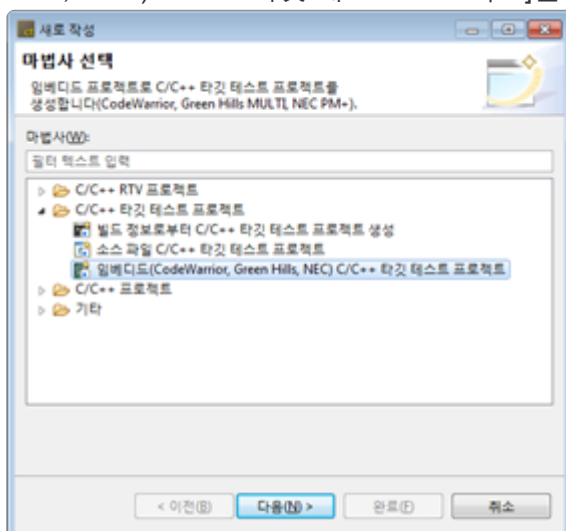


4. 테스트 대상 소스 파일을 선택하고 [완료] 버튼을 클릭합니다.

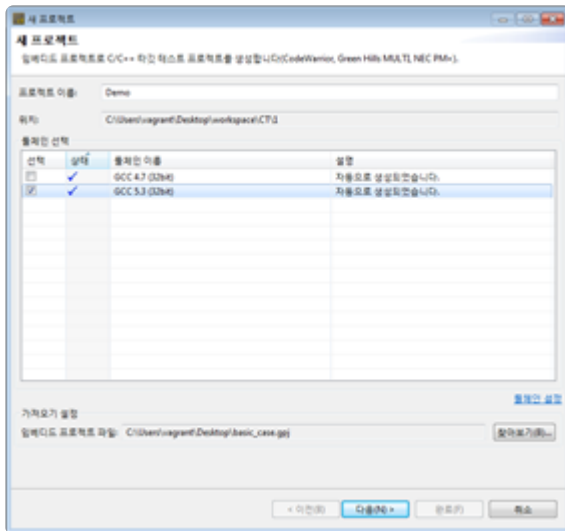


## 임베디드(CodeWarrior, Green Hills, NEC) C/C++ 타겟 테스트 프로젝트

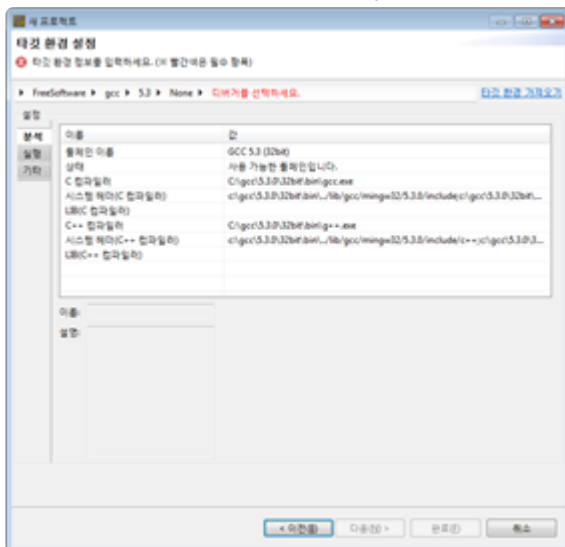
1. 전역 메뉴에 있는 [파일] -> [새로 만들기] -> [기타...]를 선택하고 [임베디드(CodeWarrior, Green Hills, NEC) C/C++ 타겟 테스트 프로젝트]를 클릭하여 프로젝트를 생성합니다.



- 프로젝트 이름을 작성하고 사용할 툴체인을 선택합니다. 툴체인을 선택하고 난 후, 테스트할 임베디드 프로젝트 파일을 가져옵니다. 현재 CT 2023.12에서 지원하는 임베디드 프로젝트는 CodeWarrior, Green Hills, NEC 입니다.

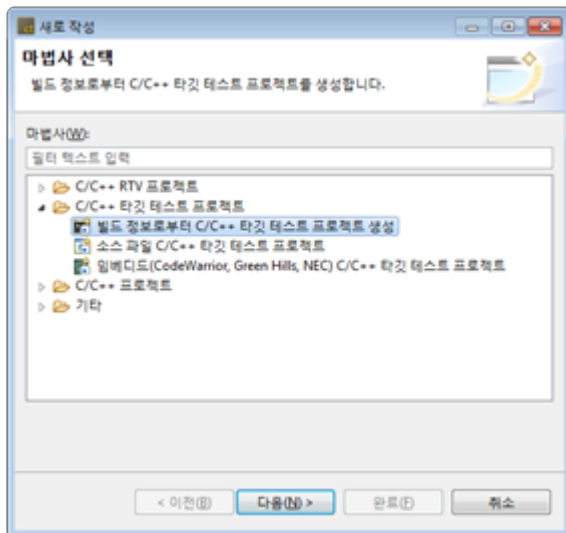


- 타겟 환경의 기본 정보를 선택한 후 각 카테고리별 상세 설정을 입력합니다.

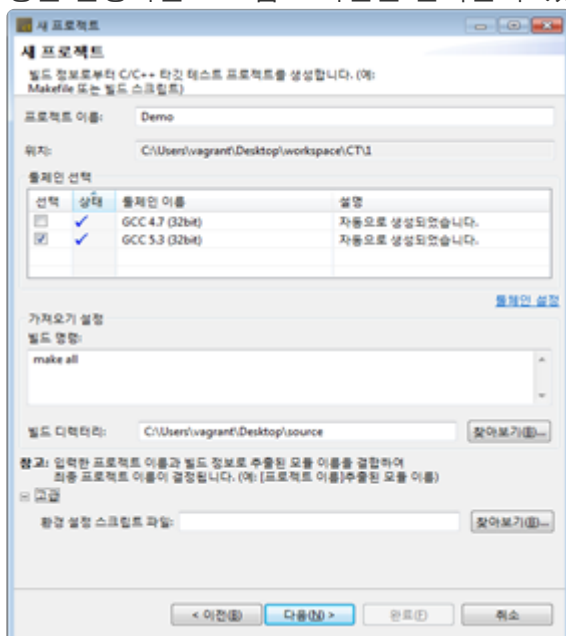


## 빌드 정보로부터 C/C++ 타겟 테스트 프로젝트 생성

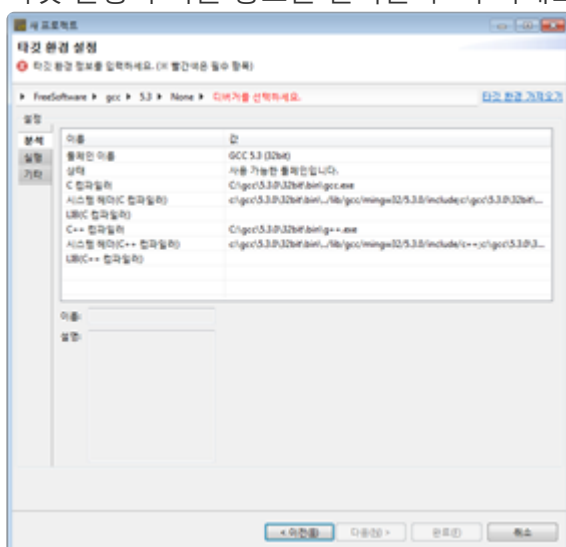
- 전역 메뉴에 있는 [파일] -> [새로 만들기] -> [기타...]를 선택하고 [빌드 정보로부터 C/C++ 타겟 테스트 프로젝트 생성]을 클릭하여 프로젝트를 생성합니다.



- 프로젝트 이름을 작성하고 사용할 툴체인을 선택합니다. 툴체인을 선택하고 난 후, 작성된 Makefile 및 빌드 스크립트 경로를 빌드 디렉터리로 지정하고 빌드 명령을 작성합니다. [고급] 메뉴에서 빌드 환경을 설정하는 스크립트 파일을 선택할 수 있습니다.



- 타겟 환경의 기본 정보를 선택한 후 각 카테고리별 상세 설정을 입력합니다.





컴파일러 경로가 유효하지 않은 툴체인은 타깃 테스트 프로젝트 생성 시 선택할 수 없습니다.



## 26.3. 타깃 테스트 실행 설정하기

타깃 테스트 실행 관련 설정은 [프로젝트] -> [특성]의 타깃 테스트 페이지에서 확인할 수 있습니다.

### 커버리지 종류

타깃 테스트 수행 시 측정할 커버리지 종류(구문, 분기, MC/DC)를 선택할 수 있습니다.

### 커버리지 측정 대상

테스트 대상 함수 외에 추가로 커버리지를 측정할 함수를 선택할 수 있습니다.

테스트 대상 함수가 호출하는 모든 함수	테스트 대상 함수가 호출하는 모든 함수에 대해 커버리지를 측정합니다.
이전 테스트 실행 시 호출된 함수	이전 테스트 수행 시 커버리지가 측정된 함수들을 포함합니다.

### 테스트 케이스 실행

테스트 케이스 실행 방식을 선택할 수 있습니다.

한 번에 실행하기	모든 테스트케이스를 한 번에 타깃에 로드하고 실행합니다. 테스트케이스 실행이 이전 테스트케이스 실행 결과에 영향을 받습니다.
각각 실행하기	테스트케이스 하나씩 타깃에 로드하고 실행하기를 반복합니다. 한 번에 실행하는 것보다 시간이 오래 걸리지만, 적은 메모리를 사용합니다.

## 26.4. 타깃 로그 수집기

타깃 로그 수집기는 타깃 테스트 결과를 수집하여 CT 2023.12로 전송하는 서버입니다. 타깃 테스트 결과 수집 방법은 크게 통신과 파일 스캔으로 나뉘며, 지원하는 통신 프로토콜과 스캔 가능한 파일 형식은 다음과 같습니다.

통신 프로토콜	TCP, UDP, UART
파일 형식	타깃 로그(text), 메모리 덤프(Hex, Intel HEX)

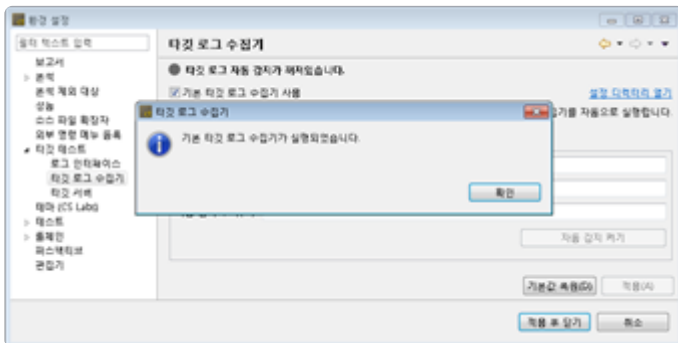
### 설치

CT 2023.12 설치 시 타깃 로그 수집기가 자동으로 설치됩니다.

### 실행

[환경 설정] -> [타깃 테스트] -> [타깃 로그 수집기]에서 [기본 타깃 로그 수집기 사용]을 체크하면 타깃 로그 수집기가 타깃 테스트 결과 수집을 시작합니다.

TargetLogCollector.zip을 압축 해제하여 사용할 경우, 압축 해제 디렉터리안에 TargetLogCollector.exe를 windows 명령프롬프트에서 실행합니다.



### 설정

타깃 로그 수집기 설치 경로에 'settings.ini' 파일을 통해 설정할 수 있으며, 옵션은 다음과 같습니다.

<b>[LogReceiveServer]</b>	타깃 테스트 결과 수신 관련 설정
<b>port</b>	TCP, UDP 통신 포트
<b>protocol</b>	타깃 테스트 결과 수신을 위한 통신 프로토콜
<b>timeout</b>	연결 타임아웃
<b>lastString</b>	데이터의 마지막을 나타내는 문자열
<b>serialPort</b>	Serial 통신 포트 (Windows: COM#, Linux: /dev/ttyS#)
<b>baudRate</b>	Serial 통신 설정

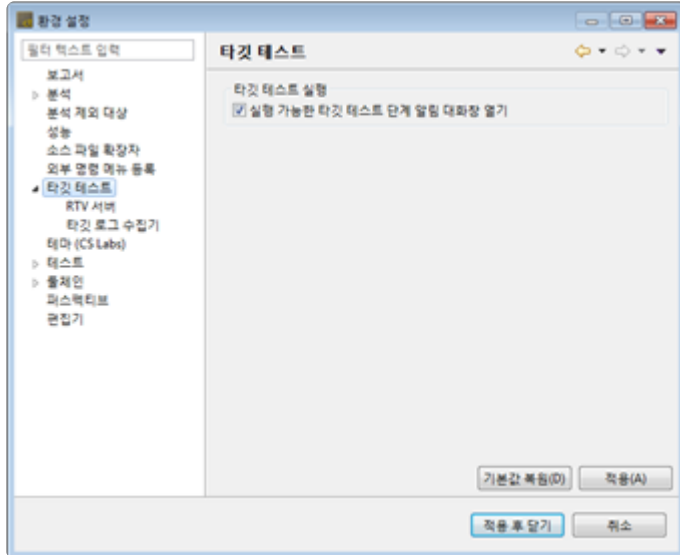
<b>dataBits</b>	Serial 통신 설정
<b>stopBits</b>	Serial 통신 설정
<b>parity</b>	Serial 통신 설정
<b>flowControl</b>	Serial 통신 설정
<b>[ScanLog]</b>	파일 스캔 관련 설정
<b>dir</b>	스캔 대상 디렉터리
<b>fileExtension</b>	스캔 대상 파일 확장자(공백일 경우 모든 파일 스캔)
<b>[LogSendServer]</b>	CT 2023.12와 통신 관련 설정
<b>port</b>	타깃 로그 파일 전송 포트

- \* **timeout:** 마지막 수신으로부터 지정한 시간(초) 동안 데이터 전송이 없는 경우, 수신한 데이터를 저장합니다.
- dir:** 지정한 디렉터리에 타깃 로그 파일(text, hex)을 입력하면 타깃 로그 수집기가 인식합니다.

## 26.5. 환경 설정

### 타깃 테스트

타깃 테스트 실행 시 실행 가능한 타깃 테스트 단계 알림 대화창 열기 여부를 설정할 수 있습니다.



### 타깃 로그 수집기

사용할 타깃 로그 수집기 정보와 타깃 테스트 결과 자동 감지 여부를 설정합니다. [자동 감지 켜기] 버튼을 클릭하여 타깃 로그 자동 감지를 켤 경우, 지정한 자동 감지 주기마다 타깃 로그 수집기로부터 새로운 타깃 테스트 결과를 받아옵니다.



타깃 로그 수집기 설정	설명
자동 감지 주기(초)	새로운 타깃 테스트 결과 확인 주기
아이피(IP)	타깃 로그 수집기가 실행 중인 서버 또는 PC의 아이피
포트(Port)	타깃 로그 수집기의 통신 포트

! 기본 타깃 로그 수집기를 사용할 경우, 자동 감지 주기(초)만 설정합니다.

## 27. CT RTV(Remote Target Verifier)

---

### CT RTV 소개

CT RTV는 리눅스 환경에서 자동화된 테스트를 수행할 수 있도록 도와줍니다.

리눅스 환경에 RTV 서버를 설치하고 기본 설정을 마치면, RTV 클라이언트를 사용하여 호스트 환경에서 테스트를 수행하는 것처럼 쉽게 RTV 테스트를 수행할 수 있습니다.

리눅스 환경에 RTV 서버를 설치하는 방법은 본 문서 [문제 해결](#) 페이지 하단에 있는 기술지원 연락처를 통해 문의해주시기 바랍니다.

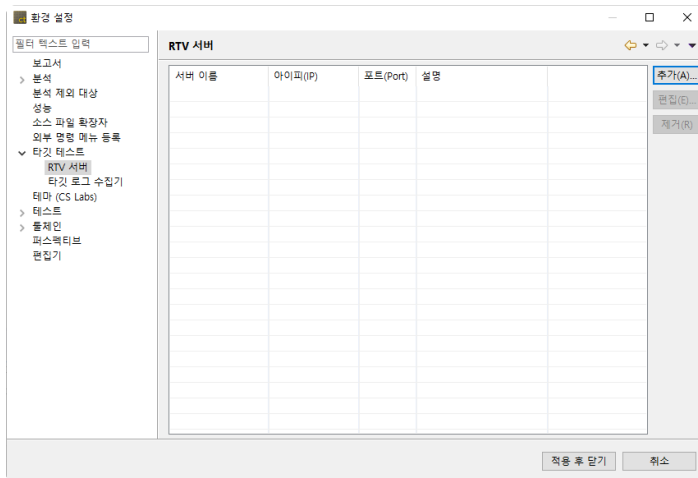
### RTV 지원 OS

서버: Linux (RHEL, Ubuntu, Debian, Fedora 기반, 그 외 문의)

클라이언트: Windows 7 / 10 / 11

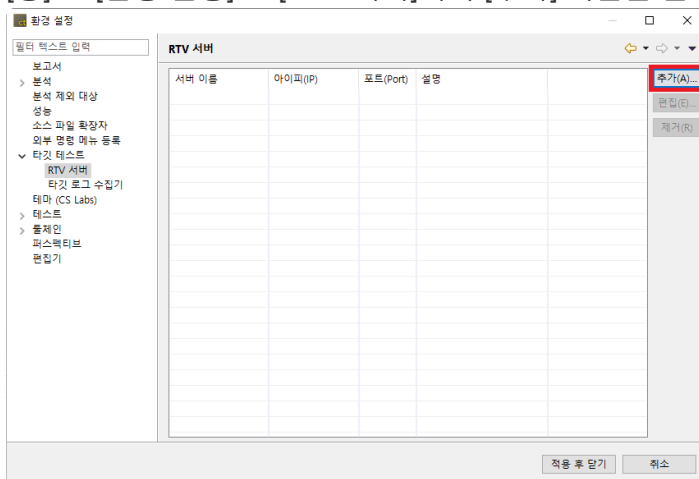
## 27.1. RTV 서버 설정

CT RTV의 기능을 사용하기 위해서는 RTV 서버가 설치된 RTV 서버를 추가해야 합니다.  
RTV 서버는 [창] -> [환경 설정] -> [RTV 서버]에서 추가할 수 있습니다.



### RTV 서버 추가

1. [창] -> [환경 설정] -> [RTV 서버]에서 [추가] 버튼을 클릭합니다.



2. [RTV 서버 추가] 창에서 RTV 서버 정보를 입력합니다.

RTV 서버 추가

RTV 서버 추가

RTV 서버 정보를 입력하세요.

서버 이름:

아이피(IP):

포트(Port):

설명:

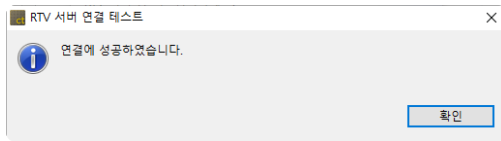
연결 테스트(C)

확인

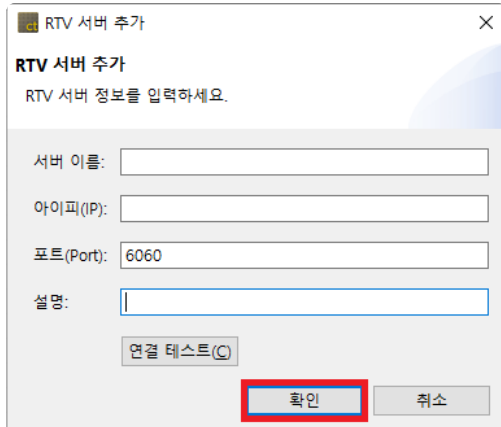
취소

! 서버 이름은 Windows의 파일 이름으로 사용할 수 있는 문자만 허용하며, RTV 서버를 추가한 뒤에 수정할 수 없습니다.

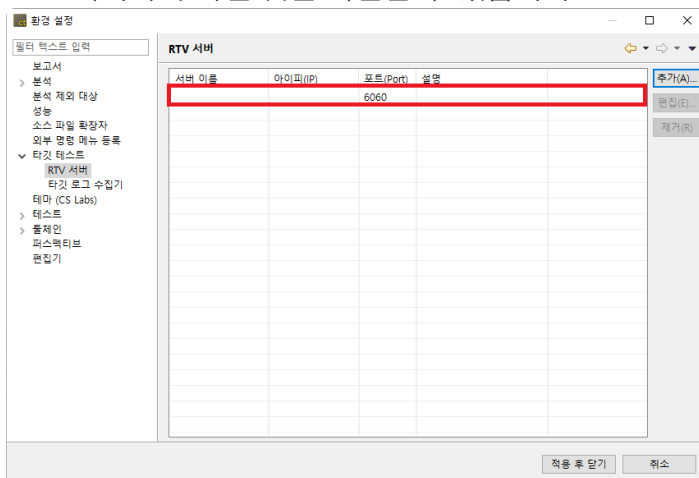
3. [연결 테스트] 버튼을 클릭하여 입력한 RTV 서버와의 연결 상태를 확인합니다.



4. RTV 서버와 연결 테스트에 성공했다면, [확인] 버튼을 클릭하여 RTV 서버를 추가합니다.

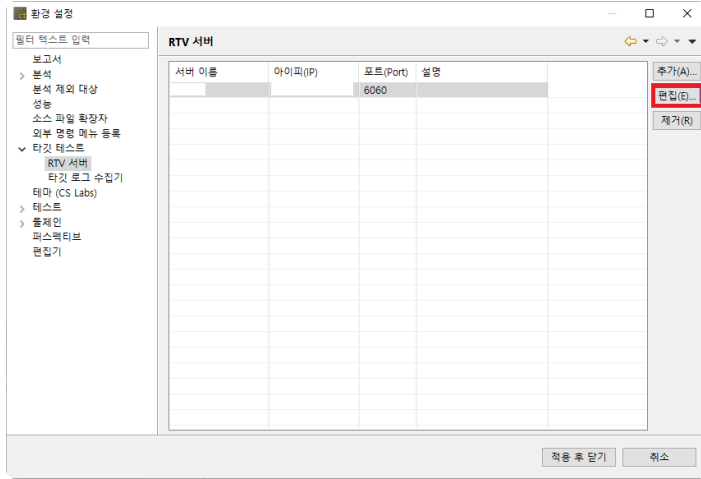


5. RTV 서버가 추가된 것을 확인할 수 있습니다.

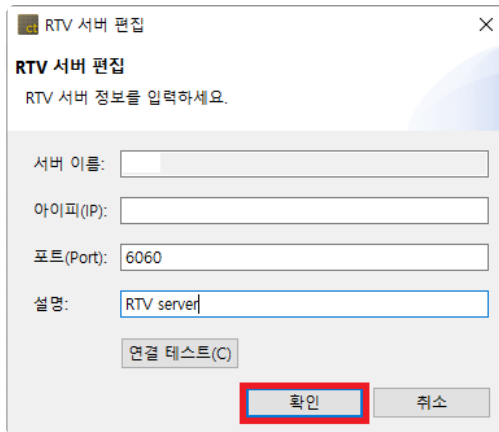


## RTV 서버 편집

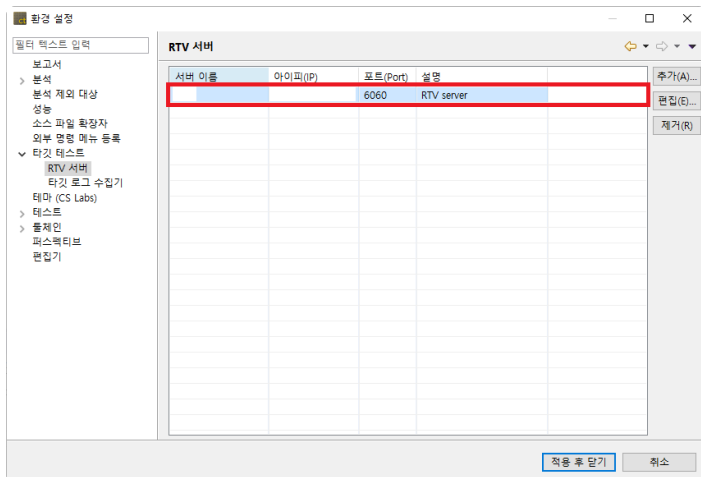
1. RTV 서버 목록에서 서버를 더블 클릭하거나, 서버를 선택한 뒤에 [편집] 버튼을 클릭합니다.



2. [RTV 서버 편집] 창에서 서버 정보를 편집한 뒤에, [확인]을 클릭합니다.



3. RTV 서버 정보가 편집된 것을 확인할 수 있습니다.



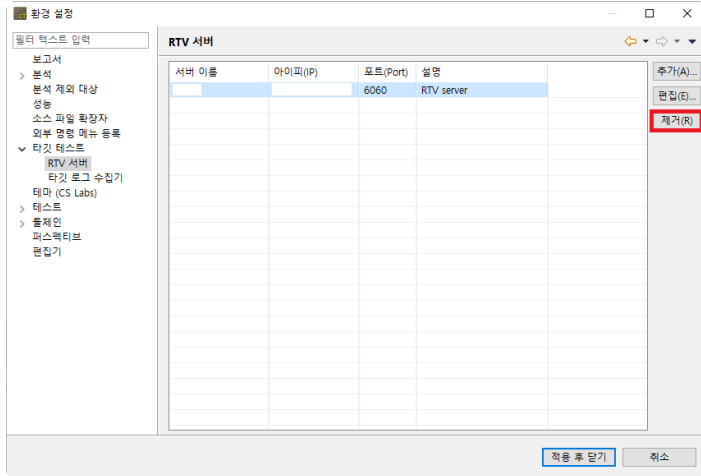
## RTV 서버 제거

RTV 서버를 더 이상 사용하지 않을 경우, 제거할 수 있습니다.

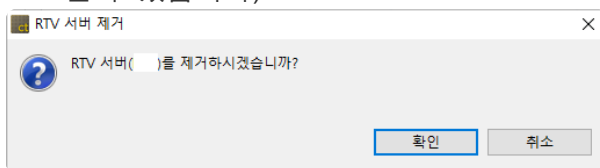
RTV 서버를 제거하면, RTV 서버와 연관된 틀체인 또는 프로젝트의 리소스가 클라이언트 환경에서 삭제됩니다(해당 RTV 서버로부터 추가한 RTV 틀체인이 존재하면 삭제할 수 없음).

1. RTV 서버 목록에서 서버를 선택한 뒤에 [제거] 버튼을 클릭합니다.

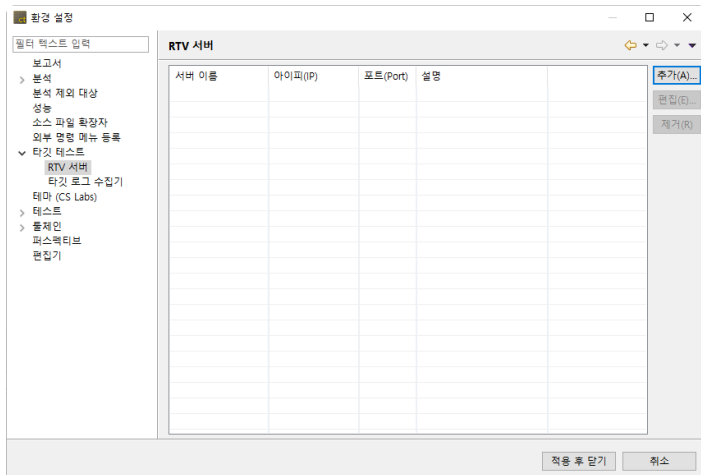




2. [RTV 서버 제거] 창에서 [확인] 버튼을 클릭합니다(RTV 서버 리소스 크기에 따라 1~2분 정도 시간이 소요될 수 있습니다).



3. RTV 서버가 제거된 것을 확인할 수 있습니다.



## 27.2. RTV 툴체인 설정

### RTV 툴체인

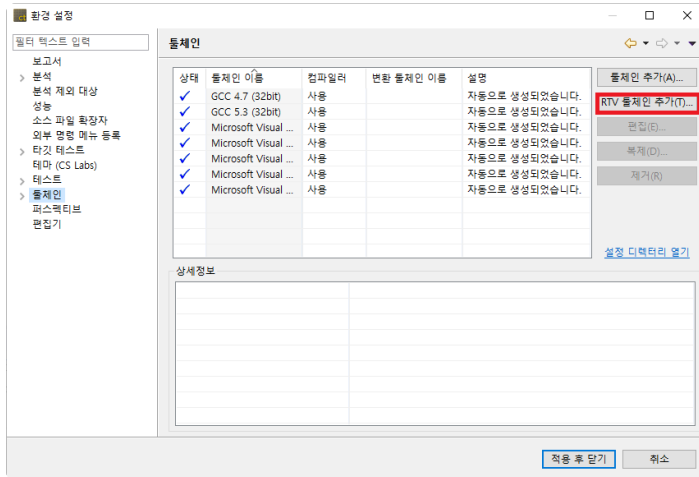
CT RTV로 RTV 환경에서 테스트를 빌드하고 실행하려면, 테스트 대상 소스 파일의 툴체인(컴파일러) 정보를 설정해야 합니다.

이 장에서는 RTV 서버가 설치된 RTV 환경에서 미리 추출해놓은 RTV 툴체인 정보를 클라이언트에서 추가, 편집, 삭제하는 방법을 설명합니다.

(RTV 환경에서 툴체인 정보를 추출하는 방법은 본 문서 [문제 해결](#) 페이지 하단에 있는 기술지원 연락처를 통해 문의해주시기 바랍니다.)

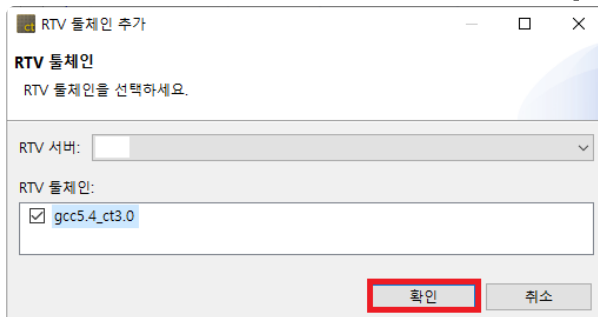
### RTV 툴체인 추가

1. [창] -> [환경 설정] -> [툴체인] 창에서 [RTV 툴체인 추가] 버튼을 클릭합니다.

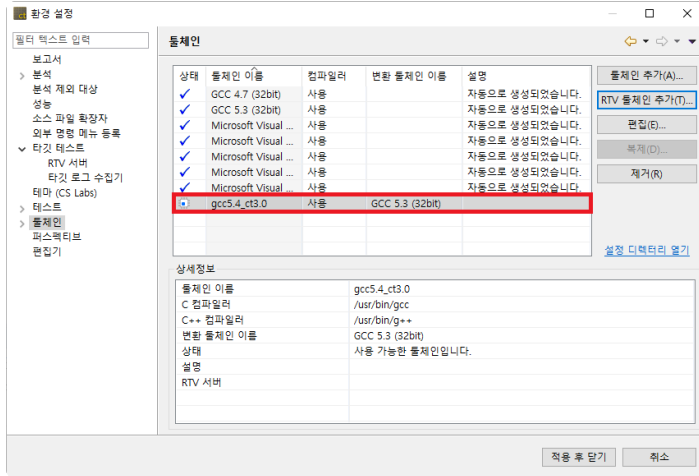


✿ RTV 서버가 없다면, [RTV 서버 추가] 창이 먼저 나타납니다.

2. [RTV 툴체인 추가] 창에서 RTV 서버를 선택하면 서버에 추출되어 있는 RTV 툴체인 목록을 확인할 수 있습니다. 추가하려는 툴체인을 선택한 뒤에 [확인] 버튼을 클릭합니다.



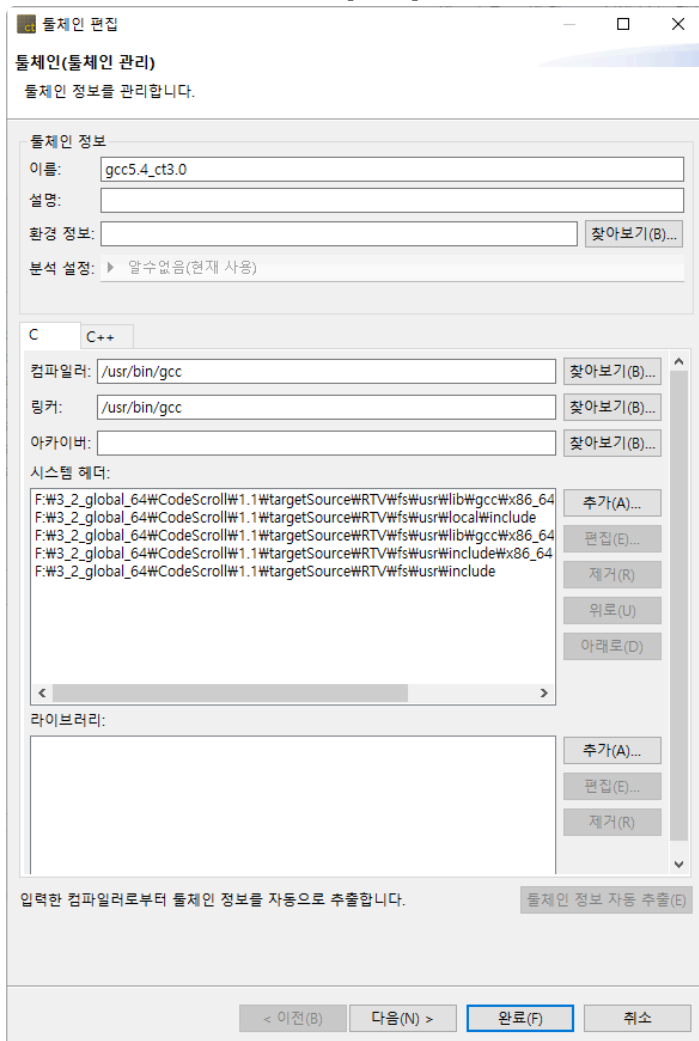
3. RTV 서버로부터 툴체인 정보를 가져옵니다(시스템 헤더 크기에 따라 1~2분 정도 시간이 소요될 수 있습니다). 작업이 완료되면 RTV 툴체인이 추가된 것을 확인할 수 있습니다.



4. 추가된 RTV 툴체인을 선택하면 상세정보 화면에서 RTV 서버의 정보가 표시되는 것을 확인할 수 있습니다.

## RTV 툴체인 편집

1. 편집할 툴체인을 선택하고 [편집] 버튼을 클릭합니다.



2. RTV 툴체인 정보를 편집합니다.  
3. [완료] 버튼을 클릭합니다.

## RTV 톨체인 복제

RTV 톨체인은 복제 기능을 제공하지 않습니다.

## RTV 톨체인 삭제

삭제할 톨체인을 선택한 후 [삭제] 버튼을 클릭합니다.



삭제하려는 톨체인과 연관된 프로젝트가 있다면, 삭제 이후에 해당 프로젝트를 정상적으로 사용할 수 없습니다. 프로젝트의 모듈 특성 페이지에서 톨체인을 다시 설정해야 합니다.

## RTV 톨체인 가져오기/내보내기

RTV 톨체인은 RTV 서버를 통해서 추가할 수 있기 때문에 가져오기/내보내기 기능은 제공하지 않습니다.

## 27.3. RTV 프로젝트 만들기

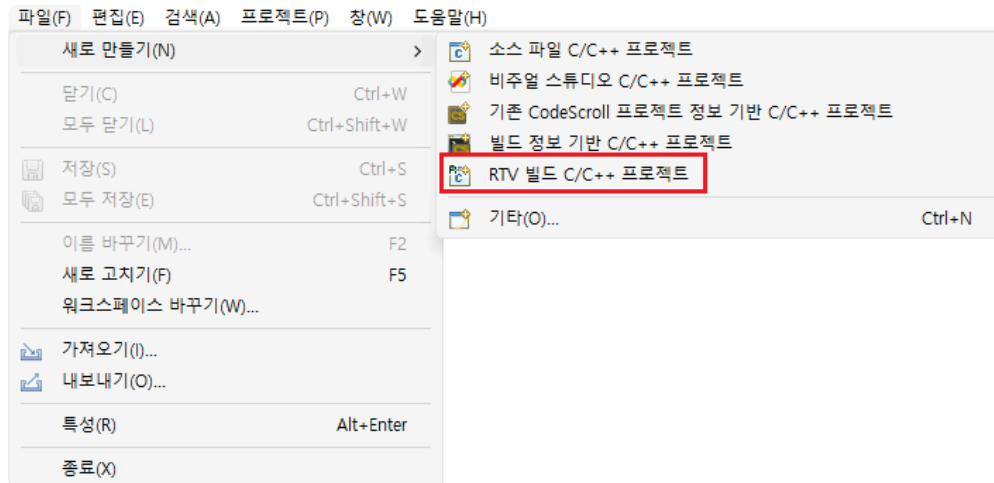
RTV 환경에 존재하는 소스 파일을 테스트하기 위해 RTV 프로젝트를 생성합니다.

