

# LabVIEW™

## LabVIEW 기본 매뉴얼

## 전세계 기술 지원 및 제품 정보

ni.com

### National Instruments 본사

11500 North Mopac Expressway Austin, Texas 78759-3504 USA 전화 : 512 683 0100

### 전세계 사무소

남아프리카 공화국 27 0 11 805 8197 , 네덜란드 31 0 348 433 466,  
노르웨이 47 0 66 90 76 60, 뉴질랜드 0800 553 322, 대만 886 02 2377 2222,  
덴마크 45 45 76 26 00, 독일 49 0 89 741 31 30, 러시아 7 095 783 68 51 , 레바논 961 0 1 33 28 28,  
말레이시아 1800 887710, 멕시코 01 800 010 0793, 벨기에 32 0 2 757 00 20,  
브라질 55 11 3262 3599, 스웨덴 46 0 8 587 895 00, 스위스 41 56 200 51 51,  
스페인 34 91 640 0085, 슬로베니아 386 3 425 4200, 싱가포르 1800 226 5886,  
영국 44 0 1635 523545, 오스트리아 43 0 662 45 79 90 0, 이스라엘 972 0 3 6393737,  
이탈리아 39 02 413091, 인도 91 80 51190000, 일본 81 3 5472 2970, 중국 86 21 6555 7838,  
체코 420 224 235 774, 캐나다 800 433 3488, 타이 662 278 6777, 포르투갈 351 210 311 210,  
폴란드 48 22 3390150, 프랑스 33 0 1 48 14 24 24, 핀란드 385 0 9 725 725 11,  
한국 82 02 3451 3400, 호주 1800 300 800

### 전세계 사무소

추가적인 지원 정보는 , *기술 지원과 전문 서비스* 부록을 참조하십시오 . National Instruments 문서에 대한 문의사항은 National Instruments 웹 사이트의 ni.com/info 에서 info code feedback 을 입력하십시오 .

# 중요한 정보

## 보증

귀하가 National Instruments 소프트웨어를 받은 미디어에 대하여 영수증 또는 다른 문서에 의해 확인되는 발송일로부터 90 일 동안 재로나 기술에 있어서의 결함으로 인하여 programming instructions의 실행에 있어서 오류가 없을 것을 보증합니다. 만약 National Instruments가 보증 기간동안 그러한 결함에 대한 통지를 받는다면 National Instruments는 programming instructions를 실행하지 못하는 소프트웨어 미디어에 대해서는 회사의 적절한 판단에 따라 이를 수리하거나 교체할 것입니다. National Instruments는 소프트웨어의 작동이 중단되지 않거나 에러가 발생하지 않을 것이라고 보증하지는 않습니다.

어떠한 설비가 보증 작업의 대상이 되기 위하여는 공장에서부터 '제품 반환 공인' 넘버가 붙여져야 하며 그 상자의 바깥 부분에 명확히 그 넘버가 표시되어 있어야 할 것입니다. National Instruments는 반환하기 위해 발송하는 비용을 소유자 측에 지불할 것이며 이는 보증에 의해 보호되는 내용입니다.

National Instruments는 이 문서에 있는 정보가 정확하다고 믿습니다. 이 문서의 기술적인 정확성은 면밀하게 검토되었습니다. 기술적인 오류나 오타가 있는 경우에는 National Instruments는 이 문서의 이번 '판'을 보유한 분에게 사전의 통지를 하지 않고 이 문서의 이후의 '판'을 변경할 권한을 보유합니다. 이 문서를 읽는 분은 에러가 의심된다면 National Instruments와 상담하여야 합니다. 어떤 경우에도 National Instruments는 이 문서와 그 안에 포함되어 있는 정보로부터 발생하는 또는 그와 관련하여 발생하는 손해에 대하여 아무런 책임이 없습니다.

National Instruments는 이 문서에 규정되어 있는 사항을 제외한 다른 사항들에 대해서는 명시적으로든 묵시적으로든 아무런 보증을 하지 않으며 특히 시장성 (MERCHANTABILITY)이나 특정 목적에 대한 적합성에 대하여는 어떠한 보증도 인정하지 않습니다. National Instruments측의 과실이나 부주의로 인한 손해를 회복하기 위한 고객의 권리는 고객이 그에 대해 지불한 액수로 한정될 것입니다. National Instruments는 데이터나 이익의 손실로 인한 손해, 제품 사용으로 인한 손해, 우발적 손해나 간접손해에 대하여는 그 손해의 가능성에 대하여 통고를 하였다 하더라도 그에 대하여 아무런 책임을 지지 않습니다. 부주의를 포함하여 계약상 책임 또는 불법행위상의 책임 등 소송의 형태에 관계없이 National Instruments의 책임 제한이 인정될 것입니다. National Instruments에 대한 소송은 어떠한 소송이라도 그 소송의 원인 발생일로부터 1년 이내에 제기되어야 할 것입니다. 합리적인 이유 없이 지체된 손해배상청구에 대해서는 National Instruments는 책임을 지지 않습니다. 이 문서에서 규정한 보증은 소유자가 National Instruments의 설치, 작동, 유지에 관한 지시를 따르지 않거나 소유자의 제품 변경, 소유자의 남용, 오용, 부주의한 사용; 전력 공급 중단 또는 전압 변화, 화재, 홍수, 사고, 제 3자의 소송 또는 합리적인 통제 범위를 넘는 다른 외부적 사건사고로 야기된 손해, 결함, 기능 장애 또는 서비스 오류들에는 인정되지 않습니다.

## 저작권

저작권법상 이러한 출판물은 National Instruments Corporation의 서면에 의한 사전 동의 없이는 그 일부나 전부를 사진 복사, 녹음, 정보 검색 시스템에 저장하는 것, 번역 등을 포함하여 전자적이거나 기계적인 방법을 막론하고 어떠한 방법이나 형태로도 재발행되거나 전달되는 것이 금지되어 있습니다.

USI (Xerxes C++, ICU, HDF5)에서 사용되는 부분에 관하여는 다음의 저작권이 적용됩니다.

조건과 거부의 목록에 대하여는 USICopyrights.chm을 참조하십시오.

이 제품은 Apache Software Foundation (<http://www.apache.org/>)이 개발한 소프트웨어를 포함합니다. Copyright © 1999 Apache Software Foundation. 판권 소유.

Copyright ©1995-2003 International Business Machines Corporation and others. 판권 소유.

일리노이 대학 이사회의 NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities Copyright 1998, 1999, 2000, 2001, 2003. 판권 소유.

## 상표

National Instruments, NI, ni.com과 Lab VIEW는 National Instruments Corporation의 상표들입니다. National Instruments의 상표들에 관한 더 많은 정보를 원하신다면 [ni.com/legal](http://ni.com/legal)에서 *Terms of Use* 란을 참조하십시오.

Fire Wire®은 Apple Computer, Inc.의 등록된 상표입니다. 이 문서에서 언급된 다른 제품과 회사의 이름들은 각각 해당 회사의 상표이거나 상호들입니다.

National Instruments Alliance Partner Program의 멤버들은 National Instruments와는 다른 독자적인 사업 기구들이며 National Instruments와 어떠한 대리관계나 파트너쉽 또는 joint-venture 관계를 가지고 있지 않습니다.

## 특허권

적절한 위치에서 내쇼날인스트루먼트의 특허권을 참조할 수 있습니다: 소프트웨어의 Help >> Patents, CD의 patents.txt 파일, 또는 [ni.com/patents](http://ni.com/patents).

## NATIONAL INSTRUMENTS 제품 사용에 관한 경고

(1) NATIONAL INSTRUMENTS의 제품들은 외과적인 이식 조직에 사용되거나 그와 관련하여 사용되는 것 또는 작동하지 않는 경우 사람에게 중대한 손상을 야기할 것으로 합리적으로 예견되는 임의의 생명 유지 시스템의 중요한 요소로서 사용되기에 적합할 정도의 신뢰성을 테스트 받지 않았고 그러한 요소로 설계된 것이 아닙니다.

(2) 앞서 설명한 것을 포함하여 어떠한 어플리케이션의 경우에도 소프트웨어 제품 작동의 신뢰성은 전력 공급에 있어서의 불안정, 컴퓨터 하드웨어 기능장애, 컴퓨터 작동 시스템 소프트웨어의 적합성, 활용을 향상시키기 위해 사용되는 컴파일러와 개발 소프트

웨어의 적합성, 설비의 오류, 소프트웨어와 하드웨어의 조화 문제, 전기 모니터링 장치나 조절 장치의 기능 장애 또는 오류, 전기 시스템 (하드웨어 또는 소프트웨어)의 일시적인 오류, 예견되지 않은 사용이나 오용, 사용자나 활용 디자이너의 측면에서의 오류 (이상과 같은 맞지 않는 요인들은 이하에서 집합적으로 "시스템 오류"라고 합니다.) 등을 포함하여 부정적인 요인들에 의하여 손상될 수 있습니다.

이 시스템 오류가 재산이나 사람에게 해를 끼칠 수 있는 위험 (신체적인 손상이나 죽음을 포함한다.)을 발생시킬 수 있는 어플리케이션의 경우에 시스템 오류의 위험 때문에 한가지 형태의 전기적 시스템에만 의존하여서는 안됩니다. 손해, 손상, 죽음을 피하기 위하여 사용자 또는 어플리케이션 디자이너는 백업이나 셋 다운 장치 등을 포함하여 시스템 오류에 대하여 이를 보호하기 위한 단계를 합리적이고 신중하게 봐야 합니다.

각 마지막 사용자 시스템은 맞춤형이며 NATIONAL INSTRUMENTS' TESTING PLATFORMS 과 다르게 사용자나 어플리케이션 디자이너는 NATIONAL INSTRUMENTS 의 제품을 다른 제품들과 결합하여 NATIONAL INSTRUMENTS 가 평가하거나 고려하지 않은 방법으로 사용할 수 있기 때문에 사용자 또는 어플리케이션 디자이너는 NATIONAL INSTRUMENTS 제품들이 시스템이나 어플리케이션의 안전 수준, 적합한 디자인, 공정 등을 포함하여 시스템이나 활용에 결합될 때 마다 NATIONAL INSTRUMENTS 제품들의 적합성을 최종적으로 입증하거나 검증할 책임이 있습니다.

# 목차

---

## 이 매뉴얼에 관하여

규약.....	xi
---------	----

## 제 1 장 LabVIEW 소개

LabVIEW 문서 리소스 .....	1-1
LabVIEW 도움말.....	1-1
인쇄 문서.....	1-2
Readme 문서 .....	1-3
LabVIEW VI 템플릿, 예제 VI, 도구 .....	1-3
LabVIEW VI 템플릿 .....	1-3
LabVIEW 예제 VI.....	1-4
DAQ 설정을 위한 LabVIEW 도구 (Windows).....	1-4

## 제 2 장 버추얼 인스트루먼트 소개

프런트패널 .....	2-1
블록다이어그램 .....	2-2
터미널 .....	2-3
노드 .....	2-3
와이어 .....	2-3
구조 .....	2-4
아이콘 및 커넥터 팬 .....	2-4
VI 와 SubVI 사용자 정의 및 사용하기.....	2-5

## 제 3 장 LabVIEW 환경

시작하기 윈도우 .....	3-1
컨트롤 팔레트.....	3-1
함수 팔레트 .....	3-2
컨트롤과 함수 팔레트 탐색하기 .....	3-2
도구 팔레트.....	3-3
메뉴와 도구 모음.....	3-3
메뉴 .....	3-3
바로 가기 메뉴 .....	3-4
VI 도구 모음 .....	3-4
프로젝트 탐색기 윈도우 도구 모음.....	3-4
기본 도움말 윈도우 .....	3-5
프로젝트 탐색기 윈도우.....	3-5
탐색 윈도우 .....	3-6

작업 환경 사용자 정의하기 .....	3-6
컨트롤과 함수 팔레트를 사용자 정의하기 .....	3-6
작업 환경 옵션 셋팅 .....	3-7

## 제 4 장 프런트패널 만들기

프런트패널 컨트롤과 인디케이터 .....	4-1
컨트롤과 인디케이터의 스타일 .....	4-1
일반 및 클래식 컨트롤과 인디케이터 .....	4-1
시스템 컨트롤과 인디케이터 .....	4-2
숫자 디스플레이, 슬라이드, 스크롤 막대, 노브, 다이얼, 타임스탬프 .....	4-2
숫자 컨트롤과 인디케이터 .....	4-2
슬라이드 컨트롤과 인디케이터 .....	4-3
스크롤 막대 컨트롤과 인디케이터 .....	4-3
회전식 컨트롤과 인디케이터 .....	4-3
타임스탬프 컨트롤과 인디케이터 .....	4-4
그래프와 차트 .....	4-4
버튼, 스위치, 빛 .....	4-4
라디오 버튼 컨트롤 .....	4-5
텍스트 엔트리 박스, 라벨, 경로 디스플레이 .....	4-5
문자열 컨트롤과 인디케이터 .....	4-5
콤보 박스 컨트롤 .....	4-6
경로 컨트롤과 인디케이터 .....	4-6
배열, 행렬, 클러스터 컨트롤과 인디케이터 .....	4-6
리스트박스, 트리 컨트롤, 테이블 .....	4-6
리스트박스 .....	4-7
트리 컨트롤 .....	4-7
테이블 .....	4-7
링과 열거형 타입 컨트롤 및 인디케이터 .....	4-7
링 컨트롤 .....	4-7
열거형 타입 컨트롤 .....	4-8
컨테이너 컨트롤 .....	4-8
탭 컨트롤 .....	4-8
서브패널 컨트롤 .....	4-8
I/O 이름 컨트롤과 인디케이터 .....	4-9
웨이브폼 컨트롤 .....	4-9
디지털 웨이브폼 컨트롤 .....	4-9
디지털 데이터 컨트롤 .....	4-10
객체 또는 어플리케이션의 참조 .....	4-10
.NET 과 ActiveX 컨트롤 (Windows) .....	4-11
프런트패널 객체 설정하기 .....	4-11
옵션 원소 보이기와 숨기기 .....	4-11
컨트롤을 인디케이터로 또는 인디케이터를 컨트롤로 바꾸기 .....	4-12
프런트패널 객체 대체하기 .....	4-12

프런트패널 설정하기 .....	4-12
객체 색칠하기 .....	4-12
객체 정렬 및 간격 조절하기 .....	4-13
객체 그룹화 및 잠금 설정 .....	4-13
객체 크기 조정하기 .....	4-13
윈도우 크기를 조정하지 않고 프런트패널에 공간 추가하기 .....	4-14
라벨 붙이기 .....	4-14
텍스트 특성 .....	4-15
사용자 인터페이스 디자인하기 .....	4-15
프런트패널 컨트롤과 인디케이터 사용하기 .....	4-15
대화 상자 디자인하기 .....	4-16

## 제 5 장 블록다이어그램 만들기

블록다이어그램 객체 .....	5-1
블록다이어그램 터미널 .....	5-1
컨트롤과 인디케이터 데이터 타입 .....	5-2
상수 .....	5-3
블록다이어그램 노드 .....	5-3
다형성 VI 와 함수 .....	5-4
함수 개요 .....	5-4
함수에 터미널 추가하기 .....	5-5
내장된 VI 와 함수 .....	5-5
익스프레스 VI .....	5-5
와이어를 사용하여 블록다이어그램 객체에 연결 하기 .....	5-6
와이어 모양과 구조 .....	5-6
객체 연결하기 .....	5-7
와이어 꺾기 .....	5-7
와이어 연결 취소하기 .....	5-8
자동으로 객체 와이어하기 .....	5-8
와이어 선택하기 .....	5-8
깨진 와이어 수정하기 .....	5-8
강제 변환점 .....	5-9
블록다이어그램의 데이터 흐름 .....	5-9
데이터 의존성과 인위적인 데이터 의존성 .....	5-10
데이터 의존성 잃기 .....	5-11
흐름 파라미터 .....	5-11
데이터 흐름과 메모리 관리 .....	5-12
블록다이어그램 디자인하기 .....	5-12

## 제 6 장 VI 실행하고 디버깅하기

VI 실행하기 .....	6-1
깨진 VI 수정하기 .....	6-2
VI 가 깨진 원인 찾기 .....	6-2
깨진 VI 의 일반적인 원인 .....	6-3
디버깅 기술 .....	6-3
실행 하이라이트하기 .....	6-3
단계별 실행 .....	6-4
프로브 도구 .....	6-4
브레이크포인트 .....	6-4
에러 핸들링하기 .....	6-5
에러 클러스터 .....	6-6
에러 핸들링을 위해 While 루프 사용하기 .....	6-7
에러 핸들링을 위해 케이스 구조 사용하기 .....	6-7

## 제 7 장 VI 와 SubVI 생성하기

예제 검색하기 .....	7-1
내장된 VI 와 함수 사용하기 .....	7-1
SubVI 생성하기 .....	7-1
아이콘 생성하기 .....	7-2
커넥터 팬 만들기 .....	7-2
VI 의 섹션에서 subVI 생성하기 .....	7-3
SubVI 프런트패널 디자인하기 .....	7-4
VI 의 계층구조 보기 .....	7-4
다형성 VI .....	7-4
VI 저장하기 .....	7-6
VI 이름 붙이기 .....	7-6
이전 버전으로 저장하기 .....	7-6
VI 사용자 정의하기 .....	7-7

## 제 8 장 루프와 구조

For 루프와 While 루프 구조 .....	8-2
For 루프 .....	8-2
While 루프 .....	8-3
타이밍 컨트롤하기 .....	8-4
오토인덱싱 루프 .....	8-4
오토인덱싱을 사용하여 For 루프 카운트 설정 하기 .....	8-5
While 루프의 오토인덱싱 .....	8-5
루프를 이용한 배열 만들기 .....	8-6
루프의 시프트 레지스터와 피드백 노드 .....	8-6

시프트 레지스터 .....	8-6
피드백 노드 .....	8-9
루프의 기본 데이터 .....	8-10
케이스, 시퀀스, 이벤트 구조 .....	8-10
케이스 구조 .....	8-10
케이스 선택자의 값과 데이터 타입 .....	8-11
입력과 출력 터널 .....	8-12
여러 핸들링을 위해 케이스 구조 사용하기 .....	8-12
시퀀스 구조 .....	8-12
이벤트 구조 .....	8-14

## 제 9 장 문자열, 배열, 클러스터를 이용한 데이터의 그룹화

데이터를 문자열로 그룹화하기 .....	9-1
프런트패널의 문자열 .....	9-1
문자열 디스플레이 타입 .....	9-2
테이블 .....	9-2
문자열 편집, 포맷, 분석하기 .....	9-2
문자열 포맷과 분석 .....	9-3
배열과 클러스터를 이용하여 데이터 그룹화하기 .....	9-3
배열 .....	9-3
제약 .....	9-4
인덱스 .....	9-4
배열의 예 .....	9-4
배열 컨트롤, 인디케이터, 상수 생성하기 .....	9-7
여러 차원 배열 생성하기 .....	9-7
배열 함수 .....	9-8
배열의 기본 데이터 .....	9-10
클러스터 .....	9-10
클러스터 원소의 순서 .....	9-10
클러스터 함수 .....	9-11
클러스터 컨트롤, 인디케이터, 상수 생성하기 .....	9-11

## 제 10 장 그래프와 차트

그래프와 차트의 종류 .....	10-1
웨이브폼 그래프와 차트 .....	10-2
웨이브폼 그래프 .....	10-2
웨이브폼 차트 .....	10-3
웨이브폼 데이터 타입 .....	10-3
XY 그래프 .....	10-3
강도 그래프와 차트 .....	10-4
강도 차트 .....	10-5
강도 그래프 .....	10-6
디지털 웨이브폼 그래프 .....	10-7

디지털 웨이브폼 데이터 타입 .....	10-10
3D 그래프 .....	10-10
그래프와 차트 사용자 정의 .....	10-13
여러 X, Y 스케일 사용 .....	10-13
오토 스케일 .....	10-13
여러 X, Y 스케일 포맷하기 .....	10-13
그래프 팔레트 사용하기 .....	10-14
그래프와 차트의 모양 사용자 정의하기 .....	10-14
그래프 사용자 정의하기 .....	10-15
그래프 커서 사용하기 .....	10-16
그래프 주석 사용하기 .....	10-16
3D 그래프 사용자 정의하기 .....	10-17
차트 사용자 정의하기 .....	10-18
차트 히스토리 길이 설정하기 .....	10-18
차트 업데이트 모드 설정하기 .....	10-18
오버레이와 다층 플롯 사용하기 .....	10-19

## 제 11 장 파일 I/O

파일 I/O 의 기초 .....	11-1
파일 I/O 포맷 선택하기 .....	11-2
일반적인 파일 I/O 작업을 위한 VI 와 함수 사용 하기 .....	11-3
스토리지 VI 사용하기 .....	11-5
텍스트와 스프레드시트 파일 생성하기 .....	11-6
데이터를 파일에 기록하기와 포맷하기 .....	11-7
파일에서 데이터를 스캔하기 .....	11-7
2 진 파일 생성하기 .....	11-7
데이터로그 파일 생성하기 .....	11-7
파일에 웨이브폼 쓰기 .....	11-8
파일에서 웨이브폼 읽어오기 .....	11-8

## 제 12 장 VI 문서화 및 인쇄하기

VI 문서화하기 .....	12-1
VI 인쇄하기 .....	12-2

## 부록 A 기술 지원과 전문 서비스

### 용어집

### 색인

# 이 매뉴얼에 관하여

이 매뉴얼을 사용하기 전에, *LabVIEW 시작하기* 매뉴얼을 길잡이로 사용하여 LabVIEW의 그래픽 프로그래밍 환경 및 데이터 수집과 인스트루먼트 컨트롤 어플리케이션을 만드는데 사용하는 기본적인 LabVIEW의 특징에 익숙해질 수 있습니다.

이 매뉴얼은 사용자가 테스트와 측정, 데이터 수집, 인스트루먼트 컨트롤, 데이터로깅, 측정 분석, 리포트 생성 등의 어플리케이션을 구현하기 위해 사용하는 LabVIEW의 프로그래밍 개념, 기술, 특징, VI, 함수에 대해 설명합니다.

이 매뉴얼은 *LabVIEW 도움말*의 내용 중 일부이며, 이 매뉴얼의 모든 내용은 LabVIEW 도움말에 포함되어 있습니다. 이 매뉴얼에서 설명된 개념에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.

이 매뉴얼에는 각 팔레트, 도구, 메뉴, 대화 상자, 컨트롤, 인디케이터, 또는 내장 VI나 함수에 대한 특정한 정보가 포함되어 있지 않습니다. 이러한 아이템에 대한 추가적인 정보와 LabVIEW 기능 사용 및 특정한 어플리케이션 구현에 대한 단계별 설명은 *LabVIEW 도움말*을 참조하십시오. *LabVIEW 도움말*과 도움말에 접근하는 방법에 대한 추가적인 정보는 1장 *LabVIEW 소개*의 *LabVIEW 문서 리소스* 섹션을 참조하십시오.

## 규약

이 매뉴얼은 다음의 규약을 사용합니다:

»

» 기호는 메뉴 항목이나 대화 상자 옵션을 거쳐 최종 작업을 수행하도록 사용자를 이끌어 줍니다. 시퀀스 **파일»페이지 셋업»옵션**은 **파일** 메뉴를 풀다운하여 **페이지 설정** 아이템을 선택하고 마지막 대화 상자에서 **옵션**을 선택하라는 의미입니다.



이 아이콘은 사용자에게 유용한 정보를 알려주는 팁을 나타냅니다.



이 아이콘은 사용자에게 중요한 정보를 알려주는 노트를 나타냅니다.



이 기호는 부상, 데이터 손실, 또는 시스템 충돌을 미연에 방지하기 위한 주의사항을 나타냅니다.

### 굵은체

굵은체 텍스트는 메뉴 아이템 및 대화 상자 옵션과 같이 소프트웨어에서 선택하거나 클릭해야 하는 아이템을 나타냅니다. 매개변수 이름, 프런트패널에 있는 컨트롤 및 인디케이터, 대화상자, 대화상자를 이루는 부분들, 메뉴 이름 및 팔레트 이름도 굵은 글꼴로 표시됩니다.

이 매뉴얼에 관하여

- 이탤릭* 이탤릭 텍스트는 변수, 강조, 상호 참조, 또는 중요한 개념에 대한 소개를 나타냅니다. 또한 이탤릭 텍스트는 제공해야 하는 단어나 값을 위한 자리 표시자인 텍스트를 나타냅니다.
- 고정 폭 이 폰트의 텍스트는 사용자가 키보드로 입력해야 하는 텍스트나 문자, 코드의 섹션, 프로그래밍 예제, 구문 예제를 표시합니다. 또한 이 폰트를 디스크 드라이버, 경로, 디렉토리, 프로그램, 서브프로그램, 서브루틴, 디바이스 이름, 동작, 변수, 파일 이름, 확장자의 적절한 이름에 사용합니다.
- 고정 폭 굵은체 이 폰트의 굵은체 텍스트는 컴퓨터가 자동적으로 화면에 인쇄하는 메시지와 응답을 표시합니다. 또한, 이 폰트는 프로그램 코드에서 다른 예제와 구분되는 부분을 강조할 때 사용되기도 합니다.
- 고정 폭 *이탤릭* 이 폰트의 이탤릭 텍스트는 사용자가 지정해야 하는 단어나 값을 위한 자리 표시자인 텍스트를 나타냅니다.
- 플랫폼** 이 폰트의 텍스트는 특정한 플랫폼을 나타내며 이 텍스트 다음의 내용은 해당 플랫폼에만 적용된다는 것을 나타냅니다.
- 마우스 오른쪽 버튼 클릭 **(Mac OS)** 에서 <Command>-클릭은 마우스 오른쪽 버튼 클릭과 같은 기능을 합니다.

# LabVIEW 소개

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) 는 그래픽 방식의 프로그래밍 언어로서 어플리케이션을 작성하는데 텍스트 대신 아이콘을 사용합니다. 프로그램 실행 순서가 명령에 의해 결정되는 텍스트 기반의 프로그래밍 언어와 달리, LabVIEW 는 VI 와 함수의 실행 순서가 블록다이어그램에서 노드를 통과하는 데이터의 흐름으로 결정되는 데이터 흐름 프로그래밍을 사용합니다. VI 또는 버추얼 인스트루먼트는 물리적인 인스트루먼트와 유사하게 작동하는 LabVIEW 프로그램입니다.

LabVIEW 에서는 도구와 객체 세트를 사용하여 사용자 인터페이스를 생성합니다. 사용자 인터페이스는 프런트패널이라고 합니다. 사용자 인터페이스를 구성한 다음 이 프런트패널 객체를 컨트롤하기 위하여 그래픽 형태의 함수를 사용하여 코드를 추가할 수 있습니다. 이 그래픽적인 소스 코드를 G 코드 또는 블록다이어그램 코드라고도 합니다. 블록다이어그램에는 이 코드가 포함됩니다. 어떤 면에서 블록다이어그램은 흐름도와 비슷합니다.

