

# 중급 LabVIEW

랩뷰교육원

곽두영

와이파이: lvedu 또는 lvedu5G

Pw: lvedu.kr

# 강사 소개

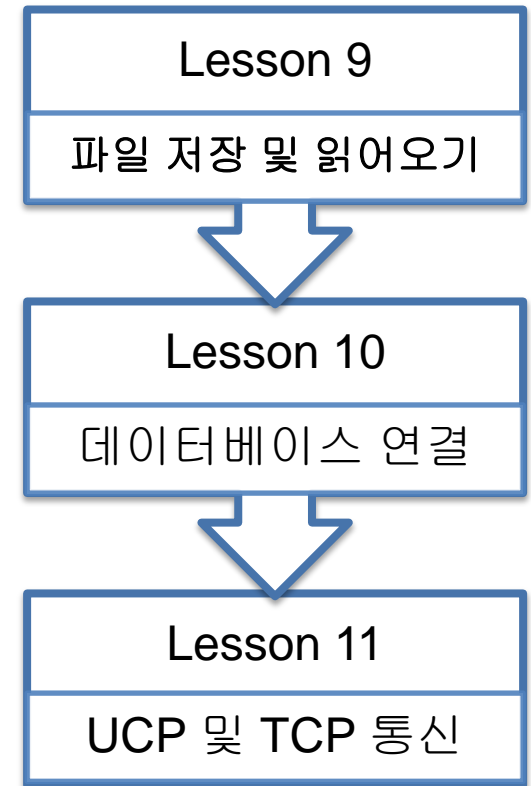
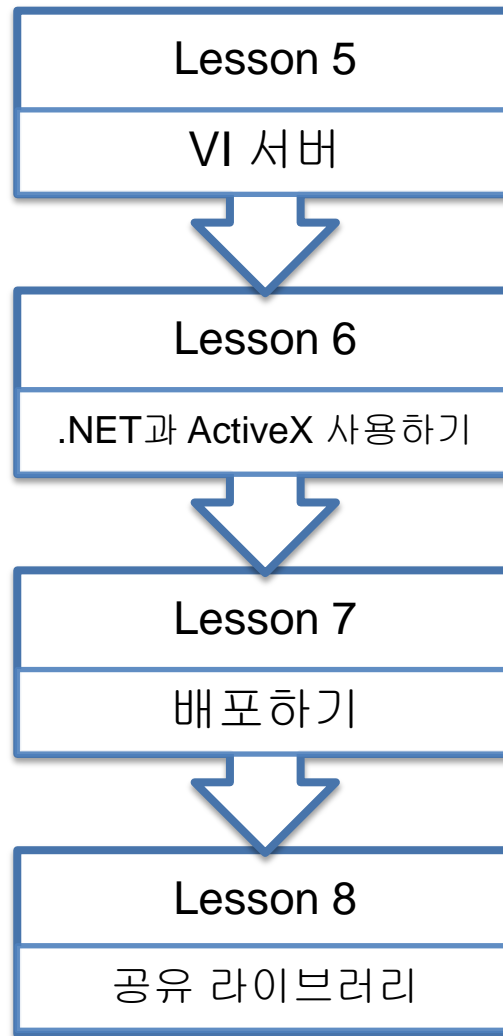
곽두영

- 1999년 2월 ~ 2001년 2월: 포항 가속기 연구소
- 2001년 3월 ~ 2012년 12월: National Instrument Korea (기술지원부 본부장)
- 2013년 1월 ~ 2014년 12월: Sunin CNS
- 2015년 1월 ~ 현재: 랩뷰교육원
- LabVIEW 서적 출판 – 2002년부터 21권

# 랩뷰교육원 소개

- 2015년 1월 개원 – LabVIEW 교육 센터 및 출판사
- 2018년 11월 – [도서] 초급 LabVIEW (개정판)
- 2016년 1월 – [도서] 중급 LabVIEW
- 2018년 12월 – [도서] LabVIEW 이미지 수집 및 분석 (개정판)
- 2015년 3월 – [도서] LabVIEW 데이터 수집 및 분석
- 2017년 11월 – [도서] LabVIEW FPGA 및 Real Time (개정판)
- 2015년 7월 – [도서] LabVIEW HMI 및 PLC 통신
- 2015년 10월 – [도서] LabVIEW 모터 제어

# 과정 내용



가장 널리 사용되는 LabVIEW 디자인 패턴을 익힌다.  
사용자 인터페이스 이벤트 핸들러 디자인 패턴을 익힌다.

# CHAPTER 1. 디자인 패턴 I

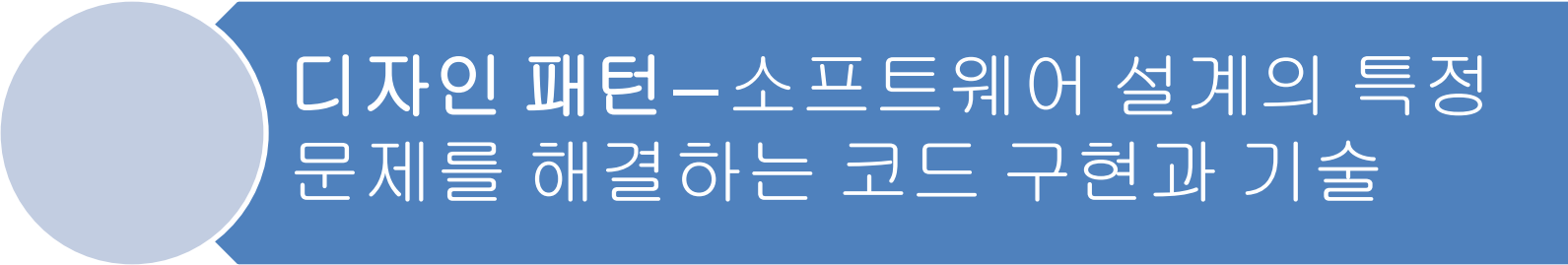
---

# 1. LabVIEW 디자인 패턴

프레임워크  
디자인 패턴

# 디자인 패턴을 사용하는 이유?

- 소프트웨어 개발에 있어 유용함이 입증된 기술.
- 프로그램을 처음부터 새로 개발할 필요가 없음.
- 코드 작성에 관여하지 않은 다른 사람도 코드를 읽고 수정하기 쉬움.