RTV 환경에서 소스 파일을 빌드할 때, RTV 프로젝트 생성에 필요한 정보를 추출할 수 있습니다.

RTV 환경에 미리 추출해놓은 소스 파일 빌드 정보(RTV 서버 유틸리티 사용)를 이용하거나, 대상 소스 파일을 빌드하는 동시에 정보를 추출하여 RTV 프로젝트를 생성하는 방법을 설명합니다.

RTV 프로젝트를 생성하려면 [프로젝트 생성 마법사]를 실행합니다.

[파일] -> [새로 만들기] 또는 단축 아이콘을 클릭합니다.



### RTV 빌드 C/C++ 프로젝트

RTV 환경에 미리 추출해놓은 소스 파일 빌드 정보(RTV 서버 유틸리티 사용)를 이용하여 RTV 프로젝트를 생성합니다.

1. [프로젝트 이름]에 프로젝트 이름을 입력합니다.

**새 프로젝트**

새 프로젝트

RTV 서버로부터 새 프로젝트를 생성합니다.

프로젝트 이름:

위치:

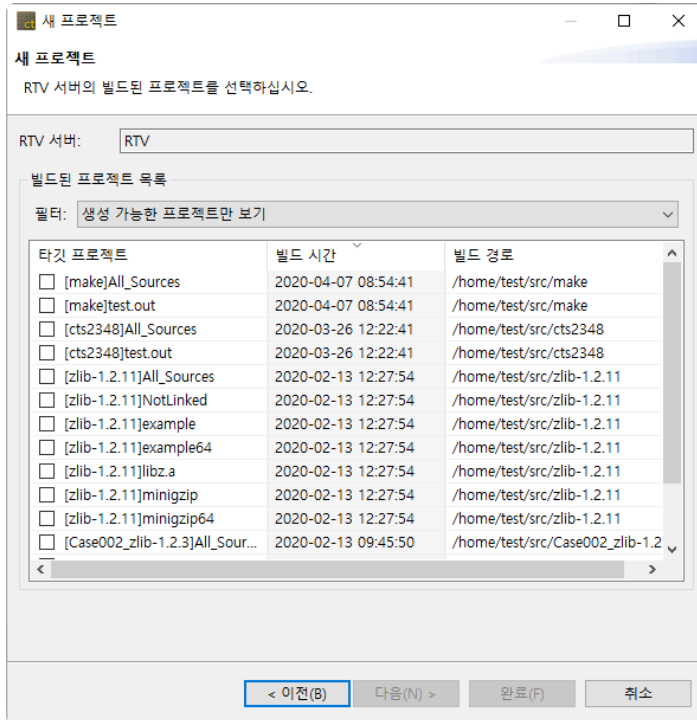
플래그인 선택

기본	플래그인 이름	RTV 서버	설명
<input type="checkbox"/>	gcc5.4_ct3.0	RTV	
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			

[플래그인 설정](#)

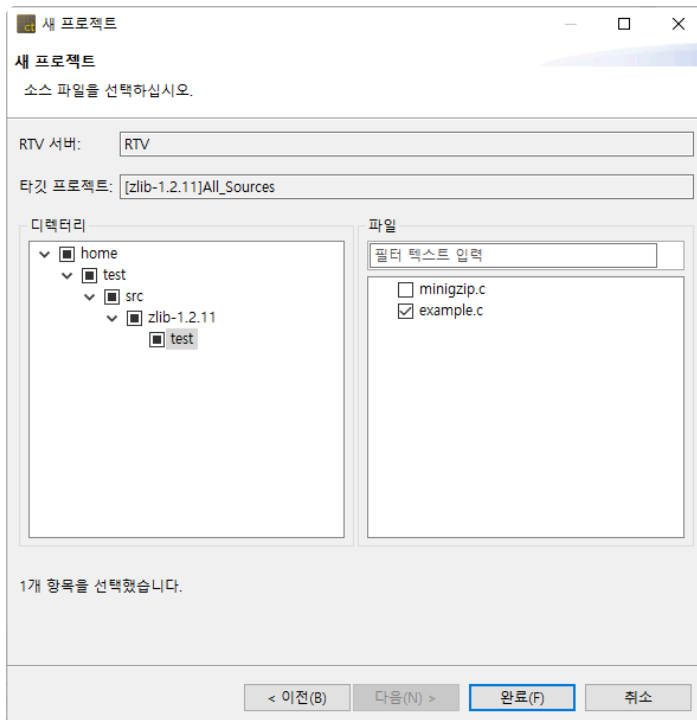
< 이전(B) 다음(N) > 완료(F) 취소

2. 툴체인 목록에 RTV 툴체인 목록이 나타납니다. 사용할 RTV 툴체인을 선택합니다.
3. 툴체인을 선택한 뒤에 [다음] 버튼을 클릭합니다. 선택한 툴체인을 가져온 RTV 서버에 미리 추출되어 있는 소스 파일 빌드 목록을 확인할 수 있습니다.



✿ 생성된 프로젝트 중 'All\_Sources\_프로젝트명'은 ctbuild를 통해 빌드 캡처 된 모듈의 모든 소스파일을 합친 프로젝트 입니다. 이 프로젝트는 빌드 여부를 고려하지 않고 합쳐진 프로젝트이기 때문에 정상적으로 테스트 수행이 되지 않을 수 있습니다.

4. 빌드된 목록에서 RTV 프로젝트를 생성할 항목을 선택한 뒤에 [다음] 버튼을 클릭합니다.



5. RTV 프로젝트를 생성할 때 포함할 소스 파일(테스트 대상)을 선택한 뒤에 [완료] 버튼을 클릭하여 프로젝트를 생성합니다.

## RTV 빌드 명령어 C/C++ 프로젝트

테스트 대상 소스 파일을 빌드하는 동시에 정보를 추출하여 RTV 프로젝트를 생성합니다. 빌드 명령어를 입력하여 RTV 서버에서 소스 파일을 빌드하면서 RTV 프로젝트를 생성합니다.

1. [프로젝트 이름]에 프로젝트 이름을 입력합니다.

새 프로젝트

새 프로젝트

RTV 서버로부터 새 프로젝트를 생성합니다.

프로젝트 이름:

위치:

툴체인 선택

기본	툴체인 이름	RTV 서버	설명
<input type="checkbox"/>	gcc5.4_ct3.0	RTV	

[툴체인 설정](#)

< 이전(B)   다음(N) >   완료(F)   취소

2. 툴체인 목록에 RTV 툴체인 목록이 나타납니다. 사용할 툴체인을 선택합니다.
3. 툴체인 선택한 뒤에 [다음] 버튼을 클릭합니다. 선택한 툴체인을 가져온 RTV 서버에서 빌드를 수행하게 됩니다.

새 프로젝트

새 프로젝트

빌드 명령어를 입력하십시오.

RTV 서버:

빌드 디렉터리:

빌드 명령:

빌드 로그

☒ 기본 위치 사용

위치:  [찾아보기\(B\)...](#)

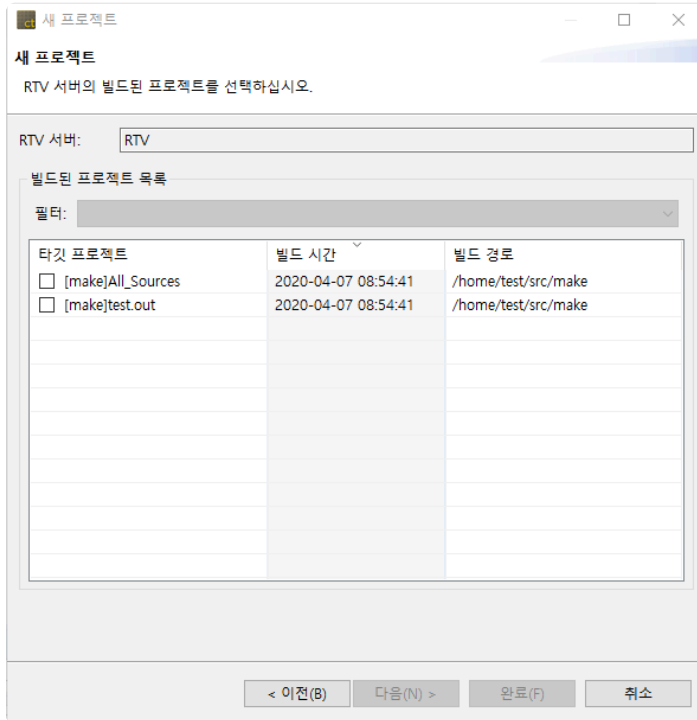
[디렉터리 열기](#)

< 이전(B)   다음(N) >   완료(F)   취소

4. [빌드 디렉터리]에 RTV 서버에서 빌드를 수행할 경로(예: make파일이 있는 디렉터리)를 입력하고, 빌

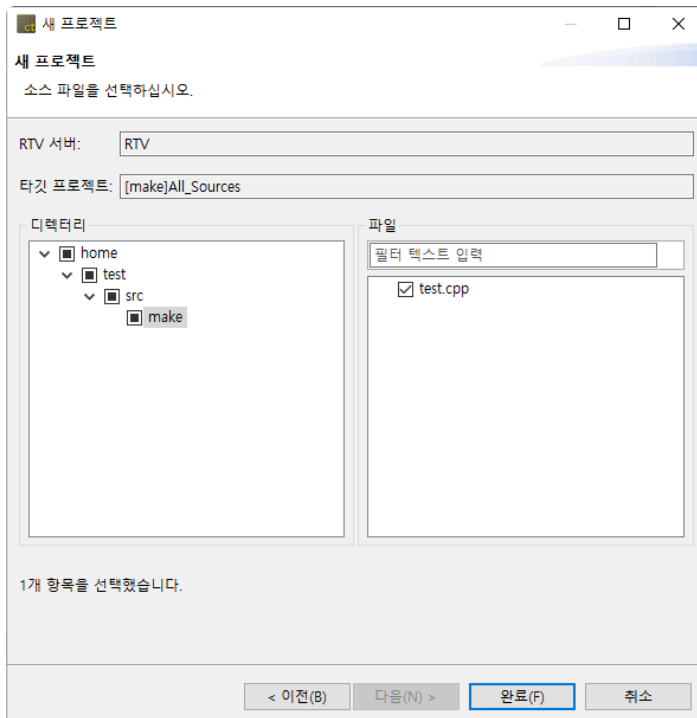
드 명령어를 입력합니다.

5. [빌드 로그]에 빌드 결과 파일을 생성할 경로를 지정합니다.
6. 빌드에 필요한 정보를 입력한 뒤에 [다음] 버튼을 클릭합니다. 입력된 빌드 명령어로 RTV 서버에서 소스 파일을 빌드합니다(빌드 대상에 따라 많은 시간이 소요될 수 있습니다).
7. 빌드가 완료되면, 빌드된 프로젝트 목록이 나타납니다.



생성된 프로젝트 중 'All\_Sources\_프로젝트명'은 ctbuild를 통해 빌드 캡처 된 모듈의 모든 소스파일을 합친 프로젝트 입니다. 이 프로젝트는 빌드 여부를 고려하지 않고 합쳐진 프로젝트이기 때문에 정상적으로 테스트 수행이 되지 않을 수 있습니다.

8. 빌드된 목록에서 RTV 프로젝트를 생성할 항목을 선택한 뒤에 [다음] 버튼을 클릭합니다.





9. RTV 프로젝트를 생성할 때 포함할 소스 파일(테스트 대상)을 선택한 뒤에 [완료] 버튼을 클릭하여 프로젝트를 생성합니다.

## RTV 빌드 명령어

빌드 정보로부터 프로젝트를 생성하려면, 빌드 명령어에 `ctbuild` 명령어를 사용해야 합니다.

### ctbuild 명령어

`ctbuild`는 소프트웨어의 빌드를 수행할 때, 빌드되는 소프트웨어의 프로젝트 구성 정보를 자동으로 추출하여 CT 2023.12의 프로젝트를 구성해주는 유틸리티 프로그램입니다. Shell에서 `make`와 같은 빌드 시스템을 이용하는 경우, 프로젝트를 자동으로 설정합니다.

### ctbuild 사용법

기존의 빌드 명령어 앞에 `ctbuild` 명령어를 붙입니다.

- 예) `make` → `ctbuild capture make`

### ctbuild 옵션

**new** : 빌드 명령어 추출 전 필요한 정보를 생성합니다. `--toolchain` 옵션으로 툴체인을 지정하고 `--project` 옵션으로 프로젝트 명을 지정합니다.

- 예) `ctbuild new --toolchain=gcc5.4 --project=PRJ`

**!** `--toolchain` 옵션은 필수 입력 사항입니다.

**capture**: 빌드 명령어 추출을 수행하여 RTV 프로젝트를 생성합니다.

- 예) `ctbuild capture make`

**convert**: STATIC 프로젝트를 RTV 프로젝트로 변환합니다.

- 예) `ctbuild convert`

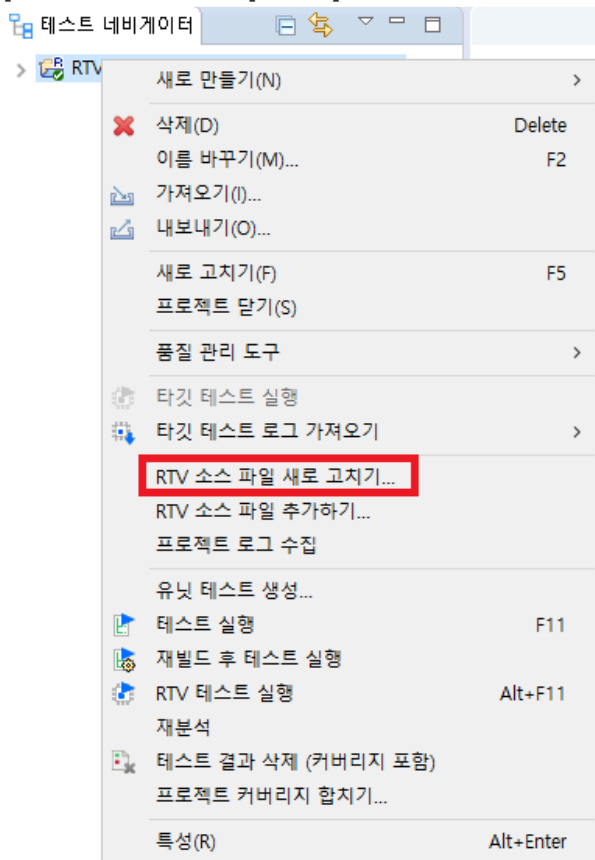
## 27.4. RTV 소스 파일 새로 고치기/추가하기

### RTV 소스 파일 새로 고치기

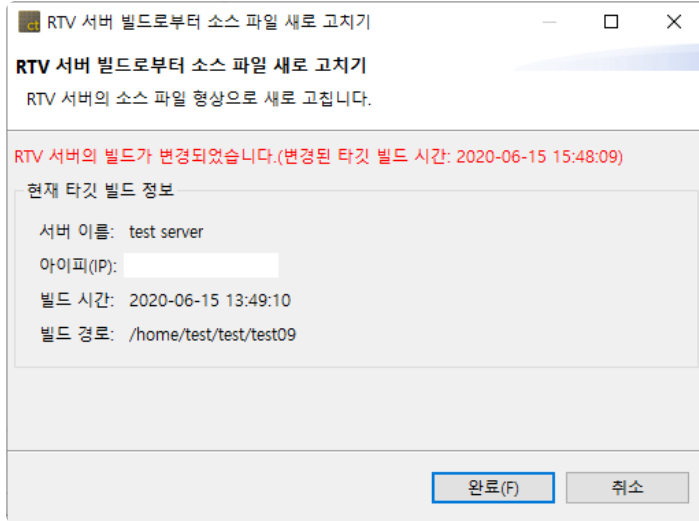
RTV 서버에서 소스 파일이 수정되고 빌드된 경우, [RTV 소스 파일 새로 고치기] 기능을 통해, 수정된 내용을 RTV 프로젝트에 반영할 수 있습니다.

! 소스 파일이 수정되었지만, 다시 빌드하지 않은 경우에는 변경된 내용이 반영되지 않습니다.  
RTV 서버의 소스 파일을 수정 후, 빌드하는 경우에는 풀 빌드를 수행해야 합니다.

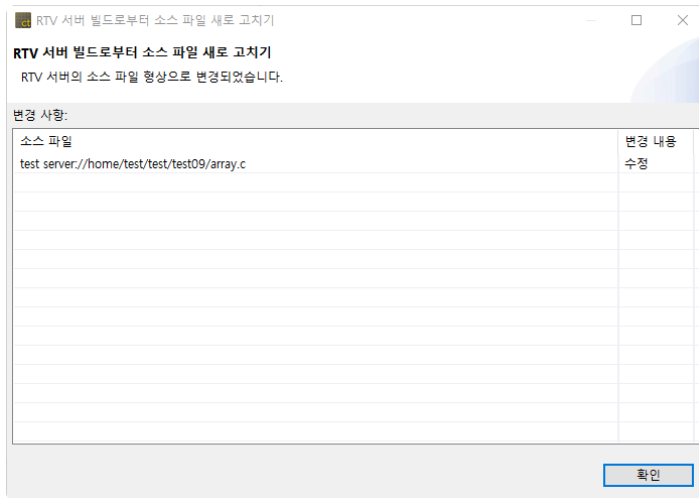
1. [테스트 네비게이터] 뷰의 [RTV 소스 파일 새로 고치기...] 컨텍스트 메뉴를 클릭합니다.



2. RTV 프로젝트를 생성할 때 선택한 소스 파일에 대한 최신 빌드 정보를 확인할 수 있습니다(빌드가 변경된 경우에 변경 알림 메시지 표시). [완료] 버튼을 클릭하여 변경사항을 확인합니다.



3. 변경사항을 확인한 뒤에 [확인] 버튼을 클릭합니다.

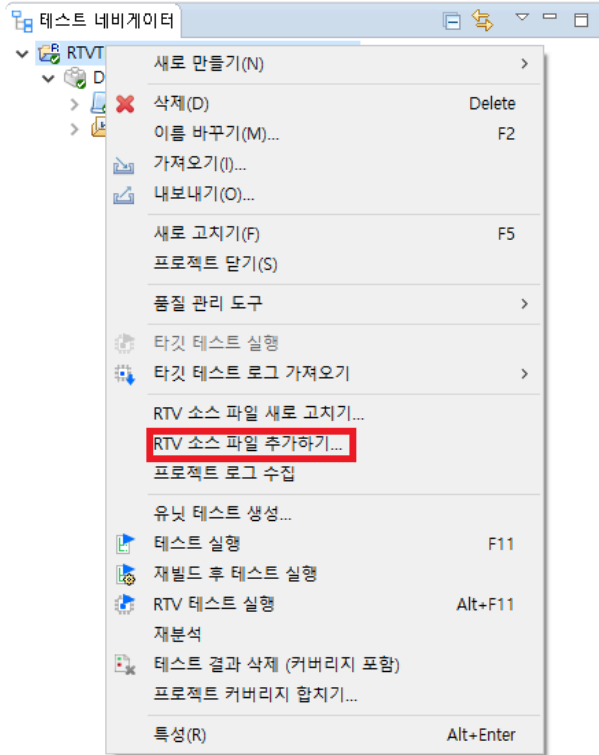


## 소스 파일 추가하기

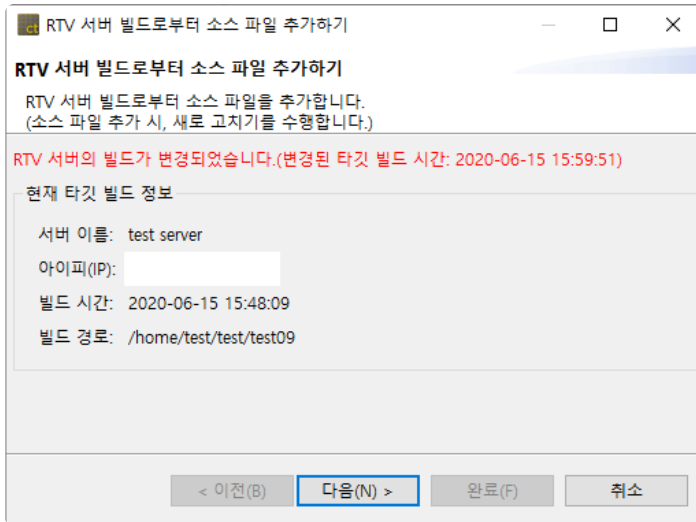
RTV 서버에서 새로운 소스 파일이 추가되고 빌드된 경우 또는 RTV 프로젝트를 생성할 때 일부 소스 파일만 선택한 경우에 [RTV 소스 파일 추가하기] 기능을 통해, 소스 파일을 RTV 프로젝트에 추가할 수 있습니다.

! 소스 파일이 추가되었지만, 다시 빌드하지 않은 경우에는 변경된 내용이 반영되지 않습니다.  
RTV 서버의 소스 파일을 추가 후, 빌드하는 경우에는 풀 빌드를 수행해야 합니다.

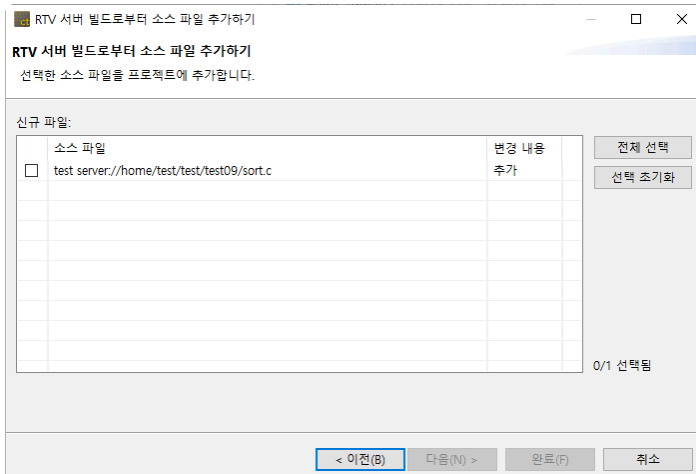
1. [테스트 네비게이터] 뷰의 [RTV 소스 파일 추가하기...] 컨텍스트 메뉴를 클릭합니다.



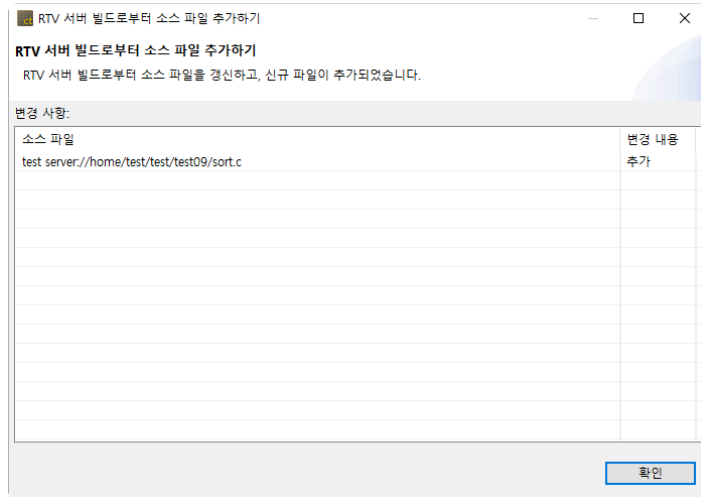
2. RTV 프로젝트를 생성할 때 선택한 소스 파일에 대한 최신 빌드 정보를 확인할 수 있습니다(빌드가 변경된 경우에 변경 알림 메시지 표시). [다음] 버튼을 클릭하여 신규 파일 목록을 확인합니다.



3. 신규 파일 목록에서 새로 추가할 파일을 선택한 뒤에 [완료] 버튼을 클릭합니다.




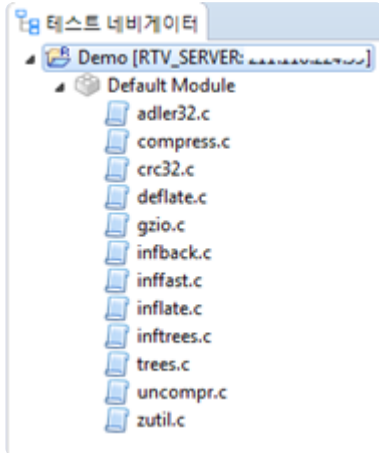
#### 4. 변경사항을 확인한 뒤에 [확인] 버튼을 클릭합니다.



## 27.5. RTV 프로젝트 정보

### 테스트 네비게이터

1. RTV 프로젝트의 아이콘은  으로 표시됩니다.
2. RTV 프로젝트의 이름 뒤에 RTV 서버의 정보([서버명: IP])가 표시됩니다.

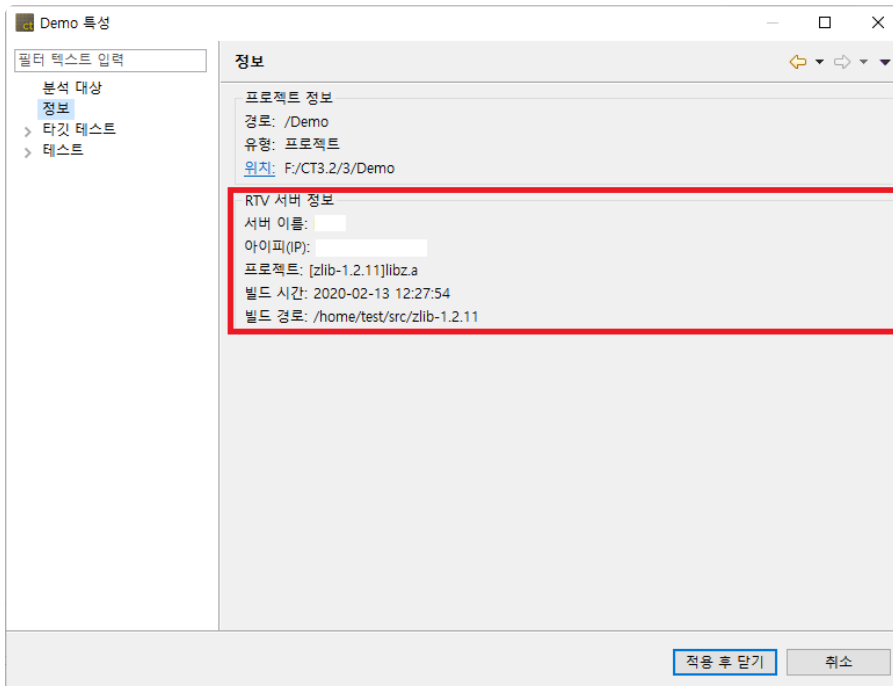


3. RTV 프로젝트의 소스 파일은 수정할 수 없습니다(RTV 프로젝트의 소스 파일 편집기는 읽기 전용).

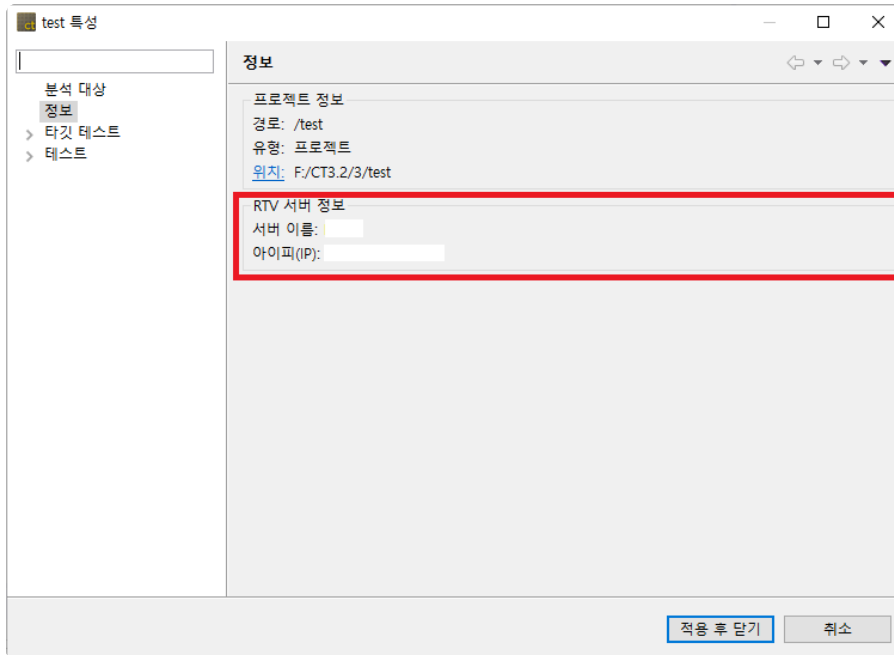
### 프로젝트 특성

#### 정보

1. RTV 프로젝트는 RTV 서버와 프로젝트 정보가 표시됩니다.



2. 일반 프로젝트는 틀체인을 RTV 틀체인으로 설정한 경우, RTV 틀체인을 가져온 RTV 서버의 정보가 표시됩니다.



## 모듈/파일 특성

### 빌드

1. RTV 프로젝트는 프로젝트를 생성할 때 설정한 RTV 툴체인을 변경할 수 없습니다.
2. 단, 최초 설정된 RTV 툴체인이 삭제된 경우에는 다른 툴체인으로 변경할 수 있습니다.
3. 일반 프로젝트는 툴체인을 RTV 툴체인으로 변경할 수 있습니다. RTV 툴체인으로 변경하면 프로젝트 하위 모든 툴체인 설정(모듈과 소스 파일)이 변경되고, RTV 테스트를 실행할 수 있게 됩니다.
4. RTV 툴체인이 설정된 프로젝트는 소스 파일의 컴파일러 설정을 변경할 수 없습니다(모듈의 링커 설정은 변경 가능).

## 27.6. RTV 테스트 실행

RTV 환경에서 테스트를 실행합니다.

RTV 프로젝트는 별도의 설정 없이 RTV 테스트를 실행할 수 있습니다. 일반 프로젝트를 RTV 환경에서 테스트하려면 해당 프로젝트의 툴체인을 테스트 대상 RTV 서버로부터 생성한 RTV 툴체인으로 설정해야 합니다.

### 전체 RTV 테스트 실행

[유닛 테스트] 뷰와 [통합 테스트] 뷰에서 선택한 테스트를 실행합니다.

RTV 테스트는 다음과 같은 방법으로 실행할 수 있습니다.

#### 1. [테스트 네비게이터] 뷰의 [RTV 테스트 실행] 컨텍스트 메뉴

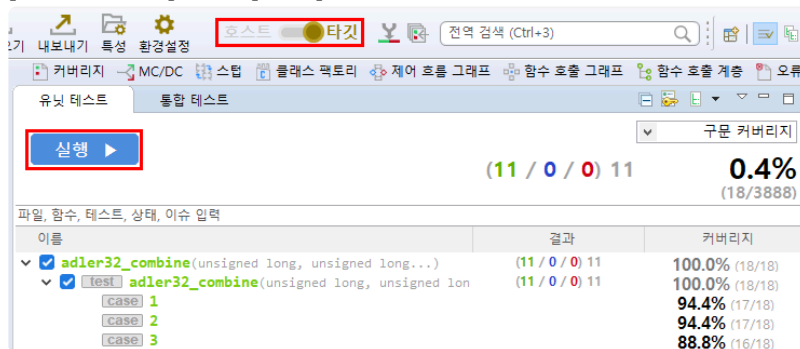


### RTV 유닛 테스트 실행

[유닛 테스트] 뷰에서 선택한 테스트를 실행합니다.

### 테스트 실행

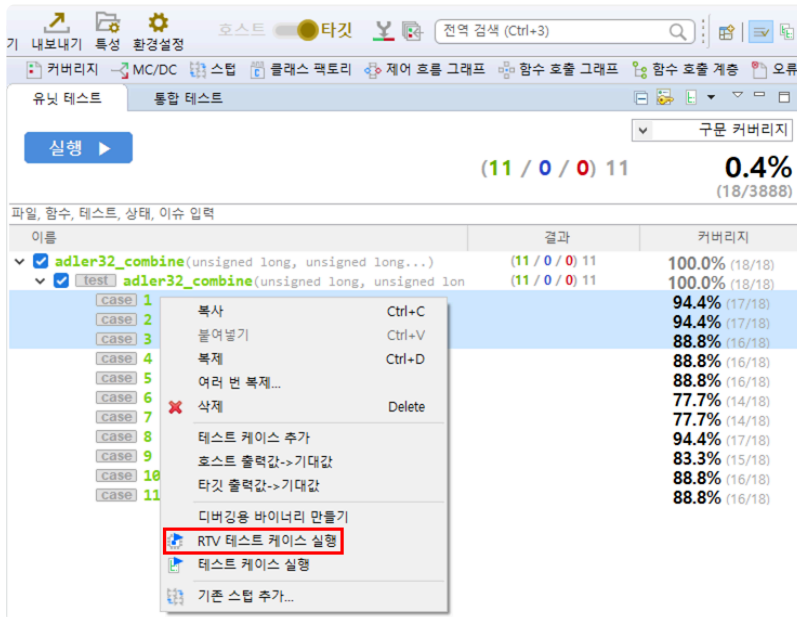
[유닛 테스트] 뷰의 [실행] 버튼





## 테스트 케이스 실행

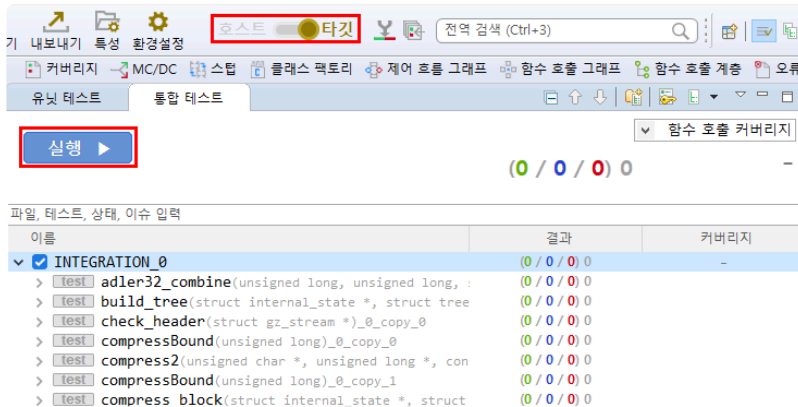
[유닛 테스트] 뷰의 [RTV 테스트 케이스 실행] 컨텍스트 메뉴(다중 선택 가능)



## RTV 통합 테스트 실행

[통합 테스트] 뷰에서 선택한 테스트를 실행합니다.

[통합 테스트] 뷰의 [실행] 버튼



## 27.7. RTV 테스트 결과 확인

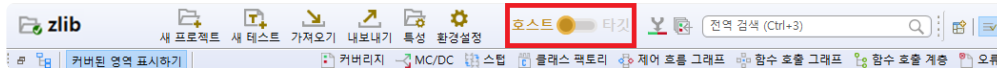
### 커버리지

최상위 톨바 메뉴의 커버리지 보기 메뉴를 통해 각 뷰와 편집기에서 표시해주는 커버리지 정보(호스트, 타깃, 호스트/타깃 병합)를 변경할 수 있습니다.

각 뷰가 나타내는 정보에 대한 상세한 설명은 [테스트 퍼스펙티브 항목](#)에서 확인할 수 있습니다.