특별한 어플리케이션을 개발하기 위해 몇가지 소프트웨어 툴킷을 추가로 구입할 수 있습니다. 모든 툴킷은 LabVIEW 와 원활하게 연결됩니다. 이 툴킷에 대한 추가적인 내용은 내쇼날인스트루먼트의 웹 사이트인 [ni.com/toolkits](http://ni.com/toolkits) 를 참고하십시오.

## LabVIEW 문서 리소스

LabVIEW 에는 신규 사용자와 기존 LabVIEW 사용자를 위한 광범위한 온라인 및 인쇄 문서가 포함되어 있습니다.

### LabVIEW 도움말

*LabVIEW 도움말*을 사용하여 LabVIEW 프로그래밍 개념, LabVIEW 사용에 대한 단계별 설명, LabVIEW VI, 함수, 팔레트, 메뉴, 도구에 대한 참조 정보에 접근합니다.

*LabVIEW 도움말*은 NI Developer Zone, 기술 지원 데이터 베이스, 그리고 제품 매뉴얼 라이브러리와 같은 내쇼날인스트루먼트 웹사이트의 기술 지원 리소스 링크를 포함하고 있습니다.

**도움말» LabVIEW 도움말 검색**을 선택해서 *LabVIEW 도움말*에 접근합니다. 또한, *LabVIEW 도움말*에서 도움말 항목이나 도움말 항목 모음을 인쇄할 수 있습니다.

도움말 항목을 인쇄하는 것에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.



**노트**

**(Mac OS)** National Instruments 는 *LabVIEW 도움말*을 보기 위해 Safari 1.0 이나 이후 버전, 또는 Firefox 1.0.2 나 이후 버전을 사용할 것을 권장합니다.

**(Linux)** National Instruments 는 *LabVIEW 도움말*을 보기 위해 Netscape 6.0 이나 이후 버전, Mozilla 1.2 나 이후 버전, 또는 Firefox 1.0.2 나 이후 버전을 사용할 것을 권장합니다.

툴킷, 모듈, 또는 드라이버와 같은 LabVIEW 애드온을 설치한 후, 해당 애드온의 문서는 *LabVIEW 도움말*에 나타나거나 **도움말 > 애드온 도움말**을 선택하여 접근할 수 있는 별도의 도움말 시스템에 나타납니다. 이 때 **애드온 도움말**은 해당 애드온에 대한 별도의 도움말 시스템의 이름입니다.

## 인쇄 문서

다음 인쇄 문서는 LabVIEW 를 사용하는데 도움이 될 수 있는 정보를 포함하고 있습니다:

- *LabVIEW 시작하기*—LabVIEW 의 그래픽 프로그래밍 환경 및 데이터 수집과 인스트루먼트 컨트롤 어플리케이션을 만드는데 사용하는 기본적인 LabVIEW 의 특징들에 익숙해지기 위하여 본 매뉴얼을 길라잡이로 사용합니다.
- *LabVIEW 도움 카드*— 이 카드는 문서 리소스, 키보드 바로 가기 키, 데이터 타입 터미널, 편집, 실행, 디버깅 도구 정보에 대한 참조로 사용합니다.
- *LabVIEW 기본 매뉴얼*— 이 매뉴얼을 사용하여 테스트와 측정, 데이터 수집, 인스트루먼트 컨트롤, 데이터로깅, 측정 분석, 리포트 생성 어플리케이션을 만드는 LabVIEW 프로그래밍 개념, 기술, 특징, VI, 함수에 대해 배울 수 있습니다. *LabVIEW 도움말*에는 이 매뉴얼의 모든 내용이 포함되어 있습니다.
- *LabVIEW 릴리즈 노트*— LabVIEW 를 설치하고 제거할 때 이 릴리즈 노트를 사용합니다. 또한, 릴리즈 노트는 LabVIEW 어플리케이션 빌더를 포함하여 LabVIEW 소프트웨어에 필요한 시스템 사양을 설명합니다.
- *LabVIEW 업그레이드 노트*— Windows, Mac OS, UNIX 에서 LabVIEW 를 최신 버전으로 업그레이드할 때 업그레이드 노트를 사용합니다. 또한 이 업그레이드 노트는 업그레이드했을 때 접하게 될 새로운 특징과 유의사항을 설명합니다.

이러한 문서는 인쇄 문서 및 labview\manuals 디렉토리에서 PDF 로 이용할 수 있습니다. PDF 파일을 보려면 Adobe Acrobat Reader 5.0.5 나 이후 버전이 설치되어 있어야 합니다. 사용자가 이러한 매뉴얼의 PDF 버전을 검색하려면 반드시 Adobe Reader with Search and Accessibility 6.x

또는 이후 버전이 설치되어 있어야 합니다. **(Mac OS)** PDF 파일을 보려면 Adobe Reader with Search and Accessibility 6.x 또는 이후 버전이 설치되어 있어야 합니다.

Acrobat Reader 를 다운로드 받기 위해서는 Adobe Systems Incorporated 웹 사이트의 [www.adobe.com](http://www.adobe.com) 를 참조합니다. 최신 설명 자료들을 업데이트하기 위해서는 [ni.com/manuals](http://ni.com/manuals) 의 내쇼날인스트루먼트 제품 매뉴얼 라이브러리를 참조합니다.

## Readme 문서

다음 readme 문서는 LabVIEW 를 사용하는데 도움이 될 수 있는 정보를 포함하고 있습니다:

- *LabVIEW Readme* — LabVIEW 의 설치와 업그레이드시 유의사항, 호환성 유의사항, LabVIEW 이전 버전에서 변경점, LabVIEW 에서 알려진 유의사항이 포함된 중요한 최신 정보를 배울 때 이 파일을 사용합니다. **시작»프로그램»National Instruments»LabVIEW 8.0»Readme** 를 선택하여 `readme.html` 를 열거나 `labview\readme` 디렉토리를 탐색하고 `readme.html` 을 열어 *LabVIEW Readme* 를 엽니다.
- *LabVIEW Application Builder User Guide* — 이 문서를 사용하여 LabVIEW 어플리케이션 빌더에 대해 배울 수 있습니다. 어플리케이션 빌더는 LabVIEW Professional Development System 에 포함되어 있으며 별도로 구매 가능합니다. **시작»프로그램»National Instruments»LabVIEW 8.0»Readme** 를 선택하여 `readme_AppBldr.html` 를 열거나 `labview\readme` 디렉토리를 탐색하고 `readme_AppBldr.html` 을 열어 *LabVIEW 어플리케이션 빌더 Readme* 를 엽니다.

## LabVIEW VI 템플릿, 예제 VI, 도구

LabVIEW VI 템플릿, 예제 VI, 도구를 이용하여 VI 를 디자인하고 작성합니다.

### LabVIEW VI 템플릿

내장된 VI 템플릿에는 일반 측정 어플리케이션을 만들기 시작할 때 필요한 subVI, 함수, 구조, 프런트패널 객체가 있습니다. VI 템플릿은 반드시 저장하되 제목없음 VI 로 열립니다. **파일»새로 만들기** 를 선택하여 **새로 만들기** 대화 상자를 디스플레이합니다. 이 대화 상자에는 내장 VI 템플릿이 나열됩니다. 또한, **시작하기** 윈도우의 **새로 만들기** 링크를 클릭하여 **새로 만들기** 대화 상자를 디스플레이할 수 있습니다.

## LabVIEW 예제 VI

LabVIEW 는 바로 사용하거나 사용자가 생성하는 VI 에 붙여서 사용할 수 있는 수백 개의 예제 VI 를 검색합니다 . 어플리케이션에 맞도록 예제를 수정하거나 , 하나 또는 여러 예제를 복사하여 생성한 VI 에 붙여넣을 수 있습니다 . **도움말>>예제 찾기**를 선택하여 NI 예제 탐색기로 예제 VI 를 탐색하거나 검색합니다 .

추가적인 예제 VI 는 [ni.com/zone](http://ni.com/zone) 의 NI Developer Zone 을 참조합니다 .

또한 *LabVIEW* *도움말*에서 VI 참조와 함수 참조 토픽의 아래에 위치한 **예제 열기**와 **관련 예제 탐색** 버튼을 사용하여 예제에 접근할 수 있습니다 . 주제와 관련된 예제 VI 를 열기 위해서 **예제 열기** 버튼을 클릭합니다 . NI 예제 탐색기를 열고 관련된 예제 VI 들을 보려면 **관련 예제 탐색** 버튼을 클릭합니다 .

또한 , 블록 다이어그램 또는 고정된 팔레트의 VI 나 함수에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **예제**를 선택하여 도움말 토픽과 해당 VI 나 함수 예제의 링크를 디스플레이할 수 있습니다 .

## DAQ 설정을 위한 LabVIEW 도구 (Windows)

Measurement & Automation Explorer (MAX) 를 사용하여 측정 디바이스를 설정할 때 도움을 받으십시오 . **도구>> Measurement & Automation Explorer**를 선택하여 MAX 를 시작하고 내쇼날인스트루먼트의 하드웨어와 소프트웨어를 설정합니다 . National Instruments 디바이스 드라이버 CD 로부터 MAX 를 설치합니다 .

다른 타입의 인스트루먼트를 컨트롤하는데 대한 정보는 *LabVIEW* *도움말* **목차** 탭의 **인스트루먼트 컨트롤하기** 모음을 참조하십시오 .

DAQ Assistant 를 이용하여 그래픽하게 채널 또는 일반 측정 태스크를 설정합니다 . NI-DAQmx 가 설치되어 있지 않으면 **함수** 팔레트에 DAQ Assistant 익스프레스 VI 가 나타나지 않습니다 . NI-DAQmx 설치에 대한 추가적인 정보는 *DAQ Getting Started Guide* 을 참조하십시오 . 다음과 같은 방법으로 DAQ Assistant 에 접근할 수 있습니다 :

- DAQ Assistant Express VI 를 블록 다이어그램에 놓습니다 .
- DAQmx global channel 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **New Channel (DAQ Assistant)** 를 선택합니다 . DAQmx task name 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **New Task (DAQ Assistant)** 를 선택합니다 . DAQmx Scale Name 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **New Scale (DAQ Assistant)** 를 선택합니다 .

- Measurement & Automation Explorer를 실행하고 **Configuration** 트리에서 **Data Neighborhood** 나 **Scales** 를 선택합니다 . **Create New** 버튼을 클릭합니다 . NI-DAQmx channel, task, 또는 scale 을 설정합니다 .

## 버추얼 인스트루먼트 소개

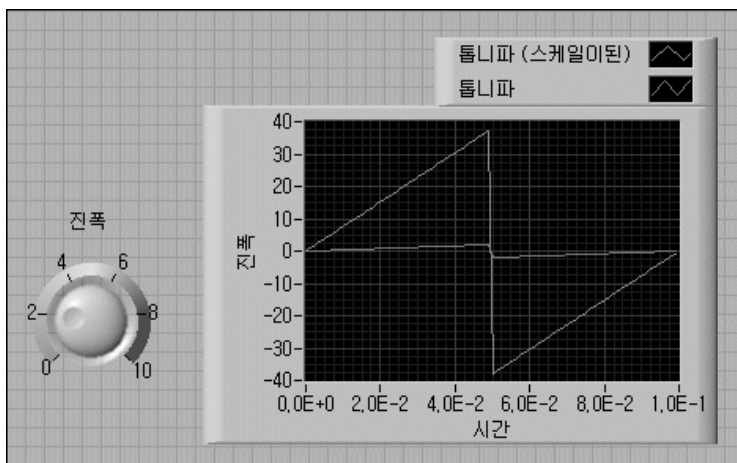
LabVIEW 프로그램은 버추얼 인스트루먼트 또는 VI 라고 불립니다 . 외관과 작동이 오실로스코프, 멀티미터와 같은 물리적 인스트루먼트와 비슷하기 때문입니다 . 모든 VI 는 함수를 사용하며 , 함수는 사용자 인터페이스 또는 다른 소스로부터의 입력을 조작하며 해당 정보를 디스플레이하거나 다른 파일 또는 컴퓨터로 옮깁니다 .

VI 는 다음의 세 구성요소를 포함합니다 :

- **프런트패널** — 사용자 인터페이스 역할을 합니다 .
- **블록다이어그램** — VI 의 기능을 정의하는 그래픽 소스 코드를 포함합니다 .
- **아이콘과 커넥터 핀** — VI 의 인터페이스를 식별하여 VI 를 다른 VI 에서 이용할 수 있도록 합니다 . 다른 VI 안의 VI 를 subVI 라고 부릅니다 . SubVI 는 텍스트 기반 프로그래밍 언어의 서브루틴에 해당합니다 .

## 프런트패널

프런트패널은 VI 의 사용자 인터페이스입니다 . 다음 그림은 프런트패널의 예제를 보여줍니다 .



각각 VI 의 대화식 입력과 출력 터미널인 컨트롤과 인디케이터를 사용하여 프런트패널을 구성합니다 . 컨트롤은 노브 , 버튼 , 다이얼 , 그리고 기타 입력

메커니즘입니다. 인디케이터는 그래프, LED, 기타 출력 디스플레이입니다. 컨트롤은 인스트루먼트의 입력 메커니즘을 시뮬레이션하고 VI의 블록 다이어그램에 데이터를 제공합니다. 인디케이터는 인스트루먼트의 출력 메커니즘을 시뮬레이션하고 블록 다이어그램에서 수집하거나 생성하는 데이터를 디스플레이합니다.

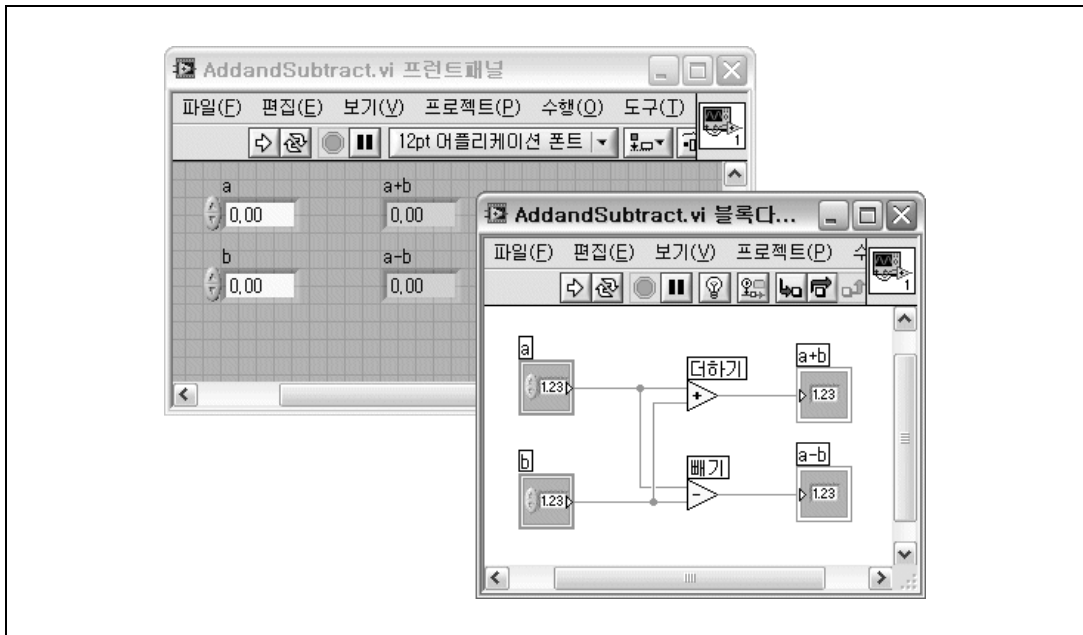
프런트패널에 대한 추가적인 정보는 제 4 장, *프런트패널 만들기*를 참조하십시오.

## 블록다이어그램

프런트패널을 만든 후, 그래픽 형태의 함수를 사용하여 프런트패널의 객체를 제어하는 코드를 추가합니다. 블록다이어그램은 G 코드 또는 블록다이어그램 코드로 알려진 그래픽 소스 코드를 포함합니다. 프런트패널 객체는 블록다이어그램에서 터미널로 나타납니다.

블록다이어그램에 대한 추가적인 정보는 제 5 장, *블록다이어그램 만들기*를 참조합니다.

다음 VI는 몇가지 주요한 블록다이어그램 객체를 보여줍니다 — 터미널, 함수, 와이어.



## 터미널

터미널은 컨트롤이나 인디케이터의 데이터 타입을 나타냅니다. 블록 다이어그램에서 컨트롤과 인디케이터를 아이콘 형태로 나타내거나 또는 데이터 타입 터미널로 나타내도록 프론트패널 컨트롤이나 인디케이터를 설정할 수 있습니다. 기본적으로, 프론트패널 객체는 아이콘 터미널로 나타납니다. 예를 들어, 다음에 보이는 노브 아이콘 터미널은 프론트패널의 노브를 나타냅니다.



터미널 아래 부분의 DBL 은 배정도 부동소수의 데이터 타입을 나타냅니다. 다음에 보이는 DBL 터미널은 배정도 부동소수 컨트롤을 나타냅니다.



LabVIEW 의 데이터 타입에 대한 추가적인 정보는 5 장, *블록다이어그램 만들기의 컨트롤과 인디케이터 데이터 타입* 섹션을 참조하십시오.

터미널은 프론트패널과 블록다이어그램 사이에 정보를 교환하는 통로입니다. 프론트패널 컨트롤 (이전 그림의 **a** 및 **b**) 로 입력하는 데이터는 컨트롤 터미널을 통해 블록다이어그램으로 들어갑니다. 그러면 이 데이터가 (더하기) 와 (빼기) 함수로 들어갑니다. (더하기) 와 (빼기) 함수가 내부 계산을 완료하면 새로운 데이터 값이 만들어집니다. 데이터 값은 인디케이터의 터미널로 흘러가서, 프론트패널의 인디케이터 (이전 그림에서 **a+b** 와 **a-b**) 를 업데이트합니다.

## 노드

노드는 입력 및 / 또는 출력을 가지며 VI 가 실행될 때 연산을 수행하는 블록 다이어그램의 객체입니다. 노드는 텍스트 기반 프로그래밍 언어에서 명령문, 연산자, 함수 및 서브루틴과 유사합니다. 이전 그림에서 (더하기) 와 (빼기) 함수는 노드의 예입니다.

노드에 대한 추가적인 정보는 제 5 장, *블록다이어그램 만들기의 블록다이어그램 노드* 섹션을 참조합니다.

## 와이어

와이어를 통해 블록다이어그램 객체 간에 데이터를 전달합니다. 이전 그림에서 와이어는 (더하기) 와 (빼기) 함수에 컨트롤과 인디케이터의 터미널을 연결합니다. 각 와이어는 단일 데이터 소스를 갖는 반면, 데이터를 읽는 많은 VI 와 함수에 연결할 수 있습니다. 와이어는 데이터 타입에 따라서 색상, 스타일, 두께가 다릅니다. 깨진 와이어는 중앙에 빨간 x 가 있는 검정색

점선 라인으로 나타냅니다. 깨진 와이어는 호환되지 않는 데이터 타입을 갖는 두 객체를 연결하려는 때와 같이 다양한 이유로 발생합니다.

배열에 대한 추가적인 정보는 5 장, *블록다이어그램 만들기의 와이어를 사용하여 블록다이어그램 객체에 연결하기* 섹션을 참조합니다.

## 구조

구조는 텍스트 기반 프로그래밍 언어의 루프 및 케이스 문을 그래픽하게 표현한 것입니다. 블록다이어그램에서 구조를 사용하여 코드의 블록을 반복하고 조건적 또는 특정한 순서로 코드를 실행합니다.

구조에 대한 추가적인 정보는 8 장, *루프와 구조를 참조하십시오.*

## 아이콘 및 커넥터 팬

VI의 프런트패널과 블록다이어그램을 작성한 후에 VI를 subVI로 사용할 수 있도록 커넥터 팬과 아이콘 생성을 합니다. 아이콘 및 커넥터 팬은 텍스트 기반 프로그래밍 언어의 함수 원형에 대응됩니다. 모든 VI의 프런트패널과 블록다이어그램 윈도우의 오른쪽 위 코너에는 다음과 같은 아이콘이 있습니다.



아이콘은 VI의 그래픽 표현입니다. 아이콘은 텍스트, 이미지, 또는 둘 다 포함할 수 있습니다. VI를 subVI로 사용하는 경우, 아이콘은 VI의 블록다이어그램에서 subVI를 식별합니다. 아이콘을 더블 클릭하면 아이콘 사용자 정의 및 편집이 가능합니다.

아이콘에 대한 추가적인 정보는 7 장, *VI와 SubVI 생성하기의 아이콘 생성하기* 섹션을 참조합니다.

다음과 같이, VI를 subVI로 사용하려면 커넥터 팬도 만들어 주어야 합니다.



커넥터 팬은 텍스트 기반 프로그래밍 언어의 함수 호출에 대한 매개변수 목록과 유사하게 VI에서 컨트롤 및 인디케이터에 대응하는 일련의 터미널입니다. 커넥터 팬은 VI에 연결할 수 있는 입력과 출력을 정의하여 VI를 subVI로 사용할 수 있도록 합니다. 커넥터 팬은 입력 터미널에서 데이터를 받고 프

런타임 컨트롤을 통하여 블록 다이어그램 코드에 데이터를 전달하며 프런트패널 인디케이터로부터 출력 터미널에 결과를 받습니다.

커넥터 팬에 설정에 대한 추가적인 정보는 7 장, *VI 와 SubVI 생성하기의 커넥터 팬 만들기* 섹션을 참조하십시오.



**노트**

하나의 VI 에 16 개 이상의 터미널을 할당하지 않는 것이 좋습니다. 터미널이 너무 많으면 VI 를 판독하기 어렵고 작업하기가 불편합니다.

## VI 와 SubVI 사용자 정의 및 사용하기

---

VI 를 만들고 아이콘과 커넥터 팬을 생성한 다음, VI 를 subVI 로 사용할 수 있습니다.

SubVI 에 대한 추가적인 정보는 7 장, *VI 와 SubVI 생성하기의 SubVI 생성하기* 섹션을 참조하십시오.

VI 의 모양과 동작을 사용자 정의할 수 있습니다.

VI 사용자 정의에 대한 추가적인 정보는 7 장, *VI 와 SubVI 생성하기의 VI 사용자 정의하기* 섹션을 참조하십시오.

## LabVIEW 환경

VI의 프런트패널과 블록다이어그램을 구현할 때 LabVIEW 팔레트, 도구, 메뉴를 사용하십시오. LabVIEW는 세 팔레트를 포함합니다: **컨트롤** 팔레트, **함수** 팔레트, **도구** 팔레트. 또한, LabVIEW는 **시작하기** 윈도우, **기본 도움말** 윈도우, **프로젝트 탐색기** 윈도우, **탐색** 윈도우를 포함합니다. **컨트롤**과 **함수** 팔레트를 사용자 정의할 수 있으며, 여러 작업 환경 옵션을 설정할 수 있습니다.

### 시작하기 윈도우

**시작하기** 윈도우는 LabVIEW를 실행할 때 나타납니다. 이 윈도우를 사용하여 새 VI를 생성하고 가장 최근에 열었던 LabVIEW 파일 중에서 선택하고, 예제를 찾고, *LabVIEW 도움말*을 실행합니다. 또한, 내셔널인스트루먼트 웹 사이트 ni.com에서 특정 매뉴얼, 도움말 항목, 리소스 등 LabVIEW를 배우는데 도움이 되는 정보와 리소스에 접근할 수 있습니다.

**시작하기** 윈도우는 기존 파일을 열거나 새 파일을 생성하면 사라집니다. **시작하기** 윈도우는 모든 열린 프런트패널과 블록다이어그램을 닫을 때 나타납니다. 또한, **보기**»**시작하기 윈도우**를 선택하여 윈도우를 디스플레이할 수 있습니다.

### 컨트롤 팔레트

이 **컨트롤** 팔레트는 프런트패널에서만 사용할 수 있습니다. **컨트롤** 팔레트는 프런트패널을 생성하기 위해 사용하는 컨트롤과 인디케이터를 포함합니다. 컨트롤과 인디케이터는 컨트롤과 인디케이터의 타입에 따라서 서브팔레트에 위치합니다.

4장, *프런트패널 만들기의 프런트패널 컨트롤과 인디케이터* 섹션을 참고하면 컨트롤과 인디케이터의 타입에 대한 더 많은 정보를 얻을 수 있을 것입니다.

**보기**»**컨트롤 팔레트**를 선택하거나 프런트패널 작업 공간에서 마우스 오른쪽 버튼을 클릭하여 **컨트롤** 팔레트를 디스플레이합니다. LabVIEW는 **컨트롤** 팔레트 위치와 크기를 간직하여, LabVIEW를 재시작할 때 같은 위치에서 같은 크기로 팔레트가 나타납니다. **컨트롤** 팔레트의 내용을 변경할 수 있습니다.

**컨트롤** 팔레트의 사용자 정의에 대한 더 많은 정보를 얻으려면 이 장의 *컨트롤과 함수 팔레트를 사용자 정의하기* 섹션을 참조하십시오.

## 함수 팔레트

**함수** 팔레트는 블록다이어그램에서만 사용할 수 있습니다. **함수** 팔레트에는 블록다이어그램을 생성하기 위한 VI와 함수들이 포함되어 있습니다. 이 VI와 함수는 VI와 함수의 타입에 따라 서브팔레트에 위치합니다.

**보기**» **함수 팔레트**를 선택하거나 블록다이어그램 작업 공간에서 마우스 오른쪽 버튼을 클릭하여 **함수** 팔레트를 디스플레이합니다. LabVIEW는 **함수** 팔레트 위치와 크기를 간직하여, LabVIEW를 재시작할 때 같은 위치에서 같은 크기로 팔레트가 나타납니다. **함수** 팔레트의 내용을 변경할 수 있습니다.


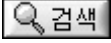
**함수** 팔레트의 사용자 정의에 대한 더 많은 정보를 얻으려면 이 장의 *컨트롤과 함수 팔레트를 사용자 정의하기* 섹션을 참조하십시오.



## 컨트롤과 함수 팔레트 탐색하기

팔레트의 객체를 클릭하여 객체를 커서 위에 놓아 해당 객체를 프론트패널 또는 블록다이어그램에 놓을 수 있도록 합니다. 또한, 팔레트의 VI 아이콘에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **VI 열기**를 선택하여 VI를 열 수 있습니다.

**컨트롤** 또는 **함수** 팔레트의 왼쪽 옆에 있는 검은 화살표를 클릭하여 서브팔레트를 펼치거나 숨깁니다. 이 화살표는 **항목 (표준)**과 **항목 (아이콘 및 텍스트)**로 팔레트 포맷을 설정한 경우에만 나타납니다.

다음과 같은 **컨트롤**과 **함수** 팔레트 도구 모음의 버튼을 사용하여 팔레트를 탐색하고, 팔레트를 설정하며, 컨트롤, VI, 함수를 검색할 수 있습니다.