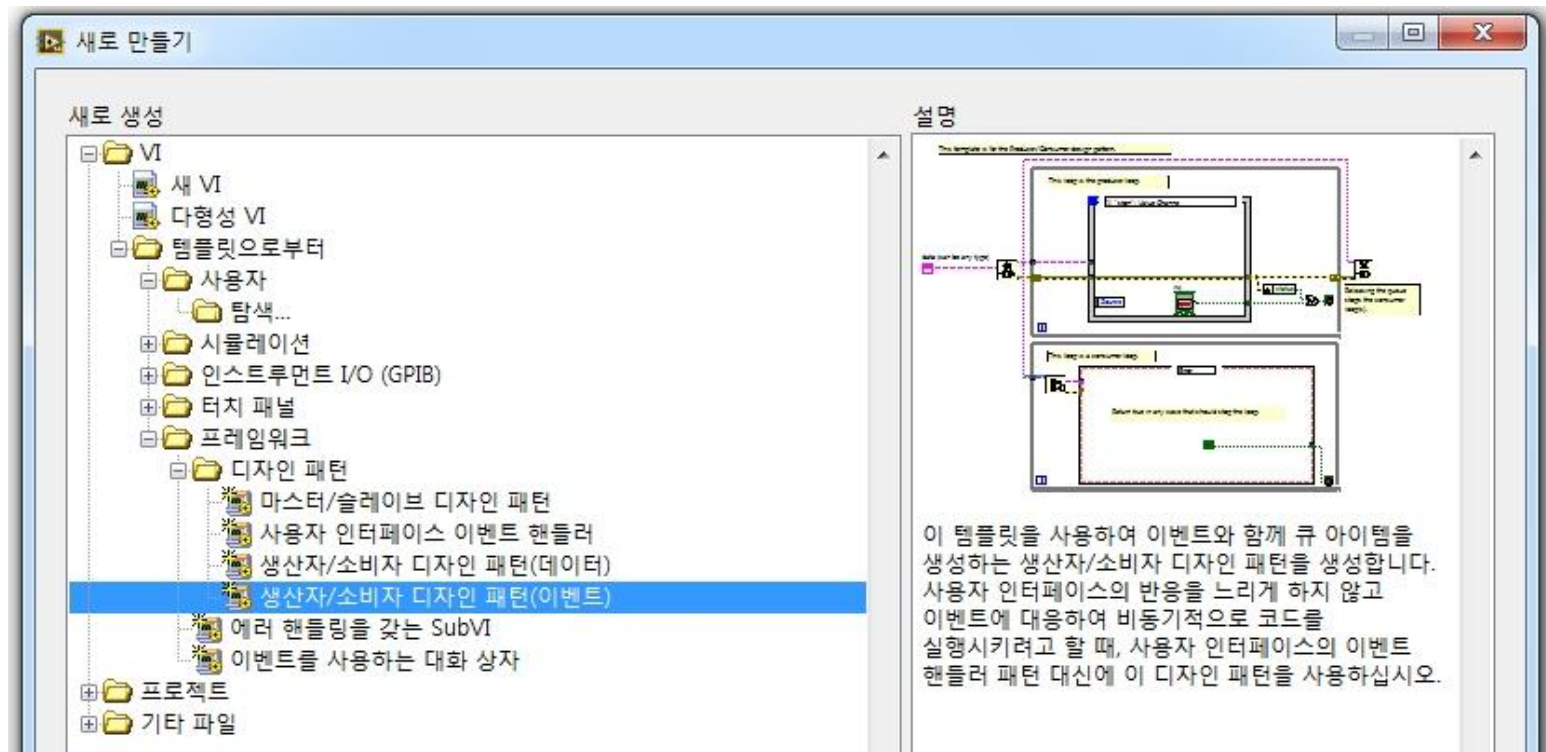


디자인 패턴—소프트웨어 설계의 특정 문제를 해결하는 코드 구현과 기술

디자인 패턴은 일반적으로 많은 개발자들의 노력을 통해 발전하고 있으며, 최대한 간단하고 관리하기 편하고 읽기 쉽도록 구성됩니다.

# LabVIEW 디자인 패턴

- LabVIEW 새로 만들기에서 VI > 템플릿으로부터 > 프레임워크 > 디자인 패턴





- 사용자 인터페이스 이벤트 핸들러
- 상태 머신
- 생산자/소비자 디자인 패턴
- 마스터/슬레이브 디자인 패턴
- 큐 메시지 핸들러

---

## 2. 사용자 인터페이스 이벤트 핸들러

Event Driven

# 사용자 인터페이스 이벤트 핸들러

- 이벤트는 어떤 일이 발생했다는 것을 알리는 비동기적 알림이다.
- 프런트패널의 사용자 동작과 블록다이어그램 실행을 동기화하려면 사용자 인터페이스 이벤트를 사용한다.
- 사용자가 특정 동작을 수행할 때마다 특정 이벤트 핸들링 케이스를 실행한다.
  - 이벤트 핸들링의 반대 개념은 폴링(Polling) 방식이다.

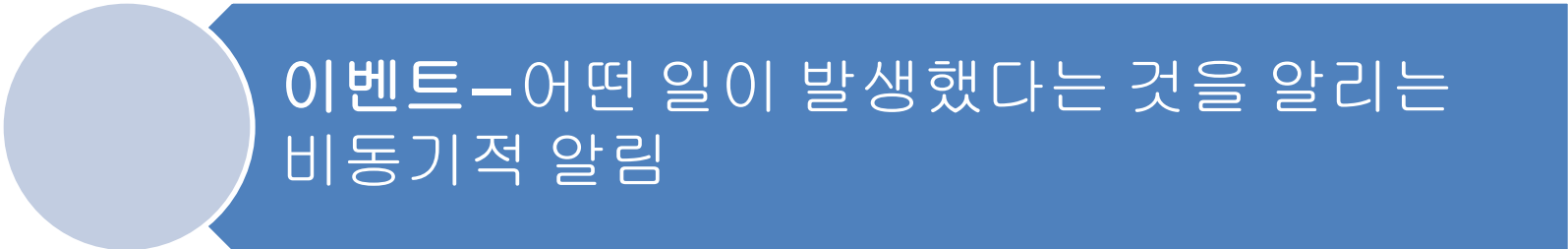
# 폴링(Polling) 방식

- 프런트패널 객체의 상태 변화를 확인하기 위해 주기적인 폴링을 해야 한다.
- 프런트패널 폴링은 상당한 **CPU** 사용이 필요하다.
- 변화가 너무 빨리 일어나면 감지하지 못할 수 있다.

# Event 방식

- 프런트패널을 폴링하지 않아도 된다.
- CPU 사용량을 줄일 수 있다.
- 블록다이어그램의 코드가 단순화된다.
- 블록다이어그램이 모든 사용자 동작에 반응할 수 있다.

# 이벤트



이벤트—어떤 일이 발생했다는 것을 알리는  
비동기적 알림

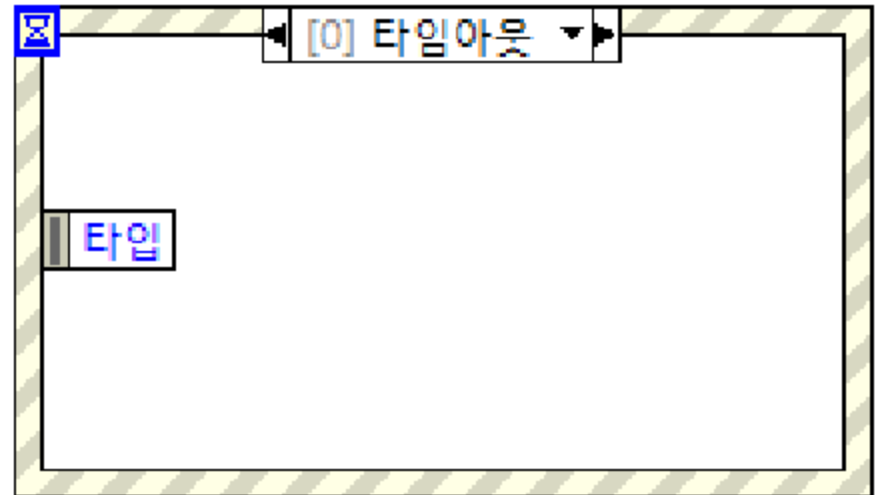
- 이벤트는 사용자 인터페이스, 외부 I/O 또는 프로그램의 다른 부분에서 발생 가능.
- 이벤트는 이벤트 소스에서 발생.
  - 예: 프런트패널 컨트롤에서 발생하는 값 변경.