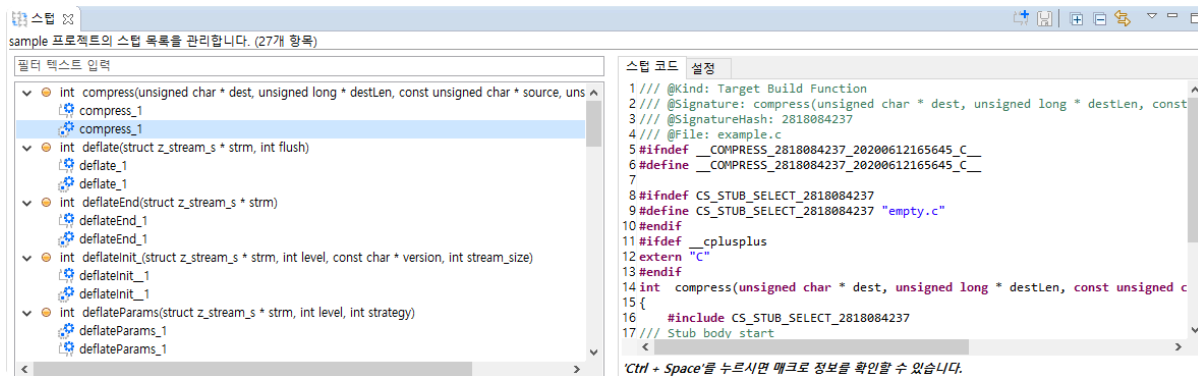
### 테스트 결과

대시보드의 호스트/타깃 설정 메뉴를 통해 테스트 결과를 변경할 수 있습니다.



### 스텝

RTV 테스트 실행 중에 생성되는 빌드 스텝은 호스트 테스트 실행 중에 생성되는 빌드 스텝과 구분됩니다.



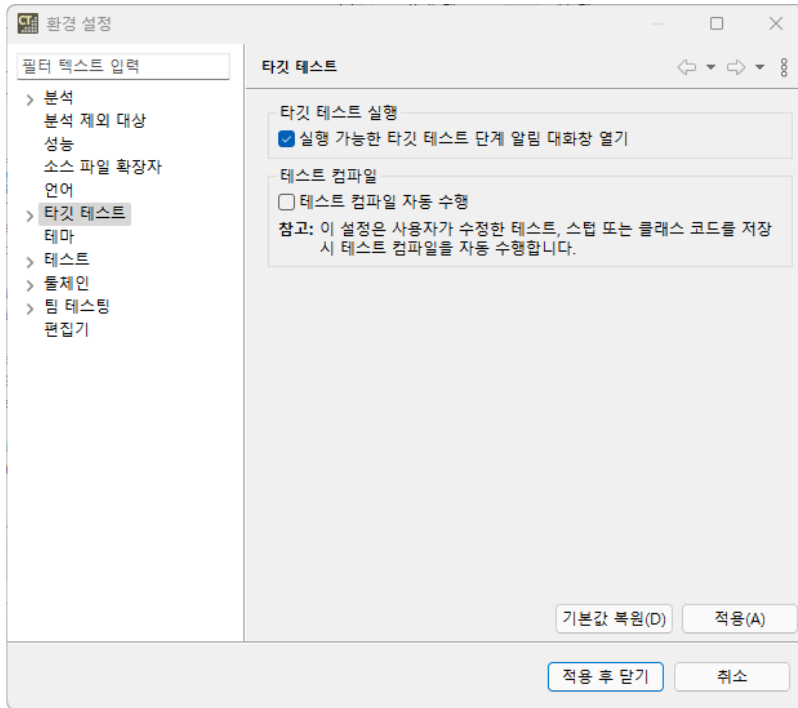
타입	설명
타깃 빌드 스텝	RTV 실행 중 자동 생성된 스텝

## 27.8. 테스트 컴파일


사용자가 테스트와 연관된 코드를 수정하는 경우 테스트 실행 전에 해당 코드에 오류가 있는지 확인할 수 있는 테스트 컴파일 기능을 제공합니다.

### 테스트 컴파일 설정

[환경설정] > [타겟 테스트]에서 테스트 컴파일 자동 수행 여부를 선택할 수 있습니다. 사용자가 테스트, 스텝, 클래스 코드를 수정하여 저장할때마다 테스트 컴파일을 수행합니다.

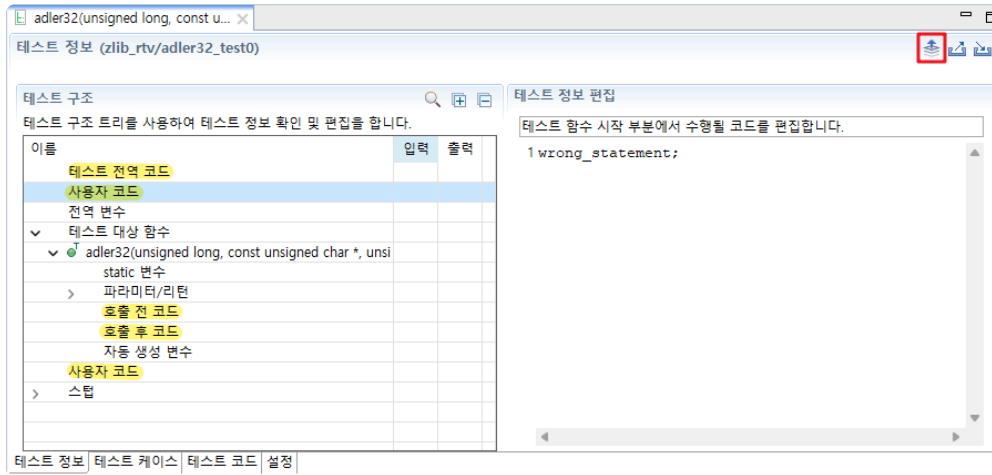


### 테스트 컴파일

테스트 컴파일 자동 수행을 선택하지 않은 경우, 각 뷰의 툴바 메뉴에서  를 클릭하여 테스트 컴파일을 할 수 있습니다.

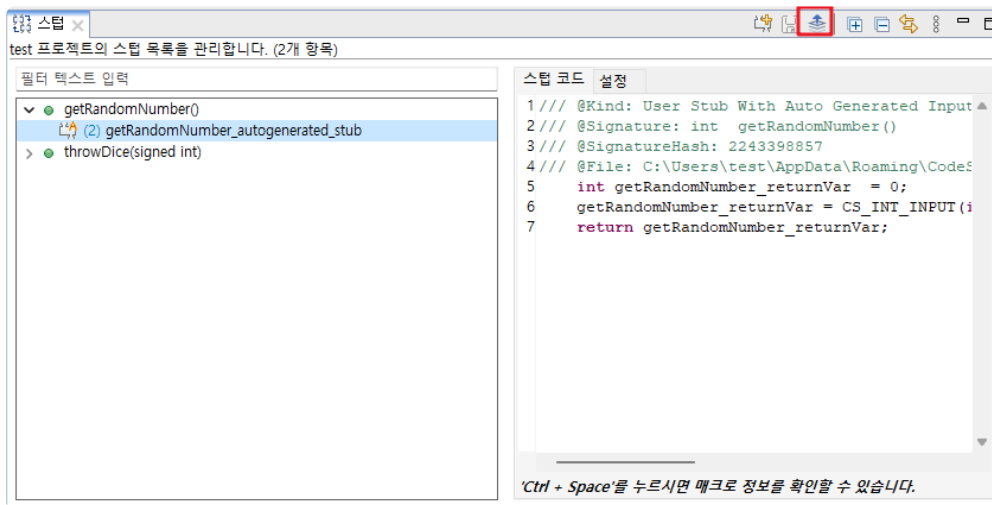
### 테스트 편집기

테스트 전역 코드, 사용자 코드, 각 함수의 호출 전/후 코드에 입력한 코드에 대하여 테스트 컴파일을 할 수 있습니다.



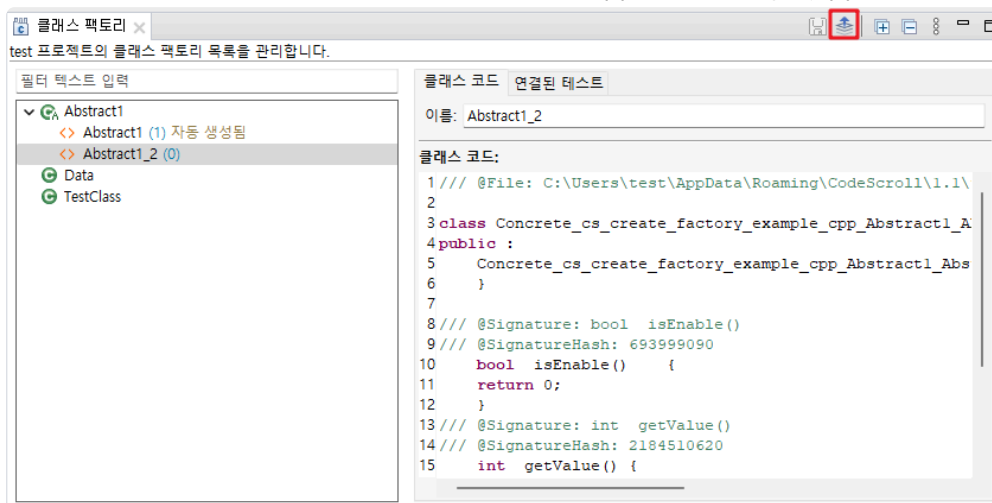
## 스텝 뷰

사용자가 작성한 스텝 코드에 대하여 테스트 컴파일을 할 수 있습니다.



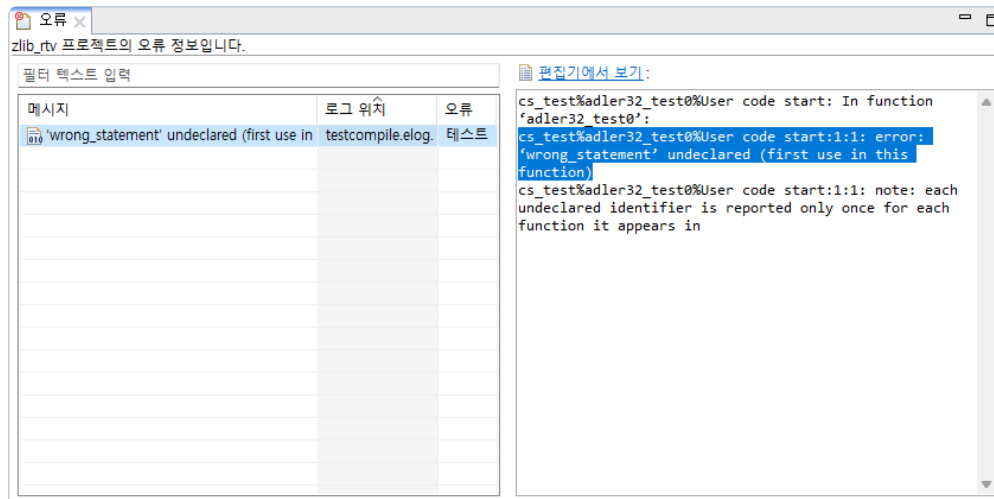
## 클래스 팩토리 뷰

사용자가 생성한 클래스 코드에 대하여 테스트 컴파일을 할 수 있습니다.



## 결과 확인

테스트 편집기, 스텝, 클래스 코드에서 테스트 컴파일을 사용할 수 있습니다. 테스트 컴파일에 오류가 발생하면, [오류 뷰]에서 확인할 수 있습니다.



## 28. CT RTV(Remote Target Verifier) Target Plug-in

---

### CT RTV Target Plug-in 소개

CT RTV Target Plug-in을 사용하면 리눅스 환경에서 개발된 소프트웨어를 실제 임베디드 타겟 환경에서 테스트할 수 있습니다. 리눅스 환경에 RTV 서버를 설치하고, CT 2023.12에서 RTV 설정과 타겟 환경 설정을 마치면 RTV 타겟 테스트를 수행할 수 있습니다. 리눅스 환경에 RTV 서버를 설치하는 방법은 본 문서 [문제 해결](#) 페이지 하단에 있는 기술 지원 연락처를 통해 문의해 주시기 바랍니다.

CT RTV Target Plug-in 프로젝트를 만들고 테스트를 실행하는 과정은 다음과 같습니다.

- [RTV 서버 설정](#)
- [RTV 톨체인 설정](#)
- [RTV 타겟 환경 설정](#)
- [RTV 타겟 프로젝트 만들기](#)
- [RTV 타겟 소스 파일 새로 고치기/추가하기](#)
- [RTV 타겟 프로젝트 정보](#)
- [RTV 타겟 테스트 실행](#)

### RTV 지원 OS

서버: Linux (RHEL, Ubuntu, Debian, Fedora 기반, 그 외 문의)

클라이언트: Windows 7 / 10 / 11

## 28.1. RTV 서버 설정

---

CT RTV Target Plug-in을 사용하기 위한 RTV 서버 설정은 본 문서 [RTV 서버 설정](#) 페이지를 참고하시길 바랍니다.

## 28.2. RTV 톨체인 설정

---

CT 2023.12의 RTV 타깃 프로젝트에서 사용하기 위한 RTV 톨체인 설정은 본 문서 [RTV 톨체인 설정](#) 페이지를 참고하시길 바랍니다.



## 28.3. RTV 타깃 환경 설정

CT RTV Target Plug-in을 사용하여 타깃 환경에서 테스트를 실행할 때 CT Target Plug-in과 마찬가지로 타깃 환경에 대한 정보를 입력해야 합니다. 타깃 환경에 대한 정보를 이용하여 CT 2023.12는 테스트 하네스를 RTV 서버에서 빌드하고 타깃에서 실행합니다. 실행 결과는 타깃 로그 수집기를 통해 자동으로 가져옵니다.

타깃 환경 설정은 프로젝트 특성 페이지 또는 타깃 테스트 프로젝트 생성 마법사에서 설정할 수 있습니다.

### 타깃 환경 설정 화면

타깃 환경 설정은 기본 정보와 상세 설정으로 나뉩니다.  
RTV 타깃 프로젝트에서는 기본 정보 설정을 하지 않습니다.

RTV target plugin does not need toolchain selection.

기본 정보를 입력하면, 입력이 필요한 상세 설정이 표시됩니다.

이름	값
language	c
툴체인 이름	gcc5.4
상태	사용 가능한 툴체인입니다.
C 컴파일러	/usr/bin/gcc
시스템 헤더(C 컴파일러)	C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSou...
LIB(C 컴파일러)	
C++ 컴파일러	/usr/bin/g++
시스템 헤더(C++ 컴파일러)	C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSou...
LIB(C++ 컴파일러)	

이름: 시스템 헤더(C++ 컴파일러) | 값:

설명:

C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt  
C:\Users\#AppData\Roaming\CodeScroll\1.1\#targetSourceWt

### 타깃 환경 상세 설정

타깃 환경 상세 설정은 분석, 빌드, 실행 그리고 기타로 나뉩니다.

카테고리	설명
분석	툴체인 정보가 표시되며, 타깃 테스트 프로젝트인 경우에는 분석을 위해 필요한 RTV 타깃 컴파일러 관련 설정이 표시됩니다.
빌드	테스트 소프트웨어를 빌드하기 위한 설정이 표시됩니다.
실행	타깃 환경에서 테스트를 실행하고 결과를 가져오기 위한 설정이 표시됩니다.
기타	그 외 설정이 표시됩니다. (프로그램 entry point 등)

각 카테고리별 필수 설정은 붉은색으로 표시됩니다. 카테고리별 필수 설정의 입력 여부에 따라 테스트 실행 버튼을 클릭했을 때의 동작이 아래와 같이 달라집니다.

필수 설정 완료된 카테고리	설명
없음	테스트 실행 불가능
분석	테스트를 실행하면 원본 소스코드에 테스트 코드를 덮어씁니다. 테스트를 수행하기 위해서는 사용자가 수동으로 테스트를 빌드하고 실행해야 합니다.
분석, 빌드	테스트를 실행하면 테스트 코드를 빌드합니다. 테스트를 수행하기 위해서는 사용자가 수동으로 테스트를 타깃에서 실행해야 합니다.
분석, 빌드, 실행	타깃 환경에서 자동으로 테스트를 실행합니다.

## 타깃 환경 상세 설정 - 빌드

RTV 타깃 프로젝트에서 빌드 스크립트 영역에 RTV 서버의 빌드 스크립트(사용자가 직접 작성) 경로를 입력합니다. 빌드 스크립트를 입력하면 RTV 서버 경로의 유효성을 확인합니다. 일정 시간 동안 입력이 없으면 유효성 검사를 통해 입력한 경로를 확인합니다. 유효하지 않은 경로를 입력할 경우 텍스트 박스가 빨간색으로 표시되고 유효한 경로를 입력할 경우 검은색으로 표시됩니다.

## 타깃 환경 상세 설정 - 실행

RTV 타깃 프로젝트에서 실행 설정은 실행 스크립트 영역과 로그 인터페이스 영역으로 나뉩니다. 스크립트 영역은 빌드 스크립트 영역과 마찬가지로 유효성 검사를 합니다. 로그 인터페이스 영역은 RTV 타깃 테스트의 수행 결과를 RTV 서버에서 가져오기 위한 설정을 합니다.

### 실행 스크립트 영역

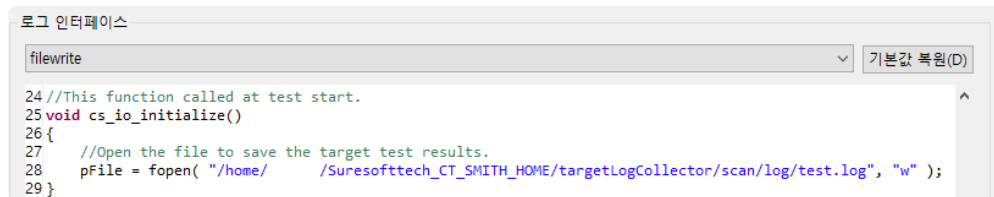
실행 스크립트 영역에는 빌드 스크립트 영역과 마찬가지로 RTV 서버의 실행 스크립트(사용자가 직접 작성) 경로를 입력합니다.

## 로그 인터페이스 영역

로그 인터페이스 영역은 타겟에서 타겟 로그의 형식으로 저장한 실행 결과를 [타겟 로그 수집기](#)를 통해 CT 2023.12로 가져올 때, 필요한 설정입니다. 타겟 로그를 CT 2023.12로 가져오는 방식은 다음 세 가지가 있습니다.

- 파일
- 소켓 통신 (TCP, UDP)
- 시리얼 통신 (UART)

filewrite 형식을 사용할 때, 타겟 로그 수집기가 타겟 로그 파일을 읽어오기 위해서 로그 인터페이스의 경로를 %SMITH\_HOME%/targetLogCollector/scan/log/%파일이름%으로 설정해야 합니다.



## 28.4. RTV 타깃 프로젝트 만들기

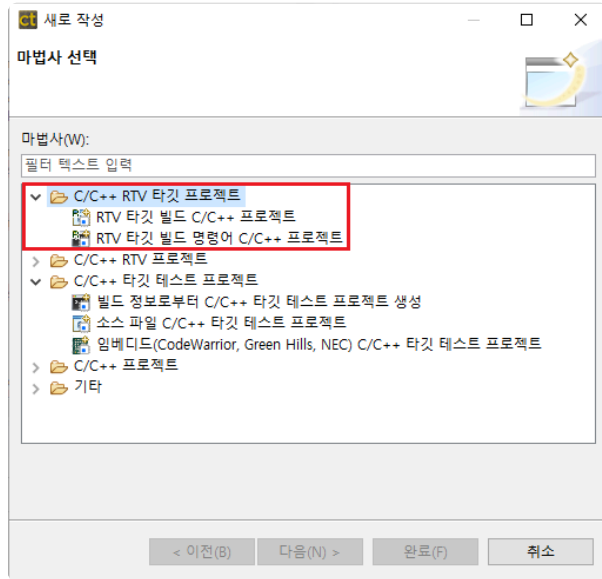
RTV 서버에 존재하는 타깃 소프트웨어의 소스 파일을 테스트하기 위해 RTV 타깃 프로젝트를 생성합니다.

RTV 환경에서 소스 파일을 빌드할 때, RTV 타깃 프로젝트 생성에 필요한 정보를 추출할 수 있습니다.

RTV 환경에 미리 추출해놓은 소스 파일 빌드 정보(CT 2023.12 RTV 서버 유틸리티 사용)를 이용하거나, 대상 소스 파일을 빌드하는 동시에 정보를 추출하여 RTV 프로젝트를 생성하는 방법을 설명합니다.

RTV 타깃 프로젝트를 생성하려면 [프로젝트 생성 마법사]를 실행합니다.

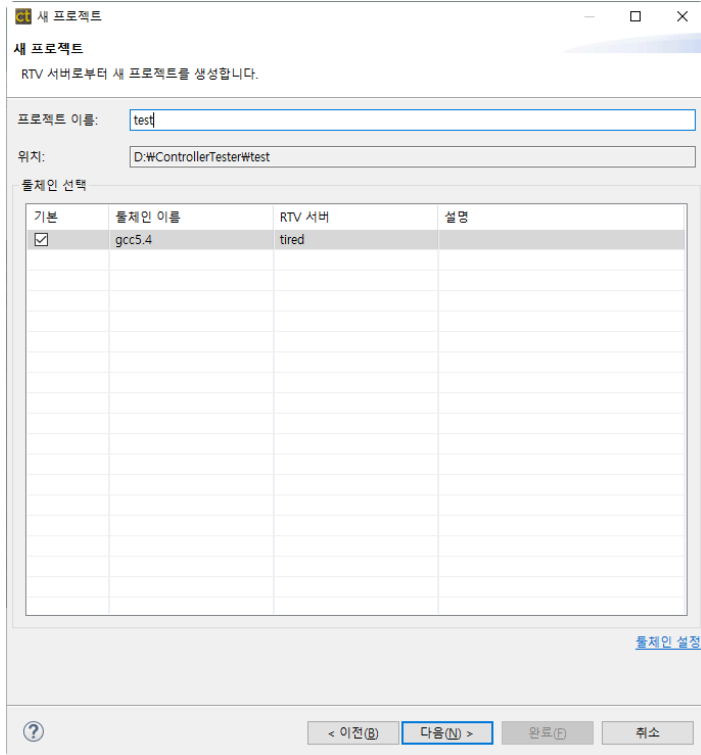
[파일] > [새로 만들기] 또는 단축 아이콘을 클릭합니다.



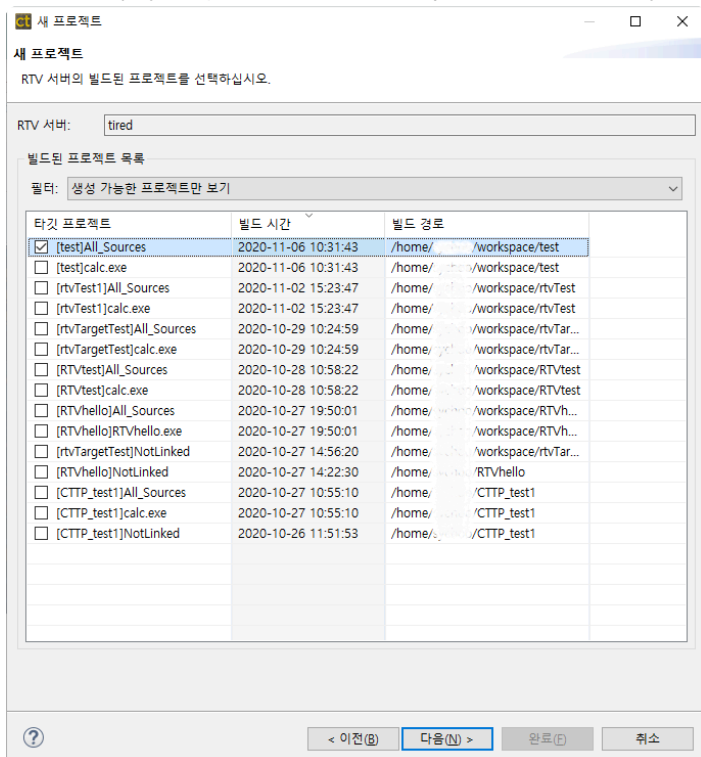
### RTV 타깃 빌드 C/C++ 프로젝트 생성

RTV 환경에 미리 추출해놓은 소스 파일 빌드 정보(CT 2023.12 RTV 서버 유틸리티 사용)를 이용하여 RTV 타깃 프로젝트를 생성합니다.

1. [프로젝트 이름]에 프로젝트 이름을 입력하고 사용할 RTV 툴체인을 선택합니다. 툴체인을 선택한 뒤에 [다음] 버튼을 클릭합니다.

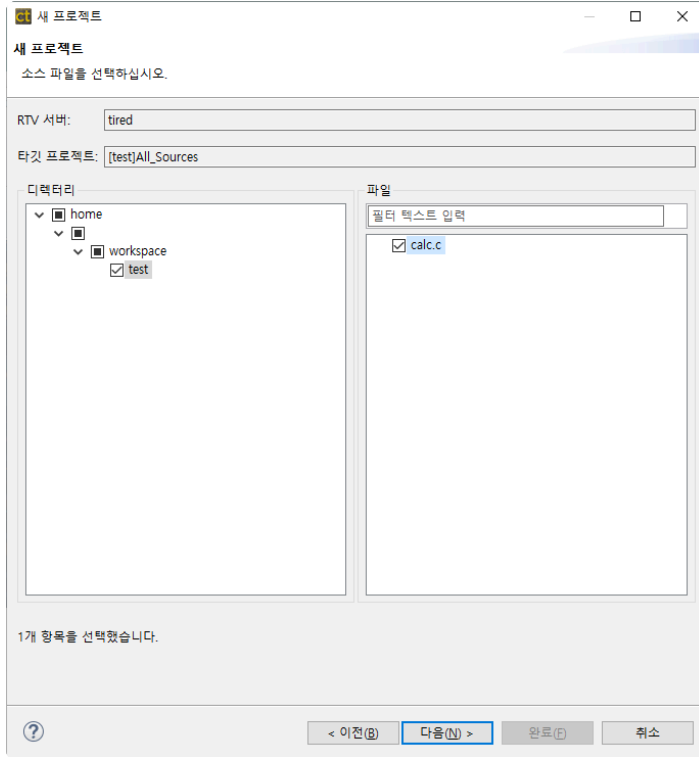


2. 선택한 틀체인을 가져온 RTV 서버에 미리 추출되어 있는 소스 파일 빌드 목록을 확인할 수 있습니다. 빌드된 목록에서 RTV 타깃 프로젝트를 생성할 항목을 선택한 뒤에 [다음] 버튼을 클릭합니다.

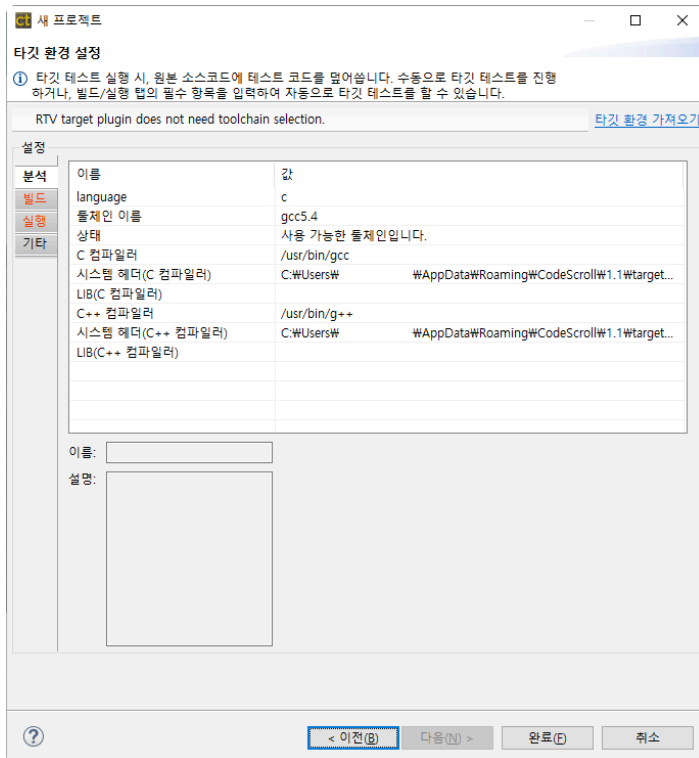


- ✿ 생성된 프로젝트 중 ‘All\_Sources\_프로젝트명’은 ctbuild를 통해 빌드 캡처 된 모듈의 모든 소스파일을 합친 프로젝트입니다. 이 프로젝트는 빌드 여부를 고려하지 않고 합쳐진 프로젝트이기 때문에 정상적으로 테스트 수행이 되지 않을 수 있습니다.

3. RTV 타깃 빌드 프로젝트를 생성할 때 포함할 소스 파일(테스트 대상)을 선택한 후 [다음] 버튼을 클릭합니다



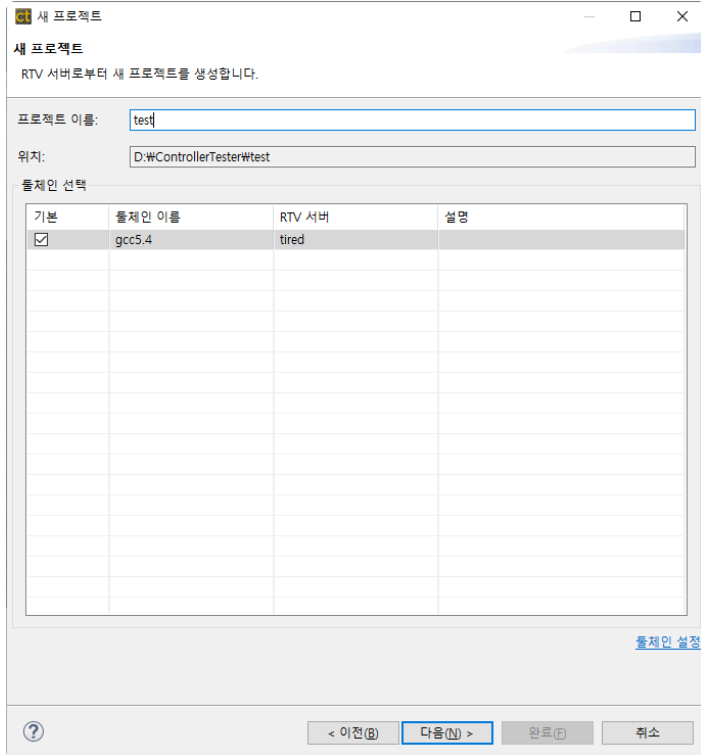
4. 타겟 환경의 상세 설정을 입력한 후 [완료] 버튼을 클릭합니다.



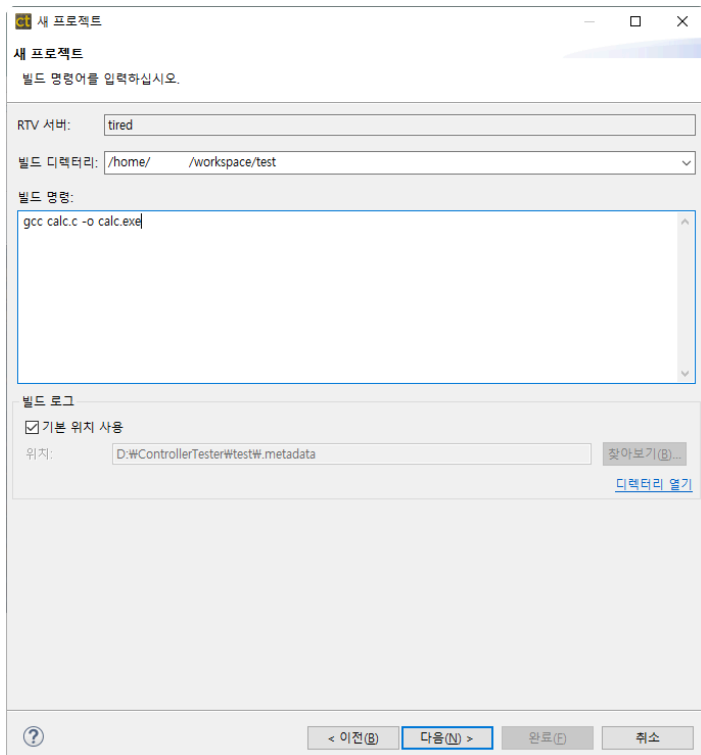
## RTV 타겟 빌드 명령어 C/C++ 프로젝트 생성

테스트 대상 소스 파일을 빌드하는 동시에 정보를 추출하여 RTV 타겟 프로젝트를 생성합니다. 빌드 명령어를 입력하여 RTV 서버에서 소스 파일을 빌드하면서 RTV 타겟 프로젝트를 생성합니다.

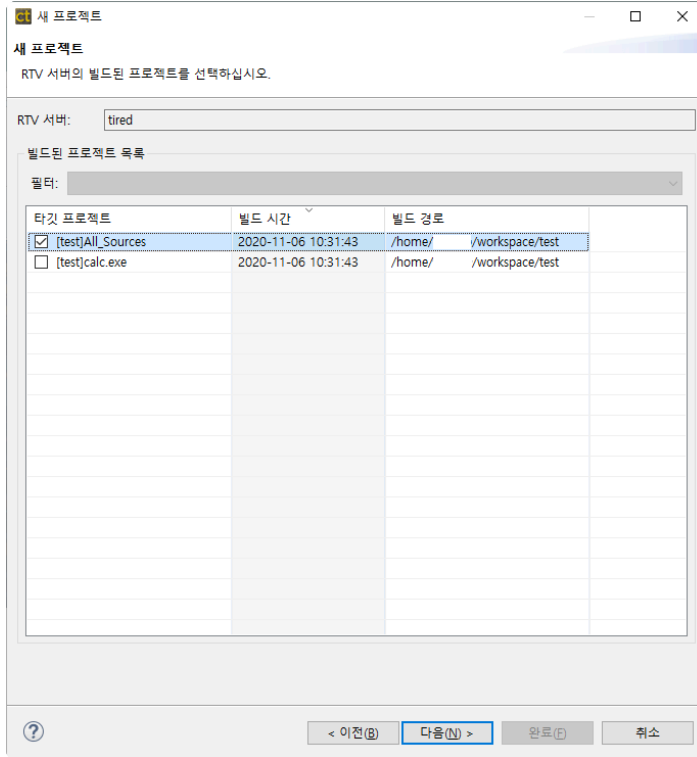
1. [프로젝트 이름]에 프로젝트 이름을 입력하고 사용할 RTV 툴체인을 선택합니다. 이름을 입력하고 툴체인을 선택한 뒤에 [다음] 버튼을 클릭합니다.



2. [빌드 디렉터리]에 RTV 서버에서 빌드를 수행할 경로(예 : make 파일이 있는 디렉터리)를 입력하고, 빌드 명령어를 입력합니다. [빌드 로그]에 빌드 결과 파일을 생성할 경로를 지정합니다. 빌드에 필요한 정보를 입력한 뒤 [다음] 버튼을 클릭하면 입력된 빌드 정보로 RTV 서버에서 소스 파일을 빌드합니다. 빌드 대상에 따라 많은 시간이 소요될 수 있습니다.

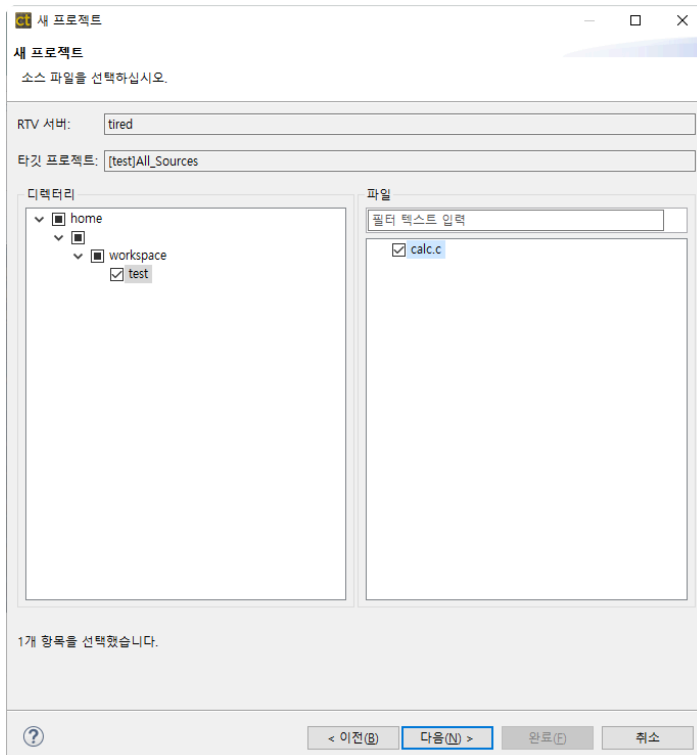


3. 빌드가 완료되면, 빌드된 프로젝트 목록이 나타납니다. 빌드된 목록에서 RTV 타깃 프로젝트를 생성할 항목을 선택한 뒤에 [다음] 버튼을 클릭합니다.