	<p><b>위</b> — 팔레트 계층구조에서 한 레벨 위로 이동합니다. 이 버튼을 클릭하고 마우스 버튼을 계속 누르면 현재 서브팔레트에 이르기 위한 경로의 각 서브팔레트를 나열하는 바로 가기 메뉴가 디스플레이됩니다. 바로 가기 메뉴에서 서브팔레트 이름을 선택하여 서브팔레트를 탐색합니다. 이 버튼은 팔레트 포맷을 <b>아이콘, 아이콘과 텍스트</b>, 또는 <b>텍스트</b>로 설정한 경우에만 나타납니다.</p>
	<p><b>검색</b> — 팔레트를 검색 모드로 변경하여 텍스트 기반 검색을 수행하여 팔레트에서 컨트롤, VI, 함수를 찾을 수 있습니다. 팔레트가 검색 모드인 상태에서 <b>돌아가기</b> 버튼을 클릭하면 검색 모드를 끝내고 팔레트로 돌아갑니다.</p>

 보기	<p><b>보기</b> — 현재 팔레트에 대한 포맷을 선택하고, 모든 팔레트의 항목을 보이거나 숨기고, <b>텍스트</b>와 <b>트리</b> 포맷의 아이টে임을 알파벳 순서로 정렬하는 옵션을 제공합니다. 바로 가기 메뉴에서 <b>옵션</b>을 선택하여 <b>옵션</b> 대화 상자의 <b>컨트롤 / 함수 팔레트</b> 페이지를 디스플레이하며, 여기서 모든 팔레트에 대한 포맷을 선택할 수 있습니다. 이 버튼은 팔레트의 왼쪽 위 코너에 위치한 압정을 클릭하여 팔레트를 고정된 경우에만 나타냅니다.</p>
	<p><b>팔레트 크기 복구</b> — 팔레트 크기를 기본 크기로 다시 조정합니다. 이 버튼은 <b>컨트롤</b>이나 <b>함수</b> 팔레트의 크기를 조정했을 때만 나타냅니다.</p>

## 도구 팔레트

**도구** 팔레트는 프런트패널과 블록다이어그램에서 사용할 수 있습니다. 도구는 마우스 커서의 특별한 작업 모드입니다. 커서는 팔레트에서 선택한 도구의 아이콘에 상응합니다. 도구를 사용하여 프런트패널과 블록다이어그램 객체를 작동하고 변경할 수 있습니다.

자동 도구 선택이 활성화되어 있는 상태에서 커서를 프런트패널이나 블록다이어그램 위의 객체로 가져가면 LabVIEW가 **도구** 팔레트에서 적합한 도구를 자동으로 선택합니다.

**보기**»**도구 팔레트**를 선택하여 **도구** 팔레트를 디스플레이합니다. LabVIEW는 **도구** 팔레트의 위치를 그대로 유지해서 LabVIEW가 새로 시작되어도 그 팔레트는 같은 위치에 나타납니다.



**팁**

<Shift> 키를 누르고 마우스 오른쪽 버튼을 클릭하여 커서의 위치에서 **도구** 팔레트의 임시 버전을 디스플레이합니다.

## 메뉴와 도구 모음

메뉴와 도구 모음의 아이টে임으로 프런트패널과 블록다이어그램 객체를 작동시키고 수정합니다.

### 메뉴

VI 윈도우 상단의 메뉴에는 **열기**, **저장**, **복사**, **붙여넣기**와 같이 다른 어플리케이션에도 공통적인 아이টে임과 LabVIEW에만 사용되는 특수한 아이টে임이 위치합니다. 또한, 일부 메뉴 아이টে임은 키보드 바로 가기를 나열합니다.

(Mac OS) 화면 상단에 메뉴가 표시됩니다.

(Windows 와 UNIX) 기본적으로 가장 최근에 사용된 항목만 디스플레이합니다. 메뉴의 아래쪽 화살표를 클릭하면 모든 아이টে임을 나타낼 수 있습니다. **도구**»**옵션**을 선택한 후 **항목** 리스트에서 **환경**을 선택하고 **축소된 메뉴**

**사용** 확인란의 확인 표시를 제거하여 모든 메뉴 항목을 디스플레이할 수 있습니다.



**노트** VI 실행 중에는 일부 메뉴 항목을 사용할 수 없습니다.

## 바로 가기 메뉴

모든 LabVIEW 객체는 관련되어 있는 바로 가기 메뉴를 가지고 있습니다. VI를 생성할 때, 바로 가기 메뉴 항목을 사용하여 프런트패널과 블록 다이어그램 객체의 모양이나 동작을 변경할 수 있습니다. 바로 가기 메뉴에 접근하려면, 객체에서 마우스 오른쪽 버튼을 클릭합니다.

(Mac OS) <Command> 를 누르고 마우스 오른쪽 버튼 클릭과 같은 동작을 수행합니다.

## 실행 모드의 바로 가기 메뉴

VI가 실행 중이거나, 또는 실행 모드에 있을 때, 모든 프런트패널의 객체는 기본적으로 바로 가기 메뉴 항목의 축소된 세트를 갖습니다. 축소된 바로 가기 메뉴 항목으로 객체의 내용을 자르고, 복사하고, 붙여넣고, 객체의 기본값으로 설정하거나 객체의 설명을 읽습니다.

일부 복잡한 기능의 컨트롤은 추가 옵션을 갖습니다. 예를 들어, 노브 바로 가기 메뉴는 바늘에 추가하고 스케일 마커의 디스플레이를 변경하는 항목을 포함합니다.

## VI 도구 모음

VI 도구 모음의 버튼을 사용하여 VI를 실행하고, VI를 일시 정지하고, VI를 강제 종료하고, VI를 디버그하고, 폰트를 설정하고, 객체를 정렬하고, 그룹을 만들고, 배포합니다.

6, VI 실행하고 디버깅하기 장에서 일부 도구 모음 버튼에 대한 추가적인 정보를 참조하거나 LabVIEW 도움말에서 도구 모음 버튼의 완전한 리스트와 설명을 참조하십시오.

## 프로젝트 탐색기 윈도우 도구 모음

**표준, 프로젝트, 빌드, 소스 컨트롤** 도구 모음의 버튼을 사용하여 LabVIEW 프로젝트에서 작업을 수행할 수 있습니다. 이 도구 모음은 **프로젝트 탐색기** 윈도우의 제일 위에서 사용 가능합니다. 모든 도구 모음을 보려면 **프로젝트 탐색기** 윈도우를 확장해야 할 수도 있습니다.

LabVIEW 프로젝트에 대한 추가적인 정보는 본 장의 **프로젝트 탐색기** 윈도우 섹션을 참조하십시오.

## 기본 도움말 윈도우

**기본 도움말** 윈도우는 LabVIEW의 각 객체 위로 커서를 가져갈 때 LabVIEW 객체에 대한 기본 정보를 디스플레이합니다. 기본 도움말 정보를 가진 객체는 VI, 함수, 상수, 구조, 팔레트, 프로퍼티, 메소드, 이벤트, 대화 상자 구성요소, **프로젝트 탐색기** 윈도우의 아이템이 있습니다. 또한 **기본 도움말** 윈도우를 사용하여 어디에 정확히 VI 나 함수의 와이어를 연결할 것인지 결정할 수 있습니다.

**기본 도움말** 윈도우를 이용한 객체의 와이어링에 대한 더 많은 정보를 얻으려면, 5 장, *블록다이어그램 만들기의 와이어를 사용하여 블록다이어그램 객체에 연결하기* 섹션을 참조하십시오.

**도움말** > **기본 도움말 보이기**를 선택하여 **기본 도움말** 윈도우를 디스플레이합니다. 다음과 같이, 도구 모음에서 **기본 도움말 윈도우 보이기** 버튼을 클릭하여 **기본 도움말** 윈도우를 디스플레이할 수 있습니다.



(Windows) 또한, <Ctrl-H> 키를 눌러서 윈도우를 디스플레이할 수 있습니다. (Mac OS) <Command-H> 키를 누르십시오. (Linux) <Alt-H> 키를 누르십시오.

**기본 도움말** 윈도우는 각 객체 설명의 분량에 맞게 크기가 조절됩니다. **기본 도움말** 윈도우의 크기를 최대로 설정할 수 있습니다. LabVIEW를 **기본 도움말** 윈도우의 위치와 크기를 유지합니다. 그러므로 LabVIEW를 다시 시작하면 윈도우는 같은 위치에 나타나고 같은 최대 크기를 가집니다.

**기본 도움말** 윈도우가 설명하는 객체에 대응하는 *LabVIEW 도움말* 토픽이 존재하는 경우, **기본 도움말** 윈도우에 파란색 **상세 도움말** 링크가 나타납니다. 또한, 다음 그림처럼 **기본 도움말** 윈도우에 **상세 도움말** 버튼이 활성화됩니다. 링크나 버튼을 클릭하여 객체에 대한 더 많은 정보를 디스플레이할 수 있습니다.



## 프로젝트 탐색기 윈도우

**프로젝트 탐색기** 윈도우를 사용하여 LabVIEW 프로젝트를 생성하고 편집할 수 있습니다. 프로젝트를 사용하여 LabVIEW 파일과 LabVIEW 외의 파일을 하나로 그룹화하고, 빌드 스펙을 생성하고, 파일을 타겟에 배포하거나

운로드합니다. **파일** > **새 프로젝트**를 선택하여 **프로젝트 탐색기** 윈도우를 디스플레이합니다.

## 탐색 윈도우

**탐색** 윈도우는 편집 모드에서 활성화된 프런트패널 또는 활성화된 블록 다이어그램의 개요를 보여줍니다. **탐색** 윈도우를 사용하여 큰 프런트패널 또는 블록다이어그램을 탐색합니다. **탐색** 윈도우에서 이미지의 영역을 클릭하여 해당 영역을 프런트패널 또는 블록다이어그램에 디스플레이합니다. **탐색** 윈도우에서 이미지를 클릭하고 끌기하여 프런트패널 또는 블록다이어그램을 스크롤할 수 있습니다. 프런트패널이나 블록다이어그램에서 보이지 않는 영역은 **탐색** 윈도우에서 희미하게 나타납니다.

**보기** > **탐색 윈도우**를 선택하여 **탐색** 윈도우를 디스플레이합니다. **(Windows)** 또한, <Ctrl-Shift-N> 키를 눌러서 윈도우를 디스플레이할 수 있습니다. **(Mac OS)** <Command-Shift-N> 키를 누릅니다. **(Linux)** <Alt-Shift-N> 키를 누릅니다.



**노트** **탐색** 윈도우는 LabVIEW Full 과 Professional Development Systems 에서만 제공됩니다.

**탐색** 윈도우를 크기 조정하여 디스플레이하는 이미지의 크기를 조정합니다. LabVIEW 는 **탐색** 윈도우의 위치와 크기를 유지하여 LabVIEW 를 다시 시작할 때, 같은 위치에서 같은 크기로 그 윈도우가 나타납니다.

## 작업 환경 사용자 정의하기

**컨트롤**과 **함수** 팔레트를 사용자 정의할 수 있습니다. 그리고 **옵션** 대화 상자를 사용하여 팔레트 포맷을 선택하고 다른 작업 환경 옵션을 설정할 수 있습니다.

### 컨트롤과 함수 팔레트를 사용자 정의하기

**컨트롤**과 **함수** 팔레트를 다음과 같이 사용자 정의할 수 있습니다:

- **컨트롤과 함수 팔레트 세트 편집** 대화 상자를 사용하여 내장 팔레트를 다시 배치하고, 서브팔레트를 생성 및 이동하는 등 팔레트 세트를 편집합니다. **도구** > **고급** > **팔레트 세트 편집**을 선택하여 **컨트롤과 함수 팔레트 세트 편집** 대화 상자를 디스플레이합니다. 수정하려는 팔레트에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴의 옵션에서 선택합니다.
- **함수** 팔레트의 아이템을 즐겨찾기 항목에 추가합니다. 고정된 **함수** 팔레트에서, 객체에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에

서 **아이템을 즐겨찾기에 추가**를 선택합니다. **항목 (표준)** 과 **항목 (아이콘 및 텍스트)** 포맷에서는, 팔레트를 확장하여 서브팔레트를 디스플레이한 후, 서브팔레트의 제목에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **즐거찾기에 서브팔레트 추가**를 선택할 수 있습니다.

## 작업 환경 옵션 셋팅

**도구»옵션**을 선택하여 LabVIEW 를 사용자 정의합니다. **옵션** 대화 상자를 사용하여 프런트패널, 블록다이어그램, 경로, 성능과 디스크 관련 내용, 정렬 눈금, 팔레트, 실행 취소, 디버깅 도구, 색, 폰트, 인쇄, **히스토리** 윈도우, 다른 LabVIEW 특징사항을 위한 옵션을 설정합니다.

**옵션** 대화 상자 왼쪽에 있는 **항목** 리스트를 사용하여 여러 항목의 옵션 중에서 선택합니다.

## 프런트패널 만들기

프런트패널은 VI의 사용자 인터페이스입니다. 일반적으로, 프런트패널을 먼저 만든 후 프런트패널에 생성한 입력과 출력에 작업을 수행하기 위해서 블록다이어그램을 디자인합니다.

블록다이어그램에 대한 추가적인 정보는 제 5 장, *블록다이어그램 만들기*를 참조하십시오.

컨트롤과 인디케이터를 가지는 프런트패널을 구성합니다. 이는 각각 VI의 대화식 입력과 출력 터미널입니다. 컨트롤은 노브, 버튼, 다이얼, 그리고 기타 입력 메커니즘입니다. 인디케이터는 그래프, LED, 그리고 기타 출력 디스플레이입니다. 컨트롤은 인스트루먼트의 입력 메커니즘을 시뮬레이션하고 VI의 블록다이어그램에 데이터를 제공합니다. 인디케이터는 인스트루먼트의 출력 메커니즘을 시뮬레이션하고 블록다이어그램에서 수집하거나 생성하는 데이터를 디스플레이합니다.

**보기**»**컨트롤 팔레트**를 선택하여 **컨트롤** 팔레트를 디스플레이한 후, **컨트롤** 팔레트에서 컨트롤과 인디케이터를 선택하여 프런트패널에 놓습니다.

## 프런트패널 컨트롤과 인디케이터

**컨트롤** 팔레트에 위치한 프런트패널 컨트롤과 인디케이터를 사용하여 프런트패널을 만듭니다. 컨트롤과 인디케이터의 타입은 슬라이드 및 노브와 같은 숫자 컨트롤과 인디케이터, 그래프, 차트, 버튼 및 스위치와 같은 불리언 컨트롤과 인디케이터, 문자열, 경로, 배열, 클러스터, 리스트박스, 트리 컨트롤, 테이블, 링 컨트롤, 열거형 타입 컨트롤, 컨테이너 등이 포함됩니다.

### 컨트롤과 인디케이터의 스타일

프런트패널 컨트롤과 인디케이터는 일반, 클래식, 또는 시스템 스타일로 나타낼 수 있습니다.

### 일반 및 클래식 컨트롤과 인디케이터

많은 프런트패널 객체는 높은 색 품질의 모양을 가집니다. 모니터 디스플레이를 최소한 16 비트 색상 이상으로 설정해야 적합한 객체의 모양을 볼 수 있습니다.

또한, **일반** 팔레트에 위치한 컨트롤과 인디케이터는 대응하는 낮은 색 객체를 가집니다. 256 컬러나 16 컬러 모니터 셋팅에서는 **클래식** 팔레트에 위치한 컨트롤과 인디케이터를 사용합니다.

## 시스템 컨트롤과 인디케이터

생성한 대화 상자의 **시스템** 팔레트에 위치한 시스템 컨트롤과 인디케이터를 사용합니다. 시스템 컨트롤과 인디케이터는 대화 상자 박스에서 사용하기 위해서 특별히 디자인되었으며, 링과 스피ن 컨트롤, 숫자 슬라이드와 진행 막대, 스크롤 막대, 리스트박스, 테이블, 문자열과 경로 컨트롤, 탭 컨트롤, 트리 컨트롤, 버튼, 체크박스, 라디오 버튼, 그리고 자동으로 상위 배경색에 일치시키는 불투명 라벨 등을 포함합니다. 이 컨트롤은 프론트패널의 일반적인 컨트롤과 모양만 다릅니다. 이 컨트롤은 시스템에 설정한 색으로 나타납니다.

시스템 컨트롤은 VI를 실행하는 플랫폼에 따라서 모양이 바뀌기 때문에, VI에 생성한 컨트롤의 모양은 모든 LabVIEW 플랫폼에서 호환 가능합니다. 다른 플랫폼에서 VI를 실행할 때, 시스템 컨트롤은 해당 플랫폼의 표준 대화 상자 컨트롤에 맞춰 색과 모양을 적용합니다.

대화 상자 디자인에 대한 정보는 이 장의 *대화 상자 디자인하기* 섹션을 참조합니다.

## 숫자 디스플레이, 슬라이드, 스크롤 막대, 노브, 다이얼, 타임스탬프

**숫자형**과 **클래식 숫자** 팔레트에 위치한 숫자 객체를 사용하여 슬라이드, 스크롤 막대, 노브, 다이얼, 숫자 디스플레이를 생성합니다. 또한, 팔레트는 색 값을 설정하는 색 상자와 색 램프, 시간과 날짜 값을 설정하는 타임스탬프도 포함합니다. 숫자 객체를 사용하여 숫자 데이터를 입력하거나 디스플레이합니다.

## 숫자 컨트롤과 인디케이터

숫자 컨트롤과 인디케이터는 숫자 데이터를 입력하고 디스플레이하는 가장 간단한 방법입니다. 숫자의 더 많은 자릿수를 확보하기 위해서 이 프론트패널 객체를 수평 방향으로 크기 조절할 수 있습니다. 다음 방법 중 하나로 숫자형 컨트롤의 값을 변경합니다:

- 수행 도구나 라벨링 도구를 이용하여 디지털 디스플레이 창의 내부를 클릭하고 키보드로 숫자를 입력합니다.
- 수행 도구를 사용하여 숫자 컨트롤의 증가 또는 감소 화살표 버튼을 클릭합니다.
- 수행 도구 또는 라벨링 도구를 사용하여 변경하려는 숫자의 오른쪽에 커서를 놓고 키보드의 위 또는 아래쪽 화살표를 누릅니다.

기본적으로, LabVIEW 는 계산기와 같이 숫자를 디스플레이하고 저장합니다. 숫자 컨트롤 또는 인디케이터는 6 자리까지 숫자로 디스플레이한 후 자동으로 지수 표기로 바꿉니다. 숫자 객체에서 마우스 오른쪽 버튼을 클릭한 후 **숫자 프로퍼티** 대화 상자의 **포맷과 정밀도** 페이지를 디스플레이하기 위해서 바로 가기 메뉴에서 **포맷 & 정밀도**를 선택하여 LabVIEW 가 지수 표기법으로 전환하기 전에 디스플레이하는 자릿수를 설정할 수 있습니다.

## 슬라이드 컨트롤과 인디케이터

슬라이드 컨트롤과 인디케이터는 스케일을 가진 숫자 객체입니다. 슬라이드 컨트롤과 인디케이터는 수직과 수평 슬라이드, 탭, 온도계를 포함합니다. 다음 방법 중 하나로 슬라이드 컨트롤의 값을 변경합니다:

- 수행 도구를 사용하여 슬라이더를 새 위치에 클릭하거나 끌기를 합니다.
- 디지털 디스플레이를 사용하여 숫자 컨트롤과 인디케이터에서와 같이 데이터를 입력합니다.

슬라이더 컨트롤 또는 인디케이터는 한 개 이상의 값을 나타낼 수 있습니다. 객체에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **슬라이더 추가**를 선택하여 슬라이더를 추가합니다. 여러 슬라이더를 가진 컨트롤의 데이터 타입은 각 숫자값을 포함하는 클러스터입니다.

클러스터에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화*의 클러스터 섹션을 참조하십시오.

## 스크롤 막대 컨트롤과 인디케이터

스크롤 막대 컨트롤은 데이터를 스크롤하기 위해서 사용하는 숫자 객체이며 슬라이드 컨트롤과 비슷합니다. 스크롤 막대에는 수직과 수평 스크롤 막대가 있습니다. 수행 도구를 사용하여 사각 스크롤 박스를 새 위치로 클릭하거나 또는 끌거나, 증가와 감소 화살표를 클릭하거나, 또는 스크롤 박스와 화살표 사이의 스페이스를 클릭하여 스크롤 막대의 값을 변경합니다.

## 회전식 컨트롤과 인디케이터

회전식 컨트롤과 인디케이터에는 노브, 다이얼, 게이지, 미터가 있습니다. 회전식 객체는 스케일을 가진 숫자 객체이므로 슬라이드 컨트롤 및 인디케이터와 비슷하게 동작합니다. 다음 방법 중 하나로 회전식 컨트롤의 값을 변경합니다:

- 수행 도구를 사용하여 바늘을 새 위치에 클릭하거나 끌고 올 수 있습니다.
- 디지털 디스플레이를 사용하여 숫자 컨트롤과 인디케이터에서와 같이 데이터를 입력합니다.

회전식 컨트롤과 인디케이터는 하나 이상의 값을 디스플레이할 수 있습니다. 객체에서 마우스 오른쪽 버튼을 클릭한 후 **바늘 추가**를 선택하여 새 바늘을 추가합니다. 여러 바늘을 가지는 컨트롤의 데이터 타입은 각 숫자값을 포함하는 클러스터입니다.

클러스터에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 클러스터* 섹션을 참조하십시오.

## 타임스탬프 컨트롤과 인디케이터

타임스탬프 컨트롤과 인디케이터를 사용하여 블록 다이어그램에서 시간과 날짜 값을 가져오거나 보냅니다. 다음 방법 중 하나를 사용하여 타임스탬프 컨트롤의 값을 변경할 수 있습니다:

- 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **포맷 & 정밀도**를 선택합니다.
- 다음과 같이, **시간 / 날짜 탐색** 버튼을 클릭하여 **시간과 날짜 설정** 대화 상자를 디스플레이합니다.



- 컨트롤에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **데이터 처리 > 시간과 날짜 설정**을 선택하여 **시간과 날짜 설정** 대화 상자를 디스플레이합니다.
- 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **데이터 처리 > 현재 시간으로 설정**을 선택합니다.

## 그래프와 차트

**그래프**와 **클래식 그래프** 팔레트의 그래프 컨트롤과 인디케이터를 사용하여 숫자 데이터를 그래프 또는 차트 형태로 플롯합니다.

LabVIEW 에서 그래프와 차트 사용하기에 대한 추가적인 정보는 제 10 장, *그래프와 차트*를 참조하십시오.

## 버튼, 스위치, 빛

**볼리언**과 **클래식 볼리언** 팔레트에 위치한 볼리언 컨트롤과 인디케이터를 사용하여 버튼, 스위치, 빛을 생성합니다. 볼리언 컨트롤과 인디케이터를 사용하여 볼리언 (참 / 거짓) 값을 입력하고 디스플레이합니다. 예를 들어, 실험에서 온도를 모니터링하는 경우, 볼리언 경고 빛을 프론트패널에 놓고 온도가 특정 레벨을 초과할 때 알려주도록 할 수 있습니다.

볼리언 컨트롤은 물리적 인스트루먼트의 동작과 보다 비슷한 프론트패널을 생성하기 위해서 볼리언 객체를 사용자 지정하도록 해주는 6 개 타입의 기계

적 동작을 갖습니다. 바로 가기 메뉴를 사용하여 불리언 객체의 모양과 클릭할 때 어떻게 동작할 것인지 사용자 정의합니다.

## 라디오 버튼 컨트롤

라디오 버튼 컨트롤 사용은 한번에 하나만 선택할 수 있는 아이템 리스트를 사용자에게 제공합니다. 사용자에게 하나의 아이템을 선택하거나 아무것도 선택하지 않을 수 있는 옵션을 제공하려는 경우, 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **선택하지 않기 허용**을 선택하여 메뉴 아이템 옆에 확인 표시를 합니다.

라디오 버튼 컨트롤의 데이터 타입은 열거형 타입이기 때문에, 라디오 버튼 컨트롤을 사용하여 케이스 구조의 케이스를 선택할 수 있습니다.

열거형 타입 컨트롤에 대한 추가적인 정보는 이 장의 *열거형 타입 컨트롤* 섹션을 참조하십시오. 케이스 구조에 대한 추가적인 정보는 제 8 장, *루프와 구조의 케이스 구조* 섹션을 참조하십시오.

라디오 버튼 컨트롤 사용의 예제는 다음 VI를 참조하십시오:

## 텍스트 엔트리 박스, 라벨, 경로 디스플레이

**문자열 & 경로**와 **클래식 문자열 & 경로** 팔레트의 문자열과 경로 컨트롤 및 인디케이터를 사용하여 텍스트 엔트리 박스와 라벨을 생성하고 파일 또는 디렉토리의 위치를 입력하거나 반환합니다.

### 문자열 컨트롤과 인디케이터

수행 도구 또는 라벨링 도구를 사용하여 프론트패널의 문자열 컨트롤에 텍스트를 입력하거나 편집합니다. 기본적으로, 새로운 텍스트 또는 변경된 텍스트는 편집 세션을 끝내기 전에는 블록다이어그램에 전달되지 않습니다. 패널의 다른 부분을 클릭하거나, 다른 윈도우로 전환하거나, 도구 모음의 **입력** 버튼을 클릭하거나, 또는 숫자 키패드의 <Enter> 키를 눌러서 편집 세션을 끝냅니다. 키보드의 <Enter> 키를 누르면 캐리지 리턴이 입력됩니다.

문자열 컨트롤 또는 인디케이터에서 마우스 오른쪽 버튼을 클릭한 후, 암호 디스플레이 또는 16 진수 디스플레이와 같은 컨트롤 또는 인디케이터의 텍스트에 대한 디스플레이 타입을 선택합니다.

문자열 디스플레이 타입에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 프론트패널의 문자열* 섹션을 참조하십시오.

## 콤보 박스 컨트롤

콤보 박스 컨트롤을 사용하여 프론트패널에서 순환할 수 있는 문자열 리스트를 생성합니다. 콤보 박스 컨트롤은 텍스트 또는 메뉴 링 컨트롤과 유사합니다. 하지만, 콤보 박스 컨트롤의 값과 데이터 타입은 링 컨트롤과 같은 숫자가 아니라 문자열입니다.

링 컨트롤에 대한 추가적인 정보는 이 장의 *링 컨트롤* 섹션을 참조하십시오.

케이스 구조에 대한 추가적인 정보는 제 8 장, *루프와 구조의 케이스 구조* 섹션을 참조하십시오.

## 경로 컨트롤과 인디케이터

경로 컨트롤과 인디케이터를 사용하여 파일 또는 디렉토리의 위치를 입력하거나 반환합니다. (**Windows 와 Mac OS**) 또한, 실행 중 놓기가 활성화되어 있는 경우 Windows 탐색기에서 경로, 폴더, 또는 파일을 끌어와 경로 컨트롤에 놓을 수 있습니다.

경로 컨트롤과 인디케이터는 문자열 컨트롤과 인디케이터와 유사하게 동작하지만, LabVIEW는 사용자의 플랫폼에 대한 표준 구문을 사용하여 경로를 포맷합니다.