# 이벤트 구조 사용하기

이벤트 구조는 [알림 기다림] 함수를 포함하는 케이스 구조처럼 작동합니다.

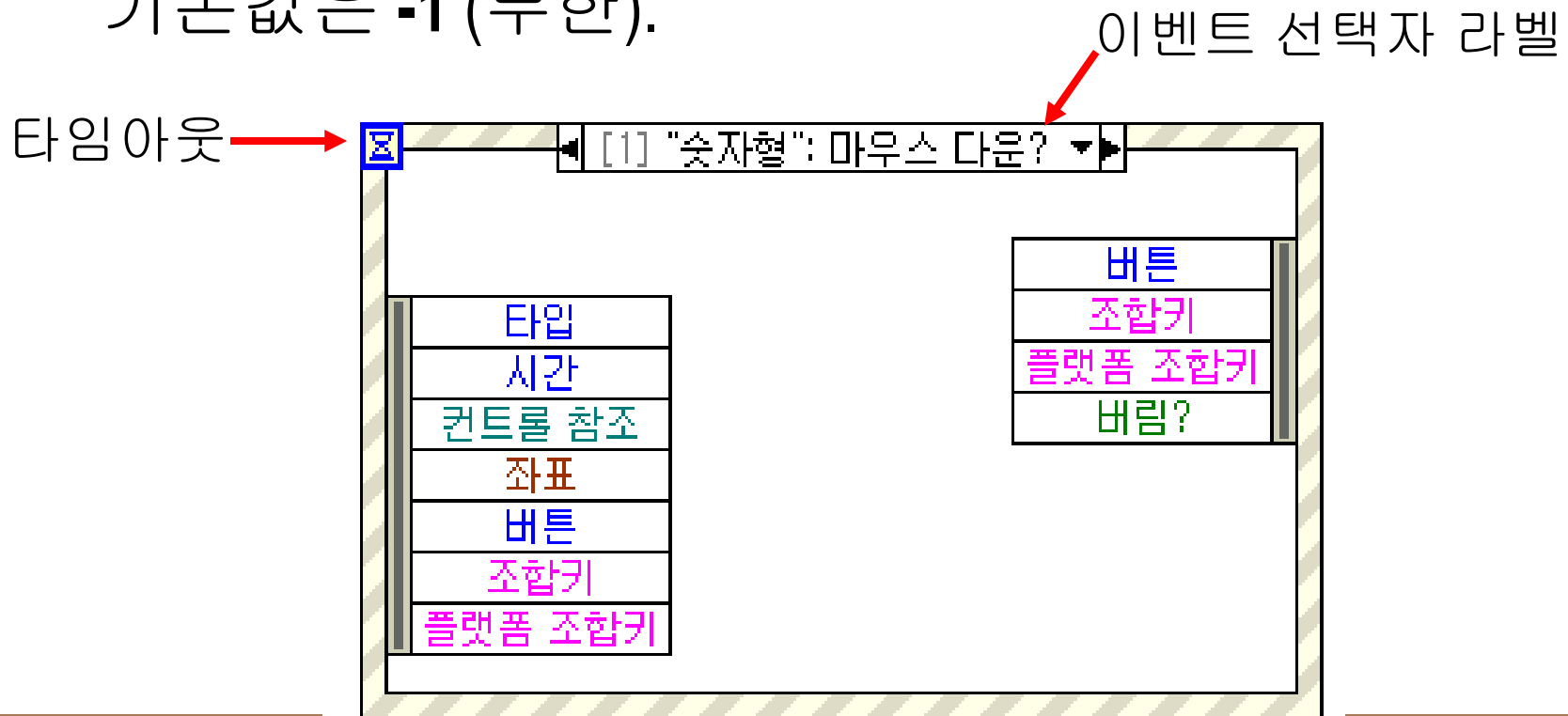
이벤트 구조를 사용하여 다음과 같은 사용자 인터페이스(정적) 이벤트 처리 가능:

- 마우스 버튼 누르기.
- 키보드 키 누르기.
- 숫자형 컨트롤의 값 변경하기.



# 이벤트 구조의 구성

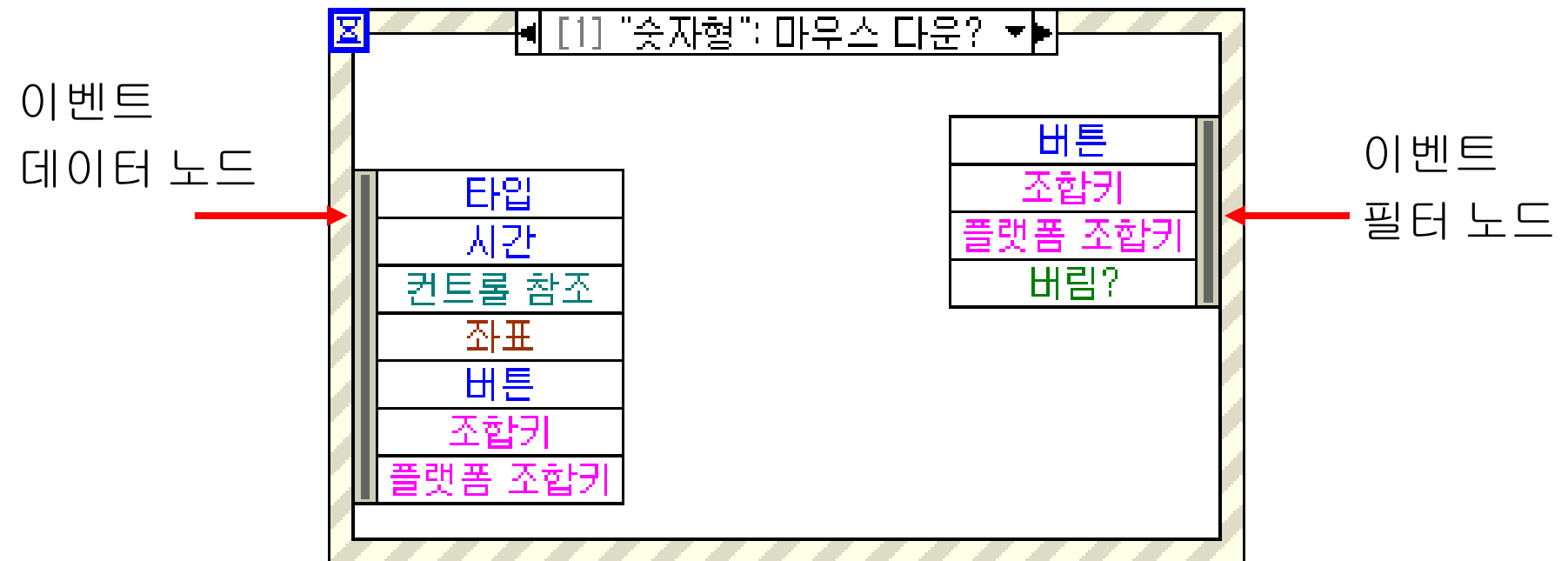
- 이벤트 선택자 라벨 — 현재 보이는 이벤트 케이스의 이름.
- 타임아웃 — 이벤트 기다리는 시간을 ms로 지정.  
기본값은 -1 (무한).