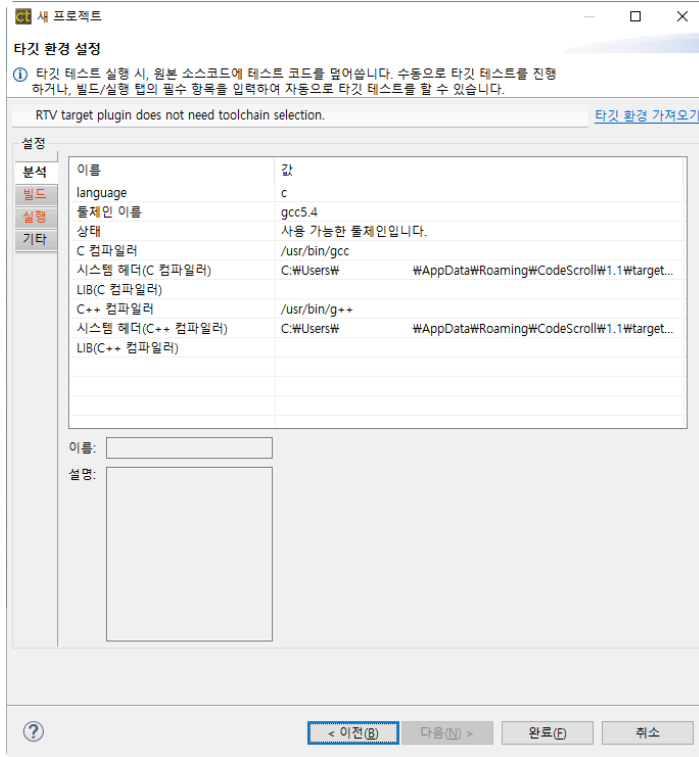
- \* 생성된 프로젝트 중 'All\_Sources\_프로젝트명'은 ctbuild를 통해 빌드 캡처 된 모듈의 모든 소스파일을 합친 프로젝트입니다. 이 프로젝트는 빌드 여부를 고려하지 않고 합쳐진 프로젝트이기 때문에 정상적으로 테스트 수행이 되지 않을 수 있습니다.

4. RTV 타겟 프로젝트를 생성할 때 포함할 소스 파일(테스트 대상)을 선택한 뒤에 [다음] 버튼을 클릭합니다.



5. 타겟 환경의 상세 설정을 입력한 후 [완료] 버튼을 클릭하여 프로젝트를 생성합니다.





## RTV 빌드 명령어

빌드 정보로부터 프로젝트를 생성하려면, 빌드 명령어에 `ctbuild` 명령어를 사용해야 합니다. 자세한 내용은 [RTV 프로젝트 만들기](#) 페이지 하단에 [RTV 빌드 명령어] 내용을 참고하시기 바랍니다.

## 28.5. RTV 타깃 소스 파일 새로 고치기/추가하기

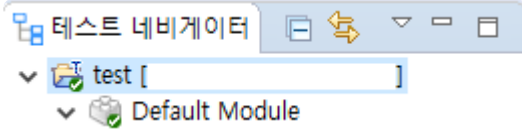
---

RTV 서버에서 소스 파일이 수정/추가된 경우 [RTV 소스 파일 새로 고치기]와 [RTV 소스 파일 추가하기] 기능을 통해 수정/추가된 내용을 RTV 타깃 프로젝트에 반영할 수 있습니다. 자세한 내용은 본 문서의 [RTV 소스 파일 새로 고치기/추가하기](#) 페이지를 참고하시길 바랍니다.

## 28.6. RTV 타깃 프로젝트 정보

### 테스트 네비게이터

- RTV 타깃 프로젝트의 아이콘은  으로 표시됩니다.
- RTV 타깃 프로젝트의 이름 뒤에 RTV 서버의 정보([서버명:IP])가 표시됩니다.

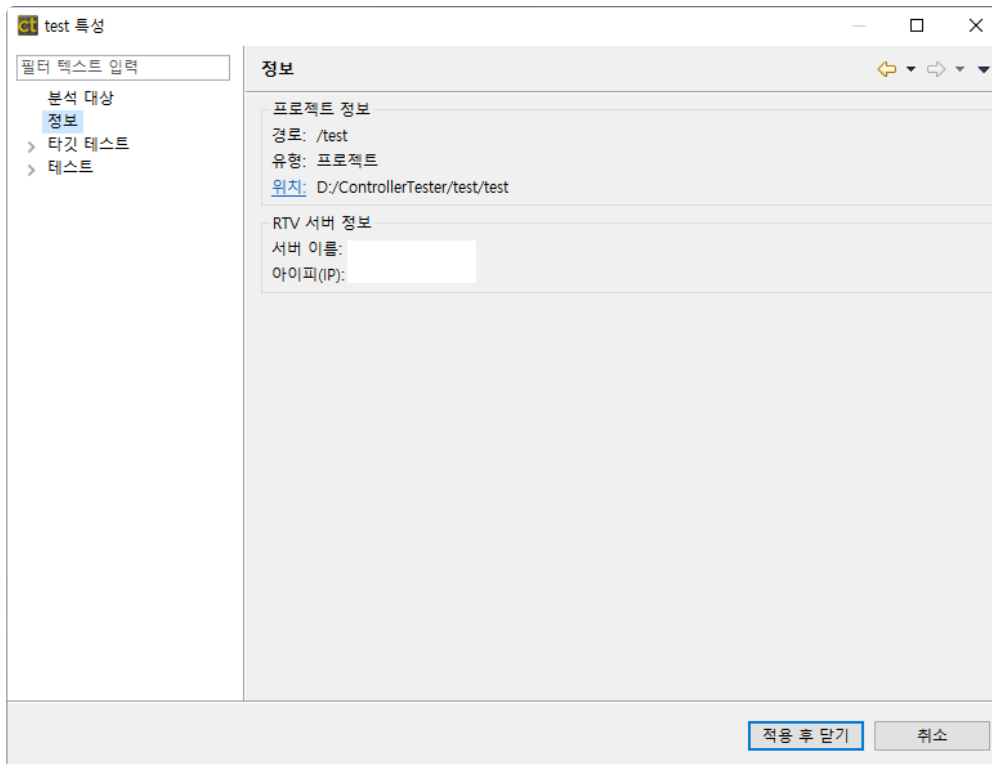


- RTV 타깃 프로젝트의 소스 파일은 수정할 수 없습니다(RTV 타깃 프로젝트의 소스 파일 편집기는 읽기 전용).

### 프로젝트 특성

#### 정보

RTV 타깃 프로젝트는 RTV 서버와 프로젝트의 정보가 표시됩니다.



#### 타깃 테스트

RTV 타깃 프로젝트에서는 타깃 테스트 실행과 타깃 환경 관련 설정을 할 수 있습니다. 타깃 테스트 실행 관련 설정은 본 문서의 [타깃 테스트 실행 설정하기](#) 페이지를, 타깃 환경 설정은 [타깃 환경 설정하기](#) 페이지를 참고하시길 바랍니다.

## 모듈/파일 특성

RTV 타깃 프로젝트의 모듈/파일 특성은 본 문서 [RTV 프로젝트 정보](#) 페이지의 [모듈/파일 특성] 내용을 참고하시길 바랍니다.

## 28.7. RTV 타겟 테스트 실행

---

### 테스트 실행

RTV Target Plug-in은 리눅스 환경에서 빌드된 바이너리 파일을 타겟 환경에서 테스트할 수 있습니다. RTV 타겟 테스트 실행은 본 문서의 [RTV 테스트 실행](#) 페이지를 참고하시길 바랍니다.

## 29. DISCOVERY Plug-in

DISCOVERY는 커버리지를 높일 수 있는 테스트 케이스를 자동으로 생성해줍니다.

### 기호 실행 기반 테스트 케이스 생성

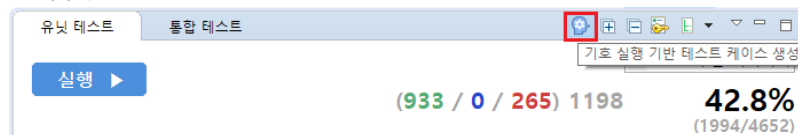
[기호 실행 기반 테스트 케이스 생성] 기능을 사용하면 커버리지를 높일 수 있는 테스트 케이스를 자동으로 생성합니다.

! release 모드로 생성된 Visual Studio 프로젝트의 경우, /D "N\_DEBUG" 매크로가 활성화되어 디스커버리를 사용할 수 없습니다.

1. 선택되어 있는 테스트를 대상으로 기호 실행을 기반으로 테스트 케이스를 생성합니다.

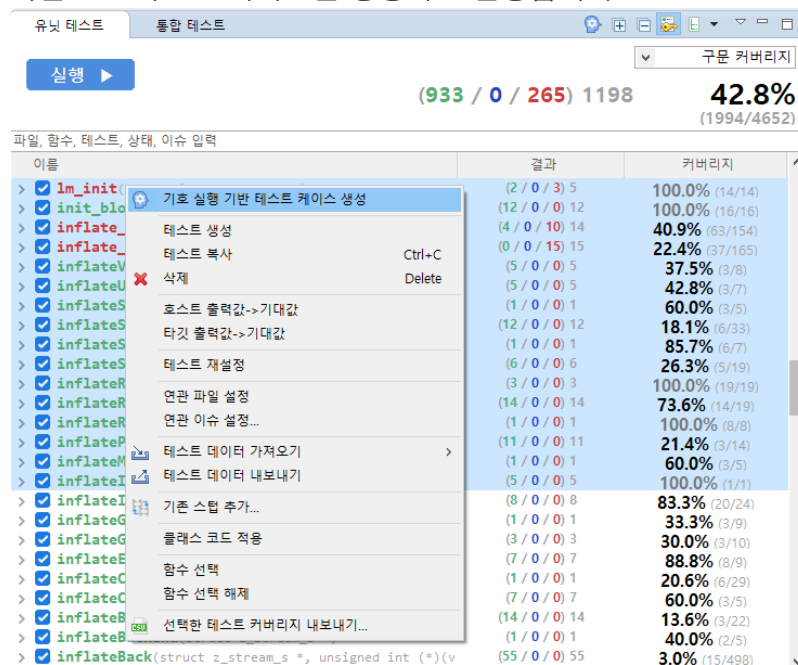
- 유닛 테스트 뷰의 툴바

유닛 테스트 뷰의 툴바에 [기호 실행 기반 테스트 케이스 생성] 클릭하면 유닛 테스트 뷰에 체크된 함수/테스트에 대하여 기호 실행을 기반으로 테스트 케이스를 생성하고 실행합니다.

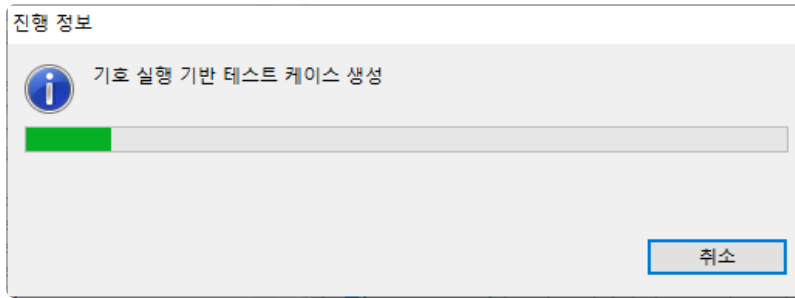


- 유닛 테스트 뷰의 컨텍스트 메뉴

함수/테스트를 우클릭하고 [기호 실행 기반 테스트 케이스 생성] 메뉴를 선택하면 기호 실행을 기반으로 테스트 케이스를 생성하고 실행합니다.

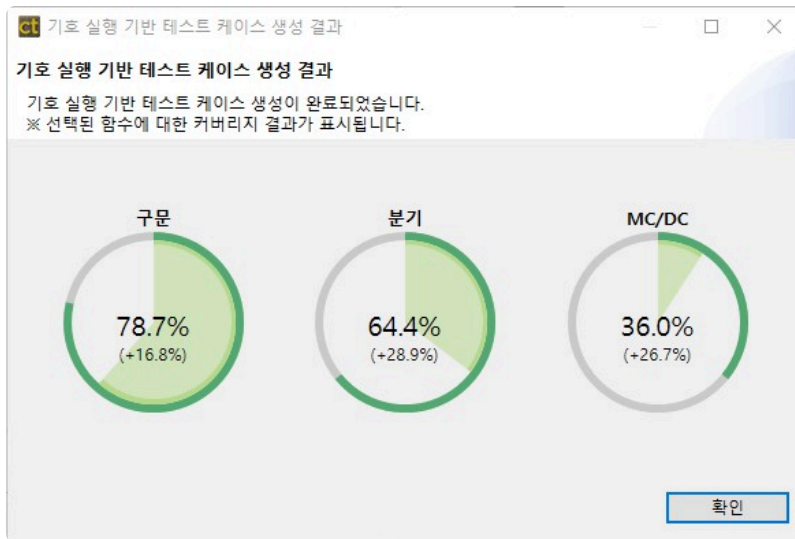


2. [기호 실행 기반 테스트 케이스 생성]을 클릭하면 진행 정보를 확인할 수 있습니다.



! 기호 실행 기반 테스트 케이스 생성 시 테스트 케이스 생성 시간이 오래 걸릴 수 있습니다.

3. 테스트 케이스 생성이 완료되면, [기호 실행 기반 테스트 케이스 생성 결과]창을 통해 커버리지를 확인할 수 있습니다.



## 30. CT CLI(Command Line Interface)

---

CT CLI는 CT 기능을 명령어 인터페이스로 제공합니다.

Command Line Interface는 CT 2023.12 설치경로/csc.exe 파일을 실행하여 프로젝트에 대한 실행(-e), 수행 결과 내보내기(-r), 삭제(-d), 도움말(-h)의 기본 옵션들을 사용합니다.

CLI 환경에서 실행하는 과정은 다음과 같습니다.

- [프로젝트 생성 및 테스트 실행](#)
- [프로젝트 가져오기](#)
- [프로젝트 내보내기](#)
- [분석 정보 및 VPES 데이터 내보내기](#)
- [프로젝트 삭제](#)
- [DISCOVERY 실행하기](#)
- [결합 위치 찾기](#)
- [팀 프로젝트 가져오기](#)



## 30.1. 프로젝트 생성 및 테스트 실행

CLI 환경에서 프로젝트 생성 및 분석, 테스트 케이스 생성, 실행, 테스트 결과 보고서 생성 등 CT 2023.12의 기능들을 사용할 수 있습니다. 방법은 아래와 같습니다.

1. CT 2023.12에서 [프로젝트] – [CLI 설정 파일 만들기]를 선택하여, CLI.ini 파일을 생성합니다.
2. CLI.ini파일을 페이지 하단의 [CLI 설정 파일 편집하기]를 참고하여 편집합니다.
3. `-e -w "%workspacePath%" --ini -O "%iniFilePath%"` 명령어를 사용하여 테스트 연관 기능을 수행합니다.
  - workspacePath : 워크스페이스 경로
  - iniFilePath : CLI.ini의 경로

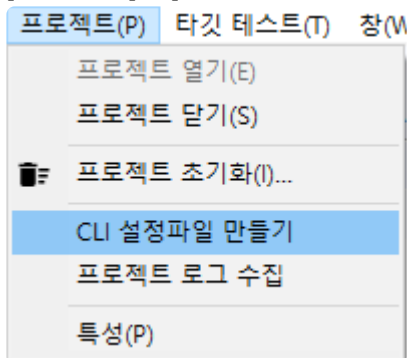
```
C:\Program Files\Suresoft\WCT 2023.6>csc.exe -e -w "C:\Users\sure\Documents\cli-workspace" --ini -O "C:\Users\sure\Desktop\CLITestProject_CLI.ini"
[MAIN-INFO] 명령어 인자를 분석 중...
[MAIN-INFO] 워크스페이스 설정
[MAIN-INFO] Install Location: C:\Program Files\Suresoft\WCT 2023.6
[MAIN-INFO] PRODUCT_VERSION: 2023.6
[MAIN-INFO] 워크스페이스: C:\Users\sure\Documents\cli-workspace
[MAIN-INFO] initialize()
[MAIN-INFO] 전역 정보를 읽었습니다.
[MAIN-INFO] execution job name: Execute UtCLI
cshome path= C:\Program Files\Suresoft\WCT 2023.6\plugins\com.codescroll.jp.core_1.0.2.201202152119\
csuthome path= C:\Program Files\Suresoft\WCT 2023.6\plugins\com.codescroll.ut_2023.6.0\
csuthomebin path= C:\Program Files\Suresoft\WCT 2023.6\plugins\com.codescroll.ut_2023.6.0\bin\
csglhome path= C:\Users\sure\AppData\Roaming\CodeScroll\W1.1\
```

\* CLI.ini 파일을 실행하기 전에 생성하고 편집해야 합니다.

### CLI 설정 파일 만들기

CLI 환경에서 CT 2023.12를 실행하기 위한 설정 파일을 생성합니다.

- [프로젝트] – [CLI 설정 파일 만들기]를 선택하면 사용자가 선택한 경로에 CLI.ini 파일이 생성됩니다.



\* 선택된 프로젝트가 없으면 프로젝트 정보가 포함되지 않은 CLI.ini 파일이 생성됩니다.

### CLI 설정 파일 편집하기

#### [Project] – 프로젝트 정보

항목	설명
----	----

create	프로젝트 생성 여부
cpi_file	create=true인 경우, 프로젝트 생성을 위한 .cpi 파일 경로
force	create=true인 경우, 같은 이름의 프로젝트가 존재하는 경우 삭제 후 생성
name	프로젝트 이름
toolchain	툴체인 이름 (RTV 프로젝트이면 작성)
target_project	타겟 프로젝트 이름 (RTV 프로젝트이면 작성)

! **target\_project** 옵션은 [프로젝트] - [특성] - [정보] - [RTV 서버 정보]의 타겟 프로젝트 이름과 같게 입력해야 합니다.

## [Analysis] – 분석

항목	설명
run	분석 수행 여부

## [InitializeTest] – 테스트 초기화

항목	설명
run	true일 경우 기존 테스트 모두 삭제

## [CreateTest] – 유닛 테스트 생성

항목	설명
run	유닛 테스트 생성 여부
target_function	유닛 테스트 생성을 위한 함수 이름으로 하나 또는 다수의 함수에 대한 유닛 테스트 생성이 가능 여러 개의 함수를 입력할 경우, 세미콜론(;) 또는 엔터(Enter)로 구분 가능 target_function 목록이 없는 경우 모든 함수에 대하여 유닛 테스트 생성
isolate	테스트 대상 함수를 분리하기 위한 스텝 생성 여부

## [CreateTestCase] – 테스트 케이스 생성

항목	설명
run	테스트 케이스 생성 여부
mode	테스트 케이스 생성 모드(flat, pairwise, random)
random_testcase_count	random일 경우 생성되는 테스트 케이스 개수 입력

## [ImportTestInfo] – 테스트 정보 일괄 가져오기

항목	설명
run	테스트 정보 가져오기 수행 여부
testinfo_file	testinfo.export 파일의 경로
include_fault_injection	결함 주입 포함 여부

## [ExecuteTest] – 유닛 테스트 실행

항목	설명
run	유닛 테스트 실행 여부
remote_target	RTV 실행 여부
run_robust	테스트 실행 중 오류 발생 시, 실행 불가능한 테스트 제외 여부
only_build	테스트 실행 없이 빌드만 수행
build_option_file_path	컴파일 커맨드 옵션 파일 경로

## [Report] – 보고서 생성

항목	설명
run	보고서 생성 여부
output_dir	보고서 출력 경로 기본 출력 경로는 프로젝트 경로
include_project_info	프로젝트 정보 포함 여부
test_env_kind	보고서 생성 종류(host: 1, target: 2)
jenkins_test_result	Jenkins 결과 파일 경로
HTML, XLSX, PDF, DOCX, PPTX	보고서 형식 지정

## 테스트 정보 개별 가져오기



[ImportTestInfo]에서 `run=false`로 설정하면 [ImportStub], [ImportTest], [ImportIntegrationTest]를 개별적으로 실행할 수 있습니다.

## [ImportStub] – 스텝 가져오기

항목	설명
run	스텝 가져오기 수행 여부

stub_file	스텝 파일 경로
stub_dir	스텝 디렉터리 경로

### [ImportTest] – 테스트 가져오기

항목	설명
run	테스트 가져오기 수행 여부
test_source_file	.utest 파일 경로 여러 개의 소스 파일 경로를 입력할 경우 세미콜론(;) 또는 엔터(Enter)로 구분 가능
test_source_dir	테스트 소스 디렉터리 경로 테스트 소스 디렉터리 경로를 입력할 경우 디렉터리 경로에 포함된 전체 테스트 가져오기 수행

### [ImportTestCase] – 테스트 케이스 가져오기

항목	설명
run	테스트 가져오기 수행 여부
testcase_dir	테스트 케이스 디렉터리 경로 테스트 케이스 디렉터리 경로를 입력할 경우 디렉터리 경로에 포함된 전체 테스트 케이스 가져오기 수행

### [ImportIntegrationTest] – 통합 테스트와 테스트 케이스 가져오기

항목	설명
run	통합 테스트와 테스트 케이스 가져오기 수행 여부
test_file	.itest 파일 경로
test_dir	.itest 파일의 공통된 상위 경로 테스트와 테스트 케이스가 같은 디렉터리에 있으면 가져오기 수행

## CPI 파일 만들기

CPI 파일은 CLI 환경에서 프로젝트를 생성하기 위한 설정 파일입니다. CLI.ini 파일의 [Project] – create 옵션을 true로 설정되어야 프로젝트가 생성됩니다.

CT 2023.12 설치 경로\plugins\com.codescroll.gp.cli\_버전\cpi 경로의 CPI 템플릿 파일에서 프로젝트를 생성하는 옵션을 설정할 수 있습니다.

**!** 워크스페이스 사용 중에는 CLI 환경에서 프로젝트를 생성할 수 없습니다.

항목	설명
----	----

KIND	프로젝트 생성 타입 설정 일반 프로젝트 =1 빌드 스크립트 가져오기 = 2 비주얼 스튜디오 가져오기 =3 임베디드 프로젝트 가져오기 = 4
NAME	프로젝트 이름
LANGUAGE	프로젝트 종류 Java = 0, C/C++ = 1

## SourceFileProject

항목	설명
IUT_LANGUAGE	대상 언어 유형 C=0, C++=1, Java=2
LINKFLAG	링크 플래그
COMMON_COMPILEFLAG	모든 TU에 적용할 컴파일 플래그
TOOLCHAIN_NAME	툴체인 이름
BINARY_KIND	바이너리 종류
SOURCE_TOP_DIR	소스 파일 최상위 디렉터리
SOURCE	소스 파일 절대 경로
COMPILEFLAG	소스 파일에만 적용할 컴파일 플래그
COMPILERPATH	소스 파일에만 적용할 컴파일러 경로

## VisualStudioProject

항목	설명
PROJECT_PATH	가져오기를 할 비주얼 스튜디오 프로젝트 파일
ENVFILE	가져오기 시 필요한 환경 변수

## BuildScriptProject

항목	설명
BUILD_COMMAND	가져오기를 수행할 빌드 스크립트
WORKING_DIRECTORY	가져오기를 수행할 작업 디렉터리
ENVFILE	가져오기 시 필요한 환경 변수

## EmbeddedProject

항목	설명
COMMON_COMPILEFLAG	모든 소스 파일에 적용할 컴파일 플래그
TOOLCHAIN_NAME	툴체인 이름
PROJECT_PATH	가져오기를 수행할 비주얼 스튜디오 프로젝트 파일

! 경로 입력 시 \는 **escape** 문자이므로 \\ 또는 /를 경로 구분자로 사용하십시오.

## 30.2. 프로젝트 가져오기

내보낸 프로젝트를 CT 2023.12로 가지고 올 수 있습니다. 프로젝트를 가져오는 방법은 아래와 같습니다.

1. `-e -w "%워크스페이스경로%" --import -O "--path '%가져올 프로젝트 경로%' --mapping-file '%PathMappingFile.csv 경로%' --include-src --include-tch"` 명령어를 통해 프로젝트를 가져옵니다.

```
C:\Program Files\Suresoft\CT 2023.6>csc.exe -e -w "C:\Users\sure\Documents\cli-workspace" --import -O "--path 'C:\Users\sure\Desktop\cliExportPath\cliTestProject_20230521120136' --include-src --include-tch"
[MAIN-INFO] 명령어 인자를 분석 중...
[MAIN-INFO] 워크스페이스 설정
[MAIN-INFO] Install Location: C:\Program Files\Suresoft\CT 2023.6
[MAIN-INFO] PRODUCT VERSION: 2023.6
[MAIN-INFO] 워크스페이스: C:\Users\sure\Documents\cli-workspace
[MAIN-INFO] initialize()
[MAIN-INFO] 전역 정보를 읽었습니다.
[MAIN-INFO] execution job name: Execute Project Import
```

- 필수 옵션이 없으면 프로젝트를 가져올 수 없습니다.

항목	설명
<code>-e, --execute</code>	(필수)실행
<code>-w, --workspace</code>	(필수)워크스페이스
<code>--import</code>	(필수)프로젝트 가져오기
<code>-O</code>	(필수)지정된 옵션으로 프로젝트 가져오기
<code>--path</code>	(필수)가져올 프로젝트 경로 지정
<code>--mapping-file</code>	매핑 파일의 경로(기본값: “내보낸 경로” + “/PathMappingFile.csv”)
<code>--include-src</code>	소스 파일 포함 가져오기
<code>--include-tch</code>	툴체인 포함 가져오기

\* `-O` 옵션의 인자는 큰따옴표로 입력받으며, `-O` 하위 옵션의 인자는 작은따옴표로 입력받습니다.

\* `--mapping-file` 옵션은 내보내기 시 프로젝트 관련 경로와 가져오는 PC의 경로가 다를 때, 해당 경로를 매핑하는데 사용합니다. PathMappingFile.csv 파일은 프로젝트 내보내기 시 내보낸 폴더에 생성됩니다.

\* `--mapping-file`에 대한 자세한 설명은 User Guide 2023.12의 [\[CLI 프로젝트 가져오기 시 경로 재설정\]](#) 을 참고하시기를 바랍니다.

## 30.3. 프로젝트 내보내기

프로젝트 내보내기에서는 선택한 프로젝트를 원하는 경로에 내보냅니다. 프로젝트를 내보내는 방법은 아래와 같습니다.

1. `-e -w "%워크스페이스 경로%" -p "%프로젝트 이름%" --export -O "--path '%내보낼 경로%' --include-src --include-tch"` 명령어를 통해 프로젝트를 내보냅니다.

```
C:\Program Files\Suresoft\CT 2023.6>csc.exe -e -w "C:\Users\sure\Documents\cli-workspace" -p "CLITestProject" --export -O "--path 'C:\Users\sure\Desktop\cliExportPath' --include-src --include-tch"
[MAIN-INFO] 환경의 인자를 분석 중...
[MAIN-INFO] 워크스페이스 설정
[MAIN-INFO] Install Location: C:\Program Files\Suresoft\CT 2023.6
[MAIN-INFO] PRODUCT VERSION: 2023.6
[MAIN-INFO] 워크스페이스: C:\Users\sure\Documents\cli-workspace
[MAIN-INFO] initialize()
[MAIN-INFO] 전역 정보를 읽었습니다.
[MAIN-INFO] execution job name: Execute Project Export
[MAIN-INFO] INFO
```

- 필수 옵션이 없으면 프로젝트를 내보낼 수 없습니다.

항목	설명
<code>-e, --execute</code>	(필수)실행
<code>-w, --workspace</code>	(필수)워크스페이스
<code>-p</code>	(필수)프로젝트
<code>--export</code>	(필수)프로젝트 내보내기
<code>-O</code>	(필수)지정된 옵션으로 프로젝트 내보내기
<code>--path</code>	(필수)프로젝트 내보내기 경로
<code>--include-src</code>	소스 파일 포함 내보내기
<code>--include-tch</code>	툴체인 포함 내보내기

\* `-O` 옵션의 인자는 큰따옴표로 입력받으며, 하위 옵션의 인자는 작은따옴표로 입력받습니다.

2. 내보내기 성공 시 “내보낸 경로”와 “결과: 성공했습니다.”라는 문구가 출력됩니다.

```
===== EXPORT START =====
[MAIN-INFO] 툴체인 설정 파일 복사...
[MAIN-PROGRESS] 40%
[MAIN-PROGRESS] 100%
[MAIN-INFO] 소스 파일을 내보내는 중...
INFO export path:C:\Users\sure\Desktop\cliExportPath\CLITestProject_20230518160615
INFO
===== EXPORT SUCCESS =====
[MAIN-INFO] result: 'Execute Project Export' succeeded.
[MAIN-INFO] 결과: 성공했습니다.
```



## 30.4. 분석 정보 및 VPES 데이터 내보내기

CLI의 `-r` 옵션은 프로젝트 결과를 메트릭, 제어 흐름 그래프 및 VPES 데이터 형식으로 내보내기를 제공합니다.

### 사용 형태

분석정보 내보내기를 위한 필수 옵션으로 워크스페이스와 프로젝트 이름, 내보낼 경로를 지정해야 합니다.

```
csc.exe [-r] [-w path] [-p name] [-o path]
```

### 수행 결과 내보내기 명령어

1. `-r -w "%워크스페이스 경로%" -p "%프로젝트 이름%" -o "%내보낼 경로%" --cfg --metric --exportvpes` 명령어를 사용하여 프로젝트 수행 결과를 내보냅니다.

```
C:\Program Files\Suresoft\CT 2023.6>csc.exe -r -w "C:\Users\sure\Documents\cli-workspace" -p "CLITestProject" -o "C:\Users\sure\Desktop\report" --cfg --metric --exportvpes
[MAIN-INFO] 환경변수 인자를 분석 중...
[MAIN-INFO] 워크스페이스 설정
[MAIN-INFO] Install Location: C:\Program Files\Suresoft\CT 2023.6
[MAIN-INFO] PRODUCT_VERSION: 2023.6
[MAIN-INFO] 워크스페이스: C:\Users\sure\Documents\cli-workspace
[MAIN-INFO] initialize()
[MAIN-INFO] 전역 정보를 읽었습니다.
[MAIN-INFO] report job name: Report CFG Image
[MAIN-INFO] result: 'Report CFG Image' succeeded.
[MAIN-PROGRESS] 33%
[MAIN-INFO] report job name: Report Metrics
[MAIN-INFO] result: 'Report Metrics' succeeded.
[MAIN-PROGRESS] 66%
[MAIN-INFO] report job name: Export VPES data
[MAIN-INFO] 데이터 작성 중...
```

- `cfg`, `metric`, `uploadvpes` 및 `exportvpes` 옵션을 사용하여 내보내려는 출력 형식을 선택합니다.
- 4가지 옵션 중 아무것도 선택하지 않으면 내보내는 파일이 생성되지 않습니다.

항목	설명
<code>-r --report</code>	(필수)수행 결과 내보내기
<code>-w, --workspace</code>	(필수)워크스페이스
<code>-p, --project</code>	(필수)프로젝트 이름
<code>-o, --output</code>	(필수)결과를 내보낼 경로 지정
<code>--cfg, --callgraph</code>	제어 흐름 그래프 이미지를 내보내기
<code>--metric</code>	메트릭 보고서를 내보내기
<code>--uploadvpes</code>	VPES에 데이터 업로드
<code>--exportvpes</code>	VPES에 데이터 내보내기

2. `--cfg, --callgraph` 세부 옵션

예시 : `-r -w "%워크스페이스 경로%" -p "%프로젝트 이름%" -o "%내보낼 경로%" --cfg --idPath`

항목	설명
----	----

--idPath	내보낸 이미지 경로 및 이름에 ID만 포함
----------	-------------------------

### 3. --metric 세부 옵션

예시 : -r -w "%워크스페이스 경로%" -p "%프로젝트 이름%" -o "%내보낼 경로%" --metric -O "-f example-cli -e html -t module"

항목	설명
-f, --fileNameSuffix	내보낼 파일 이름 지정
-e, --extension	내보낼 파일 형식 지정 (html, pdf, doc, xls, ppt)
-t, --type	내보낼 파일 모델 유형 지정(module, file, class, function)

### 4. --uploadvpes 세부 옵션

예시 : -r -w "%워크스페이스 경로%" -p "%프로젝트 이름%" --uploadvpes -O "%CT 2023.12 설치 경로\plugins\com.codescroll.gp.cli\ini\vpes\_access.info%"

 uploadvpes, exportvpes 두 옵션을 동시에 사용할 수 없습니다.

## 30.5. 프로젝트 삭제

CLI의 `-d` 옵션은 CLI 환경에서 프로젝트를 삭제할 수 있습니다.

### 사용 형태

프로젝트 삭제의 필수 옵션으로 워크스페이스와 프로젝트 이름을 입력해 주어야 합니다.

```
csc.exe [-d] [-w path] [-p name]
```

### 삭제 명령어

1. `-d -w "%워크스페이스 경로%" -p "%프로젝트 이름%"` 명령어를 사용하여 프로젝트를 삭제합니다.

```
C:\Program Files\Suresoft\WCT 2023.6>csc.exe -d -w "C:\Users\sure\Documents\cli-workspace" -p "CLITestProject"
[MAIN-INFO] 명령어 인자를 분석 중...
[MAIN-INFO] Project Delete
Warning: NLS missing message: ProjectDeleteUtilCommand_10 in: com.codescroll.jp.cli.tools.messages
[MAIN-INFO] 프로젝트 CLITestProject를 삭제하였습니다.(C:\Users\sure\Documents\cli-workspace)
```

- 삭제된 프로젝트는 복구할 수 없습니다.

항목	설명
<code>-d, --delete</code>	(필수)실행
<code>-w, --workspace</code>	(필수)워크스페이스
<code>-p, --project</code>	(필수)프로젝트

## 30.6. DISCOVERY 실행하기

CLI 환경에서 `-e` 옵션을 사용해 DISCOVERY 기능을 수행합니다. DISCOVERY는 커버리지를 높일 수 있는 테스트 케이스를 자동으로 생성합니다. 방법은 아래와 같습니다.

### 사용 형태

필수 옵션으로 워크스페이스와 프로젝트 이름, `discovery` 옵션을 입력해 주어야 합니다.

```
csc.exe [-e] [-w path] [-p name] --discovery
```

### DISCOVERY 실행 명령어

- `-e -w "%워크스페이스경로%" -p "%프로젝트 이름%" --discovery` 명령어를 통해 DISCOVERY를 실행합니다.

```
C:\Users\sure\Downloads\CodeScroll (10)\CodeScroll>csc.exe -e -w "D:\CT-WorkSpace\CT-CLI-DISCOVERY" -p "Discovery-test" --discovery
[MAIN-INFO] 명령어 인자를 분석 중...
[MAIN-INFO] 워크스페이스 설정
[MAIN-INFO] Install Location: C:\Users\sure\Downloads\CodeScroll (10)\CodeScroll
[MAIN-INFO] PRODUCT_VERSION: 2023.6
[MAIN-INFO] 워크스페이스: D:\CT-WorkSpace\CT-CLI-DISCOVERY
[MAIN-INFO] initialize()
[MAIN-INFO] 전역 정보를 읽었습니다.
[MAIN-INFO] execution job name: Execute Discovery
log4j:WARN No appenders could be found for logger (com.codescroll.ct.cli.discovery.execute.DiscoveryExecutor).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
INFO
```

- 실행이 완료 후 CT 2023.12에서 기호 실행 기반 테스트 케이스가 추가된 걸 확인할 수 있습니다.



✿ DISCOVERY는 분석 및 테스트 케이스가 생성된 프로젝트를 대상으로 실행이 가능합니다.

✿ DISCOVERY에 대한 자세한 설명은 [\[DISCOVERY Plug-in\]](#) 페이지에서 제공됩니다.

## 30.7. 결함 위치 찾기

CLI 환경에서 -e 옵션을 사용해 결함 위치 찾기 기능을 수행합니다. 결함 위치 찾기는 오류 또는 실패한 테스트 케이스로부터 결함 위치를 찾아냅니다. 방법은 아래와 같습니다.