## 배열, 행렬, 클러스터 컨트롤과 인디케이터

**배열, 행렬 & 클러스터**와 **클래식 배열, 행렬 & 클러스터** 팔레트에 위치한 배열, 행렬, 클러스터 컨트롤 및 인디케이터를 사용하여 다른 컨트롤과 인디케이터의 배열, 행렬, 클러스터를 생성합니다. 배열은 같은 타입의 데이터 원소를 그룹화합니다. 클러스터는 혼합된 타입의 데이터 원소를 그룹화합니다. 행렬은 선형 대수 연산과 같은 일부 수학 연산에서 실수 또는 복소수 스칼라 데이터의 행이나 열을 그룹화합니다.

배열과 클러스터에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 배열과 클러스터를 이용하여 데이터 그룹화하기* 섹션을 참조하십시오.

## 리스트박스, 트리 컨트롤, 테이블

**리스트 & 테이블**과 **클래식 리스트 & 테이블** 팔레트에 위치한 리스트박스 컨트롤을 사용하여 사용자에게 선택할 수 있는 아이템의 리스트를 제공합니다.

## 리스트박스

리스트박스가 하나 또는 여러 개의 선택을 허용하도록 설정 할 수 있습니다. 여러 열 리스트박스를 사용하여 아이템의 크기와 생성된 날짜와 같이 각 아이템에 대한 추가적인 정보를 디스플레이합니다.

## 트리 컨트롤

트리 컨트롤을 사용하여 사용자가 선택할 수 있는 아이템의 계층 리스트를 제공합니다. 트리 컨트롤에 입력한 아이템을 아이템이나 노드의 그룹으로 구성합니다. 노드 옆의 확장 기호를 클릭하여 노드를 확장하고 노드의 모든 아이템을 디스플레이합니다. 또한 노드 옆의 기호를 클릭하여 노드를 다시 줄입니다.



### 노트

트리 컨트롤은 오직 LabVIEW Full 과 Professional Development Systems 에서만 생성하고 편집할 수 있습니다. VI 가 트리 컨트롤을 포함하는 경우, 이 VI 는 모든 LabVIEW 패키지에서 실행 가능하지만, Base Package 에서는 이 컨트롤의 설정을 바꿀 수 없습니다.

트리 컨트롤을 사용한 예제는 `labview\examples\general\controls\Tree Control Directory.llb` 안에 있는 `Directory Hierarchy in Tree Control VI` 를 참조합니다.

## 테이블

테이블 컨트롤을 사용하여 프론트패널에 테이블을 생성합니다.

테이블 컨트롤 사용에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클래스터를 이용한 데이터의 그룹화의 테이블* 섹션을 참조하십시오.

## 링과 열거형 타입 컨트롤 및 인디케이터

**링 & 열거형**과 **클래식 링 & 열거형** 팔레트에 위치한 링과 열거형 타입 컨트롤 및 인디케이터를 사용하여 돌아가면서 선택할 수 있는 문자열의 리스트를 생성합니다.

## 링 컨트롤

링 컨트롤은 숫자값을 문자열 또는 그림에 연관시키는 숫자 객체입니다. 링 컨트롤은 사용자가 돌아가면서 선택할 수 있는 풀다운 메뉴로 나타납니다.

링 컨트롤은 트리거 모드와 같이 상호 배타적인 아이템을 선택할 때 유용합니다. 예를 들어, 링 컨트롤을 사용하여 사용자가 연속, 단일, 외부 트리거에서 선택하도록 합니다.

## 열거형 타입 컨트롤

열거형 타입 컨트롤을 사용하여 사용자에게 선택할 수 있는 아이템 리스트를 제공합니다. 열거형 타입 컨트롤, 또는 열거형은 텍스트 또는 메뉴 링 컨트롤과 비슷합니다. 하지만, 열거형 타입 컨트롤의 데이터 타입은 컨트롤의 숫자값과 문자열 라벨에 대한 정보를 포함합니다. 링 컨트롤의 데이터 타입은 숫자형입니다.

## 컨테이너 컨트롤

**컨테이너**와 **클래식 컨테이너** 팔레트에 위치한 컨테이너 컨트롤을 사용하여 컨트롤과 인디케이터를 그룹화하거나 현재 VI의 프론트패널에 다른 VI의 프론트패널을 디스플레이합니다. **(Windows)** 또한 컨테이너 컨트롤을 사용하여 프론트패널에 .NET과 ActiveX 객체를 디스플레이할 수 있습니다.

.NET과 ActiveX 컨트롤에 대한 추가적인 정보는 이 장의 *.NET과 ActiveX 컨트롤 (Windows)* 섹션을 참조하십시오.

## 탭 컨트롤

탭 컨트롤을 사용하여 좁은 영역에서 프론트패널 컨트롤과 인디케이터를 겹칩니다. 탭 컨트롤은 페이지와 탭으로 구성됩니다. 탭 컨트롤의 각 페이지에 프론트패널 객체를 놓고 탭을 선택자로 사용하여 다른 페이지를 디스플레이합니다.

탭 컨트롤은 여러 프론트패널 객체를 함께 사용하거나 특정 수행 순서에서 사용할 때 유용합니다. 예를 들어, 테스트를 시작하기 전 사용자에게 여러 가지 셋팅을 먼저 설정하도록 요구한 후, 사용자가 진행 중에 테스트의 특성을 수정할 수 있도록 허용하며, 마지막으로 사용자가 적절한 데이터만을 디스플레이하고 저장할 수 있도록 하는 VI를 만듭니다.

블록다이어그램에서, 기본적으로 탭 컨트롤은 열거형 타입 컨트롤입니다. 탭 컨트롤에 놓인 컨트롤과 인디케이터의 터미널은 일반 블록다이어그램 터미널과 동일하게 나타납니다.

열거형 타입 컨트롤에 대한 추가적인 정보는 이 장의 *열거형 타입 컨트롤* 섹션을 참조하십시오.

## 서브패널 컨트롤

서브패널 컨트롤을 사용하여 다른 VI의 프론트패널을 현재 VI의 프론트패널에 디스플레이합니다. 예를 들어, 서브패널 컨트롤을 사용하여 사용자 인터페이스가 마법사처럼 동작하도록 디자인할 수 있습니다. **이전** 또는 **다음** 버튼을 최상위 레벨 VI의 프론트패널에 놓고 서브패널 컨트롤을 사용하여 마법사의 각 단계마다 다른 프론트패널을 로드할 수 있습니다.



**노트** 서브패널 컨트롤은 오직 LabVIEW Full 과 Professional Development Systems 에서만 생성하고 편집할 수 있습니다. VI 가 서브패널 컨트롤을 포함하는 경우, 모든 LabVIEW 패키지에서 VI 를 실행할 수는 있지만, Base Package 에서는 컨트롤을 설정할 수 없습니다.

서브패널 컨트롤 사용에 대한 예제는 `labview\examples\general\controls\subpanel.11b` 를 참조하십시오.

## I/O 이름 컨트롤과 인디케이터

I/O 와 클래식 I/O 팔레트의 I/O 이름 컨트롤과 인디케이터를 사용하여 설정한 DAQ 채널 이름, VISA 리소스 이름, IVI 논리 이름을 I/O VI 에 전달하여 인스트루먼트 또는 DAQ 디바이스와 통신을 합니다.

I/O 이름 상수는 **함수** 팔레트에 있습니다. 상수는 블록다이어그램에 고정 데이터 값을 제공하는 블록다이어그램의 터미널입니다.



**노트** 모든 I/O 이름 컨트롤이나 상수는 모든 플랫폼에서 사용 가능합니다. 이것은 임의의 플랫폼에서 플랫폼 특성의 디바이스와 통신할 수 있는 I/O VI 를 개발할 수 있도록 합니다. 그러나, 해당 디바이스를 지원하지 않는 플랫폼에서 플랫폼에 특정한 I/O 컨트롤과 함께 VI 를 실행하려는 경우, 에러가 발생합니다.

**(Windows) 도구** 메뉴에서 열 수 있는 Measurement & Automation Explorer 를 사용하여 DAQ 채널 이름, VISA 리소스 이름, 그리고 IVI 논리적 이름을 설정하십시오.

**(Mac OS 와 Linux)** 인스트루먼트에 대한 설정 유틸리티를 이용하여 VISA 리소스 이름과 IVI 로직 이름을 설정하십시오. 설정 유틸리티에 대한 추가적인 정보는 인스트루먼트의 문서를 참조하십시오.

## 웨이브폼 컨트롤

웨이브폼 컨트롤을 사용하여 웨이브폼의 개별 데이터 원소를 다룹니다. 웨이브폼 컨트롤은 웨이브폼의 데이터, 시작 시간, 델타  $t$  를 운반합니다.

웨이브폼 데이터 타입에 대한 추가적인 정보는 제 10 장, *그래프와 차트의 웨이브폼 데이터 타입* 섹션을 참조하십시오.

## 디지털 웨이브폼 컨트롤

디지털 웨이브폼 컨트롤을 사용하여 디지털 웨이브폼의 개별 원소를 다룹니다.

디지털 웨이브폼 데이터 타입에 대한 추가적인 정보는 10 장, *그래프와 차트*의 *디지털 웨이브폼 데이터 타입* 섹션을 참조하십시오.

## 디지털 데이터 컨트롤

디지털 데이터 컨트롤은 행과 열로 정렬된 디지털 데이터를 디스플레이합니다. 디지털 데이터 컨트롤을 사용하여 디지털 웨이브폼을 만들거나 디지털 웨이브폼에서 뽑아낸 디지털 데이터를 디스플레이합니다. 디지털 웨이브폼 데이터 컨트롤을 디지털 데이터 인디케이터에 연결하여 디지털 웨이브폼의 샘플과 신호를 봅니다.

## 객체 또는 어플리케이션의 참조

**참조 번호**와 **클래식 참조 번호** 팔레트에 있는 참조 번호 컨트롤을 사용하여 파일, 디렉토리, 디바이스, 네트워크 연결과 함께 작동합니다. 컨트롤 참조 번호를 사용하여 프론트패널 객체 정보를 subVI 에 전달합니다.

참조 번호, 즉 **refnum** 은 파일, 디바이스 또는 네트워크 연결과 같은 객체에 대한 유일한 식별자입니다. 파일, 디바이스, 또는 네트워크 연결을 열 때, LabVIEW 는 파일, 디바이스, 또는 네트워크 연결에 연계된 참조 번호를 생성합니다. 파일, 디바이스, 또는 네트워크 연결 열기에 수행하는 모든 작업은 각 객체를 식별하기 위해서 참조 번호를 사용합니다. 참조 번호 컨트롤을 사용하여 VI 내부 또는 외부로 참조 번호를 전달합니다. 예를 들어, 참조 번호 컨트롤 또는 인디케이터를 사용하여 참조 번호가 참조하는 파일을 닫거나 다시 열지 않고 파일의 내용을 수정합니다.

참조 번호는 열린 객체에 대한 임시 포인터이기 때문에, 참조 번호는 객체가 열려 있는 동안에만 유효합니다. 객체를 닫으면, LabVIEW 는 참조 번호와 객체의 연결을 끊어서 이 참조 번호는 사용할 수 없게 됩니다. 객체를 다시 열면, LabVIEW 는 처음 참조 번호와는 다른 새 참조 번호를 생성합니다. LabVIEW 는 참조 번호와 관련된 객체에 메모리를 할당합니다. 참조 번호를 닫아 객체를 메모리에서 해제합니다.

LabVIEW 는 객체에서 읽어 오거나 쓴 현재 위치와 사용자 접근 등급과 같은 각 참조 번호에 관련된 정보를 기억하고 있기 때문에, 한 객체에 대하여 서로 독립적인 작업을 동시에 수행할 수 있습니다. VI 가 객체를 여러 번 여는 경우, 매번 열 때마다 다른 참조 번호를 반환합니다. LabVIEW 는 VI 의 실행이 끝나면 자동으로 참조 번호를 닫습니다. 그러나 메모리와 기타 리소스를 가장 효율적으로 사용하려면 사용이 끝난 직후 참조 번호를 닫는 것이 좋은 프로그래밍 방법입니다. 참조 번호를 연 순서와 반대로 닫습니다. 예를 들어, 객체 A 의 참조 번호를 열고 객체 A 에서 메소드를 실행시켜 객체 B 의 참조 번호를 얻은 경우, 객체 B 의 참조 번호를 먼저 닫고 객체 A 의 참조 번호를 닫습니다.

## .NET 과 ActiveX 컨트롤 (Windows)

.NET & ActiveX 팔레트에 있는 .NET 과 ActiveX 컨트롤을 사용하여 일반적인 .NET 또는 ActiveX 컨트롤을 다룹니다. 추후 사용을 위해 .NET 또는 ActiveX 컨트롤을 이 팔레트에 추가할 수 있습니다. **도구» .NET & ActiveX » .NET 컨트롤을 팔레트에 추가** 또는 **도구» .NET & ActiveX » ActiveX 컨트롤을 팔레트에 추가**를 선택하여 .NET 이나 ActiveX 컨트롤 세트를 각각 사용자 컨트롤로 변환하여 **.NET & ActiveX 팔레트**에 추가합니다.



### 노트

.NET 객체를 생성하고 통신하기 위해서는 .NET framework 1.1 서비스 팩 1 또는 이후 버전이 필요합니다. 내쇼날인스트루먼트는 또한 LabVIEW 프로젝트 내에서만 .NET 객체를 사용할 것을 강력히 권장합니다.

## 프론트패널 객체 설정하기

**프로퍼티** 대화 상자 또는 바로 가기 메뉴를 사용하여 컨트롤과 인디케이터가 프론트패널에 어떻게 나타나는지 또는 어떻게 동작하는지를 설정합니다. 기본 도움말을 포함하며 객체에 여러 프로퍼티를 동시에 설정할 수 있는 대화 상자를 사용하여 프론트패널 컨트롤과 인디케이터를 설정하려고 할 때 **프로퍼티** 대화 상자를 사용합니다. 바로 가기 메뉴를 사용하여 일반 컨트롤과 인디케이터 프로퍼티를 빨리 설정합니다. **프로퍼티** 대화 상자와 바로 가기 메뉴에서 사용할 수 있는 옵션은 프론트패널 객체에 따라 다릅니다. 바로 가기 메뉴를 사용하여 설정한 옵션은 **프로퍼티** 대화 상자에 반영되며, **프로퍼티** 대화 상자를 사용하여 설정한 옵션은 바로 가기 메뉴에 반영됩니다.

프론트패널의 컨트롤 또는 인디케이터에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **프로퍼티**를 선택하여 객체에 대한 **프로퍼티** 대화 상자에 접근합니다. VI가 실행되는 동안 컨트롤 또는 인디케이터의 **프로퍼티** 대화 상자에 접근할 수 없습니다.

또한 사용자 컨트롤 또는 인디케이터를 생성하여 사용 가능한 프론트패널 객체 세트를 확장할 수 있습니다. 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **고급»사용자 정의**를 선택하여 컨트롤 또는 인디케이터를 사용자 정의합니다. 생성한 사용자 컨트롤 또는 인디케이터를 디렉토리 또는 LLB에 저장한 후 다른 프론트패널에서 사용자 컨트롤 또는 인디케이터를 사용할 수 있습니다.

## 옵션 원소 보이기와 숨기기

프론트패널 컨트롤과 인디케이터는 라벨, 캡션, 디지털 디스플레이 등의 보이거나 숨길 수 있는 옵션 원소를 가집니다. 프론트패널 객체에 대한 **프로퍼티** 대화 상자의 **모양** 페이지에서 컨트롤 또는 인디케이터의 보이는 원소를 설정합니다. 또한, 객체에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메

뉴에서 **보이는 아이템**을 선택한 후 사용 가능한 옵션에서 선택하여 보이는 원소를 설정할 수 있습니다.

## 컨트롤을 인디케이터로 또는 인디케이터를 컨트롤로 바꾸기

LabVIEW 는 객체의 초기 설정을 **컨트롤** 팔레트에 있는 컨트롤과 인디케이터의 일반적인 용도에 따라서 지정하고 있습니다. 예를 들어, 토글 스위치는 일반적으로 입력 메커니즘으로 사용되기 때문에 프런트패널에 토글 스위치를 놓으면 프런트패널에 컨트롤로 나타납니다. 프런트패널에 LED 를 놓으면, LED 는 일반적으로 출력 디바이스로 사용되기 때문에 인디케이터로 나타납니다.

일부 팔레트는 같은 타입이나 객체의 클래스에 대해서 컨트롤과 인디케이터를 포함합니다. 예를 들어, **숫자** 팔레트는 숫자 컨트롤과 숫자 인디케이터를 포함합니다. 숫자 입력 또는 숫자 출력을 가질 수 있기 때문입니다.

객체에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **인디케이터로 변경**을 선택하여 컨트롤을 인디케이터로 변경하고 객체에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **컨트롤로 변경**을 선택하여 인디케이터를 컨트롤로 변경할 수 있습니다.

## 프런트패널 객체 대체하기

프런트패널 객체를 다른 컨트롤 또는 인디케이터로 대체할 수 있습니다. 객체에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **대체**를 선택할 때, 임시 **컨트롤** 팔레트가 나타납니다. 임시 **컨트롤** 팔레트에서 컨트롤 또는 인디케이터를 선택하여 프런트패널의 현재 객체를 대체합니다.

## 프런트패널 설정하기

프런트패널 객체의 색을 변경하고, 프런트패널 객체를 정렬 및 배포하는 등 프런트패널을 사용자 정의할 수 있습니다.

### 객체 색칠하기

많은 객체의 색을 바꿀 수 있지만 모든 객체에서 가능한 것은 아닙니다. 대부분의 프런트패널 객체 및 프런트패널과 블록다이어그램 작업 공간의 색을 변경할 수 있습니다. 시스템 컨트롤과 인디케이터의 색은 변경할 수 없습니다. 이들 객체는 사용자 시스템에서 설정한 색으로 나타나기 때문입니다.

색칠 도구를 사용하여 객체 또는 작업공간에서 마우스 오른쪽 버튼을 클릭한 후 프런트패널 객체 또는 프런트패널과 블록다이어그램 작업 공간의 색을 바꿉니다. 또한 **도구>옵션**을 선택한 후 **항목** 리스트에서 **색**을 선택하여 몇몇 객체에 대한 기본색을 바꿀 수 있습니다.

중요한 정보를 읽을 때 색은 사용자에게 혼란을 줄 수 있기 때문에 사용하는 경우 논리적이고 일관되도록 주의하여 사용해야 합니다.

## 객체 정렬 및 간격 조절하기

**편집**»**패널 눈금 정렬 활성화**를 선택하여 프론트패널의 눈금 정렬을 활성화하고 객체를 놓을 때 객체를 정렬합니다. 또는, **편집**»**패널 눈금 정렬 비활성화**를 선택하여 눈금 정렬을 비활성화하고 보이는 눈금을 사용하여 수동으로 객체를 정렬합니다. <Ctrl-#> 키를 눌러서 눈금 정렬을 활성화하거나 비활성화할 수 있습니다. 프랑수어 키보드에서는, <Ctrl-“> 키를 누릅니다.

**(Mac OS)** <Command-\*> 키를 누르십시오. **(Linux)** <Alt-#> 키를 누르십시오.

블록다이어그램에서도 정렬 눈금을 사용할 수 있습니다.

**도구**»**움선**을 선택한 후 **항목** 리스트에서 **정렬 눈금**을 선택하여 눈금을 숨기거나 사용자 정의합니다.

객체를 놓은 후에 객체를 정렬하려면, 객체를 선택한 후 도구 모음의 **객체 정렬** 풀다운 메뉴를 선택하거나 **편집**»**아이템 정렬**을 선택합니다. 객체의 간격을 균일하게 하려면, 객체를 선택한 후 도구 모음의 **객체 간격 조절** 풀다운 메뉴를 선택하거나 **편집**»**아이템 간격 조절**을 선택합니다.

## 객체 그룹화 및 잠금 설정

위치 도구를 사용하여 그룹화 및 잠금 설정을 하려는 프론트패널 객체를 선택합니다. 도구 모음의 **순서 재설정** 버튼을 클릭한 후 풀다운 메뉴에서 **그룹** 또는 **잠금**을 선택합니다. 그룹화된 객체는 위치 도구를 이용하여 위치를 옮기거나 크기를 바꿀 때, 상호 간의 상대적인 배열과 크기를 유지합니다. 잠긴 객체는 프론트패널에서 그 위치를 유지하며 잠금 해제를 하기 전에는 삭제할 수 없습니다. 객체들을 선택하여 그룹화와 잠금 설정을 동시에 할 수 있습니다. 위치 도구 이외의 다른 도구들도 그룹화되고 잠금 설정된 객체에서 사용할 수 있습니다.

## 객체 크기 조정하기

대부분의 프론트패널 객체는 크기를 바꿀 수 있습니다. 위치 도구를 크기 조정이 가능한 객체 위로 움직일 때, 크기 조정 핸들 또는 원이 객체를 크기 조정할 수 있는 포인트에서 나타납니다. 객체를 크기 조정할 때, 폰트 크기는 변하지 않습니다. 객체의 그룹을 크기 조정하면 그룹 내의 모든 객체들의 크기가 조정됩니다.

디지털 숫자 컨트롤과 인디케이터 같은 일부 객체는 크기 조정할 때 수평 또는 수직으로만 크기 조정됩니다. 노브와 같은 다른 객체는 크기 조정할 때 같

은 비율을 유지합니다. 위치 커서는 똑같이 나타나지만, 객체를 둘러싸고 있는 경계 점선은 한 방향으로만 움직입니다.

객체를 크기 조정할 때 수동으로 증가 방향을 제한할 수 있습니다. 확대를 수직 또는 수평으로 제한하거나 객체의 현재 비례를 유지하려면, <Shift> 키를 누르면서 크기 조정 핸들이나 원을 끕니다. 중심점 주위로 객체를 크기 조정하려면, <Ctrl> 키를 누르면서 크기 조정 핸들이나 원을 끕니다.

**(Mac OS)** <Option> 키를 누릅니다. **(Linux)** <Alt> 키를 누릅니다.

여러 객체를 같은 크기로 조정하려면, 객체를 선택한 후 도구 모음의 **객체 크기 조정** 풀다운 메뉴를 선택합니다. 모든 선택된 객체를 가장 크거나 또는 가장 작은 객체의 폭 또는 높이로 크기 조정할 수 있으며, 모든 선택된 객체를 픽셀 단위의 특정 크기로 크기 조정할 수 있습니다.

## 윈도우 크기를 조정하지 않고 프론트패널에 공간 추가하기

윈도우 크기를 조정하지 않고 프론트패널에 공간을 추가할 수 있습니다. 뿔뿔하거나 조밀하게 그룹화된 객체 사이에 공간을 늘리려면, <Ctrl> 키를 누른 상태에서 위치 도구를 사용하여 프론트패널 작업 공간을 클릭합니다. 키를 누르고 있는 동안, 삽입하려는 크기만큼의 영역을 끌어서 키웁니다.

**(Mac OS)** <Option> 키를 누릅니다. **(Linux)** <Alt> 키를 누릅니다.

점선으로 표시된 사각형 경계는 삽입될 공간을 정의합니다. 마우스 버튼과 <Ctrl> 키를 놓아서 공간을 추가합니다.

## 라벨 붙이기

라벨을 사용하여 프론트패널과 블록다이어그램의 객체를 식별합니다.

LabVIEW에는 고유 라벨과 독립 라벨의 두 가지 라벨이 있습니다. 고유 라벨은 특정 객체에 속하여 함께 이동하며 해당 객체만을 설명합니다. 고유 라벨을 독립적으로 이동시킬 수 있지만, 라벨을 가진 객체를 움직일 때 라벨은 객체와 함께 움직입니다. 고유 라벨을 숨길 수는 있지만, 라벨을 가진 객체에 독립적으로 복사하거나 삭제할 수는 없습니다. 숫자 컨트롤 또는 인디케이터에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **보이는 아이템** > **단위 라벨**을 선택하여 숫자 컨트롤과 인디케이터에 단위 라벨이라고 불리는 별개의 고유 라벨을 디스플레이할 수 있습니다.

독립 라벨은 어떤 객체에도 속하지 않으며, 독립적으로 생성, 이동, 회전, 또는 삭제할 수 있습니다. 독립 라벨을 사용하여 프론트패널과 블록다이어그램을 설명합니다.

독립 라벨은 블록 다이어그램의 코드를 설명하거나 프론트패널에 사용자 설명을 나열할 때 유용합니다. 빈 공간을 더블 클릭하거나 라벨링 도구를 사용하여 독립 라벨을 생성하거나 라벨을 편집합니다.

## 텍스트 특성

LabVIEW는 컴퓨터에 이미 설치된 폰트를 사용합니다. 도구 모음의 **텍스트 셋팅** 폴다운 메뉴를 사용하여 텍스트의 속성을 변경합니다.

**텍스트 셋팅** 폴다운 메뉴에는 다음과 같은 내장 폰트가 있습니다:

- **어플리케이션 폰트 — 컨트롤과 함수** 팔레트 및 새 컨트롤의 텍스트에 사용되는 기본 폰트
- **시스템 폰트** — 메뉴에 사용되는 폰트
- **대화 상자 폰트** — 상자의 텍스트에 사용되는 폰트

**텍스트 셋팅** 폴다운 메뉴에서 선택을 하기전에 객체 또는 텍스트를 선택한 경우, 선택한 모든 것들에 변경이 적용됩니다. 선택한 것이 없을 경우, 변경은 기본 폰트에 적용됩니다. 기본 폰트를 변경해도 기존 라벨의 폰트는 변경되지 않습니다. 오직 이후부터 생성하는 라벨에만 영향을 줍니다.

내장 폰트를 포함하는 VI를 다른 플랫폼으로 옮기면, 폰트는 가장 가까운 폰트로 바뀝니다.

**텍스트 셋팅** 폴다운 메뉴는 또한 **크기**, **스타일**, **자리맞춤**, **색** 서브메뉴 아이템을 가집니다.

## 사용자 인터페이스 디자인하기

VI가 사용자 인터페이스 또는 대화 상자로 사용될 경우, 프론트패널 모양과 배치는 중요합니다. 사용자가 쉽게 어떤 작업을 수행할지 식별할 수 있도록 프론트패널을 디자인합니다. 인스트루먼트 또는 다른 디바이스와 비슷하게 보이는 프론트패널을 디자인할 수 있습니다.

### 프론트패널 컨트롤과 인디케이터 사용하기

컨트롤과 인디케이터는 프론트패널의 주요 구성요소입니다. 프론트패널을 디자인할 때, 어떻게 사용자가 VI와 상호 작용할지를 고려하고 컨트롤과 인디케이터를 논리적으로 그룹화합니다. 일부 컨트롤이 연관된 경우, 장식 경계를 추가하여 컨트롤을 둘러싸거나 클러스터로 묶습니다. **장식** 팔레트에 있는 장식을 사용하여 상자, 라인, 화살표로 프론트패널의 객체를 그룹화하거나 분리합니다. 이 객체는 단지 장식을 위한 것이며 데이터를 디스플레이하지 않습니다.