# 이벤트 구조의 구성(계속)

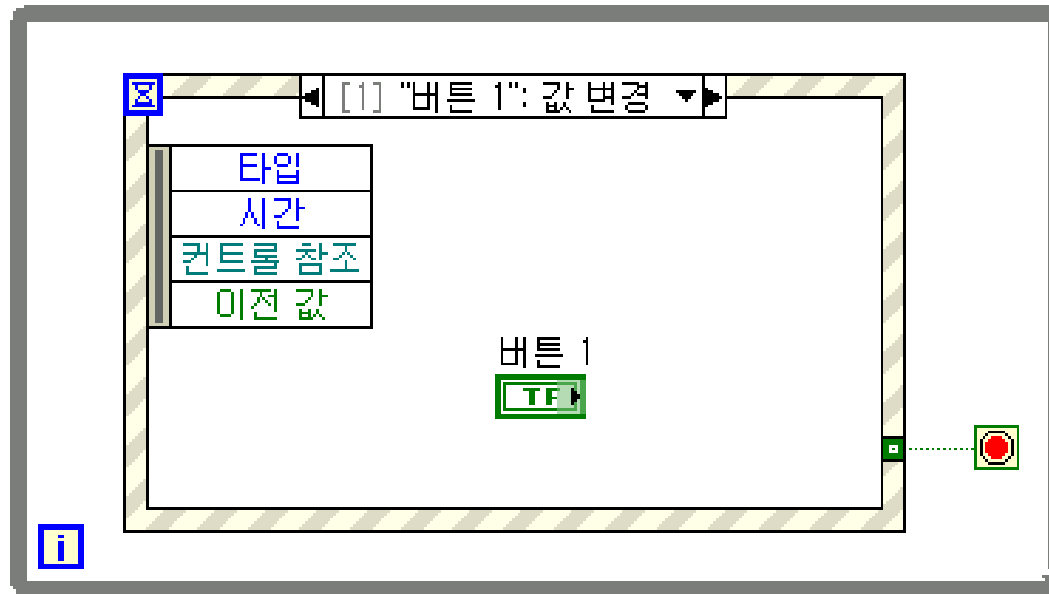
- 이벤트 데이터 노드-이벤트 발생 시 LabVIEW가 제공하는 데이터 식별. [이름으로 풀기] 함수와 유사.
- 이벤트 필터 노드-이벤트 데이터 노드의 데이터 중에서 이벤트 케이스가 수정할 수 있는 데이터 식별.



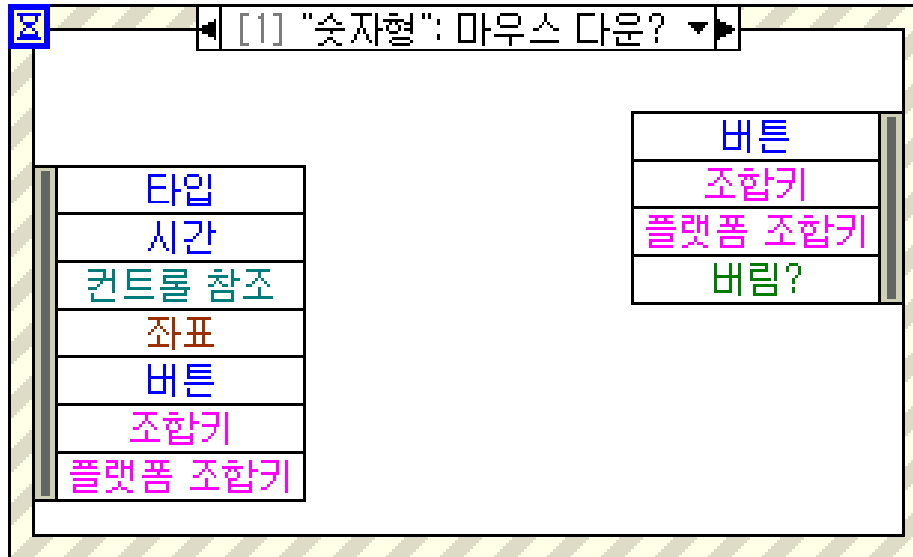
# 이벤트 구조 사용하기

일반적으로 이벤트 구조는 While 루프 안에 놓습니다.

- 이벤트 구조는 While 루프가 한 번 반복될 때마다 한 개의 이벤트를 처리합니다.
- 이벤트가 발생하지 않는 동안 이벤트 구조는 유휴 상태를 유지합니다.



# 이벤트 구조 설정하기



이벤트 구조의 경계에서 마우스 오른 쪽 버튼을 클릭한 후 바로 가기 메뉴에서 이 케이스에 의해 **핸들되는 이벤트 편집**을 선택하여 대화 상자를 열고 각 이벤트 설정.