### 사용 형태

필수 옵션으로 워크스페이스와 프로젝트 이름, faultlocalization 옵션을 입력해 주어야 합니다.

```
csc.exe [-e] [-w path] [-p name] -e --faultlocalization
```

### 결함 위치 찾기 실행 명령어

- -e -w "%워크스페이스경로%" -p "%프로젝트 이름%" --faultlocalization 명령어를 통해 결함 위치 찾기를 실행합니다.

```
C:\Users\jhhyeo\Downloads\CodeScroll (4)\CodeScroll>csc.exe -e -w "H:\bitbucket_rep\cli-test\working_dir\faultlocalization\FaultLocalizationTest001"
-p "FaultLocalizationTest001" --faultlocalization
[MAIN-INFO] 명령어 인자를 분석 중...
[MAIN-INFO] 워크스페이스 설정
[MAIN-INFO] Install Location: C:\Users\jhhyeo\Downloads\CodeScroll (4)\CodeScroll
[MAIN-INFO] PRODUCT_VERSION: 2023.12
[MAIN-INFO] 워크스페이스: H:\bitbucket_rep\cli-test\working_dir\faultlocalization\FaultLocalizationTest001
[MAIN-INFO] initialize()
[MAIN-INFO] 전역 정보를 읽었습니다.
[MAIN-INFO] execution job name: Execute Fault Localization
Load JDBC Driver...OK
Serialization query: 210
log4j:WARN No appenders could be found for logger (com.codescroll.ct.cli.faultlocalization.execute.CliFaultLocalizationExecutor).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
INFO
===== FaultLocalization START =====

[MAIN-INFO]
[MAIN-PROGRESS] 100%
INFO
===== FaultLocalization SUCCESS =====

[MAIN-INFO] result: 'Execute Fault Localization' succeeded.
[MAIN-INFO] 결과: 성공했습니다.
```

- 실행이 완료 후 CT 2023.12의 [결함 위치 뷰]에서 결함 위치에 대한 정보를 제공합니다.

✿ 결함 위치 찾기에 대한 자세한 설명은 [\[결함 위치 찾기\]](#) 페이지에서 제공됩니다.

## 30.8. 팀 프로젝트 가져오기

팀 테스트 서버에서 프로젝트를 가져올 수 있습니다. 팀 프로젝트를 가져오는 방법은 아래와 같습니다.

```
-e -w "%워크스페이스경로%" --import-team-project -O "--project-name '%팀 프로젝트 이름%' --address '%팀 테스트 서버 주소%' --port '%팀 테스트 서버 포트%' "
```

```
C:\Users\jhyeo\Downloads\CodeScroll (4)\CodeScroll>csc.exe -e -w "H:\bitbucket_rep\cli-test\working_dir\import-team\ImportTeamProject001" --import-team-project -O "--project-name 'ImportTeamProject001' --address --port
[MAIN-INFO] 명령어 인자를 분석 중...
[MAIN-INFO] 워크스페이스 설정
[MAIN-INFO] Install Location: C:\Users\jhyeo\Downloads\CodeScroll (4)\CodeScroll
[MAIN-INFO] PRODUCT_VERSION: 2023.12
[MAIN-INFO] 워크스페이스: H:\bitbucket_rep\cli-test\working_dir\import-team\ImportTeamProject001
[MAIN-INFO] initialize()
[MAIN-INFO] 전역 정보를 읽었습니다.
[MAIN-INFO] execution job name: Execute Team Project Import
log4j:WARN No appenders could be found for logger (com.codescroll.ct.cli.team.execute.TeamProjectImportExecutor).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
INFO
===== IMPORT TEAM PROJECT START =====
```

- 필수 옵션이 없으면 프로젝트를 가져올 수 없습니다.

항목	설명
-e, --execute	(필수)실행
-w, --workspace	(필수)워크스페이스
--import-team-project	(필수)팀 프로젝트 가져오기
-O	(필수)지정된 옵션으로 팀 프로젝트 가져오기
--project-name	(필수)팀 프로젝트 이름
--address	(필수)팀 서버 주소
--port	(필수)팀 서버 포트
--exclude-test	테스트 제외하고 가져오기
--run-test-with-commit	테스트 실행 후 커밋
--test-env	테스트 실행 환경 (1: host, 2: target, 기본값: 1)

✿ -O 옵션의 인자는 큰따옴표로 입력받으며, -O 하위 옵션의 인자는 작은따옴표로 입력받습니다.

## 31. CT 팀 테스트 서버 plug-in

CT 2023.12은 한 개의 소프트웨어를 여러 사용자가 테스트할 수 있는 팀 테스트 기능을 제공합니다. CT 2023.12 프로젝트를 팀 테스트 서버에 올려두고 여러 사용자가 프로젝트를 가져와서 테스트할 수 있습니다. 여러 사용자가 테스트한 결과를 팀 테스트 대시보드에서 확인할 수 있습니다.

팀 테스트를 할 수 있는 프로젝트는 다음과 같습니다.

- C/C++ 프로젝트
- RTV C/C++ 프로젝트

✳ 타깃 C/C++ 프로젝트나 RTV 타깃 C/C++ 프로젝트는 팀 테스트를 할 수 없습니다.

### 용어

이 단락에서는 팀 테스트 기능에 사용하기 위해 알아야 할 용어를 설명합니다.

- 팀 테스트 서버(팀 서버): 다른 사용자와 공유할 프로젝트가 올라가 있는 서버
- 로컬: 사용자가 작업 중인 CT
- 팀 프로젝트: 팀 테스트 서버에 올려 다른 사용자와 공유할 수 있는 프로젝트
- 자원
  - 구성 자원: 프로젝트를 구성하는 설정 정보(프로젝트 특성 정보 등)
  - 공유 자원: 테스트, 스텝, 결함 주입, 클래스 코드 등 모든 사용자가 공유하여 사용하는 테스트 자원
- 리비전: 서버의 수정 기록. 프로젝트 생성 시 리비전은 1이며 커밋할 때마다 증가
- 커밋: 로컬의 변경 사항과 테스트 결과를 서버에 반영하는 작업
- 업데이트: 서버의 최신 형상을 로컬에 반영
- 충돌: 업데이트할 대상 자원이 로컬에서 수정이 있는 경우

### 기능

CT 2023.12에서 제공하는 팀 테스트 기능은 다음과 같습니다.

- [CT 팀 테스트 서버 설치하기](#)
- [CT 팀 테스트 서버 실행하기](#)
- [환경설정](#)
- [팀 프로젝트](#)
  - [팀 프로젝트 생성하기](#)
  - [팀 프로젝트 특성](#)
  - [팀 프로젝트 가져오기](#)
  - [팀 프로젝트 변환하기](#)
- [커밋](#)
- [업데이트](#)
- [팀 테스트](#)
  - [팀 테스트 가져오기](#)

- [팀 테스트 삭제](#)



## 31.1. CT 팀 테스트 서버 설치하기

### 설치 요구 사항


	최소 사양	권장 사양
<b>OS</b>	Microsoft Windows 7 (64bit)	Microsoft Windows 10 (64bit)
<b>CPU</b>	Intel Core i5 3470(4코어 이상)	Intel Core i7 9700(8코어 이상)
<b>RAM</b>	4GB	16GB
저장 공간	HDD 100GB	SSD 500GB

\* 지원하는 OS는 Windows 7, 10, 11과 Ubuntu 20.04입니다.

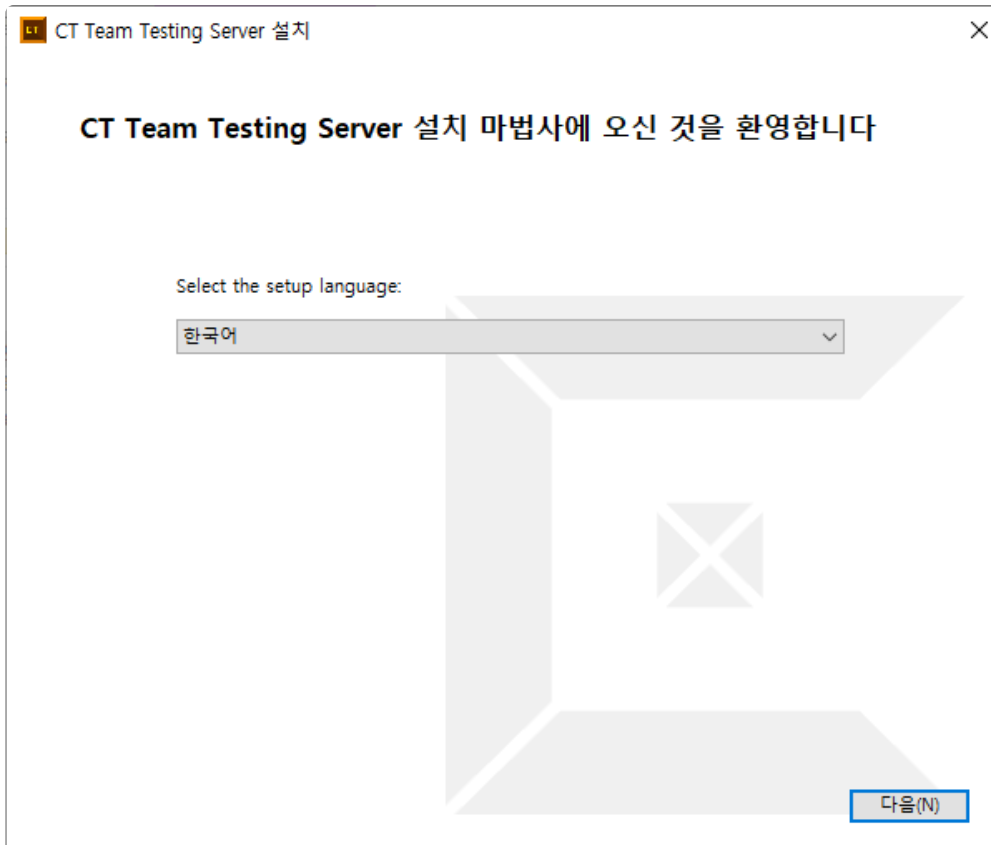
### 팀 테스트 서버 설치하기

설치 패키지를 이용하여 팀 테스트 서버를 설치하는 방법은 다음과 같습니다.

1. ct-team-testing-server-xxxx-setup.exe 파일을 실행합니다.

이름	수정한 날짜	유형	크기
 ct-team-testing-server-2023-setup	2023-12-08 오후 1:59	응용 프로그램	910,539KB

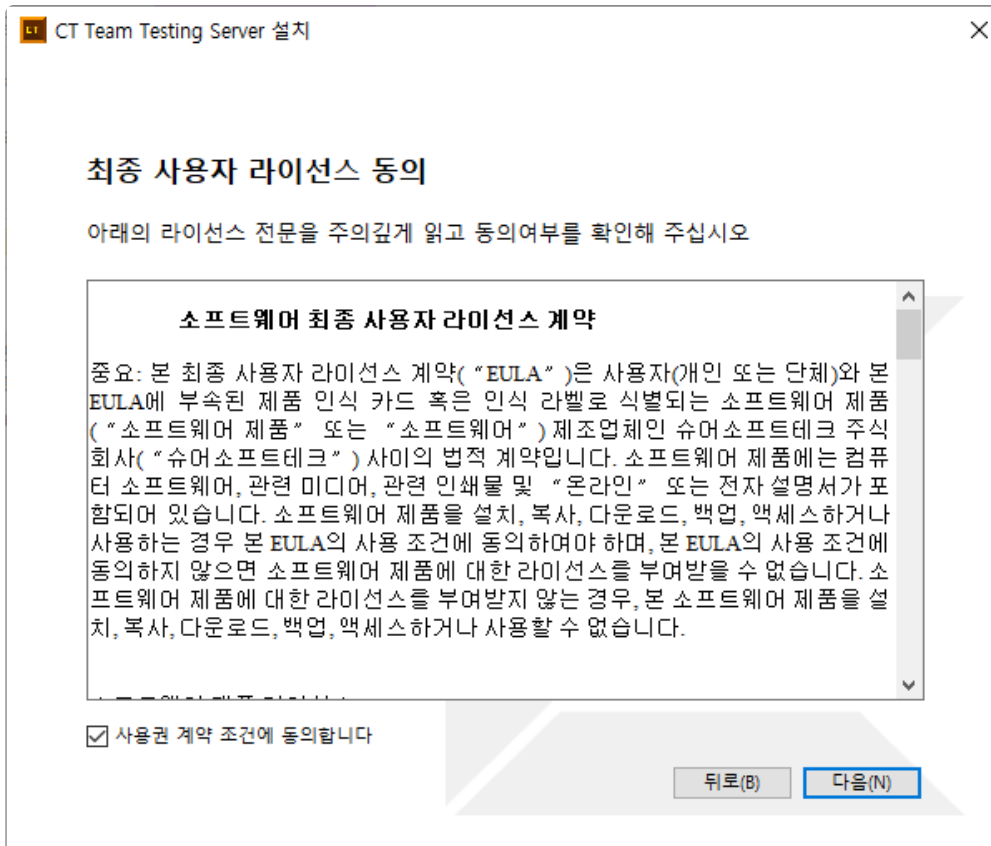
2. 설치 마법사의 언어를 선택한 후 [다음]을 클릭합니다.



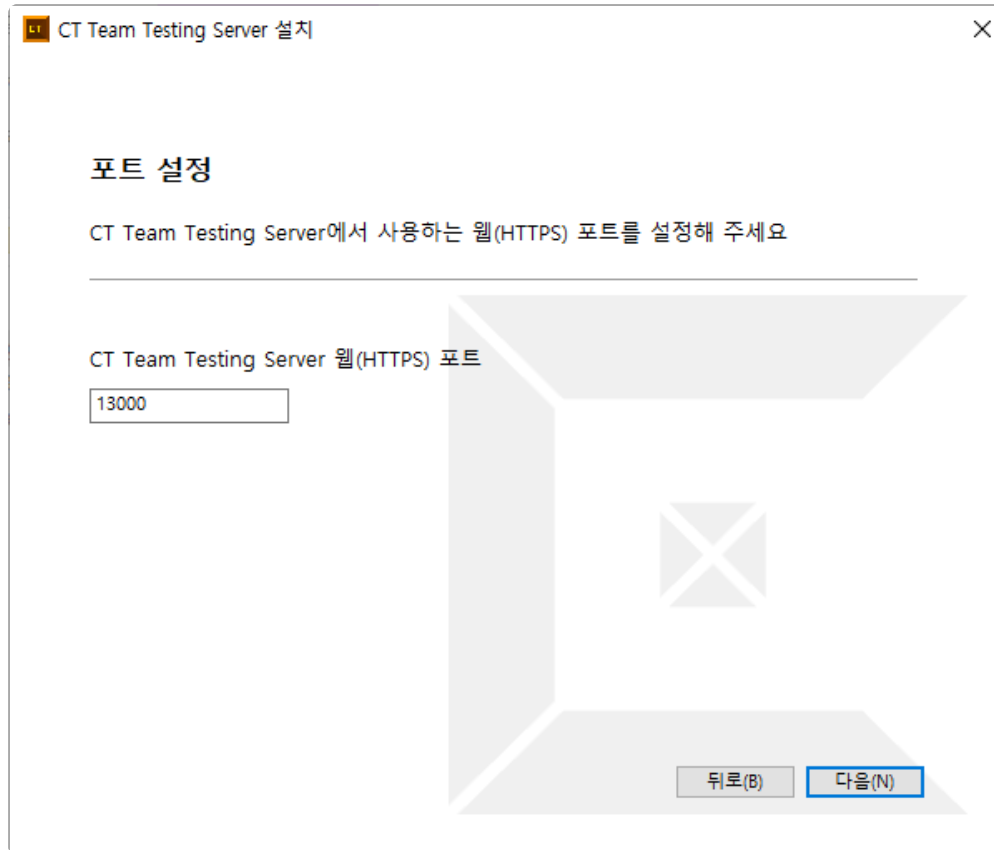
3. 설치 마법사가 실행되고 설치 정보를 수집합니다. [다음]을 클릭합니다.



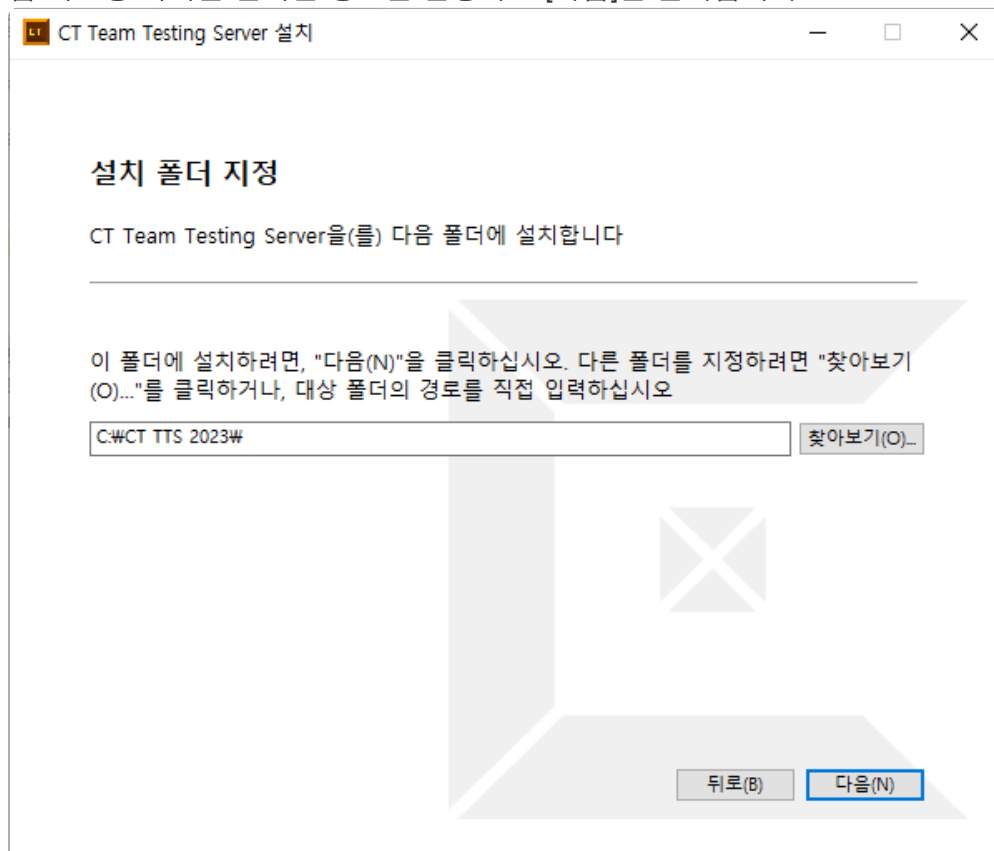
4. 최종 사용자 라이선스를 동의하고 [다음]을 클릭합니다.



5. 팀 테스트 서버에서 사용할 웹 포트를 입력하고 [다음]을 클릭합니다. 기본값은 13000입니다.

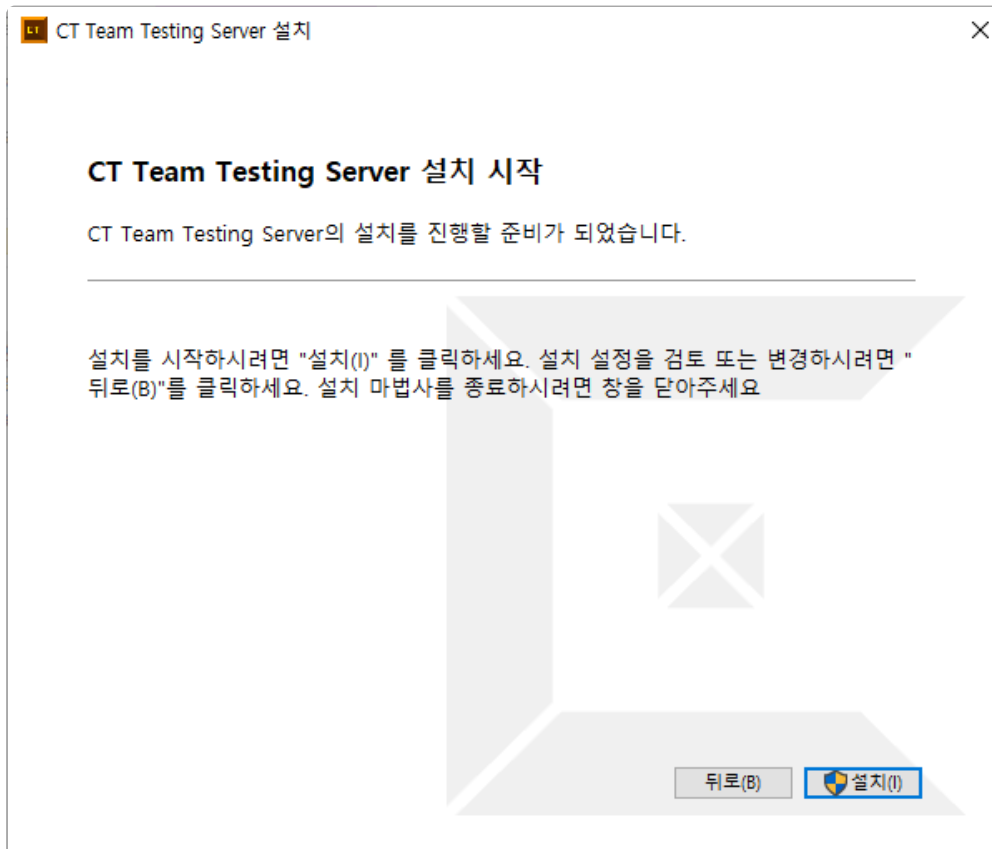


6. 팀 테스트 서버를 설치할 경로를 설정하고 [다음]을 클릭합니다.

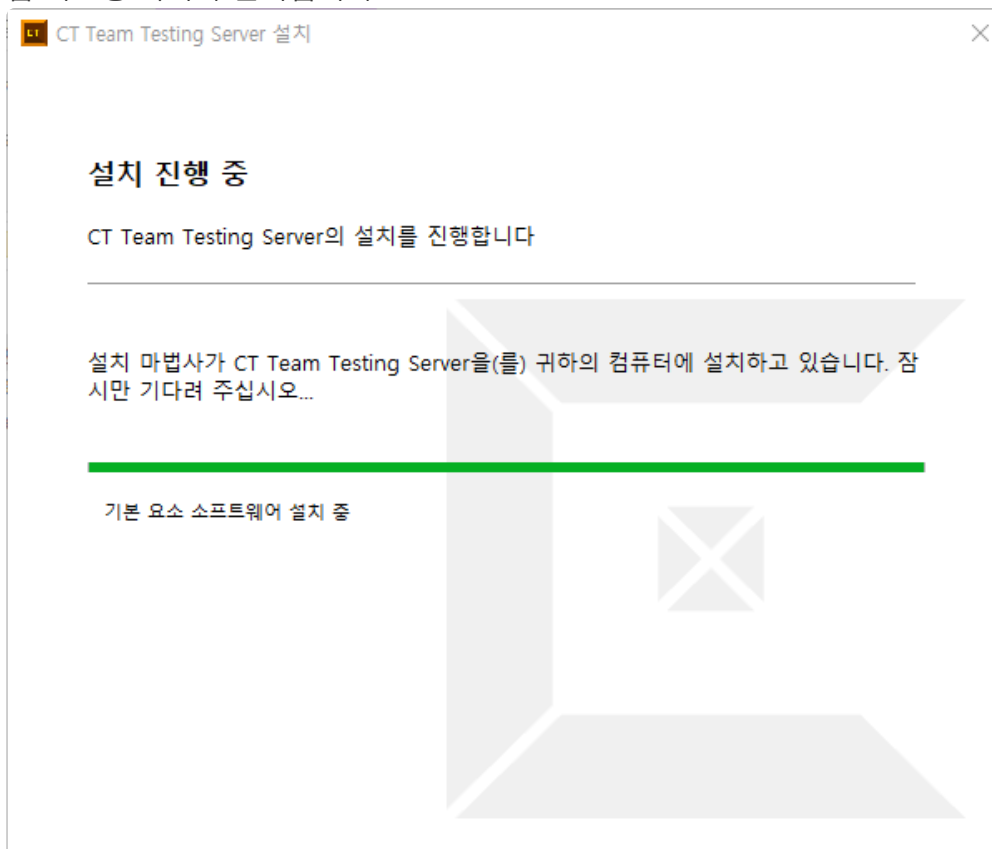


! 영어가 아닌 다른 언어가 포함된 경로에는 팀 테스트 서버를 설치할 수 없습니다.

7. 설치에 필요한 정보를 모두 수집하였습니다. [설치]를 클릭합니다.



8. 팀 테스트 서버가 설치됩니다.



9. 설치 완료 후 팀 테스트 서버를 바로 실행할 때는 [CT Team Testing Server 실행]을 클릭하고 설치를 종료할 때는 [마침]을 클릭합니다.



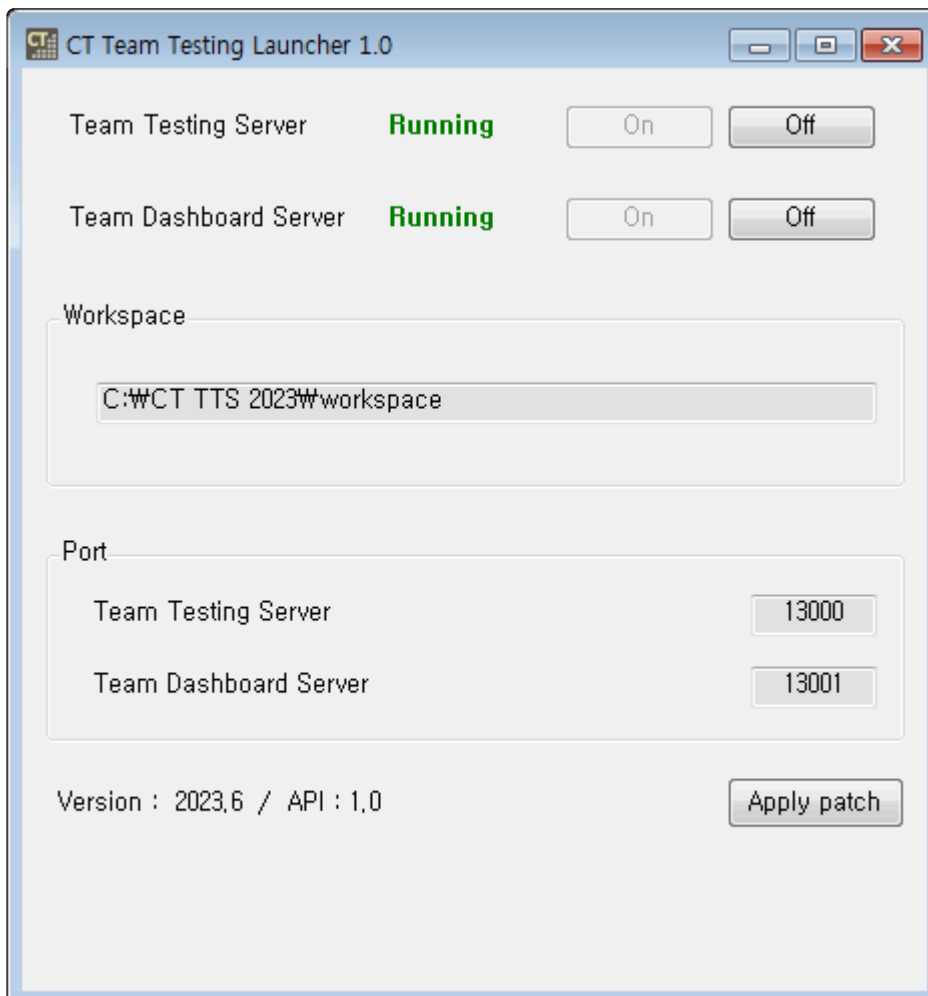
## 31.2. CT 팀 테스트 서버 실행하기

### Launcher 실행하기

{설치 경로}\bin\Launcher.exe를 이용하여 팀 테스트 서버와 팀 테스트 대시보드 서버를 켜고 끄거나 설정을 변경합니다.

<div> <div></div> <div> <div></div> <div>내 PC</div> </div> <div> <div></div> <div>로컬 디스크 (C:)</div> </div> <div> <div></div> <div>CT Team Testing Server</div> </div> <div> <div></div> <div>bin</div> </div> </div>			
이름	수정한 날짜	유형	크기
dashboard	2023-06-01 오후 1:35	파일 폴더	
tts	2023-06-01 오후 12:31	파일 폴더	
configuration.json	2023-06-01 오후 12:32	JSON 파일	1KB
CoverageRecalculator.exe	2023-05-31 오전 9:41	응용 프로그램	1,773KB
<b>Launcher.exe</b>	2023-05-31 오전 9:41	응용 프로그램	47KB
Launcher.exe.config	2023-05-25 오후 1:20	VisualStudio.confi...	1KB
Launcher.pdb	2023-05-31 오전 9:41	VisualStudio.pdb....	74KB
Newtonsoft.Json.dll	2023-05-25 오후 1:20	응용 프로그램 확장	696KB
Newtonsoft.Json.xml	2023-05-25 오후 1:20	XML 파일	686KB

### 팀 테스트 서버 실행하기

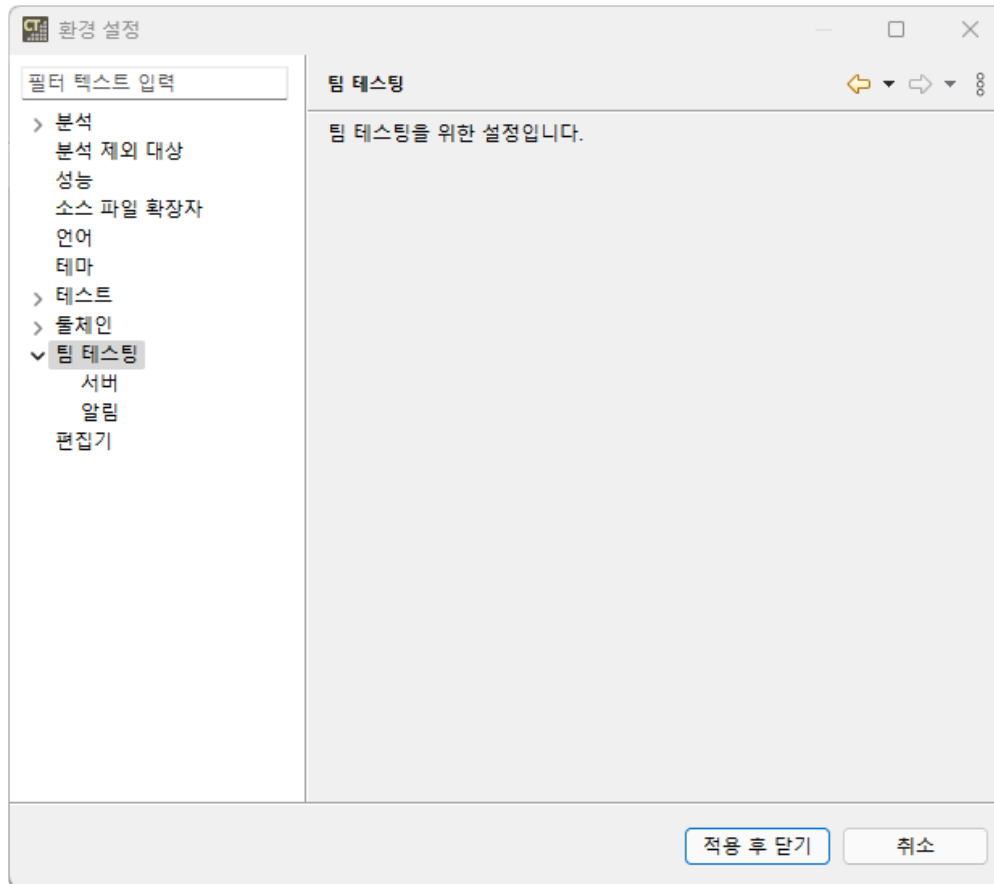


- Launcher를 실행하면 팀 테스트 서버와 팀 테스트 대시보드 서버가 켜집니다. [On]/[Off] 버튼을 이용하여 수동으로 켜고 끌 수 있습니다.
- [Version]에서 팀 테스트 서버의 버전을 확인할 수 있으며 [Apply patch] 버튼을 클릭하여 패치를 적용할 수 있습니다.

 패치 적용 후에는 서버가 재실행됩니다.

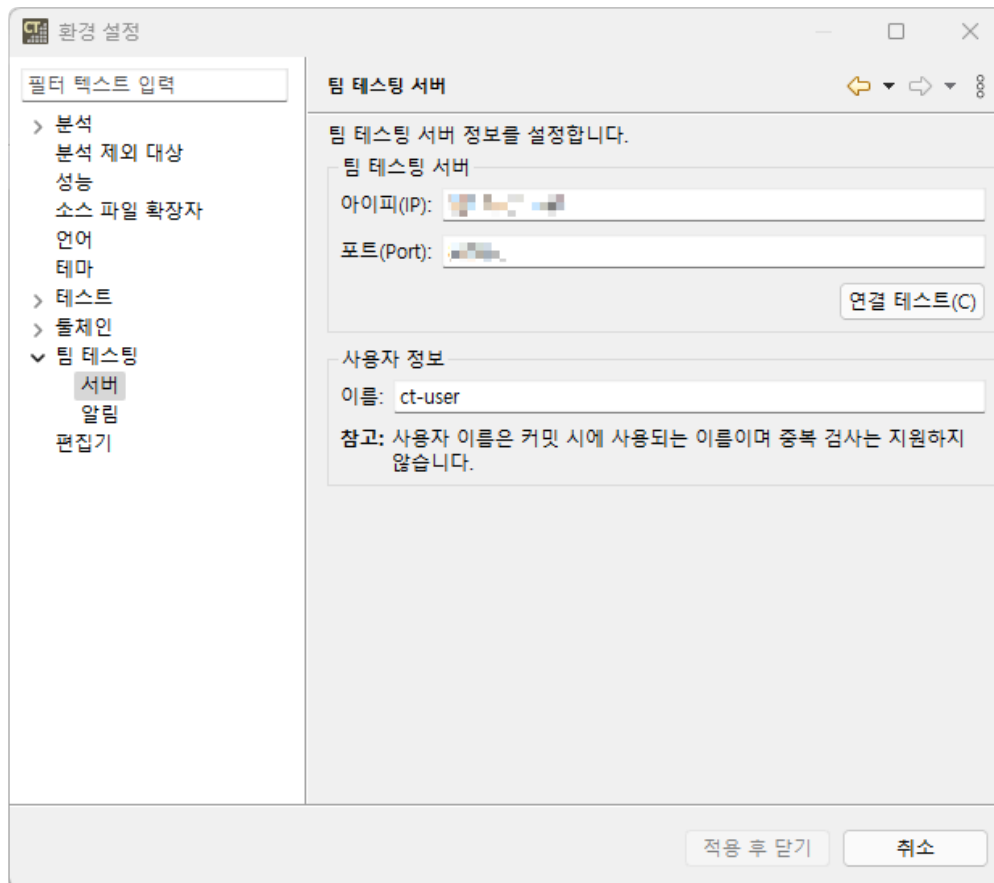
## 31.3. 환경설정

팀 프로젝트를 생성하거나 가져오기 전에 [환경설정] > [팀 테스트]에서 팀 테스트 정보를 입력해야 합니다.