## 대화 상자 디자인하기

**파일** > **VI 프로퍼티**를 선택하고 **항목** 폴다운 메뉴에서 **윈도우 모양**을 선택하여 메뉴 모음과 스크롤 막대를 숨기고 각 플랫폼에서 표준 대화 상자과 같이 보이고 동작하도록 VI를 생성합니다.

VI가 같은 스크린 위치에 연속적으로 나타나는 대화 상자를 포함하는 경우, 첫번째 대화 상자의 버튼이 두번째 대화 상자의 버튼과 같은 곳에 있지 않도록 구성합니다. 사용자가 첫번째 대화 상자의 버튼을 더블 클릭할 때 무심코 다음 대화 상자의 버튼을 클릭할 수 있기 때문입니다.

생성하는 대화 상자에 있는 **시스템** 팔레트의 시스템 컨트롤을 사용합니다.

# 블록다이어그램 만들기

프런트패널을 만든 후, 그래픽 형태의 함수를 사용하여 프런트패널의 객체를 제어하는 코드를 추가합니다. 블록다이어그램은 G 코드 또는 블록다이어그램 코드로 알려진 그래픽 소스 코드를 포함합니다.

## 블록다이어그램 객체

블록다이어그램의 객체에는 터미널과 노드가 있습니다. 이 객체들을 와이어로 연결하여 블록다이어그램을 만듭니다. 각 터미널의 색과 기호는 대응하는 컨트롤과 인디케이터의 데이터 타입을 나타냅니다. 상수는 블록다이어그램에 고정 데이터 값을 제공하는 블록다이어그램의 터미널입니다.

## 블록다이어그램 터미널

프런트패널 객체는 블록다이어그램에서 터미널로 나타납니다. 블록다이어그램의 터미널에서 더블 클릭을 하면, 대응하는 프런트패널의 컨트롤 또는 인디케이터가 하이라이트됩니다.

터미널은 프런트패널과 블록다이어그램 사이에 정보를 교환하는 통로입니다. 프런트패널 컨트롤에 입력한 데이터 값은 컨트롤 터미널을 통해서 블록다이어그램에 입력됩니다. 실행시, 출력 데이터 값은 인디케이터 터미널로 이동하여 블록다이어그램을 빠져 나오고, 프런트패널로 다시 들어가서 프런트패널 인디케이터에 나타납니다.

LabVIEW 는 구조에 컨트롤과 인디케이터 터미널, 노드 터미널, 상수, 특정한 터미널을 가지고 있습니다. 터미널에 와이어를 연결하고 다른 터미널에 데이터를 전달합니다. 블록다이어그램 객체에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **보이는 아이콘>>터미널**을 선택하여 터미널을 나타내십시오. 객체에서 마우스 오른쪽 버튼을 클릭한 후 다시 **보이는 아이콘>>터미널**을 선택하여 터미널을 숨깁니다. 바로 가기 메뉴 아이템은 확장 가능한 VI 와 함수에서는 사용할 수 없습니다.

블록다이어그램에서 프런트패널 컨트롤 또는 인디케이터가 아이콘으로 나타나거나 데이터 타입 터미널로 나타나도록 설정할 수 있습니다. 기본적으

로, 프런트패널 객체는 아이콘 터미널로 나타납니다. 예를 들어, 다음에 보이는 노브 아이콘 터미널은 프런트패널의 노브 컨트롤을 나타냅니다.



터미널 아래 부분의 DBL 은 배정도 부동소수의 데이터 타입을 나타냅니다. 다음에 보이는 DBL 터미널은 배정도 부동소수 컨트롤을 나타냅니다.



터미널에서 마우스 오른쪽 버튼을 클릭한 후 **아이콘으로 보기** 바로 가기 메뉴 아이템 옆에 있는 확인 표시를 제거하여 터미널의 데이터 타입을 디스플레이합니다. 아이콘 터미널을 사용하여 프런트패널 객체의 데이터 타입과 함께 블록다이어그램에 프런트패널 객체의 타입을 디스플레이합니다. 데이터 타입 터미널을 사용하여 블록다이어그램의 공간을 절약합니다.



**노트**

아이콘 터미널은 데이터 타입 터미널보다 크기 때문에, 데이터 타입 터미널을 아이콘 터미널로 변환할 때 의도하지 않게 다른 블록다이어그램 객체를 가릴 수 있습니다.

컨트롤 터미널은 인디케이터 터미널보다 경계가 더 두껍습니다. 또한, 프런트패널 터미널에 나타나는 화살표가 터미널이 컨트롤인지 인디케이터인지 나타냅니다. 오른쪽에 나가는 화살표가 있는 것이 컨트롤이며, 왼쪽에 들어오는 화살표가 있는 것이 인디케이터입니다.

## 컨트롤과 인디케이터 데이터 타입

일반적인 컨트롤 및 인디케이터 데이터 타입은 부동소수, 정수, 타임스탬프, 열거형, 불리언, 문자열, 배열, 클러스터, 경로, 다이내믹 웨이브폼, 참조 번호, I/O 이름 등이 있습니다. 컨트롤 및 인디케이터의 데이터 타입과 기호, 사용법의 완전한 리스트는 *LabVIEW 도움말*을 참조하십시오.

각 터미널의 색과 기호는 대응하는 컨트롤과 인디케이터의 데이터 타입을 나타냅니다. 많은 데이터 타입은 데이터를 처리할 수 있는 대응하는 함수의 세트를 가집니다. 예를 들어 **문자열** 팔레트의 ( 문자열 ) 함수는 문자열 데이터 타입에 대응됩니다.

## 기호형 숫자값

정의되지 않았거나 예상치 못한 데이터는 모든 이후의 수행을 무효로 만듭니다. 부동소수 연산은 잘못된 계산 또는 의미없는 결과를 나타내는 다음 두 기호 값을 반환합니다 :

- **NaN (숫자 아님)**은 음수의 제곱근 취하기와 같은 유효하지 않은 연산이 생성하는 부동소수 값을 나타냅니다.
- **Inf (무한대)**는 데이터 타입 범위 밖의 부동소수 값을 나타냅니다. 예를 들어, 제로로 1 을 나누면 **Inf** 가 나옵니다.

LabVIEW 는 +Inf 또는 -Inf 를 반환할 수 있습니다. +Inf 는 데이터 타입의 최대값을 나타내며 -Inf 는 데이터 타입의 최소값을 나타냅니다.

LabVIEW 는 정수값의 오버플로우 또는 언더플로우 조건을 확인하지 않습니다.

## 상수

상수는 블록다이어그램에 고정 데이터 값을 제공하는 블록다이어그램의 터미널입니다. 일반적인 상수는 원주율 ( $\pi$ ) 과 무한대 ( $\infty$ ) 와 같이 고정된 값을 가지는 상수입니다. 사용자 정의 상수는 VI 를 실행하기 전에 정의하고 편집하는 상수입니다.

대부분의 상수는 팔레트의 맨 위 또는 맨 아래에 위치합니다.

VI 또는 함수의 입력 터미널에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **상수»생성**을 선택하여 사용자 정의 상수를 생성합니다.

수행 도구 또는 라벨링 도구를 사용하여 상수를 클릭하고 그 값을 편집합니다. 자동 도구 선택이 활성화된 경우, 상수를 더블 클릭하여 라벨링 도구로 바꾼 후 값을 편집합니다.

## 블록다이어그램 노드

노드는 입력 그리고 / 또는 출력을 가지며 VI 가 실행될 때 연산을 수행하는 블록다이어그램의 객체입니다. 노드는 텍스트 기반 프로그래밍 언어의 명령문, 연산자, 함수, 서브루틴과 유사합니다. LabVIEW 는 다음과 같은 타입의 노드를 가집니다 :

- **함수** — 연산자, 함수 또는 명령문과 비교할만한 내장된 실행 원소.
- **SubVI** — 서브루틴과 비슷한 다른 VI 의 블록다이어그램에서 사용되는 VI.

블록다이어그램에서의 subVI 사용에 대한 추가적인 정보는 제 7 장, *VI와 SubVI 생성하기*의 *SubVI 생성하기* 섹션을 참조하십시오.

- **익스프레스 VI** — 일반적인 측정 작업을 수행하도록 디자인된 subVI입니다. 설정 대화 상자를 이용하여 익스프레스 VI를 설정할 수 있습니다.

익스프레스 VI 사용에 대한 추가적인 정보는 이 장의 *익스프레스 VI* 섹션을 참조하십시오.
- **구조** — For 루프, While 루프, 케이스 구조, 플랫 시퀀스 구조와 다층 시퀀스 구조, Timed 구조, 이벤트 구조와 같이 실행을 컨트롤하는 원소.

구조 사용에 대한 추가적인 정보는 제 8 장, *루프와 구조*를 참조하십시오.

블록다이어그램 노드의 완전한 리스트는 *LabVIEW 도움말*을 참조하십시오.

## 다형성 VI와 함수

다형성 VI와 함수는 다른 데이터 타입의 입력 데이터에 따라 조정됩니다. 대부분의 LabVIEW 구조는 일부 VI 및 함수와 같이 다형성입니다.

함수의 다형성 정도는 다양합니다. 다형성이 아니거나, 일부 또는 전체 입력이 다형성일 수 있습니다. 일부 함수의 입력은 숫자값과 불리언 값을 받습니다. 일부는 숫자값 또는 문자열을 받습니다. 또한 몇몇은 스칼라 숫자값 뿐만 아니라 숫자값의 배열, 숫자값의 클러스터, 숫자값의 클러스터 배열 등을 받습니다. 어떤 함수 입력은 오직 1 차원 배열만을 받을 수 있으며, 이 때 배열 원소는 모든 타입이 될 수 있습니다. 일부 함수는 복소수 값을 포함한 모든 타입의 데이터를 받습니다.

배열과 클러스터에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 배열과 클러스터를 이용하여 데이터 그룹화하기* 섹션을 참조하십시오.

## 함수 개요

함수는 LabVIEW의 본질적인 실행 원소입니다. **함수** 팔레트의 함수 아이콘은 연한 노란색 배경과 검은색 전경을 가집니다. 함수는 프런트패널 또는 블록다이어그램을 가지지는 않지만 커넥터 팬을 가집니다. 함수를 열거나 편집할 수 없습니다.

## 함수에 터미널 추가하기

일부 함수의 터미널 개수를 바꿀 수 있습니다. 예를 들어, 10 개의 원소를 가지는 배열을 만들려면, 반드시 10 개의 터미널을 ( 배열 만들기 ) 함수에 추가해야만 합니다.

위치 도구를 사용하여 함수의 상위 또는 하위 경계를 각각 위 아래로 끌어서 함수에 터미널을 추가할 수 있습니다. 또한 위치 도구를 사용하여 함수에서 터미널을 제거할 수 있지만, 이미 연결된 터미널은 제거할 수 없습니다. 터미널을 제거하기 위해서는 반드시 기존의 와이어를 먼저 삭제해야만 합니다.

객체 연결에 대한 추가적인 정보는 이 장의 *와이어를 사용하여 블록다이어그램 객체에 연결하기* 섹션을 참조하십시오.

## 내장된 VI 와 함수

**함수** 팔레트에는 또한 LabVIEW 와 함께 제공되는 VI 들이 있습니다. 사용자 어플리케이션에 subVI 로 이러한 VI 와 함수를 사용하여 개발시간을 단축할 수 있습니다. **함수** 팔레트의 **보기** 버튼을 클릭하고 바로 가기 메뉴에서 **항상 보이는 항목»모든 항목 보이기**를 선택하여 **함수** 팔레트에 모든 항목을 디스플레이합니다.

내장된 VI 와 함수 사용에 대한 추가적인 정보는 제 7 장, *VI 와 SubVI 생성하기의 내장된 VI 와 함수 사용하기* 섹션을 참조하십시오.

모든 내장된 VI 와 함수에 대한 상세한 정보는 *LabVIEW 도움말*을 참조하십시오.

## 익스프레스 VI

일반 측정 작업에 익스프레스 VI 를 사용합니다. 대화 상자에서 익스프레스 VI 를 설정하므로, 익스프레스 VI 는 최소한의 와이어링을 요구하는 노드입니다. 익스프레스 VI 의 입력과 출력은 VI 를 어떻게 설정하는지에 따라 결정됩니다. 익스프레스 VI 는 파란색 영역으로 둘러싸인 아이콘을 가지는 확장 가능 노드로 블록다이어그램에 나타납니다.

익스프레스 VI 사용에 관한 추가적인 정보는 *LabVIEW 시작하기* 매뉴얼을 참조하십시오.

# 와이어를 사용하여 블록다이어그램 객체에 연결하기

와이어를 통해 블록다이어그램 객체 간에 데이터를 전달합니다. 각 와이어는 단일 데이터 소스를 갖지만, 데이터를 읽을 수 있는 여러 VI 와 함수에 연결할 수 있습니다. 이는 텍스트 기반 프로그래밍 언어에서 필수 파라미터를 전달하는 것과 비슷합니다. 반드시 요구되는 블록다이어그램 터미널을 모두 연결해야만 합니다. 그렇지 않은 경우, VI 는 깨지고 실행되지 않을 것입니다. **기본 도움말** 윈도우를 디스플레이하여 블록다이어그램 노드가 요구하는 터미널을 확인합니다. 필수 터미널의 라벨은 **기본 도움말** 윈도우에 굵은체로 나타납니다.

깨진 VI 에 대한 추가적인 정보는 제 6 장 *VI 실행하고 디버깅하기의 깨진 VI 수정하기* 섹션을 참조하십시오.

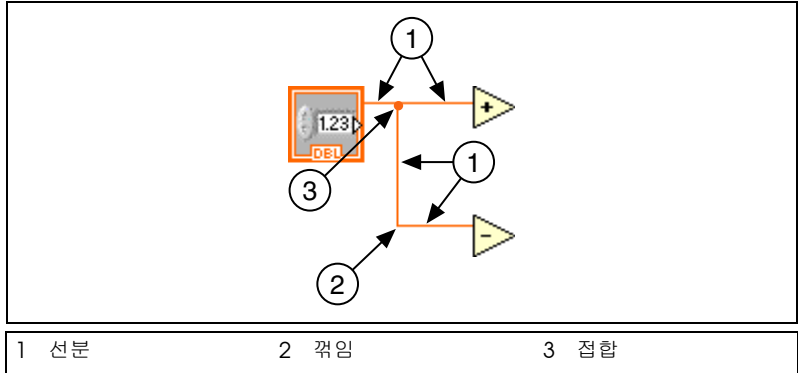
## 와이어 모양과 구조

와이어는 데이터 타입에 따라 색, 스타일, 굵기가 다릅니다. 이는 터미널의 색과 기호가 대응하는 컨트롤 또는 인디케이터의 타입을 나타내는 것과 비슷합니다. 깨진 와이어는 중앙에 빨간 x 가 있는 검정색 점선 라인으로 나타납니다. 깨진 와이어는 호환되지 않는 데이터 타입을 갖는 두 객체를 연결하려는 때와 같이 다양한 이유로 발생합니다. 깨진 와이어의 빨간색 x 옆의 화살표는 데이터 흐름 방향을 나타내며, 화살표의 색은 와이어를 따라 흐르는 데이터의 데이터 타입을 나타냅니다.

데이터 타입에 대한 추가적인 정보는 이 장의 *컨트롤과 인디케이터 데이터 타입* 섹션을 참조하십시오. 데이터 흐름에 대한 추가적인 정보는 이 장의 *블록다이어그램의 데이터 흐름* 섹션을 참조하십시오.

와이어 표시자는 와이어링 도구를 VI 또는 함수 위로 움직일 때 연결되지 않은 터미널 옆에 나타나는 와이어의 절단부입니다. 와이어 표시자는 각 터미널의 데이터 타입을 나타냅니다. 터미널의 이름을 나열하는 팁 상자도 나타냅니다. 터미널을 연결한 후에는 와이어링 도구를 노드 위로 움직여도 터미널의 와이어 표시자가 나타나지 않습니다.

와이어 선분은 와이어의 단일 수평 또는 수직 조각입니다. 와이어의 꺾임은 두 선분이 만나는 위치입니다. 둘 또는 그 이상의 와이어 선분이 만나는 지점은 접합입니다. 와이어 끝가지는 접합하는 곳에서 접합하는 곳, 터미널에서 접합하는 곳, 또는 접합하는 곳이 없을 때 터미널에서 터미널까지의 모든 와이어 선분을 포함합니다. 다음 그림은 와이어 선분, 꺾임, 접합을 나타냅니다.



## 객체 연결하기

와이어링 도구를 사용하여 블록다이어그램 노드의 터미널을 다른 블록다이어그램 노드의 터미널에 수동으로 연결합니다. 도구의 커서 포인트는 감지 않은 실타래의 끝 부분입니다. 와이어링 도구를 터미널 위로 가져가면, 터미널이 깜박입니다. 와이어링 도구를 VI 또는 함수 터미널 위로 가져갈 때, 터미널의 이름을 나열하는 팁 상자도 나타납니다. 터미널에 와이어를 연결하면 깨진 와이어가 생성될 수 있습니다. VI를 실행하기 전에 반드시 깨진 와이어를 수정해야만 합니다.

깨진 와이어를 수정하는데 관한 추가적인 정보는 이 장의 *깨진 와이어 수정하기* 섹션을 참조하십시오.

**기본 도움말** 윈도우를 사용하여 정확히 어디에 와이어를 연결할지를 결정합니다. 커서를 VI 또는 함수 위로 움직일 때, **기본 도움말** 윈도우는 VI 또는 함수의 각 터미널을 보여줍니다. **기본 도움말** 윈도우는 ( 배열 만들기 ) 함수와 같이 확장 가능한 VI와 함수의 터미널은 디스플레이하지 않습니다. **기본 도움말** 윈도우의 **옵션 터미널과 전체 경로 보기** 버튼을 클릭하여 커넥터 팬의 옵션 터미널을 디스플레이합니다.

와이어가 겹치면, 첫번째 와이어에 작은 틈이 생겨서 첫번째 와이어가 두번째 와이어 밑에 있는 것을 나타냅니다.

## 와이어 꺾기

터미널을 연결하는 동안, 커서를 수평 또는 수직 방향으로 움직여서 와이어를 90도로 꺾습니다. 여러 방향으로 와이어를 꺾으려면, 마우스 버튼을 클릭하여 와이어를 고정한 다음 커서를 새로운 방향으로 움직입니다. 반복적으로 와이어를 고정한 후 새 방향으로 움직일 수 있습니다.

## 와이어 연결 취소하기

와이어를 고정된 마지막 지점에서 실행 취소를 하려면, <Shift> 키를 누른 채로 블록다이어그램의 아무 곳이나 클릭합니다. 전체 와이어링 작업을 강제 종료하려면, 블록다이어그램의 아무 곳에서 마우스 오른쪽 버튼을 클릭합니다.

(Mac OS) <Option> 키를 누르고 클릭합니다. (Linux) 마우스 가운데 버튼을 클릭합니다.

## 자동으로 객체 와이어하기

블록다이어그램에서 선택된 객체를 다른 객체로 가까이 가져가면, LabVIEW 는 유효한 연결을 보여주는 임시 와이어를 그립니다. 객체를 블록다이어그램에 놓기 위해서 마우스 버튼을 놓으면, LabVIEW 는 자동으로 와이어를 연결합니다. 또한 이미 블록다이어그램에 있는 객체를 연결할 수도 있습니다. LabVIEW 는 가장 잘 일치하는 터미널을 연결하고 일치하지 않는 터미널은 연결하지 않습니다.

위치 도구를 사용하여 객체를 움직이는 동안 스페이스바를 눌러서 자동 와이어링을 토글합니다.

## 와이어 선택하기

위치 도구를 사용하여 한번 클릭, 더블 클릭, 또는 트리플 클릭으로 와이어를 선택합니다. 와이어를 한번 클릭하면 와이어의 한 선분이 선택됩니다. 와이어를 더블 클릭하면 와이어 끝가지가 선택됩니다. 와이어를 트리플 클릭하면 전체 와이어가 선택됩니다.

## 깨진 와이어 수정하기

깨진 와이어는 가운데 빨간 x 가 있는 검정색 점선으로 표시됩니다. 깨진 와이어는 호환되지 않는 데이터 타입을 갖는 두 객체를 연결하려는 때와 같이 다양한 이유로 발생합니다. 와이어링 도구를 깨진 와이어로 위로 이동하여 왜 와이어가 깨졌는지를 설명하는 팁 상자를 디스플레이합니다. 이 정보는 또한 와이어링 도구를 깨진 와이어 위로 이동할 때 **기본 도움말** 윈도우에도 나타납니다. 와이어에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **에러 열거**를 선택하여 **에러 리스트** 윈도우를 디스플레이합니다. 와이어가 깨진 이유에 대한 추가적인 정보를 보려면 **도움말** 버튼을 클릭합니다.

위치 도구로 와이어를 트리플 클릭한 후 <Delete> 키를 눌러서 깨진 와이어를 제거합니다. 또한, 와이어에서 마우스 오른쪽 버튼을 클릭한 후 **와이어 끝가지 삭제**, **와이어 끝가지 생성**, **연결되지 않은 끝 제거**, **와이어 정리**, **컨트롤로 변경**, **인디케이터로 변경**, **소스에서 인덱싱 활성화**, **소스에서 인덱싱 비활성화**와 같은 바로 가기 메뉴 옵션을 선택할 수 있습니다. 이 옵션은 깨진 와이어의 이유에 따라서 변합니다.

**편집>>깨진 와이어 제거**를 선택하거나 <Ctrl-B> 키를 눌러서 모든 깨진 와이어를 제거할 수 있습니다. **(Mac OS)** <Command-B> 키를 누르십시오. **(Linux)** <Meta-B> 키를 누르십시오.

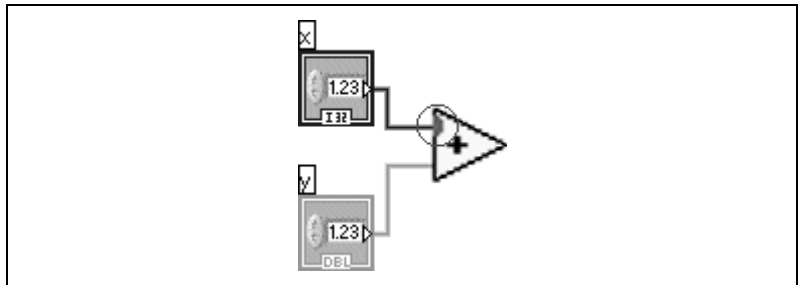


### 주의

모든 깨진 와이어를 제거할 때에는 주의하십시오. 때때로 블록다이어그램의 연결을 끝내지 않았기 때문에 깨진 와이어가 나타나기도 합니다.

## 강제 변환점

블록다이어그램 노드의 강제 변환점은 두개의 서로 다른 숫자 데이터 타입을 연결했다는 것을 알려주기 위해서 나타납니다. 점은 LabVIEW 가 노드에 전달된 값을 다른 형으로 변환했다는 것을 의미합니다. 예를 들어, ( 더하기 ) 함수는 두 개의 배정도 부동소수 입력을 요구합니다. 두 입력 중 하나를 정수로 바꿀 경우, 다음 그림과 같이 강제 변환점이 ( 더하기 ) 함수에 나타납니다.



강제 변환점은 VI 의 메모리 사용을 늘리고 실행 시간을 증가시킬 수 있습니다. 생성한 VI 의 데이터 타입이 일관되도록 노력하십시오.

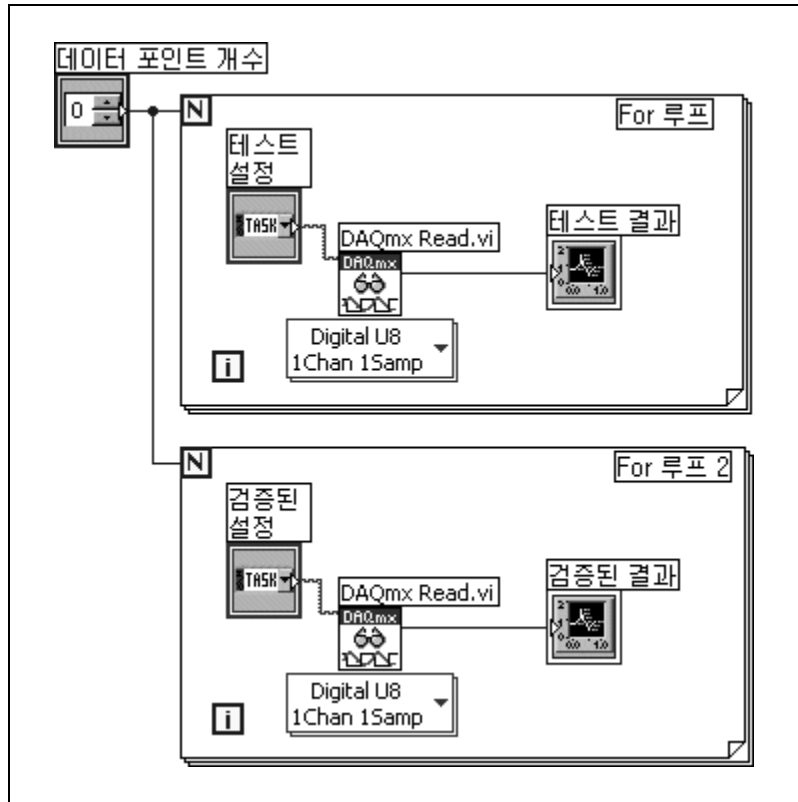
## 블록다이어그램의 데이터 흐름

LabVIEW 는 VI 실행시 데이터 흐름 모델을 따릅니다. 블록다이어그램 노드는 모든 요구되는 입력을 받을 때 실행됩니다. 노드가 실행될 때, 노드는 출력 데이터를 만들고 데이터 흐름 경로에서 다음 노드로 데이터를 전달합니다. 노드를 통한 데이터의 이동은 블록다이어그램에서 VI 와 함수의 실행 순서를 결정합니다.

Visual Basic, C++, JAVA 및 대부분의 다른 텍스트 기반 프로그래밍 언어는 프로그램 실행에 흐름 제어 모델을 따릅니다. 흐름 제어에서, 프로그램 원소의 순서는 프로그램의 실행 순서를 결정합니다.

LabVIEW 에서, 명령의 순서가 아닌 데이터의 흐름이 블록다이어그램 원소의 실행 순서를 결정합니다. 그러므로, 동시 작업을 할 수 있는 블록다이어그램을 생성할 수 있습니다. 예를 들어, 다음 블록다이어그램에서와 같이 두

개의 For 루프를 동시에 실행하고 프런트패널에 결과를 디스플레이할 수 있습니다.



## 데이터 의존성과 인위적인 데이터 의존성

흐름 제어 모델의 실행은 명령에 따라서 수행됩니다. 데이터 흐름은 데이터에 의해 실행되거나 데이터 의존적으로 실행됩니다. 다른 노드에서 데이터를 받는 노드는 항상 다른 노드가 실행을 완전히 끝낸 이후에 실행됩니다.