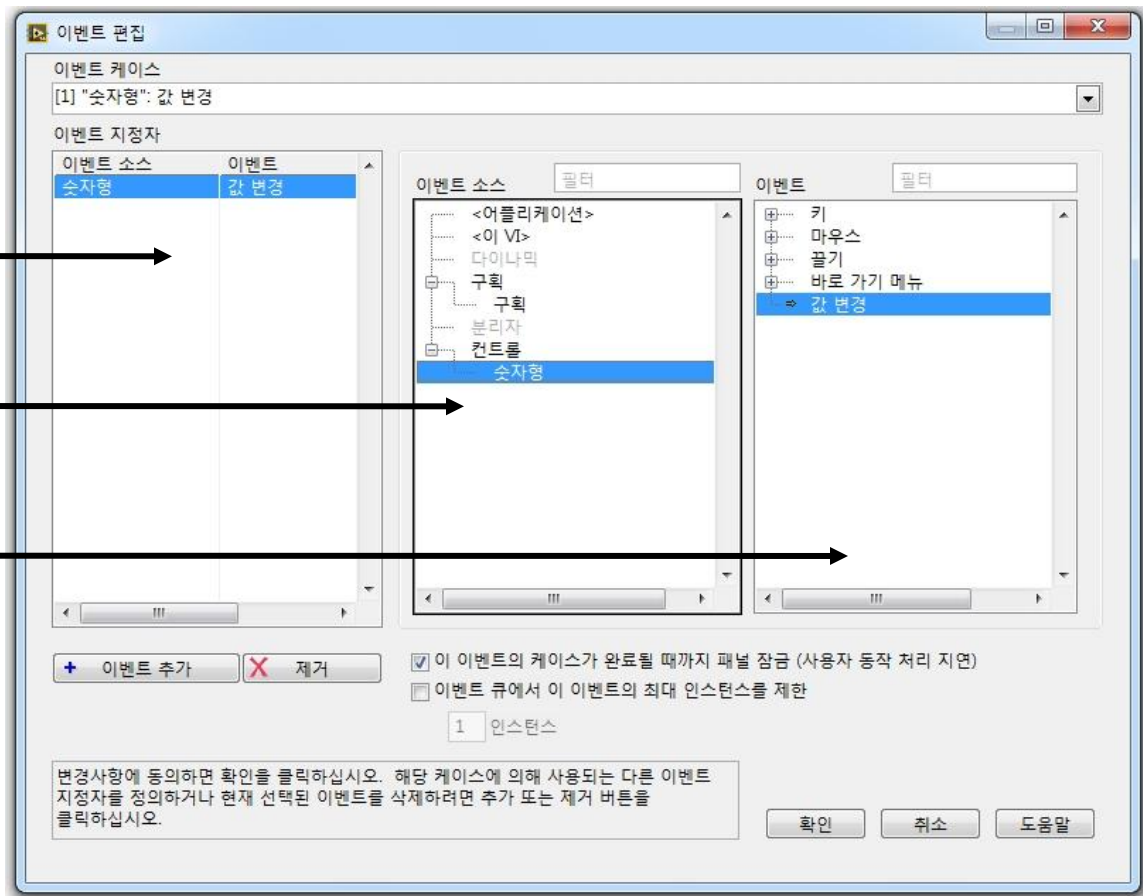
보이는 아이템	▶
도움말	
예제	
설명과 팁...	
브레이크포인트	▶
구조 팔레트	▶
✓ 자동 크기 조정	
다이어그램 정리에서 제외	
이벤트 구조 제거	
이 케이스에 의해 <b>핸들되는 이벤트 편집...</b>	
이벤트 케이스 추가...	
이벤트 케이스 복제...	
이 이벤트 케이스 삭제	
다이나믹 이벤트 터미널 보이기	
[0] 타임아웃 케이스 보이기	
케이스 재배치...	
컨트롤 찾기	
프로퍼티	

# 이벤트 편집 대화 상자

설정된 이벤트

이벤트 소스

이벤트



# 알림과 필터 이벤트

## 이벤트

- + ... 끌기
- + ... 키
- ... 마우스
  - ... 마우스 다운
  - ... 마우스 다운?
  - ... 마우스 커서 들어옴
  - ... 마우스 커서 이탈
  - ... 마우스 이동
  - ... 마우스 업
- + ... 바로 가기 메뉴
  - ... 값 변경

## → 알림 이벤트 (녹색 화살표)

사용자 동작이 이미 발생되었고  
LabVIEW가 이벤트를 처리함.

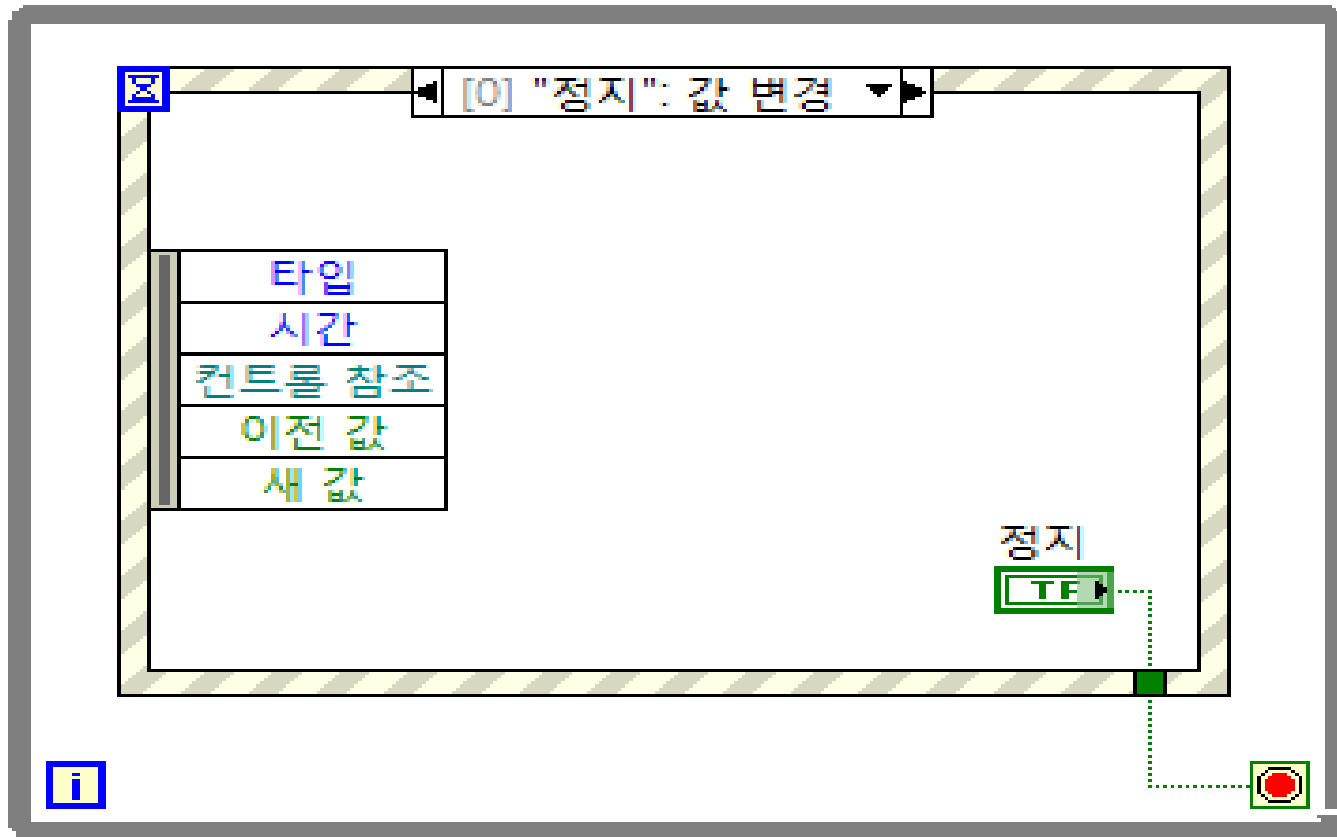
## → 필터 이벤트 (빨간색 화살표)

사용자 동작이 이미 발생되었고  
LabVIEW가 이벤트를 처리하지 않음.  
필터 이벤트로 이벤트의 기본 동작을 덮어  
쓸 수 있음.