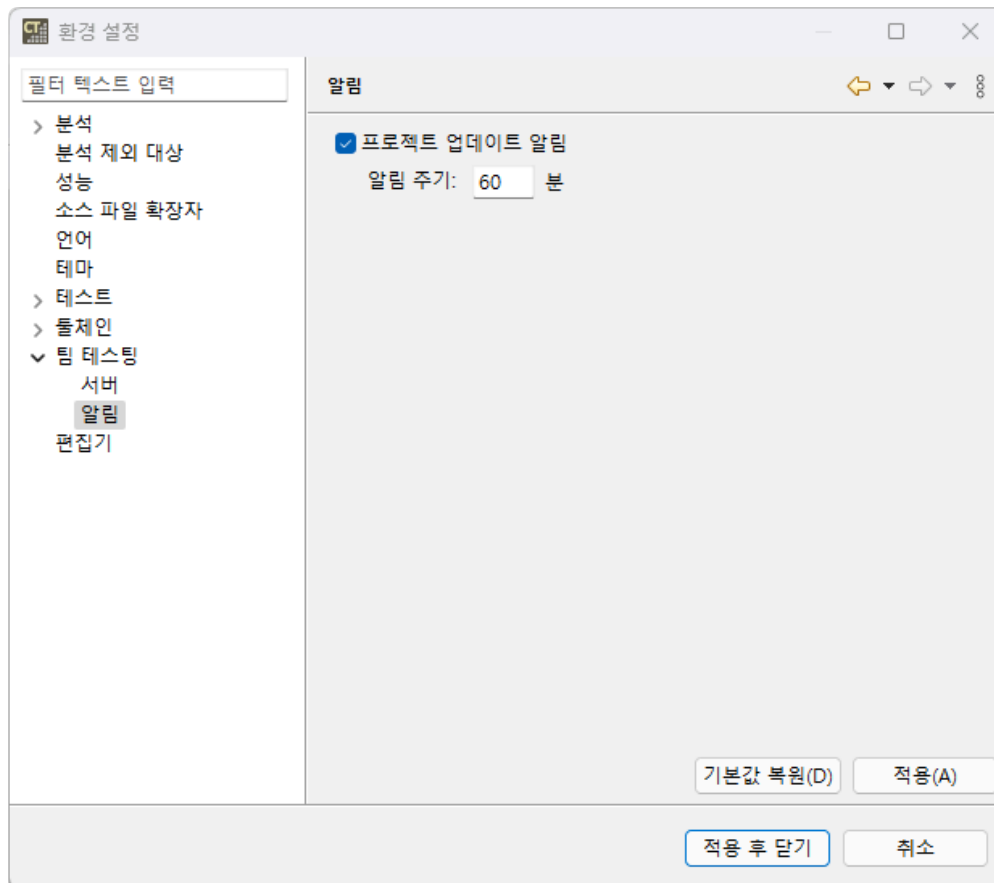


## 서버



- 팀 테스트 서버: 팀 테스트 서버의 아이피와 포트를 입력합니다. 입력 후 [연결 테스트]를 클릭하여 해당 서버와의 연결을 확인할 수 있습니다.
- 사용자 정보: 팀 테스트 서버에서 사용할 이름을 입력합니다.

## 알림



- 팀 프로젝트에 업데이트가 있을 때 알림을 띄울지 선택하고 알림 주기를 설정합니다.
- 알림 주기의 기본값은 60분이며 5분 이상으로 설정해야 합니다.

## 31.4. 팀 프로젝트

---

CT 2023.12은 팀 프로젝트를 사용하기 위해 아래와 같은 기능을 제공합니다.

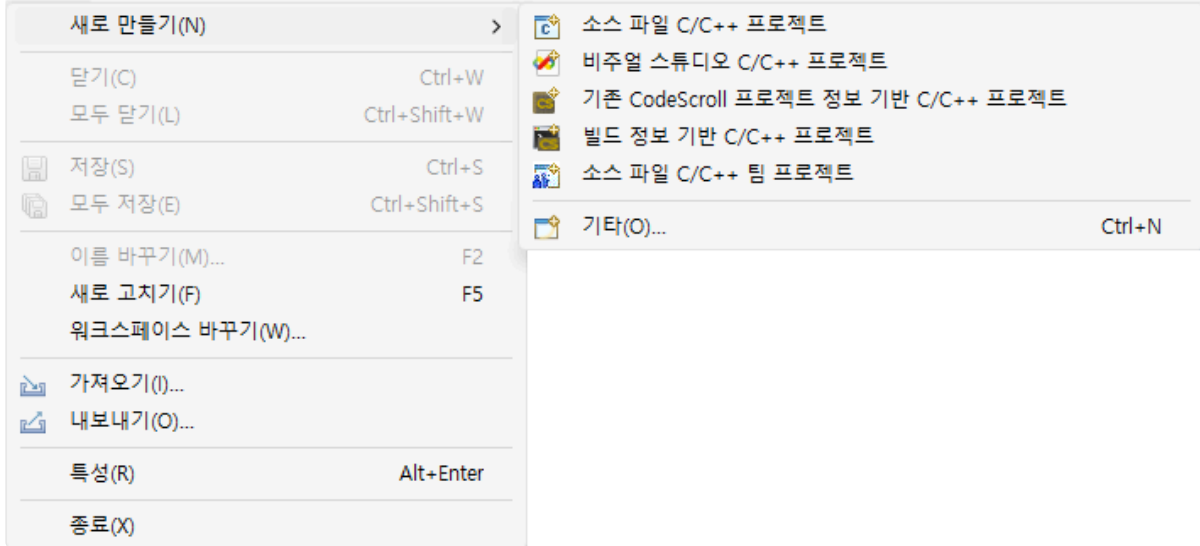
- [팀 프로젝트 생성하기](#)
- [팀 프로젝트 특성](#)
- [팀 프로젝트 가져오기](#)
- [팀 프로젝트 변환하기](#)

## 31.4.1. 팀 프로젝트 생성하기

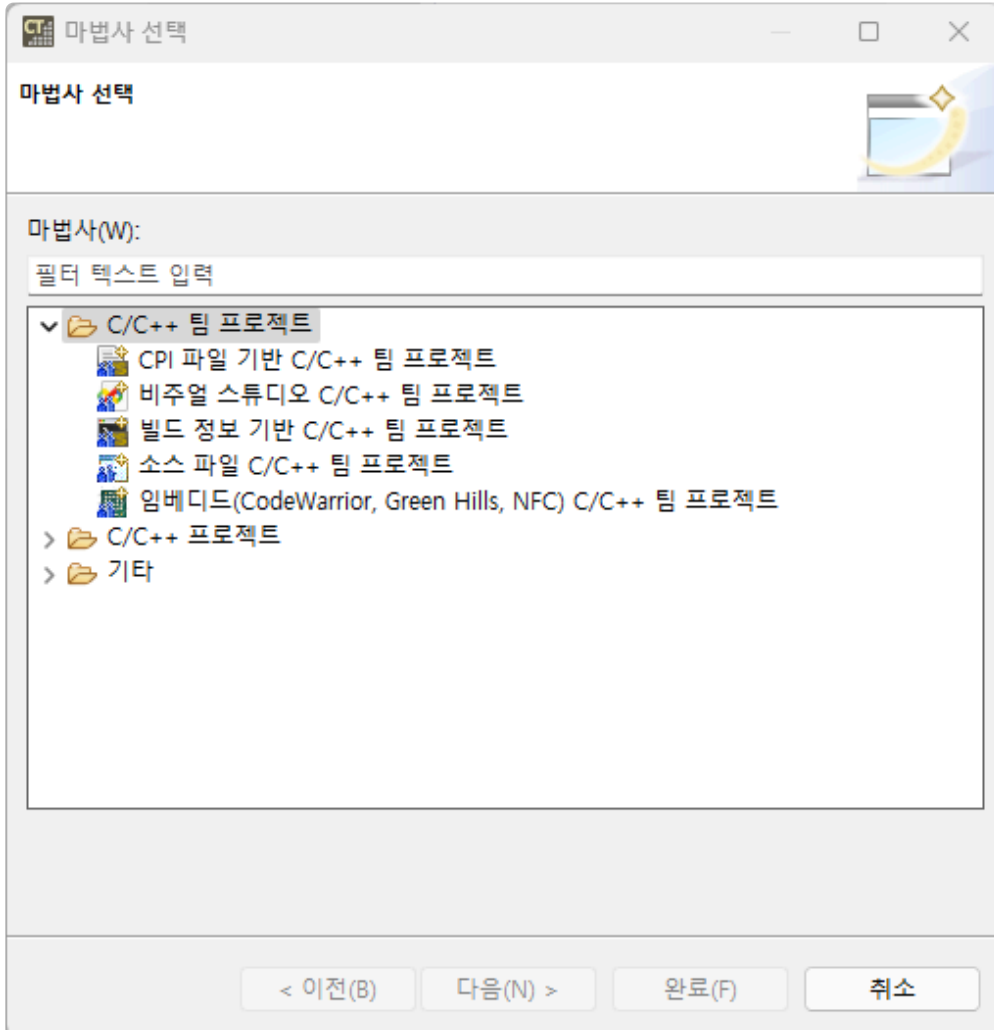
다른 사용자와 하나의 프로젝트를 테스트하기 위해 프로젝트를 서버에 생성해야 합니다.

1. 팀 프로젝트를 생성하려면 [프로젝트 생성] 마법을 실행합니다. [파일] > [새로 만들기] 또는 단축 아이콘을 클릭합니다.

파일(F) 편집(E) 검색(A) 프로젝트(P) 창(W) 도움말(H)

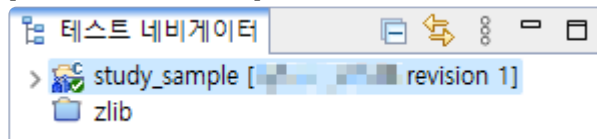


2. [프로젝트 생성] 마법사에서 생성할 팀 프로젝트를 선택하고 [다음]을 클릭합니다.



3. 환경 설정에서 입력한 팀 테스트 서버와 사용자 정보를 확인한 후 [다음]을 클릭합니다. [팀 테스트 서버 설정...]을 클릭하여 팀 테스트 서버와 사용자 정보를 수정할 수 있습니다.

4. 이후의 과정은 일반 프로젝트 생성과 같습니다. 이 문서의 [\[프로젝트 만들기\]](#) 페이지나 [\[RTV 프로젝트 만들기\]](#) 참고하시기 바랍니다.
5. [테스트 네비게이터] 뷰에서 생성된 팀 프로젝트의 서버 IP와 리비전을 확인할 수 있습니다.

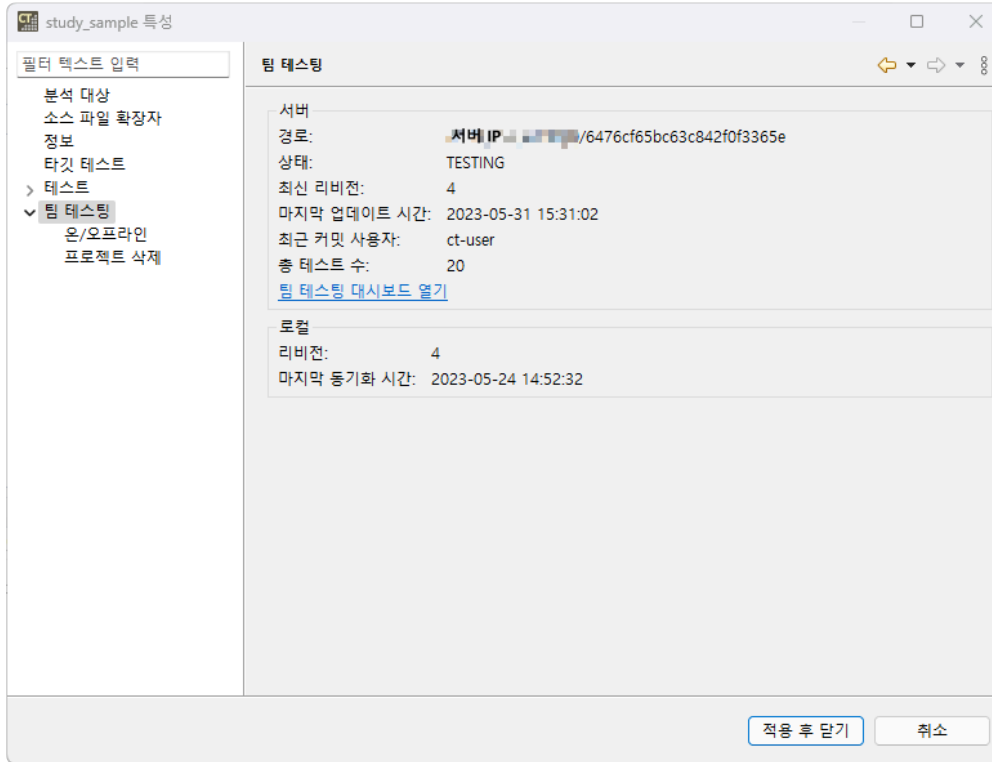


## 31.4.2. 팀 프로젝트 특성

팀 프로젝트의 정보는 프로젝트 특성 페이지에서 확인할 수 있습니다.

### 팀 테스트

서버와 로컬의 프로젝트 정보를 확인할 수 있습니다.



- 서버의 정보를 확인할 수 있습니다.
- [팀 테스트 대시보드 열기]를 클릭하여 팀 테스트 대시보드에서 프로젝트 정보를 확인할 수 있습니다. 팀 테스트 대시보드에 관한 자세한 내용은 이 문서의 [\[CT 팀 테스트 대시보드\]](#) 페이지를 참고하시기 바랍니다.
- 로컬의 리비전과 마지막 업데이트 시간을 확인할 수 있습니다.

### 온/오프라인

서버와 연결 여부를 선택할 수 있습니다. 온/오프라인 설정에 관한 자세한 내용은 User Guides 2023.12의 [\[온/오프라인 모드\]](#) 페이지를 참고하시기 바랍니다.

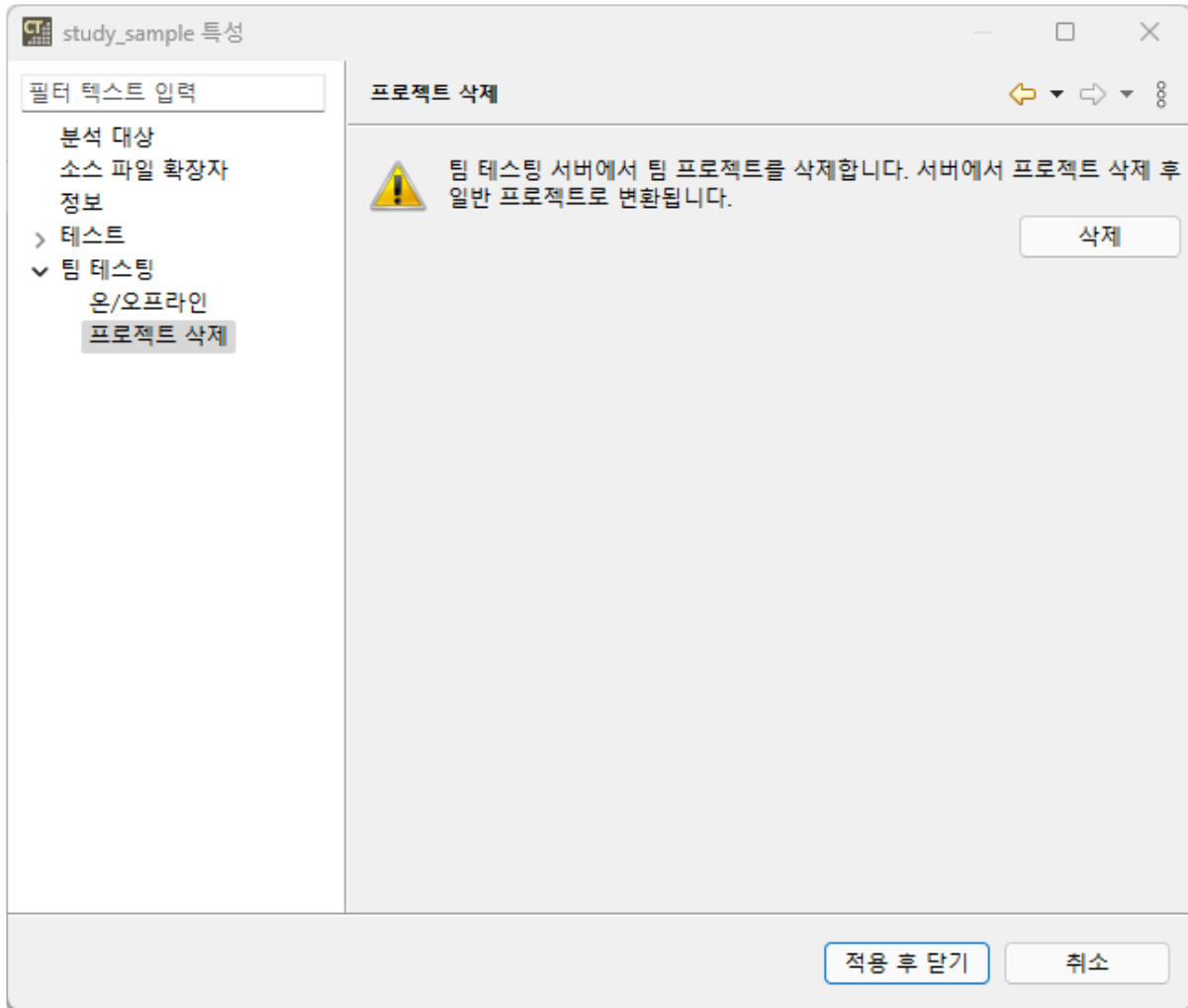


- 오프라인을 설정할 경우 일부 팀 테스트 기능을 사용할 수 없습니다.
  - 테스트 실행 후 자동 커밋
  - 프로젝트 업데이트 알림

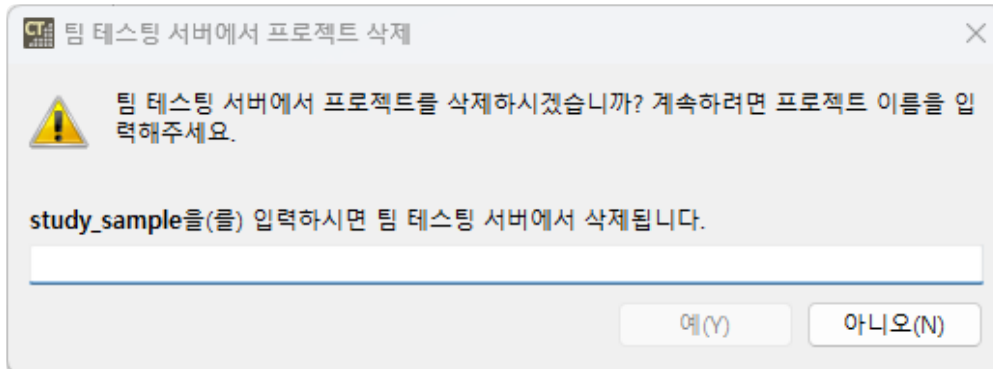
## 프로젝트 삭제

서버에 있는 팀 프로젝트를 삭제할 수 있습니다. 서버의 프로젝트를 삭제하면 로컬의 팀 프로젝트는 일반 프로젝트로 전환됩니다.

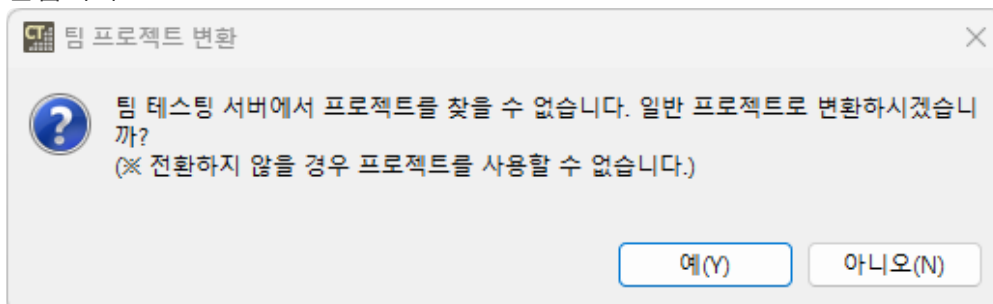
1. [팀 테스트] > [프로젝트 삭제]에서 [삭제]를 클릭합니다.



2. 프로젝트 이름을 입력한 후 [확인]을 누르면 서버에서 프로젝트가 삭제됩니다.



3. 삭제된 프로젝트와 연결된 다른 사용자가 있는 경우, 해당 사용자의 프로젝트는 일반 프로젝트로 변환됩니다.

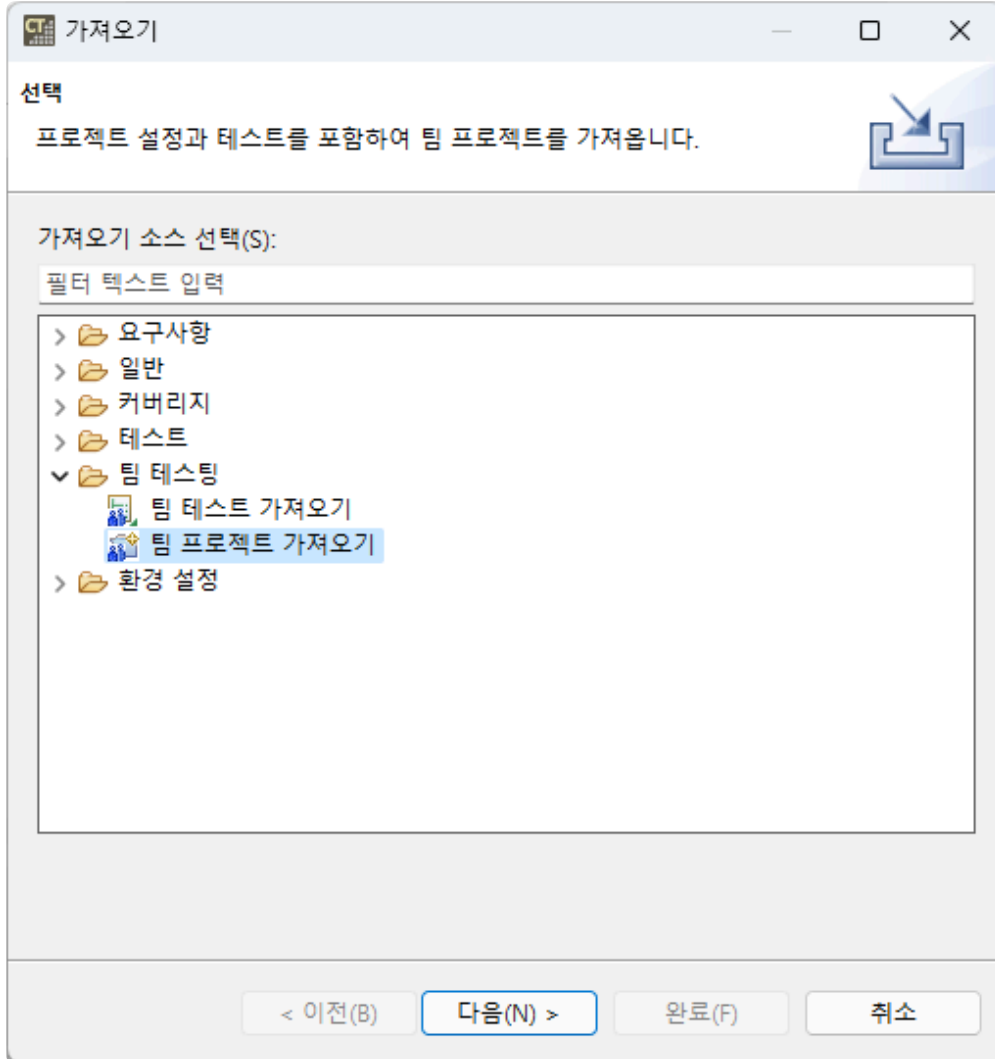




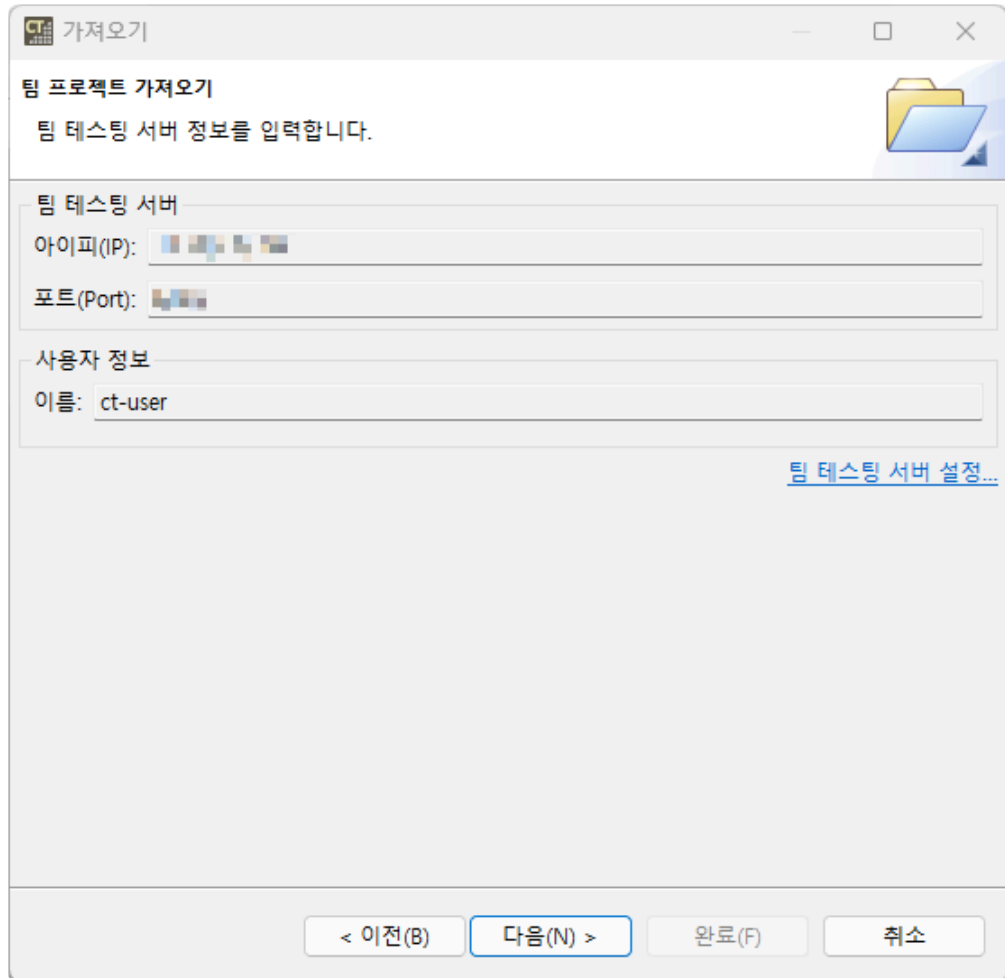
## 31.4.3. 팀 프로젝트 가져오기

다른 사용자가 팀 서버에 올려둔 프로젝트를 가져올 수 있습니다. 팀 프로젝트 가져오기의 경우, 테스트를 제외한 프로젝트의 모든 자원을 최신 리비전으로 가져옵니다. 테스트를 가져오려면 이 문서의 [\[팀 테스트 가져오기\]](#) 페이지를 참고하시기를 바랍니다.

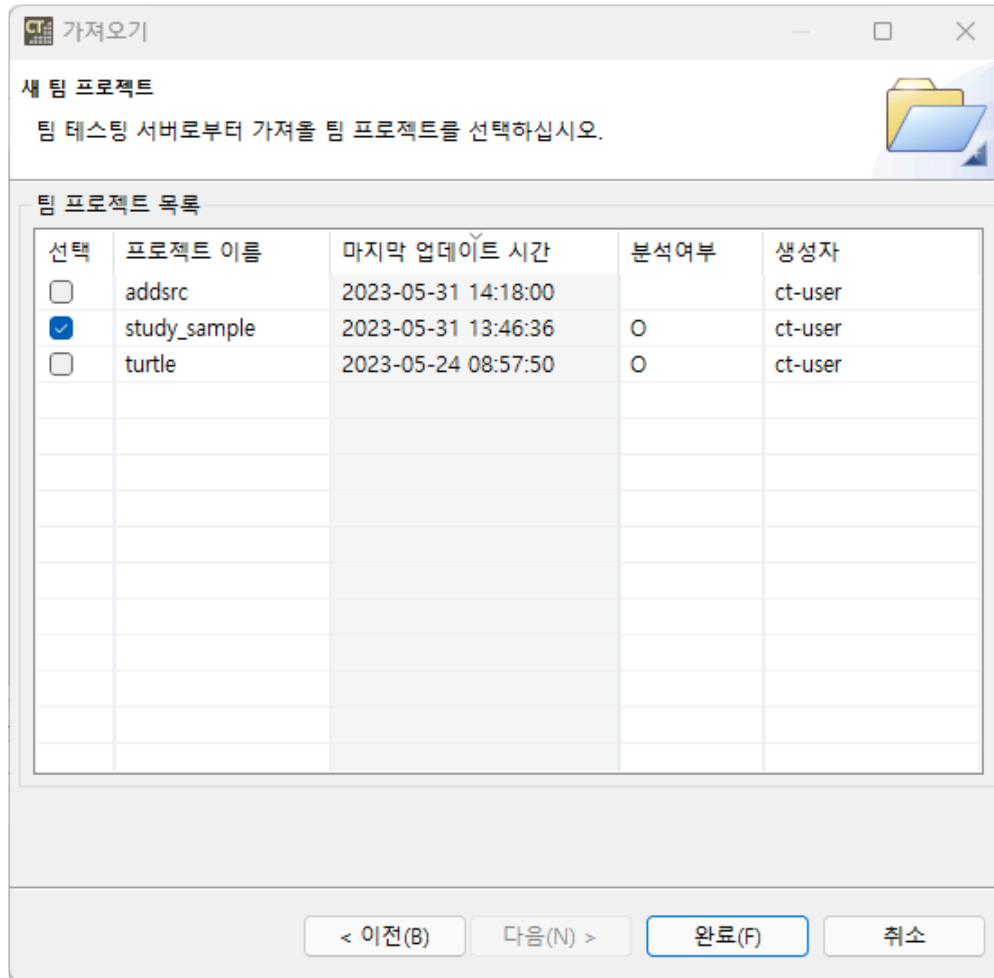
1. [가져오기] 마법사에서 [팀 테스트] > [팀 프로젝트 가져오기]를 선택하고 [다음]을 클릭합니다.



2. 팀 테스트 서버 정보를 확인하거나 입력한 후 [다음]을 클릭합니다.

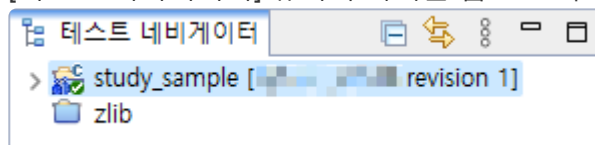


- [팀 테스트 서버 설정...]을 선택하여 환경설정 페이지에서 팀 테스트 서버 정보를 입력할 수 있습니다.
3. 서버에 있는 팀 프로젝트 목록에서 가져올 프로젝트를 선택한 후 [완료]를 클릭합니다. 목록은 마지막 업데이트 시간을 기준으로 정렬합니다.



✿ 가져올 팀 프로젝트와 같은 이름의 프로젝트가 이미 로컬에 있는 경우, 기존 프로젝트의 이름을 바꾼 뒤 팀 프로젝트를 가져오시기 바랍니다.

4. [테스트 네비게이터] 뷰에서 가져온 팀 프로젝트의 서버 IP와 리비전을 확인할 수 있습니다.



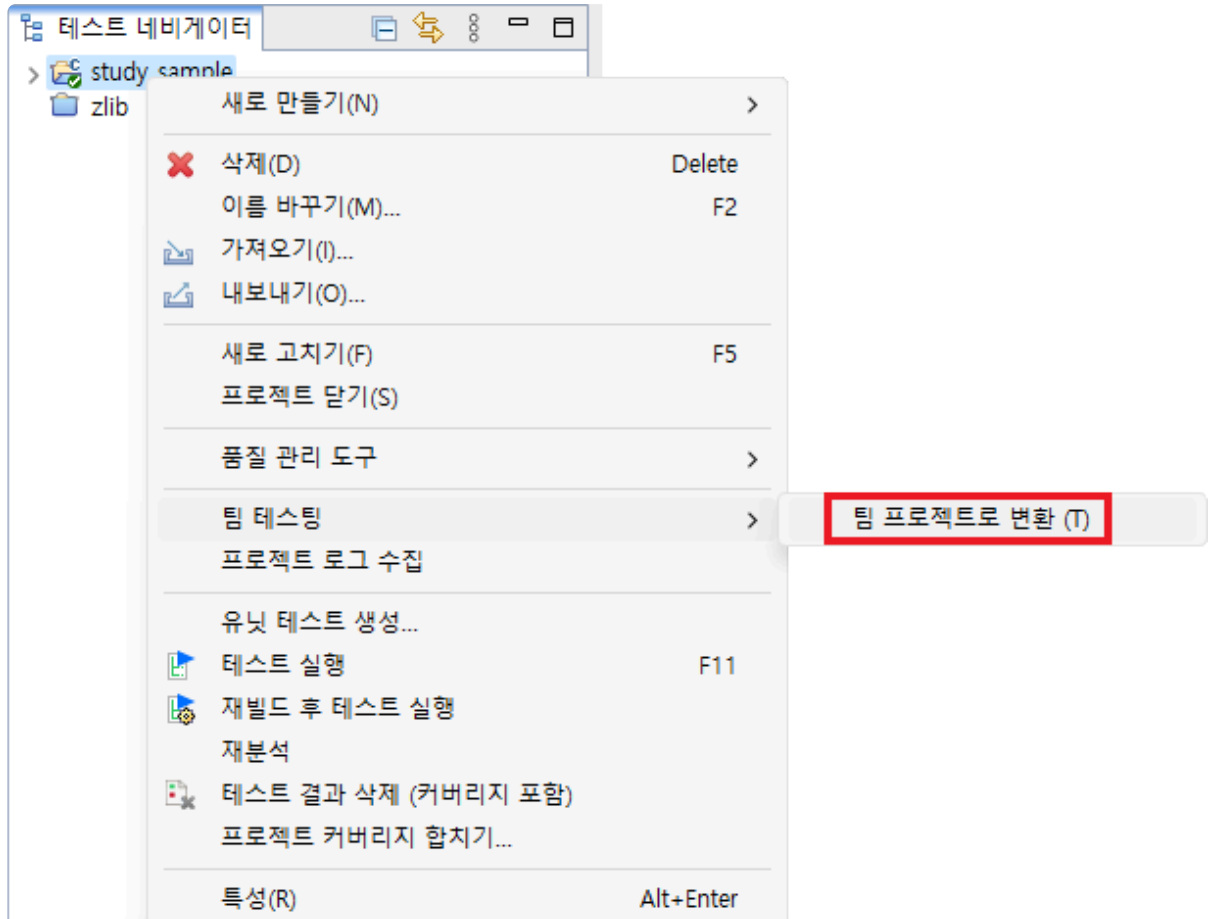
## 31.4.4. 팀 프로젝트 변환하기

일반 프로젝트를 팀 프로젝트로 변환하거나 팀 프로젝트를 일반 프로젝트로 변환할 수 있습니다. 팀 프로젝트로 변환하는 경우, 현재 프로젝트의 모든 자원과 테스트가 서버에 올라갑니다.

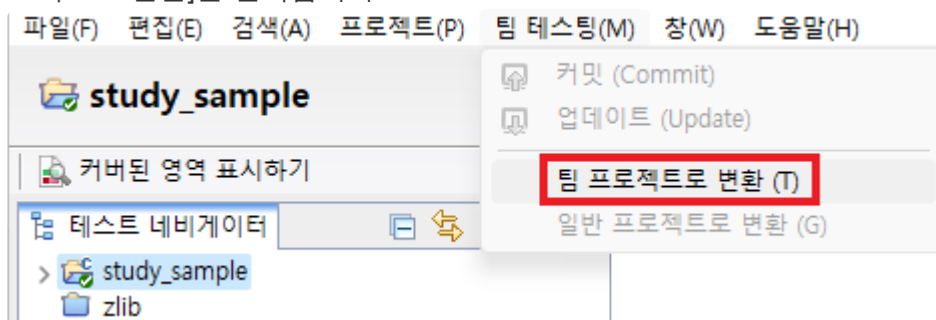
### 일반 프로젝트를 팀 프로젝트로 변환하기

1. 아래 두 가지 방법 중 하나를 이용하여 [팀 프로젝트로 변환]을 선택합니다.

- [테스트 네비게이터] 뷰에서 변환할 프로젝트를 오른쪽 클릭하여 [팀 테스트] > [팀 프로젝트로 변환]을 선택합니다.



- [테스트 네비게이터] 뷰에서 변환할 프로젝트를 선택한 후, 최상위 메뉴의 [팀 테스트] > [팀 프로젝트로 변환]을 선택합니다.

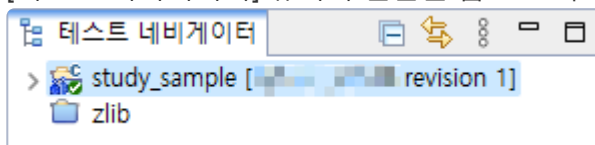


2. 프로젝트를 올릴 서버의 정보와 사용자 정보를 확인합니다.

✿ 서버 정보와 사용자 정보를 변경하는 방법은 이 문서의 [환경설정](#) 페이지를 참고하시기를 바랍니다.

3. 프로젝트 변환이 완료되었습니다.