와이어가 연결되지 않은 블록다이어그램 노드는 임의의 순서로 실행될 수 있습니다. 데이터 의존성이 없을 때 흐름 파라미터를 사용하여 실행 순서를 컨트롤할 수 있습니다. 흐름 파라미터를 사용할 수 없을 때 시퀀스 구조를 사용하여 실행 순서를 컨트롤할 수 있습니다.

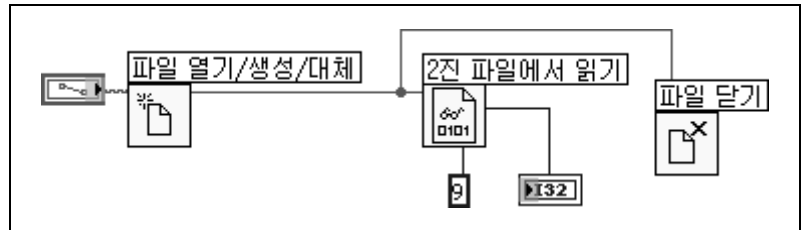
흐름 파라미터에 대한 추가적인 정보는 이 장의 *흐름 파라미터* 섹션을 참조하십시오. 시퀀스 구조에 대한 추가적인 정보는 제 8 장, *루프와 구조의 시퀀스 구조* 섹션을 참조하십시오.

또한, 인위적인 데이터 의존성을 만들 수 있으며, 이 때 (데이터를) 받는 노드는 실제로 받은 데이터를 사용하지 않습니다. 대신에 데이터를 받는 노드는 데이터의 도착을 실행의 트리거로 사용합니다. 인위적인 데이터 의존성을 사용하는 것에 대한 예제는 labview\examples\general\structs.11b 의 Timing Template (data dep) VI 를 참조하십시오.

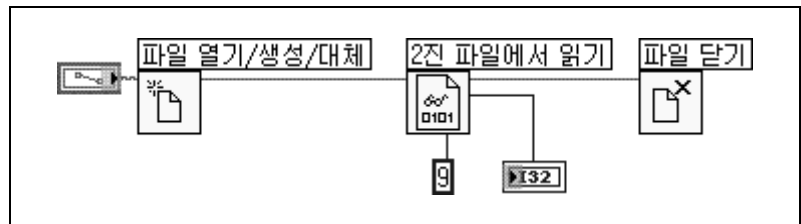
## 데이터 의존성 잃기

데이터 의존성이 없을 때 실행이 왼쪽에서 오른쪽, 위에서 아래쪽으로 진행된다고 가정하지 마십시오. 필요한 경우 데이터 흐름을 연결해서 이벤트의 순서를 명백히 정의합니다.

다음 블록다이어그램에서, (2진 파일에서 읽기) 함수가 (파일 닫기) 함수에 연결되지 않았기 때문에 (2진 파일에서 읽기) 함수와 (파일 닫기) 함수 사이에 의존성이 없습니다. 어떤 함수가 먼저 실행될지 결정할 방법이 없기 때문에 이 예제는 의도한대로 실행되지 않을 것입니다. (파일 닫기) 함수가 먼저 실행되면, (2진 파일에서 읽기) 함수는 작동할 수 없습니다.



다음 블록다이어그램은 (2진 파일에서 읽기) 함수의 출력을 (파일 닫기) 함수에 연결하여 의존성을 만듭니다. (파일 닫기) 함수는 (2진 파일에서 읽기) 함수의 출력을 받기 전까지는 실행되지 않습니다.



## 흐름 파라미터

일반적으로 참조 번호 또는 에러 클러스터인 흐름 파라미터는 대응하는 입력 파라미터와 같은 값을 반환합니다. 이 파라미터를 사용하여 데이터 의존성이 없을 때 실행 순서를 컨트롤할 수 있습니다. 실행하려는 첫번째 노드의 흐름 출력을 실행하려는 다음으로 실행하려는 노드의 대응하는 입력에 연결하여 인위적인 데이터 의존성을 생성합니다. 이러한 흐름 파라미터 없이 원

하는 순서대로 데이터 연산을 수행하려면 반드시 시퀀스 구조를 사용해야 합니다.

시퀀스 구조에 대한 추가적인 정보는 6 장, *VI 실행하고 디버깅하기의 에러 핸들링하기* 섹션을 참조하십시오. 흐름 파라미터에 대한 추가적인 정보는 8 장, *루프와 구조의 시퀀스 구조* 섹션을 참조하십시오.

## 데이터 흐름과 메모리 관리

데이터 흐름 실행은 실행 흐름 제어 모델보다 메모리 관리가 쉽습니다. LabVIEW 에서는 변수에 메모리를 할당하거나 변수에 값을 지정하지 않습니다. 대신에, 데이터 전달을 나타내는 와이어를 가진 블록다이어그램을 생성합니다.

데이터를 생성하는 VI 와 함수는 자동적으로 해당 데이터에 대한 메모리를 할당합니다. VI 또는 함수가 더 이상 데이터를 사용하지 않을 때, LabVIEW 는 할당된 메모리를 해제합니다. 새 데이터를 배열 또는 문자열에 추가할 때, LabVIEW 는 새 데이터를 관리할 수 있는 충분한 메모리를 할당합니다.

## 블록다이어그램 디자인하기

다음 지침을 사용하여 블록다이어그램을 디자인합니다:

- 왼쪽에서 오른쪽, 위에서 아래로의 배치를 사용합니다. 블록다이어그램 원소의 위치가 실행 순서를 결정하지는 않지만, 구조화된 블록다이어그램과 쉬운 이해를 위해서 오른쪽에서 왼쪽으로 연결하는 것을 피합니다. 오직 와이어와 구조가 실행 순서를 결정합니다.
- 하나 또는 두 스크린을 넘어서는 블록다이어그램을 생성하는 것을 피합니다. 블록다이어그램이 커지고 복잡해지면, 블록다이어그램을 이해하고 디버깅하기가 힘들어질 수 있습니다.
- 일부 블록다이어그램 구성요소를 다른 VI 에서 재사용할 수 있는지, 또는 블록다이어그램의 한 섹션이 같은 논리 구조로 되어있는지를 확인합니다. 그런 경우, 블록다이어그램을 특정 작업을 수행하는 subVI 로 나눕니다. SubVI 를 사용하면 블록다이어그램을 수정하고 디버깅하기가 쉽습니다.

SubVI 에 대한 추가적인 정보는 7 장 *VI 와 SubVI 생성하기의 SubVI 생성하기* 섹션을 참조하십시오.

- 에러 핸들링 VI, 함수, 파라미터를 사용하여 블록다이어그램의 에러를 관리합니다.

에러 처리에 대한 추가적인 정보는 제 6 장, *VI 실행하고 디버깅하기의 에러 핸들링하기* 섹션을 참조하십시오.

- LabVIEW 가 결과 와이어의 일부 선분을 숨길 수 있으므로, 구조 경계 아래 또는 겹쳐진 객체들 사이로 와이어를 연결하는 것을 피합니다.

- 와이어 위에 객체를 놓지 않습니다. 터미널 또는 아이콘을 와이어 위에 놓으면 연결이 없는 경우에도 연결이 존재하는 것처럼 보입니다.
- 독립 라벨을 사용하여 블록다이어그램의 코드를 설명합니다.  
독립 라벨 사용에 대한 추가적인 정보는 4 장, *프런트패널 만들기의 라벨 붙이기* 섹션을 참조하십시오.

# VI 실행하고 디버깅하기

VI를 실행하려면 모든 subVI, 함수, 구조를 터미널의 올바른 데이터 타입에 맞게 연결해야 합니다. 때로는 VI가 예상치 않은 방식으로 데이터를 생성하거나 실행될 수 있습니다. LabVIEW를 사용하여 블록다이어그램의 구성이나 블록다이어그램을 통과하는 데이터에서 문제점을 찾습니다.

## VI 실행하기

VI를 실행하면 사용자가 VI를 디자인한대로 작업을 실행합니다. 다음과 같이 도구 모음의 **실행** 버튼이 붉은 흰색 화살표로 나타나면 VI를 실행할 수 있습니다.



또한 붉은 흰색 화살표는 해당 VI에 커넥터 팬을 생성하는 경우 이 VI를 subVI로 사용할 수 있음을 나타냅니다.

커넥터 팬 생성에 대한 추가적인 정보는 7장, *VI와 SubVI 생성하기의 커넥터 팬 만들기* 섹션을 참조하십시오.

블록다이어그램 도구 모음의 **실행** 또는 **연속 실행** 버튼을 클릭하거나 단계별 실행 버튼을 클릭하면 VI가 실행됩니다. VI가 실행되는 동안 **실행** 버튼은 다음과 같이 회색 화살표로 바뀌어 VI가 실행되고 있음을 나타냅니다.



VI가 실행 중일 때는 VI를 변경할 수 없습니다.

**실행** 버튼을 클릭하면 VI를 한 번 실행합니다. VI가 데이터 흐름을 완료하면 VI는 정지합니다. 다음과 같이 **연속 실행** 버튼을 클릭하면 수동으로 정지시킬 때까지 VI를 연속적으로 실행합니다.



단계별 실행 버튼을 클릭하면 단계별로 VI를 실행합니다.

단계별 실행 버튼을 사용하여 VI 디버깅하기에 대한 추가적인 정보는 이 장의 *단계별 실행* 섹션을 참조하십시오.

## 깨진 VI 수정하기

VI 가 실행되지 않으면 깨졌거나 실행할 수 없는 VI 입니다. 생성하거나 편집하는 VI 에 에러가 있으면 다음과 같이 **실행** 버튼이 깨져서 나타납니다.



블록 다이어그램의 연결을 마쳤는데도 버튼이 여전히 깨져있는 경우, VI 는 깨진 것이며 실행할 수 없습니다.

## VI 가 깨진 원인 찾기

경고는 VI 의 실행을 막지는 않습니다. VI 의 잠재적인 문제점을 피하는데 도움을 주도록 디자인되었습니다. 그러나 에러로 인해서는 VI 가 깨질 수 있습니다. VI 를 실행하기 전에 반드시 모든 에러를 해결해야 합니다.

깨진 **실행** 버튼을 클릭하거나 **보기** > **에러 리스트**를 선택하여 VI 가 깨진 이유를 찾습니다. **에러 리스트** 윈도우는 모든 에러를 나열합니다. **에러가 있는 아이템** 섹션은 VI 및 프로젝트 라이브러리 등 에러가 있는 메모리 상의 모든 아이템 이름을 나열합니다. 둘 이상의 아이템이 같은 이름을 가지고 있는 경우, 이 섹션은 각 아이템에 대한 특정한 어플리케이션 인스턴스를 보여줍니다. **에러와 경고** 섹션은 **에러가 있는 아이템** 섹션에 선택한 VI 의 에러와 경고를 나열합니다. **세부사항** 섹션은 에러를 설명하며, 일부 경우에는 에러를 수정하는 방법을 권장합니다. **도움말** 버튼을 클릭하여 에러를 상세히 설명하고 에러를 수정하기 위한 단계별 해설을 포함하는 *LabVIEW 도움말*의 토픽을 디스플레이합니다.

**에러 보이기** 버튼을 클릭하거나 에러 설명을 더블 클릭하여 에러를 포함하는 블록 다이어그램 또는 프런트패널 영역을 하이라이트합니다.

VI 에 경고가 포함되어 있고 **에러 리스트** 윈도우의 **경고 보이기** 확인란에 확인 표시를 한 경우 다음과 같이 도구 모음에 **경고** 버튼이 나타납니다.



## 깨진 VI 의 일반적인 원인

아래의 리스트는 편집할 때 VI 가 깨지는 일반적인 원인입니다 .

- 블록다이어그램에 데이터 타입이 맞지 않거나 , 끊어지고 연결되지 않아서 깨진 와이어가 있습니다 .

깨진 와이어 수정하기에 대한 정보는 5 장 *블록다이어그램 만들기의 깨진 와이어 수정하기* 섹션을 참조하십시오 .

- 필수적인 블록다이어그램 터미널이 연결되어 있지 않습니다 .

필수 입력 및 출력 설정에 대한 추가적인 정보는 5 장 , *블록다이어그램 만들기의 와이어를 사용하여 블록다이어그램 객체에 연결하기* 섹션을 참조하십시오 .

- SubVI 가 깨졌거나 subVI 의 아이콘을 VI 의 블록다이어그램에 놓은 후 커브터 팬을 편집한 경우입니다 .

SubVI 에 대한 정보는 7 장 *VI 와 SubVI 생성하기의 SubVI 생성하기* 섹션을 참조하십시오 .

## 디버깅 기술

VI 가 깨지지 않는 않으나 예상치 않은 데이터를 얻었을 경우 다음 몇 가지 기술을 사용하여 VI 또는 블록다이어그램 데이터 흐름의 문제점을 확인하고 수정할 수 있습니다 .

### 실행 하이라이트하기

다음과 같이 **실행 하이라이트** 버튼을 클릭하여 블록다이어그램 실행의 애니메이션을 봅니다 .



실행 하이라이트는 블록다이어그램의 한 노드에서 다른 노드까지의 데이터 이동을 와이어를 따라 움직이는 방울을 사용하여 보여줍니다 . 실행 하이라이트를 단계별 실행과 함께 사용하여 데이터 값이 VI 를 통하여 어떻게 노드와 노드 사이를 움직이는지 확인합니다 .



**노트** 실행 하이라이트는 VI 의 실행 속도를 현저히 저하시킵니다 .

**에러 출력** 클러스터에 에러가 발생하는 경우 , 이 에러 값은 **에러 출력**의 옆에 빨간 경계로 나타납니다 . 만약 에러가 없으면 , **확인**이 녹색 경계를 가진 **에러 출력** 옆에 나타납니다 .

에러 클러스터에 대한 추가적인 정보는 이 장의 *에러 클러스터* 섹션을 참조하십시오 .

## 단계별 실행

VI를 단계별로 실행하여 VI가 실행될 때 블록다이어그램에서 VI의 각 단계별 작업을 봅니다. 다음과 같은 단계별 실행 버튼은 단계별 실행 모드에서의 VI 또는 subVI 실행에만 영향을 줍니다.



블록다이어그램의 도구 모음에서 **단계별 실행 건너뛰기** 또는 **단계별 실행 들어가기** 버튼을 클릭하여 단계별 실행 모드로 들어갑니다. **단계별 실행 건너뛰기**, **단계별 실행 들어가기**, **단계별 실행 나가기** 버튼 위로 커서를 이동하여 해당 버튼을 클릭한 경우 다음 단계를 설명하는 팁 상자를 봅니다. SubVI를 단계별로 실행하거나 정상적으로 실행할 수 있습니다.

실행 하이라이트를 켜고 VI를 단계별로 실행하는 경우, 다음과 같은 실행 문양이 현재 실행 중인 subVI의 아이콘에 나타납니다.



## 프로브 도구

일반 프로브를 사용하여 와이어를 통과하는 데이터를 봅니다. 와이어에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **사용자 프로브»일반 프로브**를 선택하여 일반 프로브를 사용합니다.

## 브레이크포인트

다음과 같이 브레이크포인트 도구를 사용하여 블록다이어그램의 VI, 노드, 또는 와이어에 브레이크포인트를 놓아 해당 위치에서 실행을 일시 중지시킵니다.



와이어에 브레이크포인트를 설정할 때, 실행은 데이터가 와이어를 통과한 후에 일시 정지합니다. 블록다이어그램에 브레이크포인트를 놓아서 블록다이어그램의 모든 노드가 실행된 후에 실행을 일시 정지합니다.

브레이크포인트에서 VI가 일시 중지되면, LabVIEW는 블록다이어그램을 전면에서 띄우고 선택 표시를 사용하여 브레이크포인트가 있는 노드나 와이어를 하이라이트합니다. 커서를 기존의 브레이크포인트 위로 이동할 때, 브레이크포인트 도구 커서의 검은 영역은 하얀색으로 나타납니다.

실행 도중 브레이크포인트에 도달하면, VI 는 멈추고 **일시 정지** 버튼은 빨간색으로 나타납니다. 다음 동작을 취할 수 있습니다:

- 단계별 실행 버튼을 사용하여 단계별로 실행합니다.
- 와이어에 프로브를 놓아 중간값을 확인합니다.
- 프런트패널 컨트롤의 값을 변경합니다.
- **일시 정지** 버튼을 클릭하여 다음 브레이크포인트나 VI 가 실행을 마칠 때까지 계속 실행합니다.

LabVIEW 는 VI 와 함께 브레이크포인트를 저장하지만 브레이크포인트는 VI 를 실행할 때만 활성화됩니다. **수행>브레이크포인트**를 선택한 후 **찾기** 버튼을 클릭하면 모든 브레이크포인트를 볼 수 있습니다.

## 에러 핸들링하기

아무리 만든 VI 의 완성도를 확신한다 하더라도 사용자가 접하게 될 모든 문제를 예상할 수는 없습니다. 에러 확인 메커니즘이 없는 경우, VI 가 올바르게 작동하지 않는다는 사실만 알 수 있습니다. 에러 확인을 하면 왜 그리고 어디에서 에러가 발생했는지 알 수 있습니다.

모든 종류의 입력과 출력 (I/O) 을 수행할 때 에러가 발생할 가능성을 고려하십시오. 거의 모든 I/O 함수는 에러 정보를 반환합니다. VI, 특히 I/O 실행 (파일, 시리얼, 인스트루먼트, 데이터 수집, 통신) 에 에러 확인하기를 포함시키고 적절하게 에러를 처리하는 메커니즘을 제공합니다.

기본으로, LabVIEW 는 VI 의 실행을 일시 정지하고, 에러가 발생한 subVI 또는 함수를 하이라이트하고, 에러 대화 상자를 디스플레이하여 VI 가 실행될 때 자동으로 에러를 처리합니다.

현재 VI 의 에러 핸들링을 비활성화 하려면, **파일>VI 프로퍼티**를 선택하고 **항목** 폴다운 메뉴에서 **실행**을 선택합니다. 사용자가 생성한 새로운 빈 VI 의 자동 에러 핸들링을 비활성화하려면, **도구>옵션**을 선택하고 **항목** 리스트에서 **블록다이어그램**을 선택합니다. VI 내의 subVI 나 함수의 자동 에러 핸들링을 비활성화하려면 **에러 출력** 파라미터를 다른 subVI 나 함수의 **에러 입력** 파라미터 또는 **에러 출력** 인디케이터에 연결합니다.

다른 에러 핸들링 방법을 선택할 수 있습니다. 예를 들어, 블록다이어그램에서 I/O VI 의 타임아웃으로 전체 어플리케이션이 정지하고 에러 대화 상자를 디스플레이하기를 원하지 않을 수도 있습니다. 또한 특정 시간 간격으로 그 VI 를 다시 시도하기를 원할 수도 있습니다. LabVIEW 의 경우, VI 의 블록다이어그램에서 이러한 에러 핸들링에 대한 결정을 할 수 있습니다.

**대화 상자 & 사용자 인터페이스** 팔레트의 LabVIEW 에러 핸들링 VI 와 함수 및 대부분의 VI 와 함수의 **에러 입력**과 **에러 출력** 파라미터를 사용하여 에러를 처리합니다. 예를 들어, LabVIEW 가 에러를 만났을 경우 다른 종류의 대

화 상자에 에러 메시지를 디스플레이할 수 있습니다. 에러 핸들링과 디버깅 도구를 함께 사용하여 에러를 찾아내고 관리합니다.

VI 와 함수는 숫자 에러 코드 또는 에러 클러스터로 에러를 반환합니다. 일반적으로 함수는 숫자 에러 코드를, VI 는 에러 클러스터를 사용하여 에러를 입력 또는 출력합니다.

LabVIEW 의 에러 핸들링은 데이터 흐름 모델을 따릅니다. 데이터 값이 VI 를 따라 흐르듯이 에러 정보도 VI 를 따라 흐릅니다. 에러 정보를 VI 의 시작에서 끝까지 연결합니다. VI 의 끝에 에러 핸들러 VI 를 놓아 해당 VI 가 에러 없이 실행되었는지 확인합니다. 사용하거나 만든 각 VI 의 **에러 입력**과 **에러 출력** 클러스터를 사용하여 VI 를 통해 에러 정보를 전달합니다. 에러 클러스터는 흐름 파라미터입니다.

흐름 파라미터에 대한 추가적인 정보는 5 장, *블록 다이어그램 만들기의 흐름 파라미터* 섹션을 참조하십시오.

VI 를 실행하면 LabVIEW 는 각 실행 노드에서 에러를 테스트합니다. LabVIEW 가 에러를 발견하지 못하면 노드는 정상적으로 실행됩니다. LabVIEW 가 에러를 발견하면 노드는 코드의 해당 부분을 실행하지 않고 에러를 다음 노드로 전달합니다. 다음 노드도 해당 부분을 실행하지 않고 에러를 그 다음 노드로 전달합니다. 실행 흐름의 끝에서 LabVIEW 는 에러를 보고합니다.

## 에러 클러스터

**에러 입력**과 **에러 출력** 클러스터는 다음의 원소의 정보를 포함합니다:

- **상태**는 에러가 발생했을 때 참을 보고하는 불리언 값입니다.
- **코드**는 에러를 숫자로 표현하는 32 비트 부호있는 정수입니다. 제로가 아닌 에러 코드가 표시되고 **상태**에 거짓이 나타나면 에러가 아닌 경고를 나타냅니다.
- **소스**는 어디에서 에러가 발생했는지 나타내는 문자열입니다.

또한 불리언 데이터를 받는 일부 VI, 함수, 구조도 에러 클러스터를 인식합니다. 예를 들어, 에러 클러스터를 선택, LabVIEW 종료, 또는 정지 함수의 불리언 입력에 연결할 수 있습니다. 에러가 발생하는 경우, 에러 클러스터는 참값을 해당 함수에 전달합니다.

클러스터에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 클러스터* 섹션을 참조하십시오.

## 에러 핸들링을 위해 While 루프 사용하기

에러 클러스터를 While 루프의 조건 터미널에 연결하여 While 루프의 반복을 멈춥니다. 에러 클러스터를 조건 터미널에 연결하면 에러 클러스터 **상태** 파라미터의 참 또는 거짓값만 터미널에 전달됩니다. 에러가 발생하면 While 루프는 멈춥니다.

에러 클러스터가 조건 터미널에 연결될 때, **참이면 정지**나 **참이면 계속** 바로 가기 메뉴 아이템은 **에러 발생시 정지**와 **에러 발생시 계속**으로 변경됩니다.

While 루프 사용에 대한 추가적인 정보는 8 장, *루프와 구조의 While 루프* 섹션을 참조하십시오.

## 에러 핸들링을 위해 케이스 구조 사용하기

에러 클러스터를 케이스 구조의 선택자 터미널에 연결할 때, 케이스 선택자 라벨은 에러와 에러 없음의 두 케이스를 디스플레이하고, 케이스 구조의 경계는 색이 변경됩니다 — 에러의 경우에는 빨강, 에러 없음의 경우에는 녹색. 에러가 발생하는 경우, 케이스 구조는 에러 서브다이아그램을 실행합니다.

케이스 구조 사용에 대한 추가적인 정보는 8 장, *루프와 구조의 케이스 구조* 섹션을 참조하십시오.

에러 핸들링 템플릿 VI 를 가진 subVI 를 사용하여 에러 핸들링을 위한 케이스 구조를 가진 VI 를 생성합니다.

템플릿 VI 에 대한 추가적인 정보는 제 1 장, *LabVIEW 소개의 LabVIEW VI 템플릿* 섹션을 참조하십시오.

## VI 와 SubVI 생성하기

VI 는 사용자 인터페이스 또는 자주 사용하는 작업이 될 수 있습니다 . 프런트패널과 블록다이어그램을 만드는 방법을 배운 후에는 자신만의 VI 와 subVI 를 생성하고 , 생성한 VI 를 사용자 정의할 수 있습니다 .

### 예제 검색하기

새 VI 를 만들기 전에 , **도움말>예제 찾기**를 선택하여 NI 예제 탐색기를 열고 필요 사항에 맞는 예제 VI 가 있는지 검색해보도록 하십시오 . 적당한 예제 VI 를 찾을 수 없는 경우 , **새로 만들기** 대화 상자에서 템플릿 VI 를 열고 **함수** 팔레트의 내장 VI 와 함수로 템플릿을 채우십시오 .

예제 VI 와 템플릿 VI 에 대한 추가적인 정보는 1 장 , *LabVIEW 소개의 LabVIEW VI 템플릿 , 예제 VI , 도구* 섹션을 참조하십시오 .

### 내장된 VI 와 함수 사용하기

LabVIEW 에는 데이터 수집 VI 와 함수 , 다른 VI 에 접근하는 VI , 다른 어플리케이션과 통신하는 VI 등과 같이 특수한 어플리케이션을 만드는데 도움을 주는 내장 VI 와 함수가 있습니다 . 이러한 VI 를 어플리케이션에 subVI 로 사용하여 개발 시간을 줄일 수 있습니다 . 새 VI 를 만들기 전에 **함수** 팔레트를 검색하여 비슷한 VI 와 함수를 찾고 기존 VI 를 새 VI 의 시작점으로 사용하는 것을 고려하십시오 .

### SubVI 생성하기

VI 를 만든 이후에는 다른 VI 에서도 VI 를 사용할 수 있습니다 . 다른 VI 의 블록다이어그램에서 호출되는 VI 를 subVI 라고 부릅니다 . SubVI 를 다른 VI 에 다시 사용할 수 있습니다 . SubVI 를 생성하려면 , 커넥터 팬을 만들고 아이콘을 생성해야 합니다 .

SubVI 노드는 텍스트 기반 프로그래밍 언어의 서브루틴 호출에 해당합니다 . 프로그램에서 서브루틴 호출 명령문이 서브루틴이 아닌 것처럼 , 노드가 subVI 인 것은 아닙니다 . 여러 동일한 subVI 노드를 포함하는 블록다이어그램은 동일한 subVI 를 여러번 호출합니다 .

SubVI 컨트롤과 인디케이터는 호출하는 VI 의 블록다이어그램에서 데이터를 받고 데이터를 반환합니다. **함수** 팔레트에서 **VI 선택** 아이콘 또는 텍스트를 클릭한 후, VI 를 탐색하여 더블 클릭하고 VI 를 블록다이어그램에 놓아서 해당 VI 를 호출하는 subVI 를 생성합니다.

수행 도구 또는 위치 도구로 블록다이어그램에 있는 subVI 를 더블 클릭하여 subVI 를 편집할 수 있습니다. SubVI 를 저장할 때, subVI 의 변경은 현재 인스턴스만이 아닌 모든 subVI 에 대한 호출에 영향을 미칩니다.

## 아이콘 생성하기

모든 VI 는 프런트패널과 블록다이어그램 윈도우 오른쪽 위 코너에 다음에 보이는 것과 같이 아이콘을 디스플레이합니다.



아이콘은 VI 의 그래픽 표현입니다. 아이콘은 텍스트, 이미지, 또는 둘 다 포함할 수 있습니다. VI 를 subVI 로 사용할 경우, 아이콘은 VI 의 블록다이어그램에서 subVI 를 식별합니다.