# 사용자 인터페이스 이벤트 핸들러

- 사용자 인터페이스 이벤트 핸들러 디자인패턴
  - 프론트패널의 컨트롤 값 변경
  - 마우스 클릭이나 움직임
- **While** 루프 + 이벤트 구조
  - 이벤트 소스
  - 이벤트
- 이벤트가 없을 경우에는 휴면 상태로 전환된다.

# While 루프 + 이벤트 구조



- While 루프에 하나의 이벤트 구조만 넣어야 된다.
- 불리언의 레치 동작을 구현 → 해당 불리언을 이벤트에 넣어줘야 된다.

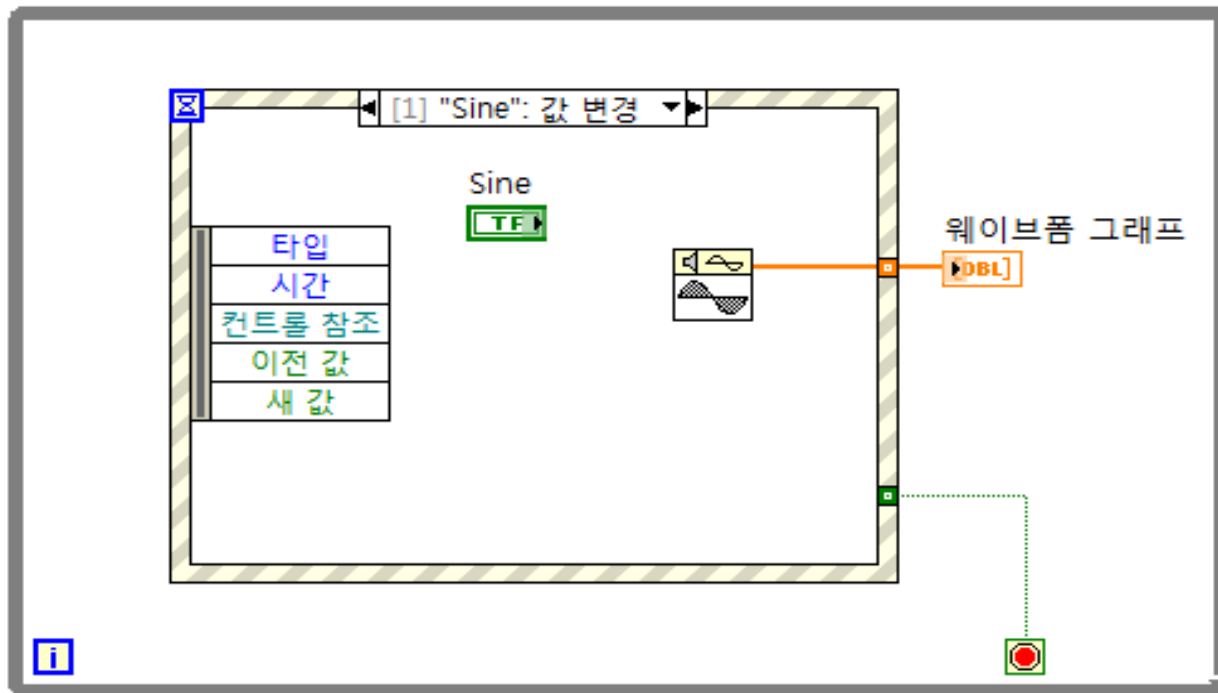
# Event 핸들러 (정리)

- 독립적으로 이벤트를 처리하는 여러 케이스
- 한번에 오직 한 개의 이벤트만 발생
- 이벤트 선택자 라벨 → 케이스 선택자와 유사하다.
- 타임 아웃 터미널 → 이벤트를 기다릴 시간(ms)을 지정한다. **Default** 값은 -1이다. (무한히 기다림)
- 이벤트 데이터 노드 → 이름으로 풀기와 유사하다. 이벤트가 발생할 때, **LabVIEW**가 전달하는 데이터들을 출력한다.