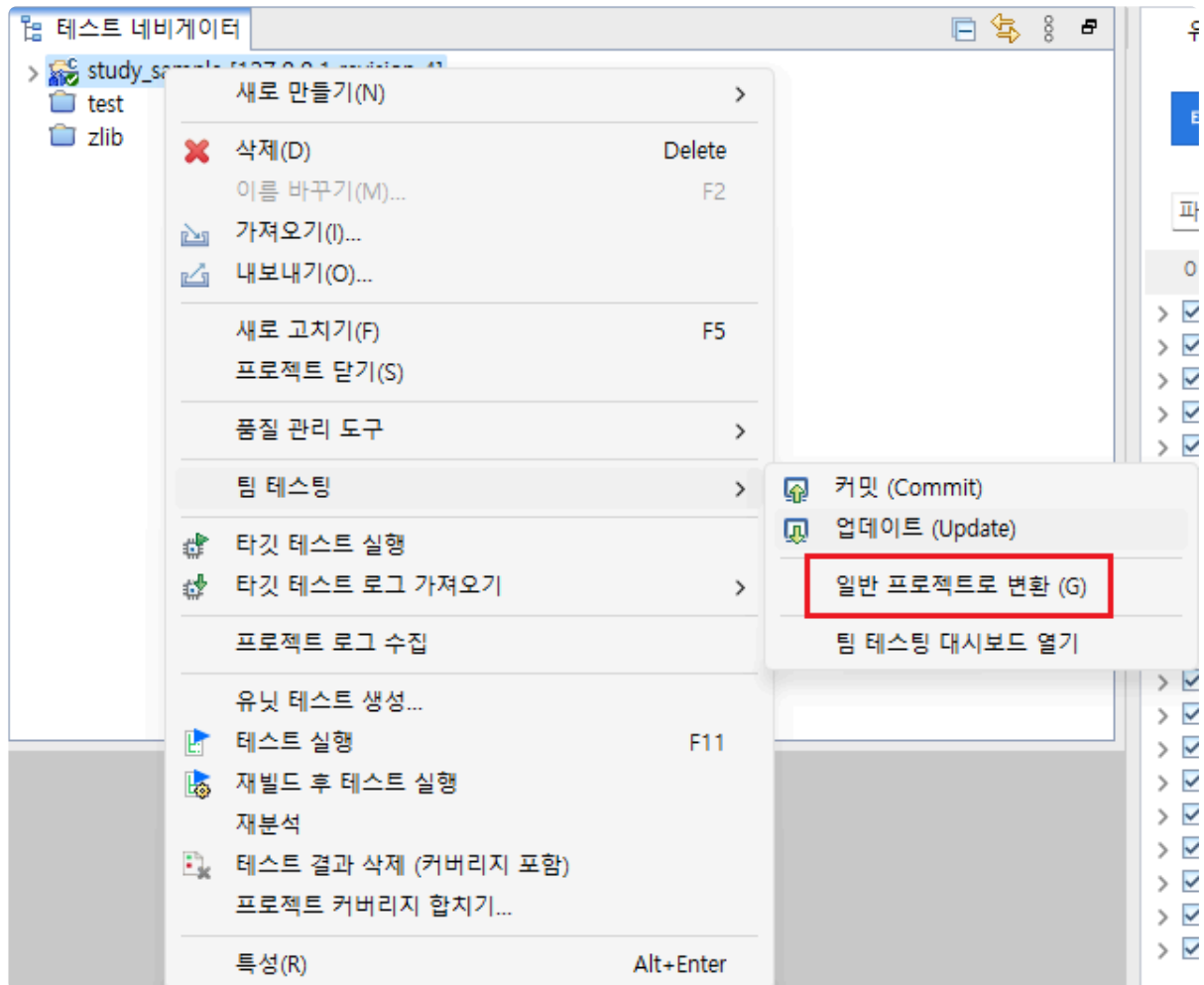
4. [테스트 네비게이터] 뷰에서 변환된 팀 프로젝트의 서버 IP와 리비전을 확인할 수 있습니다.



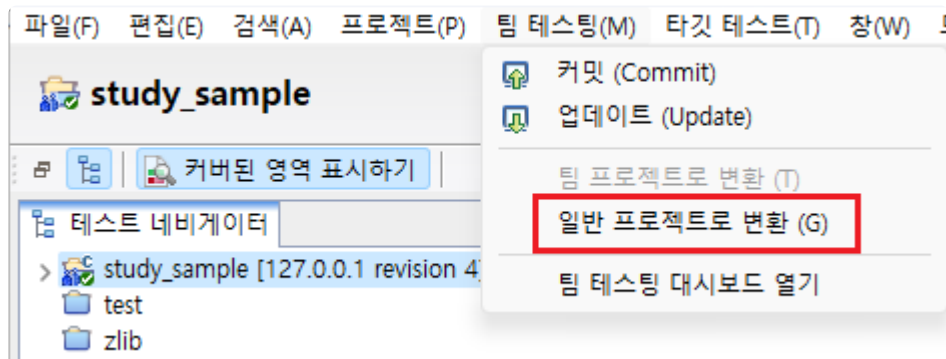
## 팀 프로젝트를 일반 프로젝트로 변환하기

팀 프로젝트를 더 이상 서버와 연결하고 싶지 않을 때, 해당 팀 프로젝트를 일반 프로젝트로 변환할 수 있습니다. 일반 프로젝트로 변환하여도 서버에 팀 프로젝트는 삭제되지 않습니다. 아래 두 가지 방법 중 하나를 이용하여 [일반 프로젝트로 변환]을 선택합니다.

- [테스트 네비게이터] 뷰에서 변환할 팀 프로젝트를 오른쪽 클릭하여 [팀 테스트] > [일반 프로젝트로 변환]을 선택합니다.



- [테스트 네비게이터] 뷰에서 변환할 팀 프로젝트를 선택한 후, 최상위 메뉴의 [팀 테스트] > [일반 프로젝트로 변환]을 선택합니다.



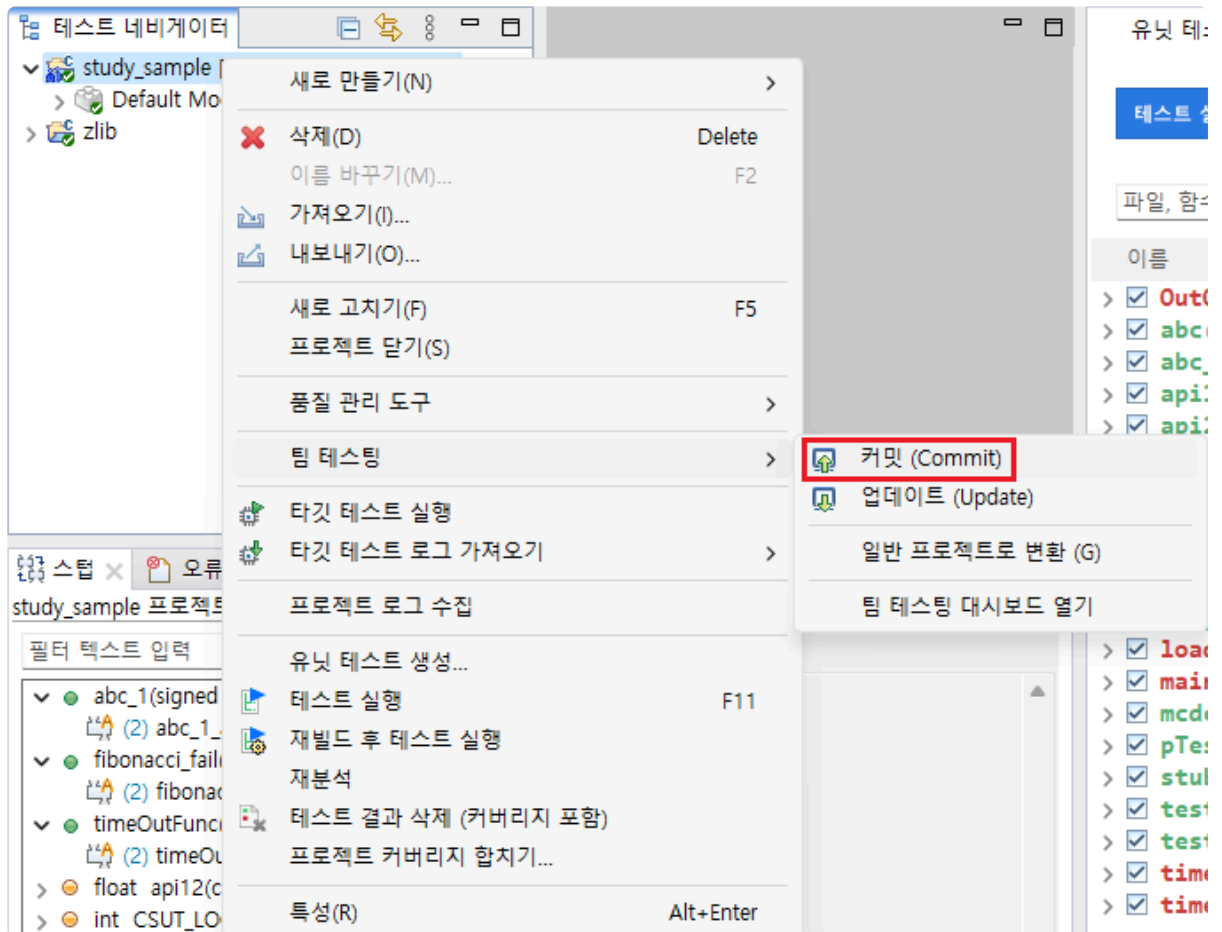
## 31.5. 커밋 (Commit)

[커밋 (Commit)] 기능을 사용하여 사용자가 로컬에서 작업한 내용을 팀 서버에 반영할 수 있습니다. 커밋을 하려면 서버의 리비전과 로컬의 리비전이 같아야 합니다. 서버와 로컬의 리비전이 다른 경우, 업데이트 대화 상자가 뜰 수 있습니다.

### 수동 커밋

1. 아래 두 가지 방법 중 하나를 선택하여 수동으로 커밋합니다.

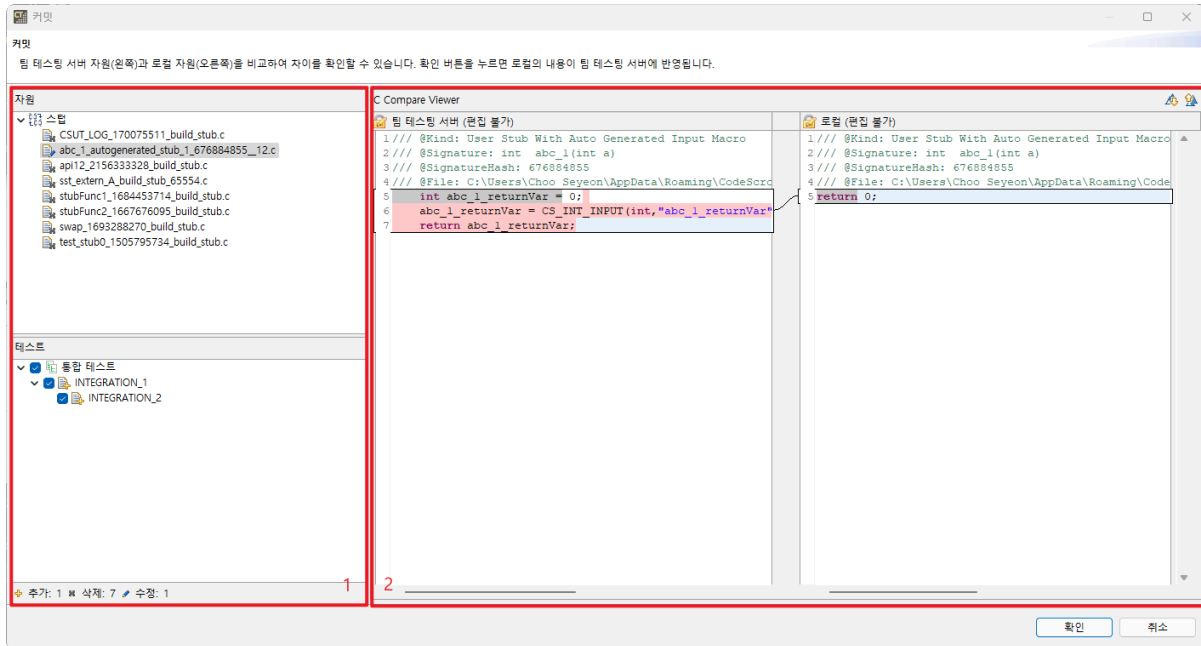
- [테스트 네비게이터 뷰]에서 프로젝트의 컨텍스트 메뉴에서 [팀 테스트] > [커밋 (Commit)]을 선택합니다.






- 도구 상단의 [팀 테스트] > [커밋 (Commit)]을 선택합니다.

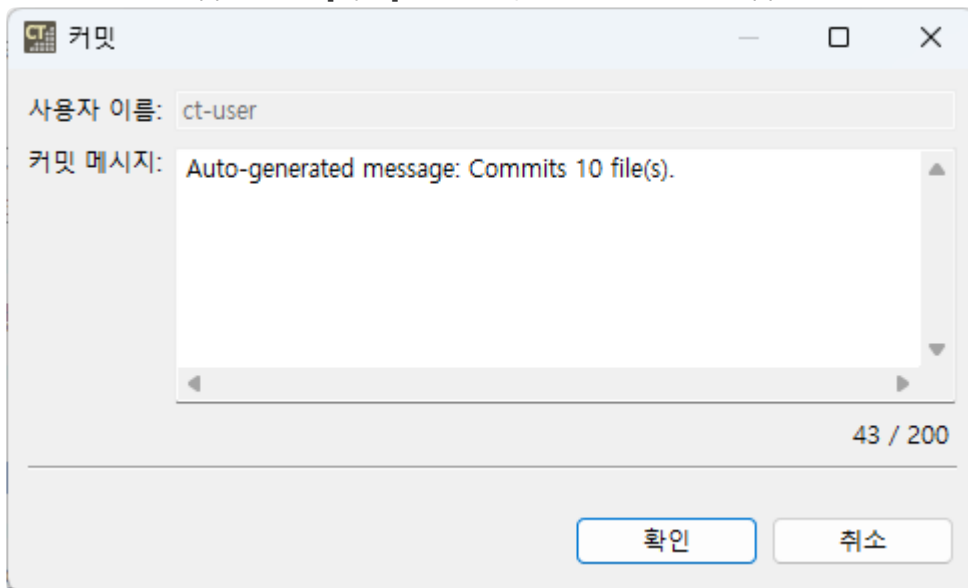


2. 커밋 대화상자에서 커밋할 내용을 확인한 후 [확인] 버튼을 클릭합니다.



- 1번 영역에서 커바할 자원과 테스트를 확인할 수 있습니다.
  - 추가, 삭제, 변경된 자원이나 테스트는 각각 , ,  로 표시됩니다.
- 2번 영역에서 서버 형상(왼쪽)과 커바할 로컬 형상(오른쪽)을 비교할 수 있습니다.
  - 서버 형상과 로컬 형상은 수정할 수 없습니다.

3. 커바 메시지를 입력한 후 [확인] 버튼을 누르면 커바가 진행됩니다.



✿ 커바하는 데이터가 많으면 시간이 오래 걸릴 수 있습니다.

## 자동 커바

테스트 실행 후, 프로젝트의 변경 사항과 테스트 결과가 자동으로 커바됩니다. 자동으로 커바될 때는 커바 대화상자가 뜨지 않습니다.



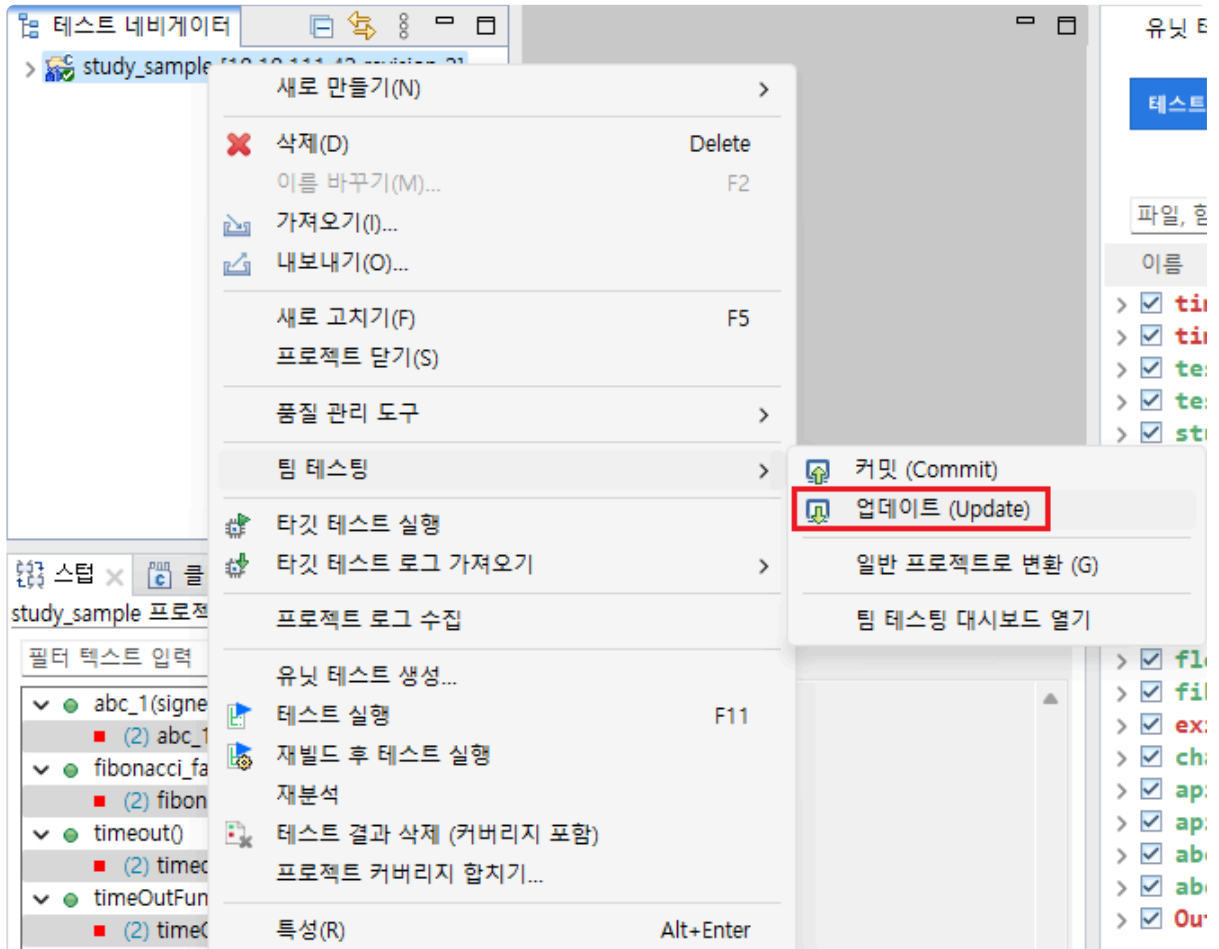
## 31.6. 업데이트 (Update)

[업데이트 (Update)] 기능을 사용하여 서버의 최신 형상으로 로컬 형상을 업데이트할 수 있습니다.

### 수동 업데이트

1. 아래 두 가지 방법 중 하나를 선택하여 수동으로 업데이트합니다.

- [테스트 네비게이터 뷰]에서 프로젝트의 컨텍스트 메뉴에서 [팀 테스트] > [업데이트 (Update)]를 선택합니다.



- 도구 상단의 [팀 테스트] > [업데이트 (Update)]를 선택합니다.



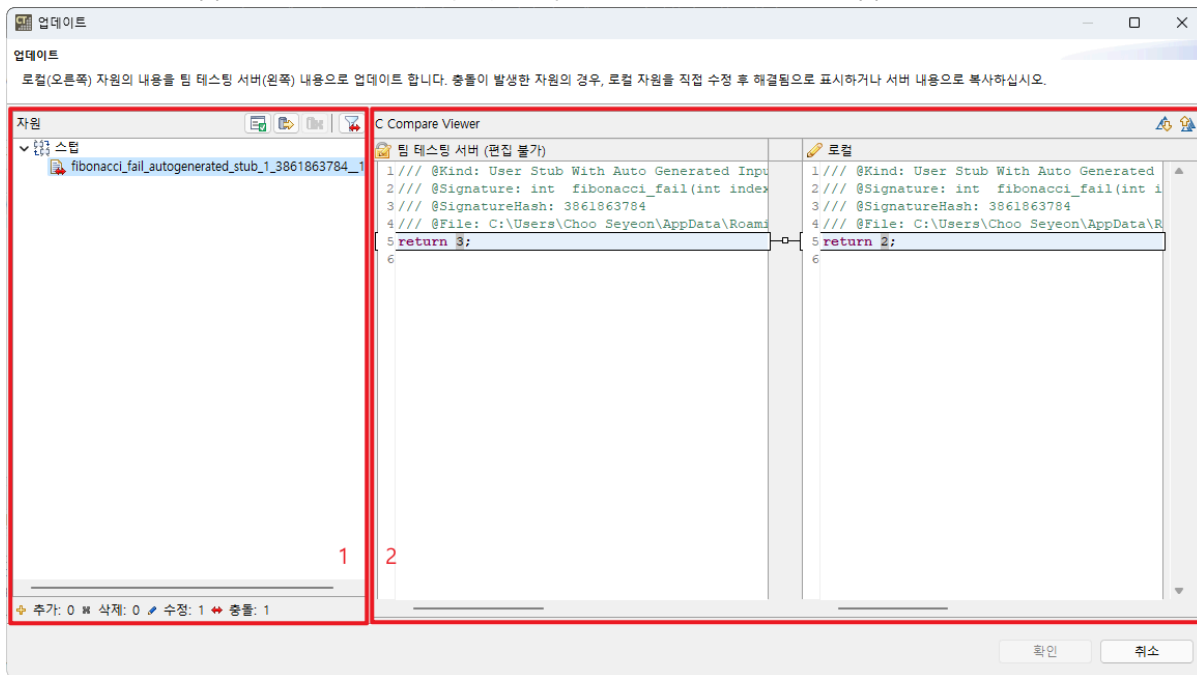
2. 구성에 변경이 있거나 자원에 충돌이 있는 경우, 또는 테스트와 연관된 공유 자원이 변경되었을 때 업데이트 대화상자가 나타납니다. 그 외에는 자동으로 업데이트가 반영됩니다. 업데이트 대화상자는 [충돌 관리] 단락에서 설명합니다.







## 자동 업데이트





다른 사용자와 테스트 환경을 똑같이 맞추기 위해서는 로컬의 리비전과 서버의 리비전을 항상 동일하게 유지하여야 합니다. 로컬 리비전이 최신이 아닌 경우, 테스트의 결과를 보장할 수 없으며 로컬의 작업 내용을 서버에 올릴 수 없습니다. 로컬의 리비전을 최신으로 유지하기 위해 CT 2023.12는 주기적으로 업데이트를 합니다. 주기적으로 업데이트 알림이 뜨며, 커밋 등 서버와의 연결이 필요한 기능을 사용하기 전에 추가로 리비전을 확인합니다.

## 충돌 관리



로컬에서 수정한 자원이 서버에서도 변경된 경우, 업데이트할 때 충돌이 발생합니다. CT 2023.12에서는 충돌을 해결하고 업데이트를 반영할 수 있도록 충돌 관리 기능을 제공합니다.



- 모든 자원의 충돌이 해결되면 [확인] 버튼이 활성화됩니다.
- 1번 영역에서는 업데이트 중 충돌이 발생한 자원과 테스트 목록을 보여줍니다.
  - 추가, 삭제, 변경, 충돌 자원이나 테스트는 각각 , , , 로 표시됩니다.
  - 충돌이 해결된 자원은 , 서버 형상으로 복사한 자원은 로 표시됩니다.

툴바 아이콘	설명
 해결됨으로 표시	해결됨으로 표시한 자산은 로컬 형상으로 저장됩니다.
 서버에서 로컬로 복사	서버 형상으로 저장됩니다. 서버에서 로컬로 복사 표시가 된 자산은 로컬 형상을 수정할 수 없습니다. 자산을 오른쪽 클릭해서 해당 기능을 사용할 수도 있습니다.
 복사 취소	서버에서 로컬로 복사를 취소하고 로컬 자원을 편집 가능한 상태로 변경합니다.
 충돌 자원만 표시	충돌 자원만 필터링합니다.

- 2번 영역에서는 서버 형상(오른쪽)과 로컬 형상(왼쪽)을 비교할 수 있습니다.

- 수정이 가능한 경우에는  , 불가능한 경우에는  로 표시됩니다.
- 로컬 형상을 수정한 후 해결됨으로 표시하면 수정된 로컬 형상이 프로젝트에 반영됩니다.

## 31.7. 팀 테스트

---

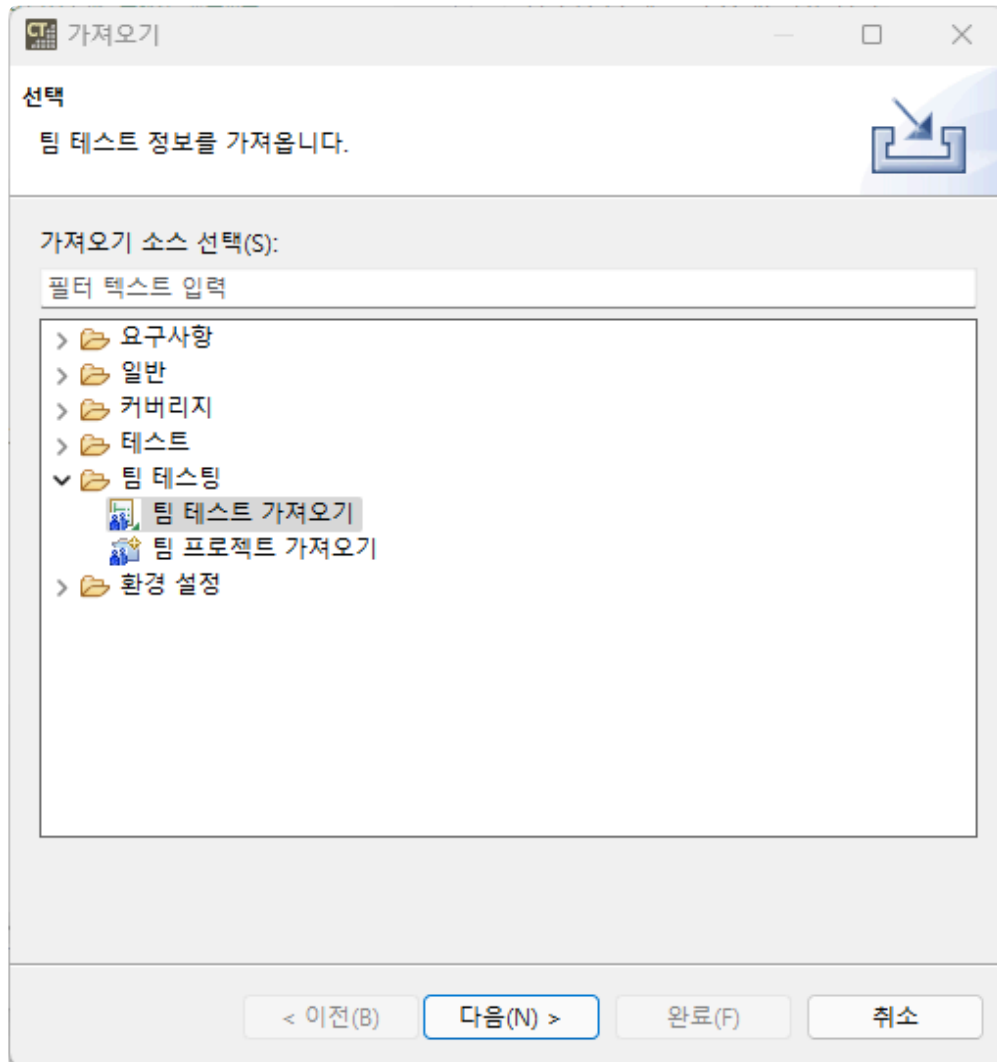
CT 2023.12은 팀 프로젝트에서 테스트를 진행하기 위한 여러 기능을 제공합니다.

- [팀 테스트 가져오기](#)
- [팀 테스트 삭제](#)

## 31.7.1. 팀 테스트 가져오기

다른 사용자가 팀 테스트 서버에 올려둔 테스트를 가져올 수 있습니다.

1. [가져오기] 마법사에서 [팀 테스트] > [팀 테스트 가져오기]를 선택하고 [다음]을 클릭합니다.



2. [가져올 테스트 선택...] 버튼을 클릭하여 팀 테스트 서버에서 가져올 테스트를 선택한 후 [완료]를 클릭합니다.

**팀 테스트 가져오기**

팀 테스트 서버로부터 가져올 테스트를 선택하십시오.

**유닛 테스트**

총 테스트	5	
이미 가져온 테스트	0	
가져올 테스트	0	

가져올 테스트 선택...

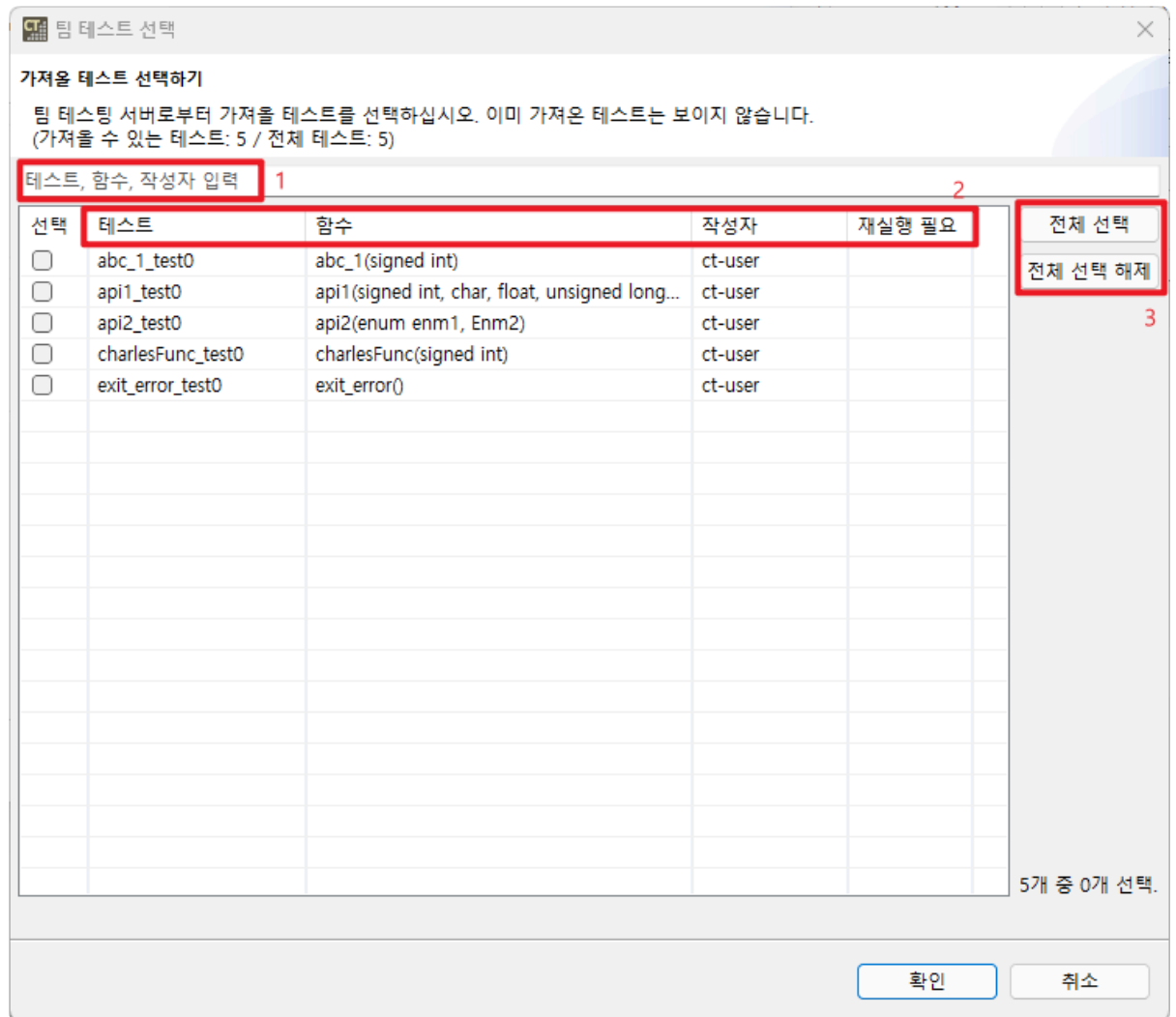
**통합 테스트**

총 테스트	0	
이미 가져온 테스트	0	
가져올 테스트	0	

가져올 테스트 선택...

< 이전(B)    다음(N) >    완료(F)    취소

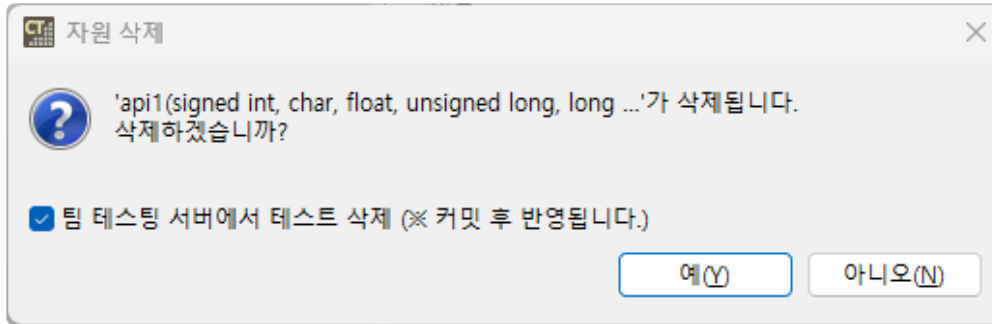
- [가져올 테스트 선택...]를 선택하면 서버에 있는 테스트를 확인할 수 있습니다.



- 유닛 테스트는 테스트별 이름, 함수의 시그니처, 작성자, 재실행 필요 여부를 확인할 수 있고 통합 테스트는 테스트별 이름, 작성자, 재실행 필요 여부를 확인할 수 있습니다.
- 1. 테스트, 함수, 작성자 이름으로 검색할 수 있습니다.
- 2. 열 제목을 선택하여 테스트, 함수, 작성자, 재실행 필요 유무로 정렬하여 확인할 수 있습니다.
- 3. [전체 선택], [전체 선택 해제] 버튼으로 모든 함수를 선택/선택 해제할 수 있습니다.

## 31.7.2. 팀 테스트 삭제

팀 프로젝트에서 테스트를 삭제하는 방법은 두 가지입니다.



- 팀 테스트 서버에서 테스트 삭제
  - [팀 테스트 서버에서 테스트 삭제] 체크박스에 체크하여 삭제하는 경우, 서버의 삭제 대상 목록에 추가되고 로컬에서 삭제됩니다. 삭제 대상 목록에 추가된 테스트는 커밋 후 서버에서 삭제됩니다.
- 로컬에서 테스트 삭제
  - [팀 테스트 서버에서 테스트 삭제] 체크박스에 체크 해제하여 삭제하는 경우, 서버에서는 테스트가 삭제되지 않습니다. 로컬에서 삭제된 테스트는 [팀 테스트 가져오기] 기능을 이용하여 다시 가져올 수 있습니다.