기본 아이콘은 LabVIEW 를 실행한 이후로 얼마나 많은 새 VI 를 열었는지를 나타내는 숫자를 포함합니다. 프런트패널 또는 블록다이어그램의 오른쪽 위 코너에 있는 아이콘에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **아이콘 편집**을 선택하거나 프런트패널의 오른쪽 위 코너에 있는 아이콘을 더블 클릭하여 기본 아이콘을 대체하는 사용자 아이콘을 생성합니다.

또한 파일 시스템 어디에서든 그래픽을 끌어다가 프런트패널 또는 블록다이어그램 오른쪽 위 코너에 떨어 뜨릴 수 있습니다. LabVIEW 는 그래픽을 32 x 32 픽셀 아이콘으로 변환합니다.

VI 아이콘에서 사용하는 표준 그래픽에 대한 정보를 참조하려면 National Instruments 웹 사이트 [ni.com/info](http://ni.com/info) 에서 정보 코드 `exkrkb` 를 입력하십시오.

## 커넥터 팬 만들기

VI 를 subVI 로 사용하려면, 다음에서 보이는 것과 같이 커넥터 팬을 만들어야 합니다.



커넥터 팬은 해당 VI 의 컨트롤과 인디케이터에 대응하는 터미널의 세트이며, 텍스트 기반 프로그래밍 언어에서 함수 호출의 파라미터 리스트와 유사

사합니다. 커넥터 팬은 VI 에 연결할 수 있는 입력과 출력을 정의하여 VI 를 subVI 로 사용할 수 있도록 합니다. 커넥터 팬은 입력 터미널에서 데이터를 받고 프런트패널 컨트롤을 통하여 블록다이어그램 코드에 데이터를 전달하며 프런트패널 인디케이터로부터 출력 터미널에 결과를 받습니다.

프런트패널 컨트롤과 인디케이터를 각 커넥터 팬 터미널에 지정하여 연결을 정의합니다. 커넥터 팬을 정의하려면, 프런트패널의 오른쪽 위 코너에 있는 아이콘에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **커넥터 보이기**를 선택하여 커넥터 팬을 디스플레이합니다. 아이콘 대신 커넥터 팬이 나타납니다. 처음에 커넥터 팬을 보면 커넥터 패턴을 볼 수 있습니다. 커넥터 팬에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **패턴**을 선택하여 다른 패턴을 선택할 수 있습니다.

커넥터 팬의 각 사각형은 터미널을 나타냅니다. 사각형을 사용하여 입력과 출력을 지정합니다. 기본 커넥터 팬 패턴은  $4 \times 2 \times 2 \times 4$  입니다. VI 가 변경되어 새 입력이나 출력이 필요한 경우를 대비하여 기본 커넥터 팬 패턴에서 할당되지 않은 여분의 터미널은 남겨둡니다.

하나의 커넥터 팬에 최대 28 개의 터미널까지 할당할 수 있습니다. 프런트패널에 프로그램적으로 사용하려는 컨트롤과 인디케이터가 28 개 이상 있을 경우, 그 중 일부를 클러스터로 그룹화한 후 클러스터를 커넥터 팬의 터미널에 지정합니다.

클러스터를 사용한 데이터 그룹화에 대한 추가적인 정보는 제 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 클러스터* 섹션을 참조하십시오.

커넥터 팬에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **패턴**을 선택하여 다른 터미널 패턴을 선택합니다. 예를 들어, 여분의 터미널이 있는 커넥터 팬 패턴을 선택할 수 있습니다. 필요할 때까지 여분의 터미널을 연결하지 않은 상태로 둘 수 있습니다. 이러한 유연성은 VI 의 계층구조에 최소한의 영향을 주면서 변경이 가능하도록 합니다.

## VI 의 섹션에서 subVI 생성하기

위치 도구를 사용하여 다시 사용하려는 블록다이어그램의 섹션을 선택하고 **편집 > SubVI 생성**을 선택하여 VI 의 섹션을 subVI 로 변환합니다. 새로운 subVI 에 대한 아이콘이 블록다이어그램의 선택 부분을 대체합니다. LabVIEW 는 새 subVI 에 대한 컨트롤과 인디케이터를 생성하고, 선택한 컨트롤과 인디케이터 터미널의 개수를 기반으로 자동적으로 커넥터 팬을 설정하며, subVI 를 기존의 와이어에 연결합니다.

선택에 의한 subVI 생성은 편리하지만, VI 의 논리적인 계층구조를 생성하기 위해서는 신중한 계획이 요구됩니다. 선택에 어떤 객체를 포함할지를 고려하고 결과 VI 의 기능이 바뀌는 것을 피하도록 합니다.

## SubVI 프런트패널 디자인하기

커넥터 팬에 나타나는 것과 같이 프런트패널에 컨트롤과 인디케이터를 놓습니다. 프런트패널의 왼쪽에 컨트롤을 놓고 오른쪽에 인디케이터를 놓습니다. 프런트패널의 왼쪽 아래에 **에러 입력** 클러스터를 놓고 오른쪽 아래에 **에러 출력** 클러스터를 놓습니다.

커넥터 팬 설정에 대한 추가적인 정보는 이 장의 *커넥터 팬 만들기* 섹션을 참조하십시오.

## VI 의 계층구조 보기

**VI 계층구조** 윈도우는 타입 정의와 글로벌 변수를 포함하여 메모리에 있는 모든 VI 의 호출 계층구조와 열려 있는 모든 LabVIEW 프로젝트 및 타겟의 그래픽 표시를 디스플레이합니다. **보기 >> VI 계층구조**를 선택하여 **VI 계층구조** 윈도우를 디스플레이합니다. 이 윈도우를 사용하여 메모리상의 VI 를 구성하는 subVI 와 다른 노드 보기를 하고 VI 계층구조에서 검색합니다.

LabVIEW 프로젝트에 대한 추가적인 정보는 3 장, *LabVIEW 환경의 프로젝트 탐색기 윈도우* 섹션을 참조하십시오.

**VI 계층구조** 윈도우는 최상위 아이콘을 디스플레이하여 메인 LabVIEW 어플리케이션 인스턴스를 나타내며, 그 아래에 프로젝트에 속하지 않거나 프로젝트를 위한 어플리케이션 인스턴스의 일부가 아닌 모든 열린 VI 가 나타납니다. 프로젝트를 추가하면 **VI 계층구조** 윈도우는 또한 다른 상위 레벨 아이콘을 디스플레이하여 프로젝트를 나타냅니다. 추가하는 각 타겟이 프로젝트에 나타납니다.

커서를 **VI 계층구조** 윈도우의 객체 위로 움직이면, LabVIEW 는 각 VI 의 이름을 톱 상자로 디스플레이합니다. 위치 도구를 사용하여 **VI 계층구조** 윈도우에서 블록다이어그램으로 VI 를 끌어서 VI 를 다른 VI 의 subVI 로 사용할 수 있습니다. 또한 한 노드 또는 여러 노드를 선택하고 클립보드에 복사하여 다른 블록다이어그램에 붙여넣을 수 있습니다. **VI 계층구조** 윈도우의 VI 에서 더블 클릭하여 그 VI 의 프런트패널을 디스플레이합니다.

SubVI 를 가지는 VI 는 아래쪽 경계에 화살표 버튼이 있습니다. 이 화살표 버튼을 클릭하여 subVI 를 보이거나 숨깁니다. 빨간색 화살표 버튼은 모든 subVI 가 숨겨져 있을 때 나타납니다. 검은색 화살표 버튼은 모든 subVI 가 디스플레이되었을 때 나타납니다.

## 다형성 VI

다형성 VI 는 단일 입력 또는 출력 터미널에서 다른 데이터 타입을 받습니다. 다형성 VI 는 같은 커넥터 팬 패턴을 가지는 VI 의 모음입니다. 모음의 각 VI 는 다형성 VI 의 인스턴스입니다.

예를 들어, ( 키 읽기 ) VI 는 다형성입니다. **기본값** 터미널은 불리언, 배정도 부동소수, 32 비트 부호있는 정수, 경로, 문자열, 또는 32 비트 부호없는 정수 데이터를 받습니다.

대부분의 다형성 VI 에서, 다형성 VI 의 입력에 연결된 데이터 타입은 사용할 인스턴스를 결정합니다. 다형성 VI 가 데이터 타입에 호환되는 인스턴스를 포함하지 않는 경우, 깨진 와이어가 나타납니다. 다형성 VI 입력에 연결한 데이터 타입이 사용할 인스턴스를 결정하지 않는다면, 그 인스턴스를 수동으로 선택해야 합니다. 다형성 VI 의 인스턴스를 수동으로 선택하면, 그 VI 는 선택한 인스턴스의 데이터 타입만을 받고 반환하기 때문에 이 VI 는 더 이상 다형성 VI 로서 작동하지 않습니다.

인스턴스를 수동으로 선택하기 위해서, 다형성 VI 에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **타입 선택**을 선택하고 사용할 인스턴스를 선택합니다. 또한 수행 도구를 사용하여 다음에서 보는 것과 같이 다형성 VI 선택자를 클릭하고 바로 가기 메뉴에서 인스턴스를 선택할 수 있습니다.

읽기 전용 ▾

블록 다이어그램의 다형성 VI 에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **보이는 아이템** > **다형성 VI 선택자**를 선택하여 선택자를 디스플레이합니다. 다루어지는 모든 데이터 타입을 다시 받도록 다형성 VI 를 변경하기 위해서, 다형성 VI 에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **타입 선택** > **자동**을 선택하거나 수행 도구를 사용하여 다형성 VI 선택자를 클릭하고 바로 가기 메뉴에서 **자동**을 선택합니다.

동일한 작업을 다른 데이터 타입에 수행할 때 다형성 VI 를 만듭니다.



**노트** 다형성 VI 는 LabVIEW Professional Development System 에서만 만들고 편집할 수 있습니다.

예를 들어, 단정도 부동소수, 숫자값의 배열, 또는 웨이브폼에 동일한 수학 연산을 수행하려는 경우, 계산, 배열 계산, 웨이브폼 계산의 3 개의 분리된 VI 를 생성할 수 있습니다. 연산을 수행할 때, 입력으로 사용하려는 데이터 타입에 따라서 3 개의 VI 중에서 하나를 블록 다이어그램에 놓습니다.

수동으로 VI 의 버전을 블록 다이어그램에 놓는 대신에, 단일 다형성 VI 를 생성하고 사용할 수 있습니다.

## VI 저장하기

**파일>저장**을 선택하여 VI 를 저장합니다. VI 를 저장할 때, 설명적인 이름을 사용하여 나중에 VI 를 쉽게 식별할 수 있도록 합니다. 또한, LabVIEW 를 업그레이드하기 편리하게 하고 필요할 때 LabVIEW 의 두가지 버전으로 VI 를 유지하는데 도움을 줄 수 있도록 LabVIEW 의 이전 버전으로 VI 를 저장할 수 있습니다.

### VI 이름 붙이기

VI 를 저장할 때, 설명적인 이름을 사용합니다. Temperature Monitor.vi 와 Serial Write & Read.vi 와 같은 설명적인 이름은 VI 와 사용법의 식별을 쉽게 합니다. VI#1.vi 와 같이 불분명한 이름을 사용할 경우, 특히 여러 VI 를 저장한 경우 VI 를 식별하는 것이 힘들 것입니다.

사용자가 다른 플랫폼에서 VI 를 사용할지를 고려합니다. \:/\*<># 과 같이 일부 OS 에서 특별한 경우에 사용하는 문자의 사용은 피합니다.



**노트** 컴퓨터에 같은 이름으로 저장된 VI 가 여러 개 있을 경우, LabVIEW 가 최상위 레벨 VI 를 실행할 때 잘못된 subVI 를 참조하지 않도록 주의깊게 VI 를 다른 디렉토리나 LLBs 에 구성합니다.

### 이전 버전으로 저장하기

LabVIEW 의 이전 버전으로 VI 를 저장하여 필요한 경우 두 개 버전의 LabVIEW 에서 VI 를 사용할 수 있도록 하고 LabVIEW 의 업그레이드를 편리하도록 할 수 있습니다. **파일>이전 버전으로 저장**을 선택하여 LabVIEW 의 이전 버전으로 저장합니다.

VI 를 이전 버전으로 저장할 때, LabVIEW 는 VI 뿐만 아니라 labview\vi.lib 디렉토리의 파일을 제외한 계층구조의 모든 VI 를 변환합니다.

중종 VI 는 이전 버전의 LabVIEW 에서는 없었던 기능을 사용하기도 합니다. 이런 경우, LabVIEW 는 가능한 많은 VI 를 저장하고 변환할 수 있는 것들에 대해서는 리포트를 만듭니다. 리포트는 **경고** 대화 상자에 즉시 나타납니다. **확인** 버튼을 클릭하여 경고를 확인하고 대화 상자를 닫습니다. **파일에 저장** 버튼을 클릭하여 경고를 나중에 보기 위해서 텍스트 파일로 저장합니다.

# VI 사용자 정의하기

어플리케이션에 맞춰서 사용하기 위해서 VI 와 subVI 를 설정할 수 있습니다. 예를 들어, 사용자 입력을 요구하는 VI 를 subVI 로 사용하려는 경우, VI 를 설정하여 호출될 때마다 프런트패널이 나타나도록 합니다.

**파일» VI 프로퍼티**를 선택하여 VI 의 모양과 특성을 설정합니다. **VI 프로퍼티** 대화 상자 상단의 **항목** 폴다운 메뉴를 사용하여 여러 다른 옵션 항목 중 하나를 선택합니다.

**VI 프로퍼티** 대화 상자에는 다음 옵션 항목이 포함됩니다:

- **일반** — 이 페이지를 사용하여 VI 가 저장된 곳의 현재 경로, 개정 번호, 개정 히스토리, 마지막으로 저장된 이후의 모든 변화를 결정합니다. 또한 이 페이지를 사용하여 VI 의 아이콘을 편집할 수 있습니다.
- **문서** — 이 페이지를 사용하여 VI 에 설명을 추가하고 도움말 파일 주제에 연결합니다.  
문서 옵션에 대한 추가적인 정보는 제 12 장, *VI 문서화 및 인쇄하기*의 *VI 문서화하기* 섹션을 참조하십시오.
- **보안** — 이 페이지를 사용하여 VI 를 잠그거나 암호 보호를 합니다.
- **윈도우 모양** — 이 페이지를 사용하여 윈도우 제목 및 스타일과 같은 VI 의 윈도우 모양을 사용자 정의합니다.
- **윈도우 크기** — 이 페이지를 사용하여 윈도우의 크기를 설정합니다.
- **실행** — 이 페이지를 사용하여 VI 가 어떻게 실행될지를 설정합니다. 예를 들어, VI 를 열면 즉시 실행되도록 설정하거나 subVI 로 호출될 때 일시 정지하도록 설정할 수 있습니다.
- **편집자 옵션** — 이 페이지를 사용하여 현재 VI 에 대한 정렬 눈금의 크기를 설정하고, 터미널에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **생성»컨트롤** 또는 **생성»인디케이터**를 선택할 때 LabVIEW가 생성하는 컨트롤 또는 인디케이터의 스타일을 변경합니다.  
정렬 눈금에 대한 추가적인 정보는 제 4 장, *프런트패널 만들기의 객체 정렬 및 간격 조절하기* 섹션을 참조하십시오.

## 루프와 구조

구조는 텍스트 기반 프로그래밍 언어의 루프 및 케이스 문을 그래픽하게 표현한 것입니다. 블록다이어그램에서 구조를 사용하여 코드의 블록을 반복하고 조건적 또는 특정한 순서로 코드를 실행합니다.

다른 노드처럼, 구조는 다른 블록다이어그램 노드로 연결해 주고, 입력 데이터가 있을 때 자동으로 실행하고, 실행이 완료되었을 때 데이터를 출력 와이어에 공급하는 터미널을 가지고 있습니다.

각 구조는 구조의 법칙에 따라 실행하는 블록다이어그램의 섹션을 둘러싸는 특징적이고, 크기 조정 가능한 경계를 가지고 있습니다. 구조의 경계선 안의 블록다이어그램 부분을 서브다이어그램이라고 부릅니다. 구조로 데이터를 주고 받는 터미널을 터널이라고 부릅니다. 터널은 구조 경계선의 연결 포인트입니다.

**구조** 팔레트에 있는 다음 구조를 사용하여 어떻게 블록다이어그램을 실행할 것인지 컨트롤합니다:

- **For 루프** — 지정된 횟수만큼 서브다이어그램을 실행합니다.
- **While 루프** — 조건이 발생할 때까지 서브다이어그램을 실행합니다.
- **케이스 구조** — 여러 개의 서브다이어그램을 가지고 있으며, 이 구조에 전달되는 입력값에 따라서 이중의 하나만이 실행됩니다.
- **시퀀스 구조** — 하나 이상의 서브다이어그램을 가지고 있으며, 이것들을 연속적인 순서에 따라서 실행합니다.
- **이벤트 구조** — 사용자와 VI의 상호 작용에 따라서 실행되는 하나 이상의 서브다이어그램을 가지고 있습니다.
- **Timed 구조** — 하나 또는 그 이상의 서브다이어그램을 시간 제한과 지연을 가지고 실행합니다.

구조의 경계선에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴를 디스플레이합니다.

# For 루프와 While 루프 구조

For 루프와 While 루프를 사용하여 반복 실행을 제어하십시오 .

## For 루프

다음과 같이 For 루프는 서브다이아그램을 지정된 횟수만큼 실행합니다 .



다음과 같이 카운트 터미널 ( 입력 터미널 ) 값은 서브다이아그램이 반복될 횟수를 나타냅니다 .



루프의 바깥쪽에서 카운트 터미널의 왼쪽이나 오른쪽에 값을 연결하여 명확하게 카운트를 설정하거나 오토인덱싱을 이용하여 내부적으로 카운트를 설정할 수 있습니다 .

카운트를 내부적으로 설정하는 방법에 대한 더 많은 정보는 이 장의 *오토인덱싱을 사용하여 For 루프 카운트 설정하기* 섹션을 참조하십시오 .

다음과 같이 반복 터미널 ( 출력 터미널 )에는 완료된 반복 횟수가 있습니다 .



반복 카운트는 항상 0에서 시작합니다 . 첫 루프 반복동안 , 반복 터미널은 0을 반환합니다 .

카운트 터미널과 반복 터미널은 모두 32 비트 부호있는 정수입니다 . 부동소수를 카운트 터미널에 연결하면 , LabVIEW 는 이 값을 반올림하여 범위 내로 강제 변환합니다 . 0 이나 음수를 카운트 터미널에 연결하면 , 루프는 실행되지 않고 출력은 그 데이터 타입의 기본값을 가집니다 .

For 루프에 시프트 레지스터를 추가하여 현재 반복에서 다음 반복으로 데이터를 전달합니다 .

시프트 레지스터를 루프에 추가하는 것에 대한 더 많은 정보는 이 장의 *시프트 레지스터* 섹션을 참조하십시오 .

## While 루프

텍스트 기반 프로그래밍 언어의 Do 루프 또는 Repeat-Until 루프와 유사하며, 다음과 같이 While 루프는 조건이 발생할 때까지 서브다이아그램을 실행합니다.



While 루프는 입력 터미널인 조건 터미널이 특정 불리언 값을 받을 때까지 서브다이아그램을 실행합니다. 다음과 같이 조건 터미널의 기본 동작과 모양은 **참이면 정지**입니다.

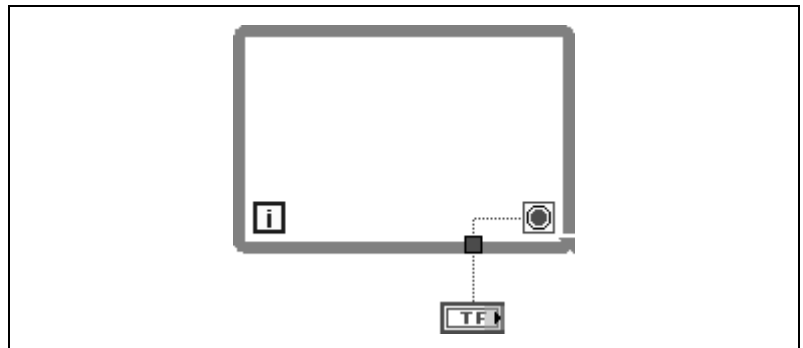


조건 터미널이 **참이면 정지**이면, 조건 터미널이 참값을 받을 때까지 While 루프가 서브다이아그램을 실행합니다. 다음과 같이, While 루프의 터미널이나 경계선에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **참이면 계속**를 선택하여 조건 터미널의 동작과 모양을 변경할 수 있습니다.



조건 터미널이 **참이면 계속**이면, 조건 터미널이 거짓값을 받을 때까지 While 루프가 서브다이아그램을 실행합니다. 또한 수행 도구를 사용하여 조건 터미널을 클릭해서 조건을 변경할 수 있습니다.

다음 그림에서 볼 수 있는 것처럼 While 루프 밖에 불리언 컨트롤의 터미널을 놓고, 루프가 시작할 때 조건 터미널이 **참이면 정지**이면 거짓으로 컨트롤을 설정해 놓으면, 무한 루프를 발생시킵니다. 또한 루프 밖의 컨트롤을 참으로 설정하고 조건 터미널을 **참이면 계속**으로 놓으면 무한 루프가 발생합니다.



컨트롤의 값을 바꾸어도 무한 루프를 멈출 수 없습니다. 그 값은 루프가 시작되기 전에 한번만 읽히기 때문입니다. 이 무한 루프를 멈추려면, 도구 모음의 **실행 강제 종료** 버튼을 클릭하여 VI를 강제 종료해야 합니다.

While 루프의 조건 터미널을 사용하여 기본적인 에러 핸들링을 수행할 수 있습니다. 에러 클러스터를 조건 터미널에 연결하면 에러 클러스터 **상태** 파라미터의 참이나 거짓값만 터미널에 전달됩니다. 그리고 **참이면 정지**와 **참이면 계속** 바로 가기 메뉴 아이템이 **에러 발생시 정지**와 **에러 발생시 계속**으로 변경됩니다.

에러 클러스터와 에러 핸들링에 대한 추가적인 정보는 6 장, *VI 실행하고 디버깅하기의 에러 핸들링하기* 섹션을 참조하십시오.

다음과 같이 반복 터미널 (출력 터미널)에는 완료된 반복 횟수가 있습니다.



반복 카운트는 항상 0에서 시작합니다. 첫 루프 반복동안, 반복 터미널은 0을 반환합니다.

While 루프에 시프트 레지스터를 추가하여 현재 반복에서 다음 반복으로 데이터를 전달합니다.

시프트 레지스터를 루프에 추가하는 것에 대한 더 많은 정보는 이 장의 *시프트 레지스터* 섹션을 참조하십시오.

## 타이밍 컨트롤하기

데이터 값을 차트에 플롯하는 속도와 같은 프로세스 실행 속도를 컨트롤할 필요가 있을 것입니다. 기다림 함수를 루프에서 사용하여, 다음 루프를 실행하기 전에 밀리초 단위의 시간만큼 기다리게 만들 수 있습니다.

## 오토인덱싱 루프

배열을 For 루프나 While 루프 입력 터미널에 연결하는 경우, 오토인덱싱을 활성화하여 해당 배열의 각 원소를 읽고 처리할 수 있습니다.

배열에 대한 추가적인 정보는 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 배열* 섹션을 참조하십시오.

배열을 루프 경계선의 입력 터미널에 연결하고 입력 터미널의 오토인덱싱을 활성화하면, 그 배열의 원소들이 첫번째 원소부터 한번에 하나씩 루프로 들어갑니다. 오토인덱싱을 비활성화하면 전체 배열이 한꺼번에 루프로 전달됩니다. 배열 출력 터미널을 오토인덱싱하면 출력 배열이 루프가 반복될 때마다만 들어지는 새로운 원소를 받아들입니다. 그러므로, 오토인덱싱된 출력 배열의 크기는 항상 반복 횟수와 같은 크기입니다. 예를 들어, 루프가 10회 실행

행되면 출력 배열은 10 개의 원소를 가집니다 . 출력 터널의 오토인덱싱을 비활성화하면 , 루프의 마지막 실행의 원소만 블록다이어그램의 다음 노드로 전달됩니다 .

루프 경계의 터널에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **인덱싱 활성화** 또는 **인덱싱 비활성화**를 선택하여 오토인덱싱을 활성화하거나 비활성화합니다 . While 루프에서 오토인덱싱은 기본적으로 비활성화되어 있습니다 .

루프 경계에 괄호 모양이 있으면 오토인덱싱이 활성화된 것입니다 . 출력 터널과 그 다음 노드를 연결하는 와이어의 굵기도 루프가 오토인덱싱을 사용하는지 여부를 나타냅니다 . 오토인덱싱을 사용하면 와이어가 스칼라가 아닌 배열을 가지고 있으므로 더 두꺼워집니다 .

루프는 1 차원 배열에 대해서 스칼라를 인덱싱하고 , 2 차원 배열에 대해서는 1 차원 배열을 인덱싱합니다 . 출력 터널에 대해서는 반대로 적용됩니다 . 스칼라 원소는 1 차원 배열로 순차적으로 쌓이고 , 1 차원 배열은 2 차원 배열로 쌓입니다 .

## 오토인덱싱을 사용하여 For 루프 카운트 설정하기

For 루프의 입력 터미널에 연결된 배열에 오토인덱싱을 활성화하면 , 사용자가 카운트 터미널에 연결할 필요가 없이 LabVIEW 가 카운트 터미널을 배열 크기에 맞추어 자동으로 설정해줍니다 . For 루프를 사용하여 한번에 한 원소씩 배열을 처리할 수 있기 때문에 , LabVIEW 는 For 루프에 연결되는 모든 배열에 대해 기본 설정으로 오토인덱싱을 활성화시킵니다 . 한번에 배열 한 원소씩 처리할 필요가 없다면 오토인덱싱을 비활성화시키십시오 .

하나 이상의 터널에 오토인덱싱을 활성화하거나 , 카운트 터미널을 연결하면 더 작은 카운트가 선택됩니다 . 예를 들어 , 각각 10 개와 20 개의 원소를 가진 2 개의 오토인덱싱된 배열을 루프에 입력하고 , 카운트 터미널에 15 를 연결한다면 , 루프는 10 회만 실행되고 루프는 두번째 배열의 처음 10 개 원소까지만 인덱싱합니다 . 다른 예제에서 , 2 개의 소스에서 나온 데이터 중에서 처음 100 개 원소를 그래프에 플롯하고자 한다면 , 카운트 터미널에 100 을 연결하십시오 . 1 개의 데이터 소스에 50 개의 원소 밖에 없다면 , 루프는 50 회 실행되고 인덱싱은 처음 50 개 원소까지만 됩니다 . ( 배열 크기 ) 함수를 사용하여 배열의 크기를 확인하십시오 .