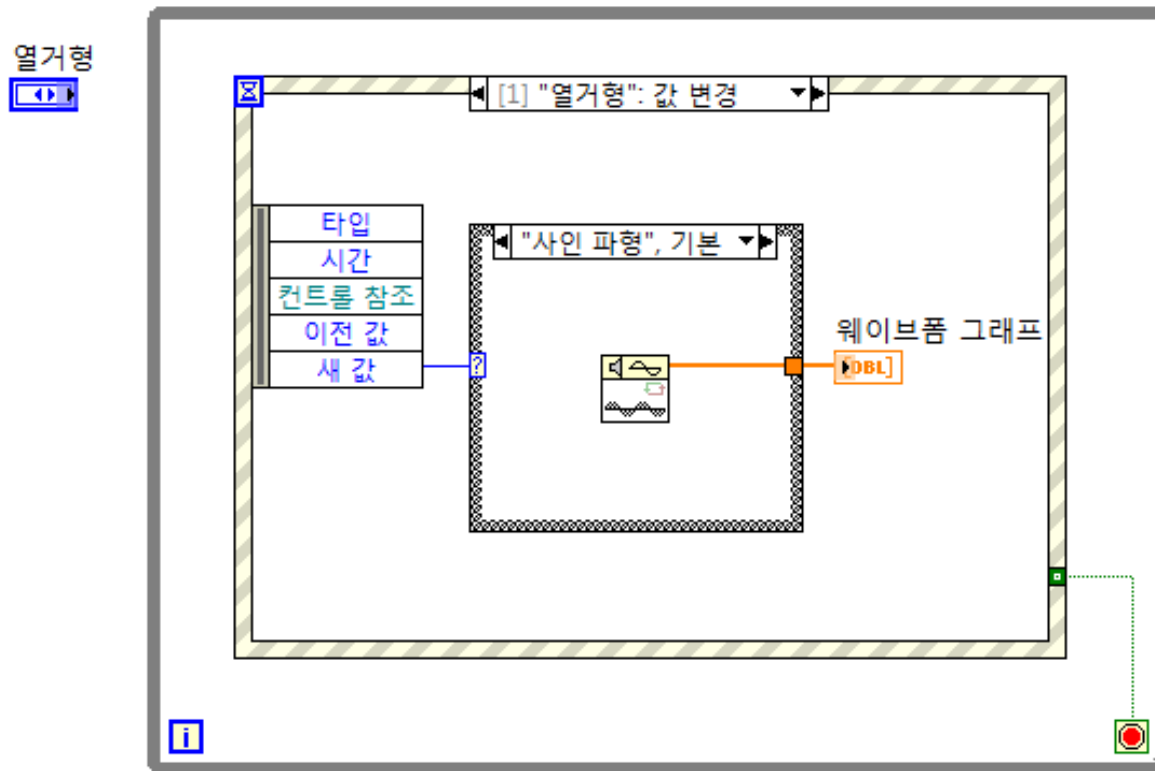
# 실습 1-1: 이벤트 구조 사용

- 놓을 때 래치 버튼



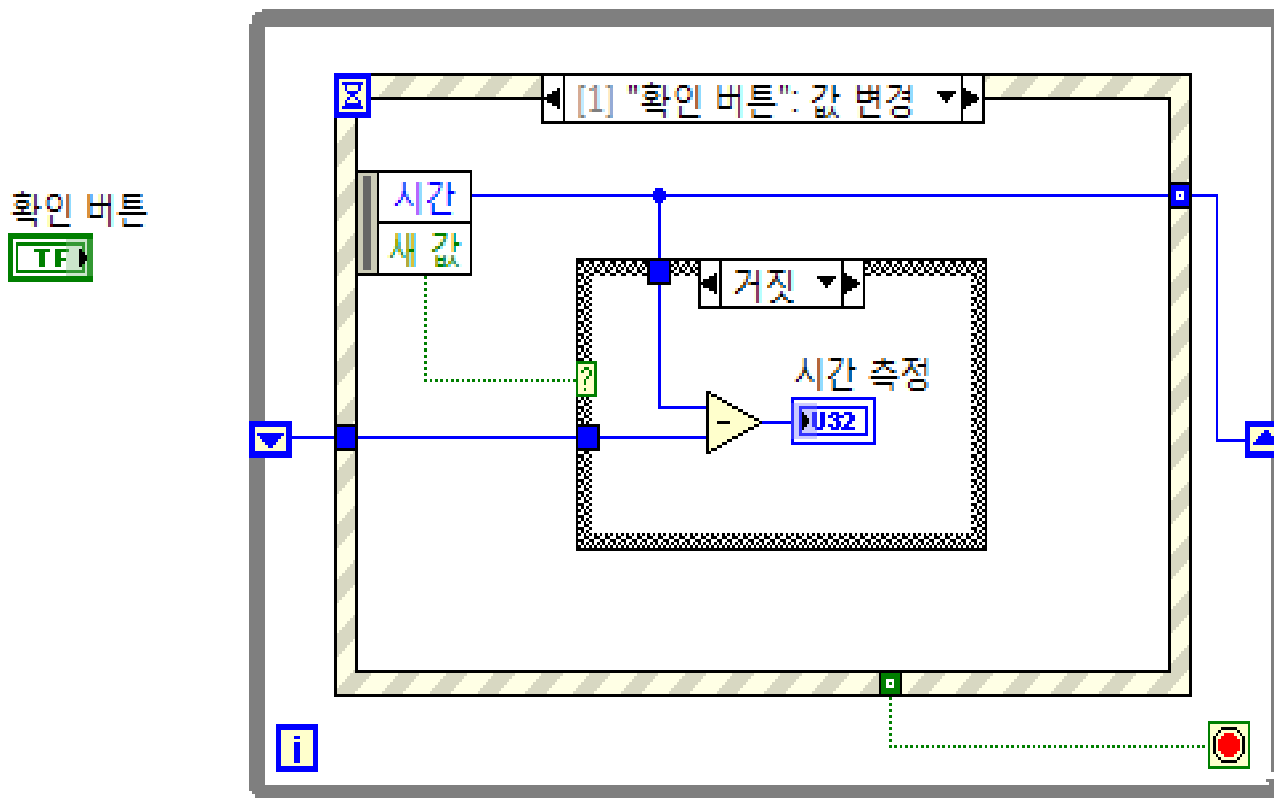
# 실습 1-2. 신호 시뮬레이션

- 열거형 컨트롤



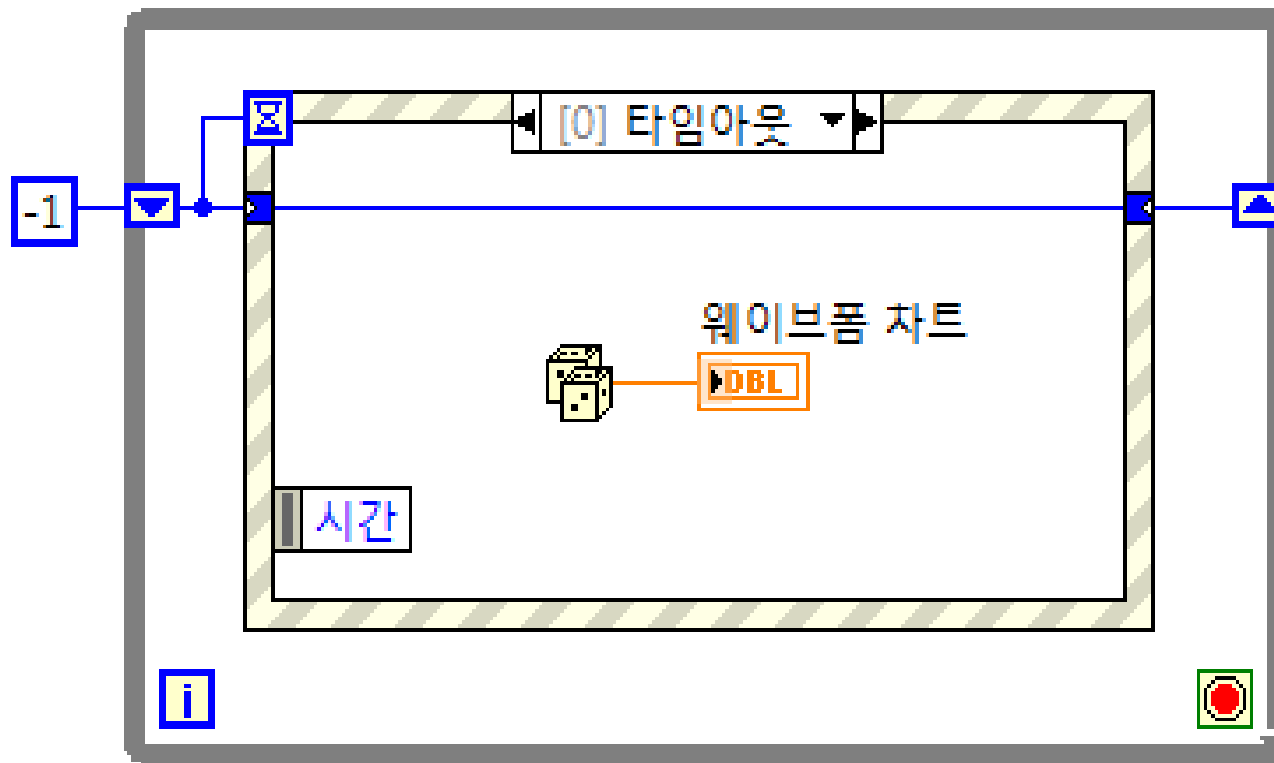
# 실습 1-3. 버튼 클릭 반응 시간

- 놓을 때까지 스위치



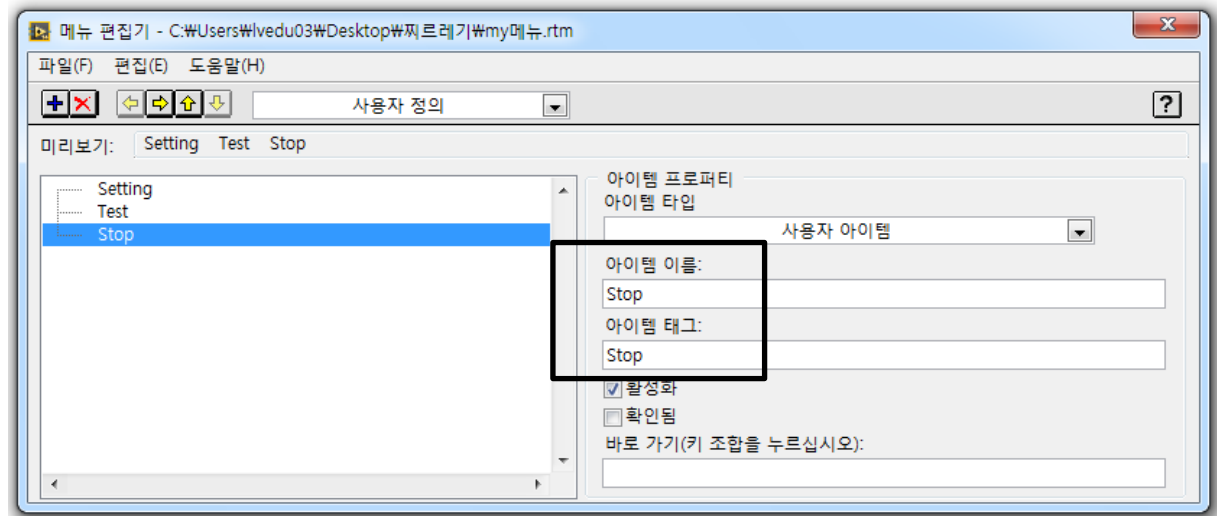
# 실습 1-4. 난수 플롯

- 타임아웃 이벤트