## 32. CT 팀 테스트팅 대시보드

---

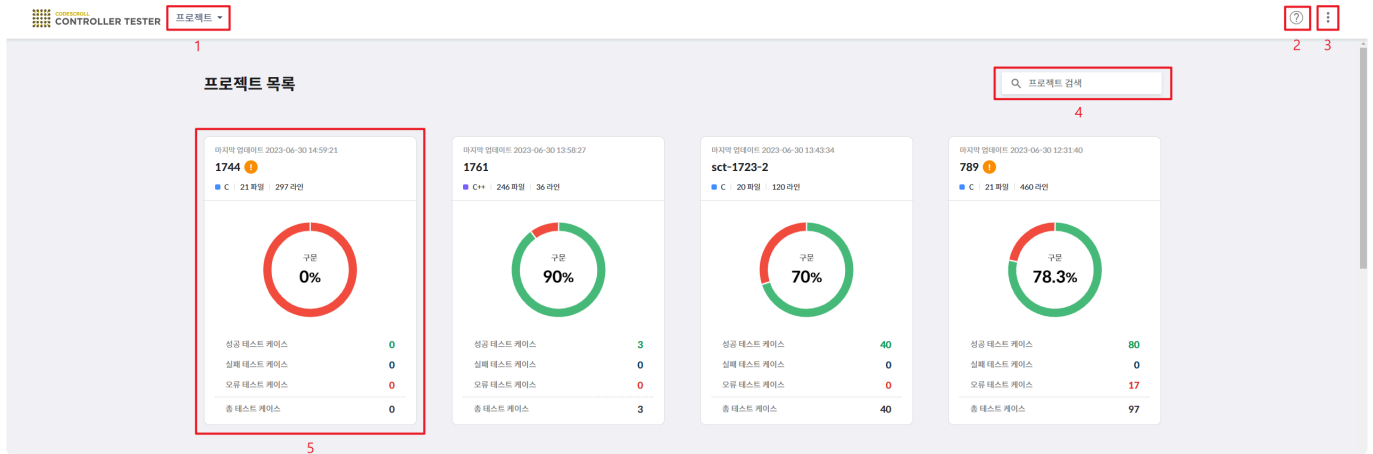
팀 테스트팅 대시보드에서는 팀 테스트팅 서버의 프로젝트 상태, 커버리지 정보, 마지막 업데이트 시간 등 팀 서버의 정보를 웹 형태로 제공합니다.

팀 테스트팅 대시보드에서 제공하는 정보는 다음과 같습니다.

- [메인 페이지](#)
- [프로젝트 정보 페이지](#)

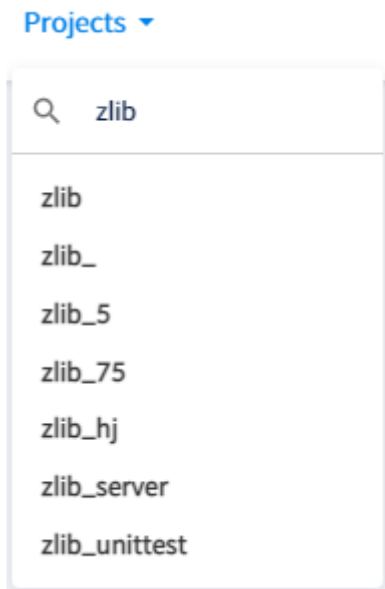
## 32.1. 메인 페이지

메인 페이지는 팀 서버에 존재하는 프로젝트를 최신 업데이트 순서로 보여줍니다. 새롭게 업로드되는 정보는 새로고침을 통해 반영할 수 있습니다.



### 1. 프로젝트 드롭다운

팀 테스트 대시보드는 드롭다운 형식으로 팀 프로젝트 목록을 제공합니다. 프로젝트 검색이 가능하며, 프로젝트를 클릭하면 프로젝트 정보 페이지로 이동합니다.

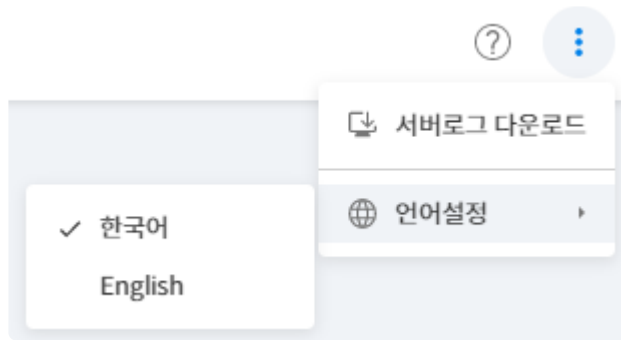


### 2. 도움말

팀 테스트 대시보드에 필요한 정보를 매뉴얼로 제공합니다.

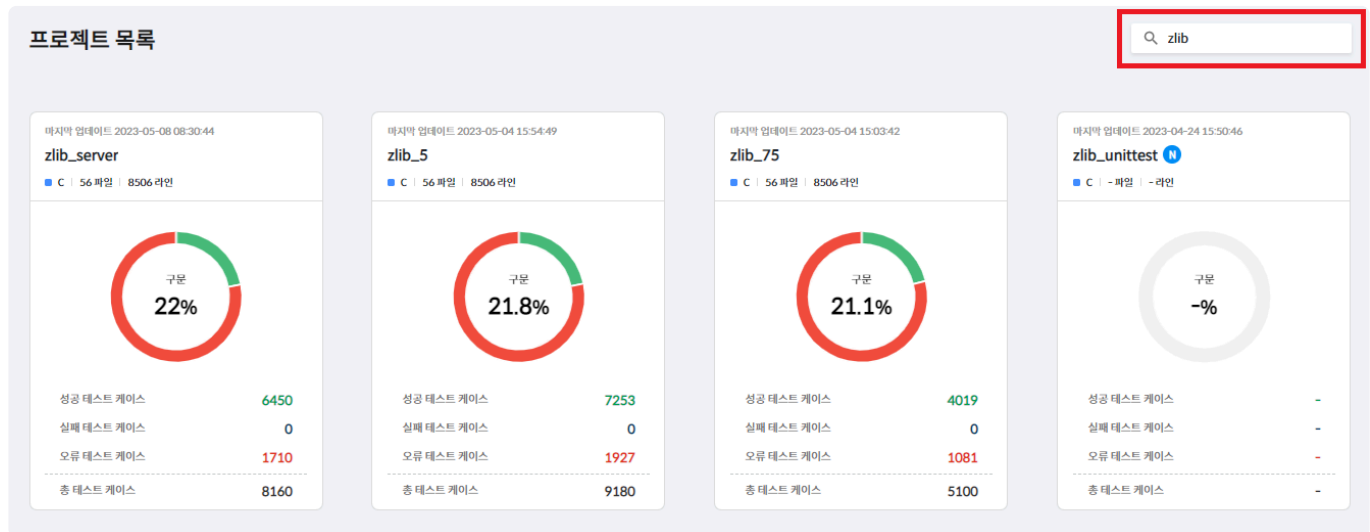
### 3. 더보기

팀 서버 로그 및 모든 프로젝트 로그를 다운로드 할 수 있고 언어 설정을 변경할 수 있습니다.




## 4. 검색 바

검색 바는 프로젝트 이름을 검색하는 기능을 제공합니다.



## 5. 프로젝트 카드

프로젝트 카드는 프로젝트의 마지막 업데이트, 프로젝트 이름 및 사용 중인 언어, 파일의 개수, 전체 라인, 커버리지, 테스트 케이스의 정보를 제공합니다.

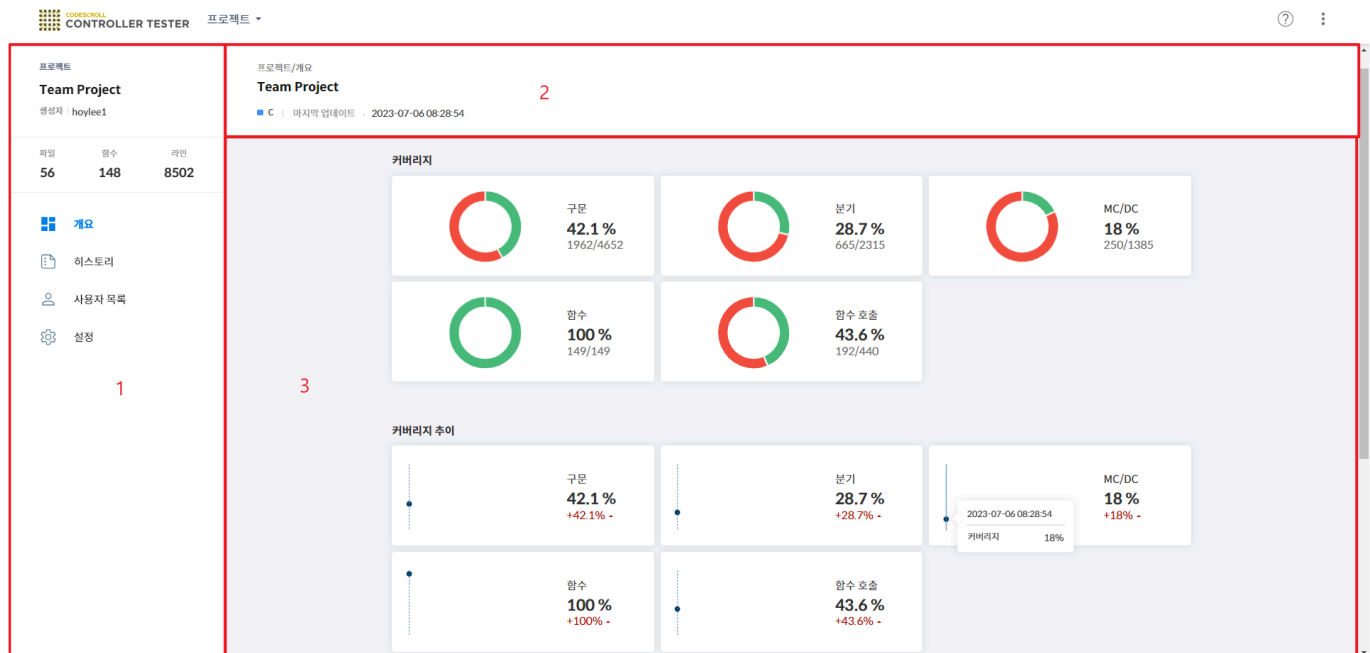
- 프로젝트에 한 번도 분석 정보가 커밋되지 않은 경우, 프로젝트 이름 옆에 N마크가 표시됩니다.
- 프로젝트 구성 변경으로 인한 테스트의 재실행이 필요한 경우  가 표시됩니다.
- 카드의 대표 커버리지는 [프로젝트 정보 페이지]에서 설정을 통해 변경할 수 있습니다.



\* 프로젝트 구성 변경으로 재실행이 필요한 테스트에 대한 자세한 설명은 [\[프로젝트 페이지 - 개요\]](#)의 [\[재실행이 필요한 테스트 목록\]](#)에서 확인할 수 있습니다.





## 32.2. 프로젝트 페이지

프로젝트 정보 페이지는 프로젝트의 세부 정보 및 테스트 결과를 텍스트 또는 그래프 형태로 제공합니다.



### 1. 사이드바 영역

사이드바 영역에서는 프로젝트의 요약 정보를 담고 있으며, 메인 콘텐츠 영역에서 표시할 개요, 히스토리, 사용자 목록, 설정을 탭 형태로 표시하며, 선택할 수 있습니다.

<div>프로젝트</div> <div>Team Project</div> <div>생성자   Ethan</div>		
파일 49	함수 245	라인 735
<div> 개요</div> <div> 히스토리</div> <div> 사용자 목록</div> <div> 설정</div>		

## 2. 메인 콘텐츠 헤더 영역

메인 콘텐츠의 주제 및 프로젝트의 이름, 언어, 마지막 업데이트된 시간을 담고 있습니다.

<div>프로젝트/개요</div> <div>Team Project</div> <div> C   마지막 업데이트 - 2023-04-26 21:00:00</div>
--

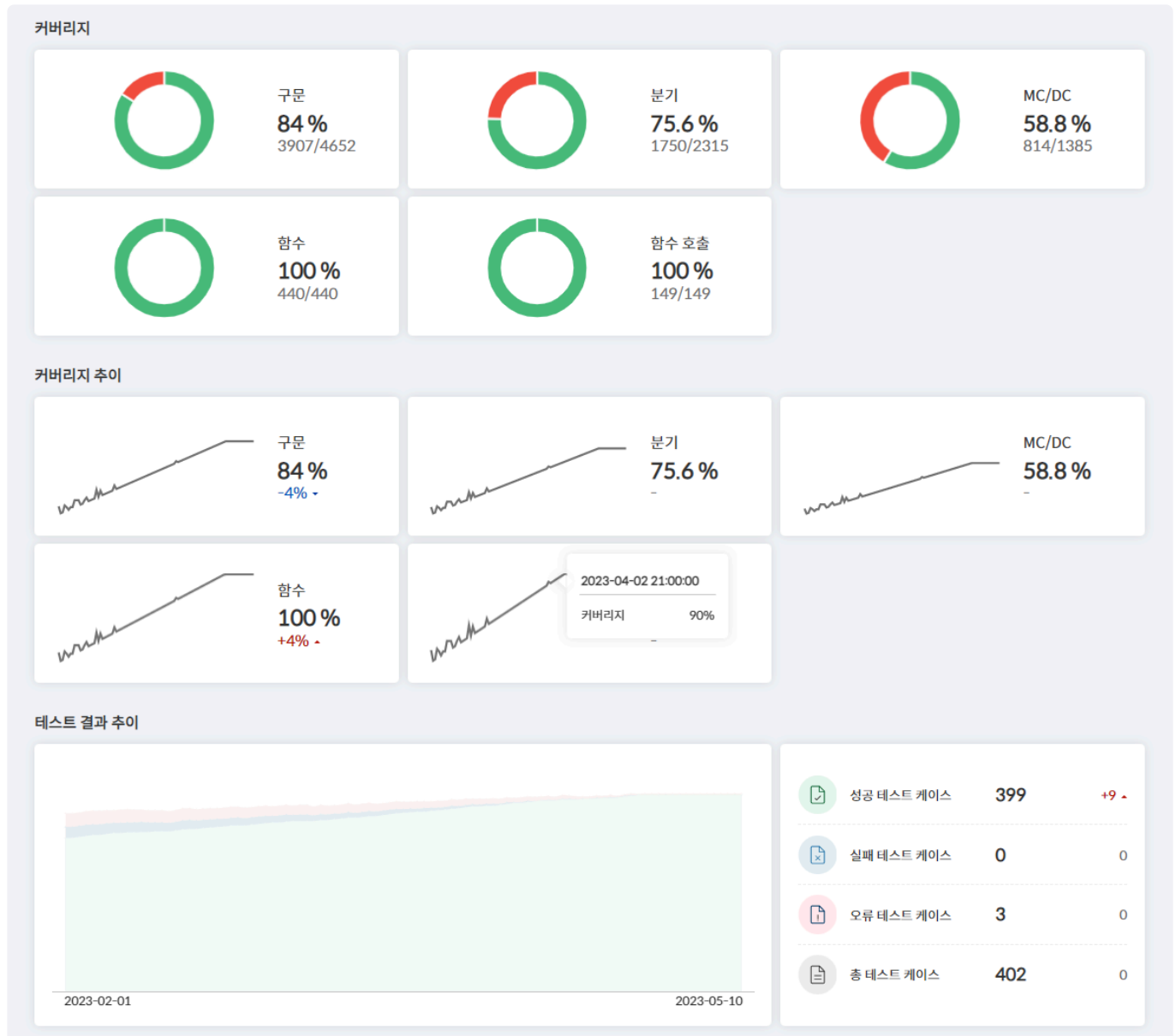
## 3. 메인 콘텐츠 영역

메인 콘텐츠 영역은 프로젝트에 대한 세부 정보를 제공합니다. 개요, 히스토리, 사용자 목록, 설정 4가지의 콘텐츠가 있습니다.

- [개요](#)
- [히스토리](#)
- [사용자 목록](#)
- [설정](#)

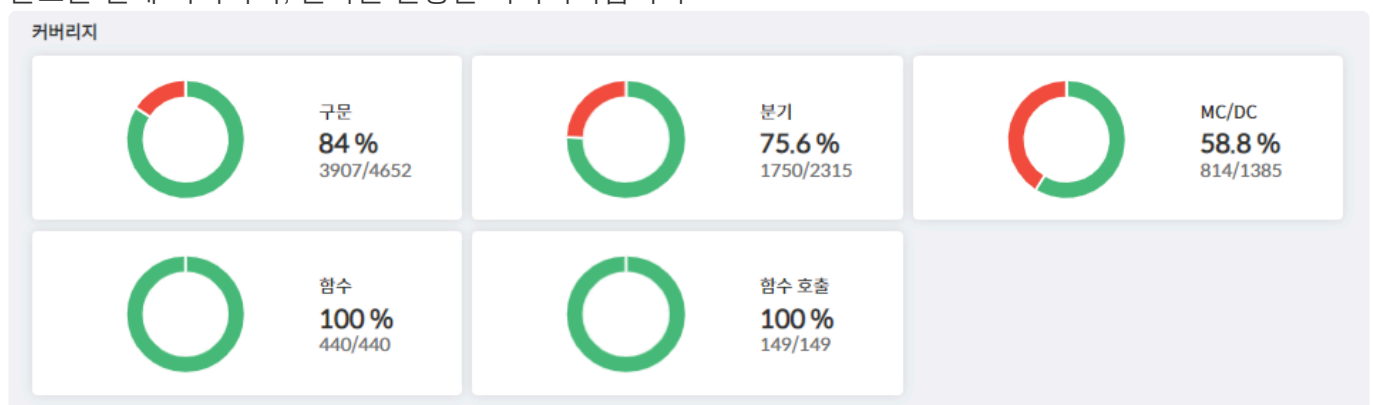
## 32.2.1. 개요

개요에서는 커버리지, 커버리지 추이, 테스트 결과 추이에 대한 정보를 제공합니다.



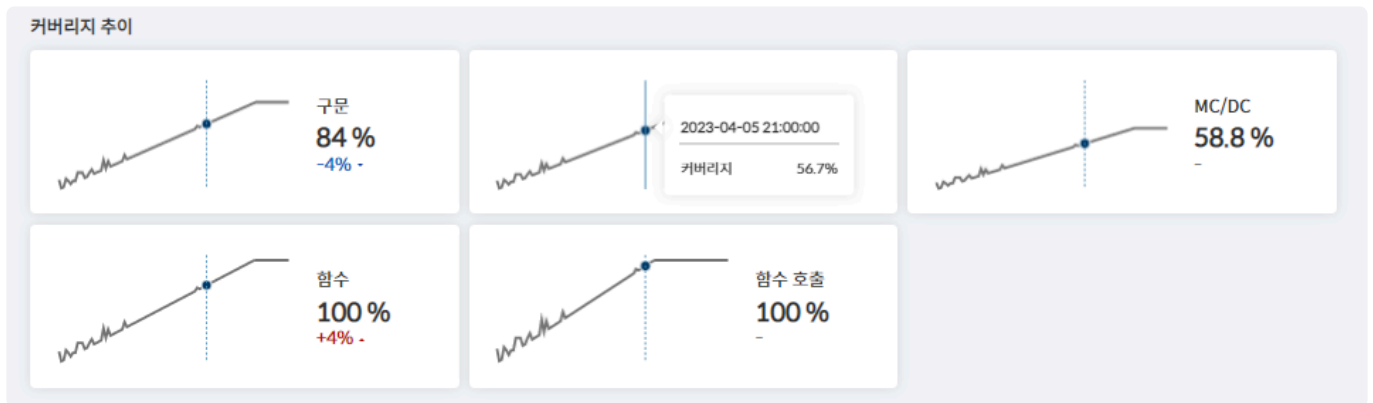
### 커버리지

프로젝트의 현재 구문, 분기, MC/DC, 함수, 함수 호출 커버리지 값을 백분율 형태로 보여줍니다. 백분율의 분모는 전체 커버리지, 분자는 달성한 커버리지입니다.



## 커버리지 추이

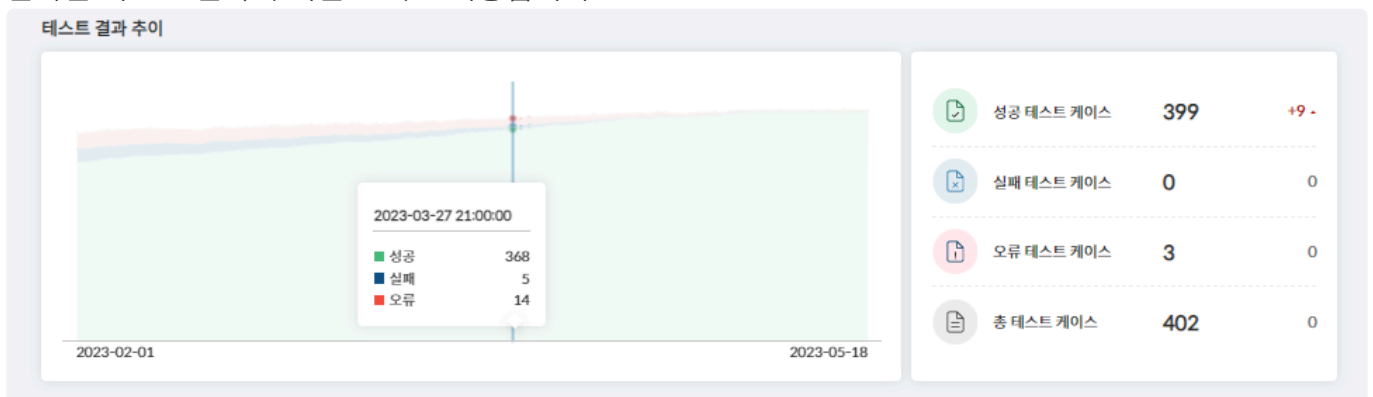
날짜별 커버리지 추이를 그래프 형태로 제공합니다.



- 그래프에 마우스를 올려 특정 날짜의 커버리지를 확인할 수 있습니다.
- 커버리지 값 아래에서 전날 대비 변화량을 확인할 수 있습니다.
- 추이 표시 범위는 [\[설정\]](#) 페이지에서 테스트 결과 추이 표시 기간을 변경할 수 있습니다.

## 테스트 결과 추이

날짜별 테스트 결과 추이를 그래프 제공합니다.



- 그래프에 마우스를 올려 특정 날짜의 테스트 결과를 확인할 수 있습니다.
- 그래프 오른쪽에서 전날 대비 변화량 제공합니다.
- 추이 표시 범위는 [\[설정\]](#) 페이지에서 테스트 결과 추이 표시 기간을 변경할 수 있습니다.

## 재실행이 필요한 테스트

재실행이 필요한 테스트가 있는 경우 상단에 “프로젝트 구성 변경으로 테스트 결과가 삭제되어 테스트 재실행이 필요합니다.”라는 경고가 뜹니다. 결과가 삭제된 테스트를 CT 2023.12에서 다시 실행하면 정확한 커버리지를 대시보드에 반영할 수 있습니다. 아래 두 가지 경우에 테스트의 결과가 삭제됩니다.



❗ 프로젝트 구성 변경으로 테스트 결과가 삭제되어 테스트 재실행이 필요합니다. (재실행 필요 테스트 수 17/23)

## 커버리지



구문  
**3 %**  
143/4652



분기  
**1.9 %**  
44/2315



MC/DC  
**1.2 %**  
17/1385



함수  
**4 %**  
6/149



함수 호출  
**0.2 %**  
1/440

- 자원을 변경하여 커밋한 경우, 해당 자원과 연관된 테스트의 결과와 커버리지가 삭제됩니다.
- 프로젝트 구성이 변경된 경우, 모든 테스트의 결과와 커버리지가 삭제됩니다.

## 32.2.2. 히스토리

히스토리에서는 프로젝트의 커밋 내역을 제공합니다.

사용자: 전체

분류: 전체

2023-04-11 ~ 2023-05-10

검색어를 입력해 주세요

2023-04-26

21:00 Ethan time : 2023-04-26 21:00:00

TEST\_DELETED

...

20:00 Ethan time : 2023-04-26 20:00:00

TEST\_FAILED TEST\_MODIFIED

...

19:00 Ethan time : 2023-04-26 19:00:00

TEST\_FAILED TEST\_DELETED TEST\_MODIFIED

...

18:00 Lily time : 2023-04-26 18:00:00

...

17:00 Gabriel time : 2023-04-26 17:00:00

TEST\_ADDED

...

16:00 Ethan time : 2023-04-26 16:00:00

...

15:00 Gabriel time : 2023-04-26 15:00:00

TEST\_MODIFIED

...

### 필터 및 검색

사용자, 분류, 날짜에 대한 필터 기능을 제공하며 커밋 메시지를 검색할 수 있습니다.

Ethan
분류: 전체
2023-04-19 ~ 2023-05-18
검색어를 입력해 주세요

2023-04-26

21:00 Ethan time: 2023-04-26 21:00:00  
TEST\_DELETED

20:00 Ethan time: 2023-04-26 20:00:00  
TEST\_FAILED TEST\_MODIFIED

19:00 Ethan time: 2023-04-26 19:00:00  
TEST\_FAILED TEST\_DELETED TEST\_MODIFIED

16:00 Ethan time: 2023-04-26 16:00:00

- 커밋 내역에서 분류 레이블을 클릭하면, 분류에서 해당 레이블로 필터링을 할 수 있습니다.

## 커밋 내역

커밋 내역을 펼치면 커밋의 세부 정보를 볼 수 있습니다.

2023-04-26

21:00 Ethan time: 2023-04-26 21:00:00  
TEST\_DELETED

커버리지
구문 84% -4% -
분기 75.6% -
MC/DC 58.8% -
함수 100% +4% -
함수 호출 100% -

테스트 결과
성공 399 +9 -
실패 0 -
오류 3 -
전체 402 -

상세 정보
소스파일 3 -
테스트 134.1 +1 -
스텝 5 +7 -
클래스 코드 0 -
결함주입 5 -
요구사항 0 -
재실행 필요 테스트 0 -

- 세부 정보로 커버리지, 테스트 결과, 자원 정보와 해당 커밋으로 변경된 수치 제공합니다.
- 자원이 변경되어 재실행 필요 테스트가 증가한 경우, 해당 커밋 내역이 주황색 테두리로 강조됩니다.

2023-06-14

12:26

hoylee

Auto-generated message: Commits 1 file(s).

!

COVERAGE\_DECREASED

CONFIGURATION\_CHANGED

...

커버리지

구문

0%

-35.7%

▼

분기

0%

-19.1%

▼

MC/DC

0%

-0.6%

▼

함수

0%

-41.9%

▼

함수 호출

0%

-34.1%

▼

테스트 결과

성공

0

-58

▼

실패

0

-

▼

오류

0

-15

▼

전체

0

-73

▼

상세 정보

소스파일

93

-

▼

테스트

37

-

▼

스텝

0

-

▼

클래스 코드

0

-

▼

결함주입

0

-

▼

요구사항

0

-

▼

재실행 필요 테스트







37



+37

▲

## 32.2.3. 사용자 목록

팀 프로젝트에 참여 중인 사용자의 정보를 제공합니다. 접속 상태 및 마지막 커밋 시간을 확인할 수 있습니다.

상태	이름 	마지막 커밋 
	Ethan	
	Gabriel	
	John	
	Lily	

 [1](#) 

## 32.2.4. 설정

설정 페이지는 프로젝트 설명, 커버리지 설정, 프로젝트 로그 다운로드 기능을 제공합니다.

- 프로젝트 설정을 변경한 후 오른쪽 상단의 [적용] 버튼을 클릭합니다.
- 대표 커버리지의 기본값은 구문 커버리지, 결과 추이 표시 기간의 기본값은 전체입니다.
- 오른쪽 상단의 [프로젝트 로그] 버튼을 클릭하여 프로젝트의 서버 로그를 다운로드할 수 있습니다.

## 33. EULA(End-User License Agreement)

### 소프트웨어 최종 사용자 라이선스 계약

**중요:** 본 최종 사용자 라이선스 계약("EULA")은 사용자(개인 또는 단체)와 본 EULA에 부착된 제품 인식 카드 혹은 인식 라벨로 식별되는 소프트웨어 제품("소프트웨어 제품" 또는 "소프트웨어") 제조업체인 슈어소프트테크 주식회사("슈어소프트테크") 사이의 법적 계약입니다. 소프트웨어 제품에는 컴퓨터 소프트웨어, 관련 미디어, 관련 인쇄물 및 "온라인" 또는 전자 설명서가 포함되어 있습니다. 소프트웨어 제품을 설치, 복사, 다운로드, 백업, 액세스하거나 사용하는 경우 본 EULA의 사용 조건에 동의하여야 하며, 본 EULA의 사용 조건에 동의하지 않으면 소프트웨어 제품에 대한 라이선스를 부여 받을 수 없습니다. 소프트웨어 제품에 대한 라이선스를 부여 받지 않는 경우, 본 소프트웨어 제품을 설치, 복사, 다운로드, 백업, 액세스하거나 사용할 수 없습니다.

### 소프트웨어 제품 라이선스

소프트웨어 제품은 지적 재산 보호 관련법뿐만 아니라, 저작권, 프로그램저작권 및 국제 저작권 규정에 따라 보호됩니다. 본 계약에서 사용되는 "컴퓨터"라는 용어는 단일 컴퓨터 시스템을 의미하며, 단일 컴퓨터 시스템의 범위에 대해서는 아래에 상술되어 있습니다.

#### 1. 라이선스 부여.

본 EULA는 사용자에게 아래 권한을 부여합니다.

**소프트웨어 설치 및 사용.** 본 EULA에 명백히 설명한 것을 제외하고, 컴퓨터에 소프트웨어 제품 복사본을 하나만 설치, 사용, 액세스, 실행 또는 상호작용("실행")할 수 있습니다. 두 대 이상의 컴퓨터에서 소프트웨어 제품을 설치, 사용 또는 실행하기 위해서는, 각 컴퓨터 상에서 사용 또는 실행되는 소프트웨어의 상호작용 여부와 무관하게 컴퓨터의 대수에 해당되는 적법한 라이선스(들)를 사용자가 부여 받아야 합니다.

**복사본 백업.** 컴퓨터에 포함된 소프트웨어 제품 백업 본이 없는 경우, 사용자가 소프트웨어 제품의 컴퓨터 소프트웨어 부분에 해당하는 단일 백업 본을 작성할 수 있습니다. 기록의 목적으로 백업 복사본을 사용할 수 있습니다. 단일 백업 본을 작성하기 위해 백업 유틸리티를 사용할 수 있습니다. 본 EULA에 명백히 제공된 것을 제외하고, 소프트웨어 제품에 동봉된 설명서를 포함한 소프트웨어 제품의 복사본을 작성할 수 없습니다.

#### 2. 기타 권리 및 제한사항 설명

**컴퓨터 혹은 단일 컴퓨터 시스템.** 소프트웨어 제품의 라이선스는 단일 컴퓨터 시스템에 적용됩니다. 단일 컴퓨터 시스템이 다수의 중앙처리장치(CPU)를 갖는 경우, 이 중앙처리장치들이 하나의 회로기관 상에 부착되어 있고 하나의 시스템 버스를 공유하는 대칭적 중앙처리 구조를 지니는 경우에 한하여 단일 컴퓨터 시스템으로 간주됩니다.

**언어 버전 선택.** 소프트웨어 제품이 한 개 이상의 언어 버전에 포함되어 있는 경우, 제공된 언어 버전 중 하나만 사용할 수 있도록 허가됩니다.

**분해모방 제한사항.** 디컴파일 및 분해. 소프트웨어 제품을 분해모방을 이용하여 디컴파일하거나 분해할 수 없습니다.

**구성요소 분리.** 소프트웨어 제품은 단일 제품으로 허가되었습니다. 한 대 이상의 컴퓨터에서 사용하기 위해 구성요소를 분리할 수 없습니다.

**임대.** 소프트웨어 제품을 임대, 리스 혹은 대여할 수 없습니다.

**소프트웨어 제품 권리 이전.** 본 EULA에 따라 하드웨어의 판매 또는 권리 이전의 일부분으로 모든 권한을 영구적으로 권리 이전할 수 있으며, 수령인은 본 EULA의 사용 조건에 동의하는 것으로 간주됩니다.

**만료.** 사용자가 본 EULA의 조항과 조건을 위반한 경우, 다른 권한의 침해 없이, 슈어소프트테크는 본 EULA의 기간 및 조건에 따라 권한을 만료시킬 수 있습니다. 이런 경우, 이미 설치된 소프트웨어 제품을 포함한 모든 소프트웨어 제품의 복사본 및 구성 부품을 폐기시켜야 합니다.

**등록 상표.** 본 EULA는 소프트웨어 제품 제조업체, 컴퓨터 제조업체, 기타 관련 공급업체의 등록 상표 또는 서비스 상표에 관련된 권한을 부여하지 않습니다.

**업그레이드.** 본 소프트웨어 제품이 슈어소프트테크에 의해 업그레이드되는 경우, 본 EULA의 사용 조건은 업그레이드된 소프트웨어에 그대로 적용되며 한 대 이상의 컴퓨터에서 사용하기 위해 분리될 수 없습니다.

### 3. 저작권.

소프트웨어 제품의 모든 권리와 지적소유권(소프트웨어 제품에 내장된 모든 이미지, 사진, 애니메이션, 비디오, 오디오, 음악, 텍스트 및 애플릿을 제한 없이 포함), 동봉된 설명서 및 모든 소프트웨어 제품의 복사본은 소프트웨어 제품 제조업체나 공급업체의 소유입니다. 소프트웨어 제품에 내장되어 있지는 않지만 소프트웨어 제품의 사용을 통해 액세스할 수 있는 내용에 대한 모든 권리와 지적 소유권은 해당 회사의 소유이며, 본 EULA는 이와 같은 내용의 사용에 대한 권한을 부여하지 않습니다. 소프트웨어 제품에 설명서가 파일로 포함되어 있는 경우, 해당 설명서 파일의 복사본을 한 부 인쇄할 수 있습니다. 소프트웨어 제품에 동봉된 인쇄물은 복사할 수 없습니다. 본 EULA에서 명백히 부여하지 않은 모든 권한은 해당 소프트웨어 제품 제조업체와 공급업체의 소유입니다.

### 4. 이중 미디어 소프트웨어 제품.

한 개 이상의 미디어로 소프트웨어 제품을 전달받을 수 있습니다. 받은 미디어의 종류 또는 크기와 관계없이, 컴퓨터에 해당 미디어를 하나만 사용할 수 있습니다. 다른 컴퓨터에서 다른 미디어를 실행할 수 없습니다. 소프트웨어 제품의 영구적인 권리 이전(위의 설명대로)을 제외하고 다른 사용자에게 다른 미디어를 임대, 리스, 대여 또는 권리 이전할 수 없습니다.

### 5. 보증.

소프트웨어 제품의 품질에 대한 보증은 첨부된 보증서에 따르며, 별도의 보증서가 첨부되지 않은 경우 슈어소프트테크는 소프트웨어 제품을 구입일로부터 1년간 정상적으로 사용할 경우 소프트웨어 제조공정상의 결함이 발생하지 않음을 보증합니다. 이는 소프트웨어 제품이 오류를 갖지 않는 소프트웨어임을 의미하지 않으며, 사용자가 작업 도중 의도하지 않았던 결과를 초래할 가능성을 배제하지 않습니다. 별도의 보증계약에 의해 명백히 기술된 경우를 제외하고, 슈어소프트테크는 소프트웨어 제품의 부정확성, 오류 또는 손실에 대해 책임지지 않으며 이로 인하여 발생하는 손상 및 손해에 대해서도 책임지지 않습니다. 슈어소프트테크는 제3자가 제기한 배상 청구나 혹은 제3자를 위해 사용자가 제기한 배상 청구에 대해 책임지지 않습니다. 본 EULA에 언급된 보증제한 규정은 사용자가 상기의 가능성에 대해 슈어소프트테크 또는 슈어소프트테크의 공인 대리업체에 미리 통고한 경우에도 유효합니다.

### 6. 수출 제한사항.

소프트웨어 제품의 권리 이전은 대한민국 내에서만 가능하며, 타 국가로 수출될 수 없습니다.



## 7. JAVA 지원 참고.

소프트웨어 제품에는 JAVA로 작성된 프로그램에 대한 지원이 포함되어 있습니다. JAVA 기술은 오류를 허용하지 않으며, JAVA 기술 오류로 인하여 사망, 부상 또는 심각한 물리적, 환경적 손상을 유발할 수 있는 핵 설비, 항공기 탐색이나 통신 시스템, 항공 제어, 의약 또는 무기 시스템과 같은 오류 방지 성능이 필요한 위험한 환경에서 온라인 제어 장치로 설계되거나 제조 또는 사용, 판매되도록 제작되지 않았습니다.

## 주의사항

© 2023. Suresofttech Inc. All rights reserved.

소프트웨어에 포함된 Suresofttech 및 기타 상표는 슈어소프트테크 주식회사의 상표입니다.

이 문서 및 관련 문서에 언급된 제품명 혹은 상표명은 제품의 저작권을 소유한 회사의 상표 및 등록상표입니다. 이 문서에 기술되어 있는 소프트웨어는 사용 계약 및 비공개 계약을 통해 제공된 것입니다.

소프트웨어는 본 계약의 사용 조건에 따라서만 사용 또는 복사될 수 있습니다.