# 런타임 메뉴

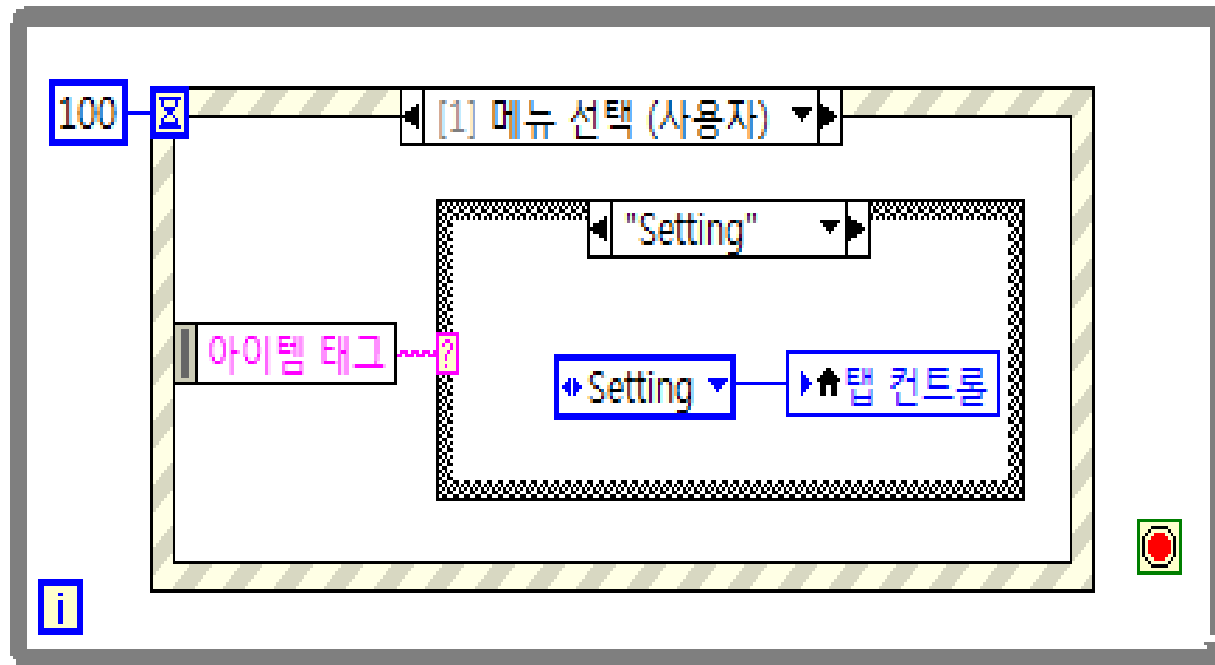
- 편집(E) > 런타임 메뉴(R)
- 메뉴 편집기
- 이벤트 구조
  - <이 VI>
  - 메뉴 선택(사용자)
  - 아이템 태그



# 실습 1-5. 런타임 메뉴

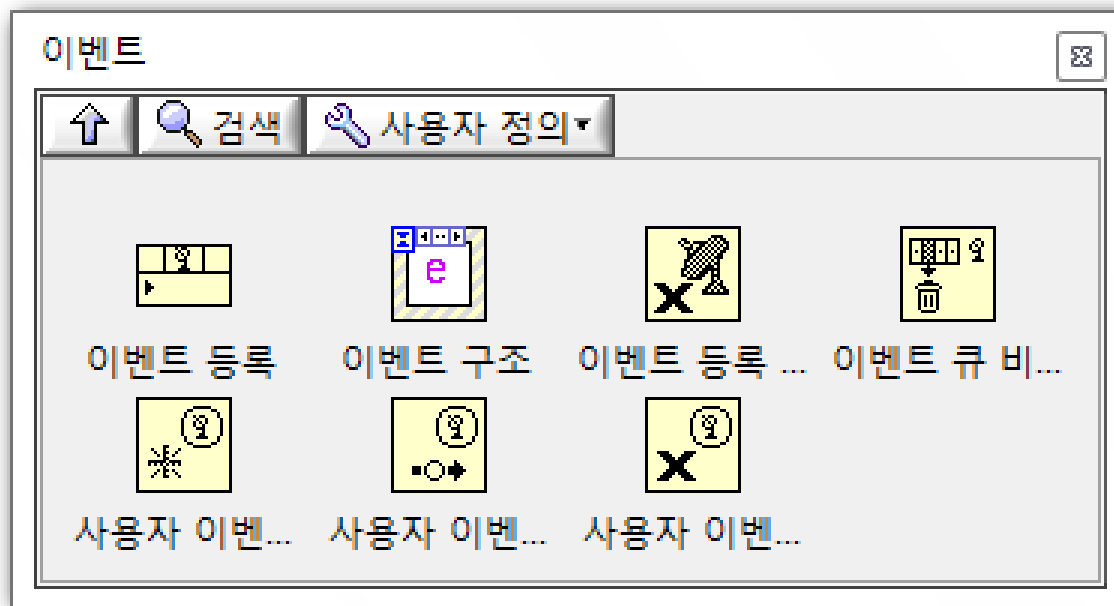
- 편집(E) > 런타임 메뉴(R)

탭 컨트롤  

# (옵션)다이나믹 이벤트 등록 및 사용

- 정적 이벤트 등록 및 사용 → 이벤트 편집
- 다이나믹 이벤트 등록 및 사용 → 이벤트 함수



# 실습 1-6. 다이내믹 이벤트

- ‘사용자 이벤트 데이터 타입’의 상수 라벨 생성