## While 루프의 오토인덱싱

While 루프의 배열 입력 오토인덱싱을 활성화하면 , While 루프도 For 루프와 같은 방법으로 배열을 인덱싱합니다 . 반면에 While 루프는 특정 조건이 발생할 때까지 실행이 반복되는 것이므로 , While 루프의 반복 횟수는 입력 배열의 크기로 제한되지 않습니다 . While 루프의 반복 횟수가 입력 배열의

크기를 넘어서면, 그 배열 원소 타입의 기본값이 루프로 전달됩니다. ( 배열 크기 ) 함수를 이용하여 이렇게 기본값이 While 루프로 전달되는 것을 피할 수 있습니다. ( 배열 크기 ) 함수는 배열에 몇 개의 원소가 있는지 보여줍니다. 반복 횟수가 배열의 크기와 같아지면 While 루프의 실행을 멈추도록 설정합니다.



**주의** 출력 배열의 크기를 미리 정할 수 없기 때문에, 출력 배열에서는 For 루프를 사용하여 오토인덱싱하는 것이 While 루프를 사용하는 것보다 더 효과적입니다. 반복을 너무 많이 하면 시스템 메모리를 넘어버릴 수도 있습니다.

## 루프를 이용한 배열 만들기

루프를 이용하여 배열 원소를 읽고 처리하는 것과 더불어, For 루프와 While 루프를 이용하여 배열을 만들 수 있습니다. 루프 내의 VI 나 함수의 출력을 루프 경계에 연결하십시오. While 루프를 사용한다면, 결과 터널에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **인덱싱 활성화**를 선택합니다. For 루프의 경우, 인덱싱이 기본 설정으로 활성화되어 있습니다. 이 터널의 출력은 루프의 각 반복에서 VI 나 함수가 반환한 모든 값의 배열입니다.

배열에 대한 추가적인 정보는 9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화*의 배열 섹션을 참조하십시오.

배열을 만드는 예제는 labview\examples\general\arrays.llb 를 참조하십시오.

## 루프의 시프트 레지스터와 피드백 노드

For 루프나 While 루프에서 시프트 레지스터나 피드백 노드를 사용하여 한 루프 반복에서 다음 반복으로 값을 전달합니다.

### 시프트 레지스터

이전 반복의 값을 루프를 통해 다음 반복으로 전달하고자 할 때 시프트 레지스터를 사용합니다. 다음과 같이 시프트 레지스터는 루프 경계의 양 옆에서 반대인 터미널 쌍으로 나타납니다.

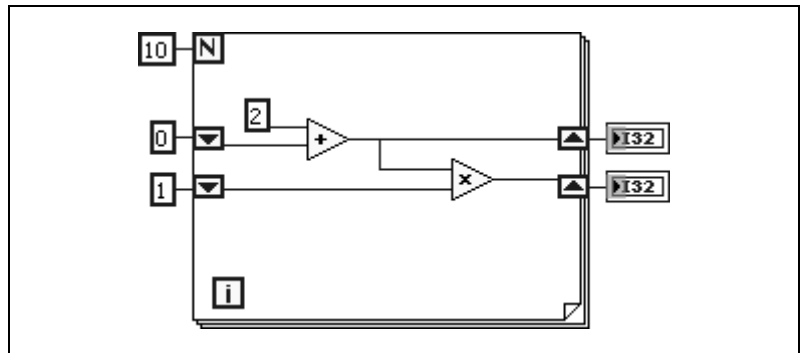


루프 오른쪽의 터미널에는 윗방향 화살표가 있고 반복이 끝날 때마다 데이터를 저장합니다. LabVIEW 는 레지스터의 오른쪽에 연결된 데이터를 다음 반복으로 전달합니다. 루프가 실행된 후에 루프의 오른쪽의 터미널은 시프트 레지스터에 저장된 마지막 값을 반환합니다.

루프의 왼쪽이나 오른쪽 경계에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **시프트 레지스터 추가**를 선택하여 시프트 레지스터를 만듭니다.

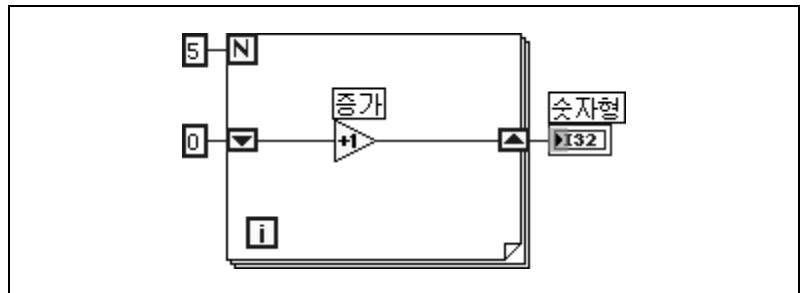
시프트 레지스터는 모든 데이터 타입을 전달하며 시프트 레지스터에 연결된 첫번째 객체의 데이터 타입에 맞게 자동으로 변경됩니다. 각 시프트 레지스터의 터미널에 연결하는 데이터는 동일 타입이어야 합니다.

루프에 여러 개의 시프트 레지스터를 둘 수 있습니다. 루프 속에서 여러 개의 실행이 이전 루프 값을 사용해야 될 경우, 다음 그림과 같이 여러 시프트 레지스터를 사용하여 구조 내 각기 다른 프로세스의 데이터 값을 저장합니다.



## 시프트 레지스터 초기화하기

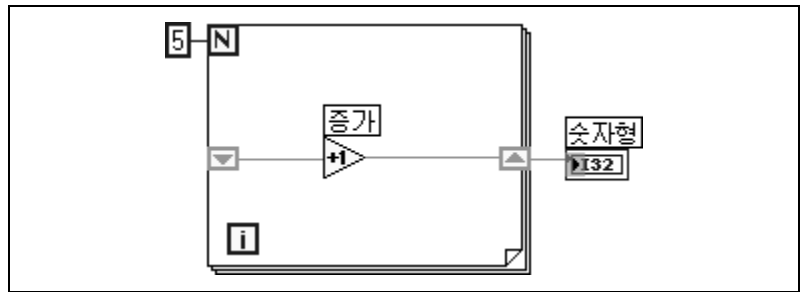
시프트 레지스터를 초기화하는 것은 VI가 실행될 때 시프트 레지스터가 루프의 처음 반복에 전달하는 값을 리셋하는 것입니다. 다음 그림과 같이 루프 왼쪽의 시프트 레지스터 터미널에 컨트롤이나 상수를 연결하여 시프트 레지스터를 초기화합니다.



이전 그림에서, For 루프는 5 번 실행하고 시프트 레지스터가 전달하는 값이 매번 하나씩 증가합니다. For 루프가 5 번 반복한 후, 시프트 레지스터는 마지막 값인 5 를 인디케이터에 전달하고 VI 를 끝냅니다. VI 를 실행할 때마다 시프트 레지스터는 0 의 값으로 시작합니다.

시프트 레지스터를 초기화하지 않는 경우, 루프는 마지막으로 실행되었을 때 시프트 레지스터에 기록된 값을 사용하며, 루프가 실행된 적이 없으면 해당 데이터 타입의 기본값을 사용합니다.

초기화되지 않은 시프트 레지스터의 사용은 VI 의 순차적인 실행에서 상태 정보를 유지하기 위하여 사용되기도 합니다. 다음 그림은 초기화되지 않은 시프트 레지스터를 보여줍니다.

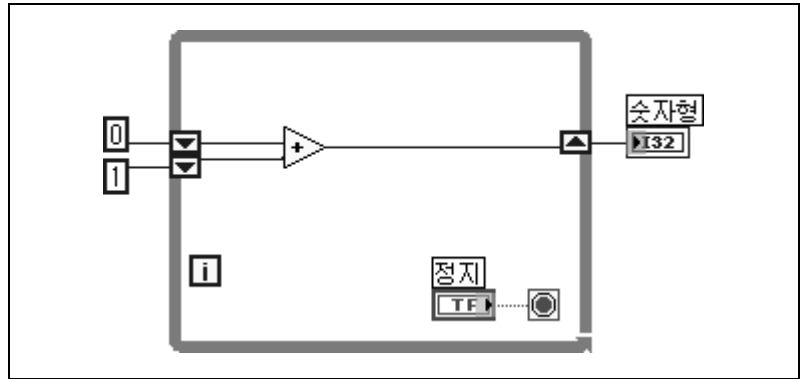


이전 그림에서, For 루프는 5 번 실행하고 시프트 레지스터가 전달하는 값이 매번 하나씩 증가합니다. 처음 VI 를 실행하면, 시프트 레지스터는 32 비트 정수의 기본값인 0 부터 시작합니다. For 루프가 5 번 반복한 후, 시프트 레지스터는 마지막 값인 5 를 인디케이터에 전달하고 VI 를 끝냅니다. 다음번에 VI 를 실행하면, 시프트 레지스터는 이전 실행에서 마지막 값이었던 5 의 초기값으로 시작합니다. For 루프가 5 번 반복하면, 시프트 레지스터는 마지막 값인 10 을 인디케이터에 전달합니다. VI 를 다시 실행한 후 시프트 레지스터는 10 의 값으로 시작합니다. 초기화되지 않은 시프트 레지스터는 VI 를 닫을 때까지 이전 반복 값을 유지합니다.

## 다층 시프트 레지스터

다층 시프트 레지스터는 이전 루프 반복의 데이터에 접근할 수 있게 해줍니다. 다층 시프트 레지스터는 여러 이전 반복의 데이터까지 저장하여 그 값을 다음 반복으로 넘길 수 있습니다. 다층 시프트 레지스터를 만들기 위하여, 시프트 레지스터의 왼쪽 터미널에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **원소 추가**를 선택합니다.

다중 시프트 레지스터는 루프의 왼쪽에만 나타납니다. 왜냐하면 오른쪽 터미널은 다음 그림에서처럼 단지 현재 루프 반복에서 생성된 데이터를 다음 루프 반복에 전달하는 역할만 하기 때문입니다.



이전 그림의 왼쪽 터미널에 다른 원소를 하나 더 추가하면, 최근 두 번의 반복 값을 다음 반복으로 전달할 수 있으며, 가장 최근 반복 값은 가장 위쪽 시프트 레지스터에 저장됩니다. 아래쪽 터미널은 더 이전 반복에서 전달된 데이터를 저장합니다.

## 피드백 노드

다음과 같이 ( 피드백 노드 ) 는 For 루프나 While 루프에서 노드나 노드 그룹의 출력을 그 노드나 노드 그룹의 입력으로 연결할 때 자동적으로 나타납니다.



또한 함수 팔레트의 ( 피드백 노드 ) 를 선택하여 For 루프 또는 While 루프의 안에 놓을 수 있습니다. ( 피드백 노드 ) 를 이용하여 루프를 가로지르는 긴 와이어링을 피합니다.

( 피드백 노드 ) 에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **초기화 터미널**을 선택하여 루프 경계에 초기화 터미널을 추가하여 루프를 초기화할 수 있습니다. 함수 팔레트에서 ( 피드백 노드 ) 를 선택하거나 초기화된 시프트 레지스터를 ( 피드백 노드 ) 로 바꾸었을 때, 루프는 초기화 터미널을 나타냅니다. ( 피드백 노드 ) 를 초기화하면 VI가 실행될 때 루프의 첫 반복에 ( 피드백 노드 ) 가 전달하는 초기값이 리셋됩니다. ( 피드백 노드 ) 를 초기화하지 않으면 ( 피드백 노드 ) 는 마지막으로 그 노드에 쓰여진 값을 전달하거나, 루프를 실행하지 않은 경우 그 데이터 타입의 기본값을 전달합니다. 초기화 터미널의 입력을 연결하지 않으면, VI가 실행될 때마다 ( 피드백 노드 ) 의 초기 입력은 이전 실행의 마지막 값입니다.

시프트 레지스터에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **피드백 노드로 대체하기**를 선택하여 시프트 레지스터를 ( 피드백 노드 ) 로 대체합니다 . 반대로 , ( 피드백 노드 ) 에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **시프트 레지스터로 대체하기**를 선택하고 ( 피드백 노드 ) 를 시프트 레지스터로 대체합니다 .

## 루프의 기본 데이터

시프트 레지스터가 초기화되어 있지 않을 때 While 루프는 기본 데이터를 생성합니다 .

For 루프의 카운트 터미널에 0 을 연결하거나 오토인덱싱이 활성화된 For 루프에 빈 배열을 입력으로 연결하는 경우 , For 루프는 기본 데이터를 생성합니다 . 루프는 실행되지 않고 오토인덱싱이 비활성화된 출력 터미널에는 이 터미널 데이터 타입의 기본값이 발생합니다 . 루프의 실행 여부와 관계없이 루프를 통해 값을 전달하기 위해 시프트 레지스터를 사용합니다 .

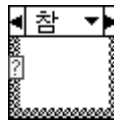
데이터 타입의 기본값에 대한 추가적인 정보는 *LabVIEW 도움말 카드*를 참조하십시오 .

## 케이스 , 시퀀스 , 이벤트 구조

케이스 , 다중 시퀀스 , 플랫 시퀀스 , 이벤트 구조에는 여러 개의 서브다이어그램이 있습니다 . 케이스 구조는 구조에 전달되는 입력 값에 따라 하나의 서브다이어그램을 실행합니다 . 다중 시퀀스 구조와 플랫 시퀀스 구조는 모든 서브다이어그램을 순차적인 순서에 따라 실행합니다 . 이벤트 구조는 사용자가 VI 와 상호작용하는 방법에 따라 서브다이어그램을 실행합니다 .

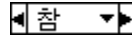
## 케이스 구조

다음과 같이 케이스 구조에는 2 개 이상의 서브다이어그램 또는 케이스가 있습니다 .



한번에 하나의 서브다이어그램만을 볼 수 있으며 , 구조는 한번에 하나의 케이스만 실행됩니다 . 입력값은 어느 서브다이어그램을 실행할 것인지 결정합니다 . 케이스 구조는 텍스트 기반 프로그래밍 언어의 switch 문이나 if...then...else 문과 유사합니다 .

다음과 같이 케이스 구조 위의 케이스 선택자 라벨에는 가운데에 그 케이스에 대응하는 선택자 값의 이름이 있고 양쪽에 증가 감소 화살표가 있습니다.



증가 및 감소 화살표를 클릭하여 사용 가능한 케이스를 스크롤합니다. 또한 케이스 이름 옆에 있는 아래 방향의 화살표를 클릭하여 풀다운 메뉴에서 케이스를 선택할 수 있습니다.

다음과 같이 입력값이나 선택자를 선택자 터미널에 연결하여 실행할 케이스를 결정합니다.



반드시 정수, 불리언 값, 문자열, 또는 열거형 타입 값을 선택자 터미널에 연결해야 합니다. 케이스 구조의 왼쪽 경계의 아무 곳이나 선택자 터미널을 둘 수 있습니다. 선택자 터미널의 데이터 타입이 불리언인 경우, 구조는 참 케이스와 거짓 케이스를 가집니다. 선택자 터미널이 정수, 문자열, 또는 열거형 타입 값인 경우, 구조는 어떤 개수의 케이스도 가질 수 있습니다.

범위를 벗어나는 값을 다루기 위하여 케이스 구조에 기본 케이스를 설정하십시오. 그렇지 않으면, 모든 가능한 입력값을 명확하게 열거해야 합니다. 예를 들면, 선택자가 정수이고 1, 2, 3의 케이스를 지정한다면, 입력값이 4나 그 외의 다른 특정하지 않은 정수값일 경우 실행하기 위해 기본 케이스를 지정해야 합니다.

## 케이스 선택자의 값과 데이터 타입

케이스 선택자 라벨에는 단일 값 또는 값의 리스트와 범위를 입력할 수 있습니다. 리스트에서,逗를 사용하여 값을 분리합니다. 숫자 범위에서, 범위를 10..20으로 지정합니다. 이는 10부터 20까지의 모든 숫자를 포함하는 것을 의미합니다. 또한 개방형 범위를 사용할 수 있습니다. 예를 들어, ..100은 100보다 작거나 같은 모든 숫자를 의미하며, 100..은 100보다 크거나 같은 숫자를 의미합니다. 또한, 리스트와 범위를 결합할 수 있는데 예를 들면, ..5, 6, 7..10, 12, 13, 14입니다. 같은 케이스 선택자 라벨에 겹치는 범위를 포함한 값을 입력하면, 케이스 구조는 그 라벨을 적절한 형태로 정리해줍니다. 앞의 예제는 ..10, 12..14로 정리됩니다. 문자열 범위의 경우, a..c의 범위에는 a와 b는 포함되지만 c는 포함되지 않습니다. a..c, c의 범위는 c의 끝 값을 포함합니다.

선택자 터미널에 연결된 객체와 다른 타입의 선택자 값을 입력하면, 값이 빨간색으로 표시되고 구조가 실행되기 전에 이 선택자 값을 삭제하거나 편집하지 않으면 VI를 실행할 수 없습니다. 또한, 부동소수는 반올림 문제가 발생할 수 있으므로, 부동소수 값을 케이스 선택자 값으로 사용할 수 없습니다. 부동소수 값을 케이스에 연결하면, LabVIEW는 가장 가까운 짝수 정수

로 반올림합니다. 케이스 선택자 라벨에 부동소수값을 입력하면, 값이 빨간색으로 표시되며 이는 구조를 실행하기 전에 이 값을 삭제하거나 편집해야 한다는 것을 나타냅니다.

## 입력과 출력 터널

케이스 구조에 여러 개의 입력과 출력 터널을 만들 수 있습니다. 입력은 모든 케이스에서 접근이 가능하지만, 모든 케이스가 각 입력을 사용할 필요는 없습니다. 반면에, 출력 터널은 모든 케이스에서 정의를 해주어야 합니다. 한 케이스에서 출력 터널을 만들면 나머지 케이스들의 경계에서도 동일한 위치에 터널이 나타납니다. 하나의 출력 터널이라도 연결되지 않으면, 구조의 모든 출력 터널이 흰색 사각형으로 나타납니다. 각 케이스에서 동일한 출력 터널에 다른 데이터 소스를 정의할 수 있지만 데이터 타입은 각 케이스에 대해 호환되어야 합니다. 출력 터널에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **연결되지 않으면 기본값 사용**을 선택하여 연결되지 않은 모든 터널의 터널 데이터 타입에 기본값을 사용합니다.

## 에러 핸들링을 위해 케이스 구조 사용하기

에러 클러스터를 케이스 구조의 선택자 터미널에 연결할 때, 케이스 선택자 라벨은 에러와 에러 없음의 두 케이스를 디스플레이하고, 케이스 구조의 경계는 색이 변경됩니다 — 에러의 경우에는 빨강, 에러 없음의 경우에는 녹색. 에러가 발생하는 경우, 케이스 구조는 에러 서브다이아그램을 실행합니다.

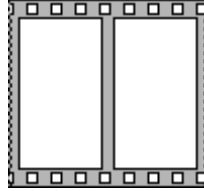
에러 처리에 대한 추가적인 정보는 6 장, *VI 실행하고 디버깅하기의 에러 핸들링하기* 섹션을 참조하십시오.

## 시퀀스 구조

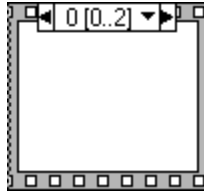
시퀀스 구조에는 순차적으로 실행되는 하나 또는 그 이상의 서브다이아그램 또는 프레임이 포함됩니다. 시퀀스 구조의 각 프레임 안에서 블록다이아그램의 다른 부분처럼, 데이터 의존성이 노드의 실행 순서를 결정합니다. 시퀀스 구조는 LabVIEW 에서 일반적으로 사용되는 구조가 아닙니다.

시퀀스 구조에는 플랫폼 시퀀스 구조와 다층 시퀀스 구조의 두가지 타입이 있습니다.

다음과 같이 플랫폼 시퀀스 구조는 한번에 모든 프레임을 디스플레이하고 왼쪽에서 오른쪽으로 프레임을 실행하며 프레임에 연결된 모든 데이터 값을 이용할 수 있고 마지막 프레임까지 실행됩니다. 프레임이 실행을 종료할 때 데이터 값은 각 프레임을 떠납니다.



다음과 같이 다층 시퀀스 구조는 각 프레임을 쌓아서 한 번에 하나의 프레임을 보여주며, 0 번 프레임, 1 번 프레임의 순으로 마지막 프레임까지 실행됩니다.



LabVIEW의 병렬 수행 능력을 충분히 활용하려면 시퀀스 구조를 지나치게 사용하지 마십시오. 시퀀스 구조는 실행의 순서를 강요하고 병렬 작업을 불가능하게 만듭니다. 예를 들어, PXI, GPIB, 시리얼 포트 및 DAQ 디바이스와 같은 I/O 장치를 사용하는 비동기 작업은 시퀀스 구조가 금지하지 않는다면 다른 작업과 동시에 실행될 수 있습니다.

실행 순서를 컨트롤 할 때는 노드 사이의 데이터 의존성을 이용하는 방법을 고려하십시오. 예를 들면, 예러 I/O와 같은 흐름 파라미터를 사용하여 실행 순서를 컨트롤할 수 있습니다.

예러 I/O에 대한 추가적인 정보는 6 장, VI 실행하고 디버깅하기의 예러 핸들링하기 섹션을 참조하십시오. 흐름 파라미터에 대한 추가적인 정보는 5 장, 블록 다이어그램 만들기의 흐름 파라미터 섹션을 참조하십시오.

## 이벤트 구조

다음과 같이 이벤트 구조는 하나 또는 그 이상의 서브다이아그램 또는 이벤트 케이스를 가지며, 이 구조가 실행되면 그 중 하나만이 실행됩니다.



이벤트 구조는 이벤트가 일어날 때까지 기다린 후, 그 이벤트를 처리할 적절한 케이스를 실행합니다. 이벤트는 사용자 인터페이스, 외부 I/O, 또는 어플리케이션의 다른 부분에서 발생할 수 있습니다. 사용자 인터페이스 이벤트에는 마우스 클릭, 키보드 입력 등이 있습니다. 외부 I/O 이벤트에는 데이터 수집이 완료되거나 예러 조건이 발생했을 때 신호를 주는 하드웨어 타이머 또는 트리거가 있습니다. 다른 타입의 이벤트를 프로그램적으로 생성할 수 있으며, 이를 사용하여 어플리케이션의 다른 부분과 통신할 수 있습니다. LabVIEW 는 사용자 인터페이스와 프로그램적으로 생성되는 이벤트는 지원하지만 외부 I/O 이벤트는 지원하지 않습니다.



### 노트

이벤트 구조는 LabVIEW Full 과 Professional Development Systems 에서만 제공됩니다. LabVIEW Base Package 에서도 이벤트 드리는 프로그래밍 기능을 가진 VI 를 실행할 수 있지만, 이벤트 처리 구성요소를 다시 설정할 수는 없습니다.

# 문자열, 배열, 클러스터를 이용한 데이터의 그룹화

데이터를 그룹화하기 위해 문자열, 배열, 클러스터를 사용합니다. 문자열은 ASCII 문자의 나열을 그룹화합니다. 배열은 같은 타입의 데이터 원소를 그룹화합니다. 클러스터는 혼합된 타입의 데이터 원소를 그룹화합니다.

## 데이터를 문자열로 그룹화하기

문자열은 디스플레이할 수 있거나 할 수 없는 ASCII 문자의 나열입니다. 문자열은 정보와 데이터에 대하여 플랫폼 독립적인 포맷을 제공합니다. 좀 더 일반적인 문자열의 어플리케이션은 다음을 포함합니다:

- 단순 문자 메시지 생성하기.
- 숫자 데이터를 문자열로 바꾸어 인스트루먼트에 전달한 후 문자열을 숫자값으로 변환하기.
- 숫자 데이터를 디스크에 저장하기. ASCII 파일안에 숫자 데이터를 저장하려면, 디스크 파일에 데이터를 쓰기 전에 반드시 숫자 데이터를 문자열로 변환해 주어야 합니다.
- 대화 상자로 사용자에게 지시 또는 요청하기.

프런트패널에서 문자열은 테이블, 문자 입력 박스, 그리고 라벨로 나타납니다. LabVIEW에는 문자열을 포맷하고 분석하고 편집하는 것을 포함하여 문자열을 다루는 내장된 VI와 함수가 있습니다.

## 프런트패널의 문자열

문자열 컨트롤과 인디케이터를 사용하여 텍스트 입력 박스와 라벨을 시뮬레이션합니다.

문자열 컨트롤과 인디케이터에 대한 추가 정보는 4장, *프런트패널 만들기*의 *문자열 컨트롤과 인디케이터* 섹션을 참조하십시오.

## 문자열 디스플레이 타입

프린트패널의 문자열 컨트롤과 인디케이터에서 마우스 오른쪽 버튼을 클릭하여 다음 테이블에서 보이는 디스플레이 타입을 선택할 수 있습니다. 또한 테이블에는 각 디스플레이 타입에 대한 예제 메시지도 있습니다.

디스플레이 타입	설명	메시지
일반 디스플레이	컨트롤의 폰트를 사용하여 인쇄 가능한 문자를 디스플레이합니다. 디스플레이할 수 없는 문자는 일반적으로 상자로 나타냅니다.	네가지 디스플레이 타입이 있습니다. \는 백슬래시입니다.
'\ ' 코드 디스플레이	모든 디스플레이 불가능 문자는 백슬래시 코드로 표기합니다.	네가지 \s 표기 \s 타입이 \s 있습니다. \n\\s 는 \s 백슬래시입니다.
양호 디스플레이	스페이스를 포함한 각 문자를 별모양 (*)으로 표기합니다.	***** *****
16 진수 디스플레이	문자 대신에 각 문자의 ASCII 값을 16 진수로 표기합니다.	B3D7 B0A1 C1F6 20C7 A5B1 E220 C5B8 C0D4 C0CC 20C0 D6BD C0B4 CFB4 D92E 0A5C B4C2 20B9 E9B5 A5BD C3C0 D4B4 CFB4 D92E

## 테이블

테이블 컨트롤을 사용하여 프린트패널에 테이블을 생성합니다. 테이블의 각 셀이 문자열이고 각 셀은 열과 행에 상주합니다. 그러므로 테이블은 문자열의 2D 배열의 디스플레이입니다.

배열에 대한 추가 정보는 이 장의 *배열* 섹션을 참조하십시오.

## 문자열 편집, 포맷, 분석하기

문자열 함수를 사용하여 다음과 같은 방법으로 문자열을 편집합니다 :

- 문자열에서 문자나 문자열의 일부를 찾거나, 불러오고, 대체합니다.
- 문자열의 모든 문자를 대문자나 소문자로 바꿉니다.
- 문자열에서 일치하는 패턴을 찾고 가져옵니다.
- 문자열에서 한 개의 라인을 가져옵니다.
- 문자열에서 문자를 회전시키거나 뒤집습니다.
- 두 개 또는 그 이상의 문자열을 합칩니다.
- 문자열에서 문자를 삭제합니다.

문자열을 프로그램적으로 편집할 때 메모리 사용을 최소화하는데 대한 추가적인 정보는 *LabVIEW 도움말*의 *LabVIEW Style Checklist*를 참고하십시오. 문자열을 편집하기 위해 문자열 함수를 사용하는 것에 대한 예제는 `labview\examples\general\strings.llb`를 참조하십시오.

## 문자열 포맷과 분석

다른 VI, 함수, 또는 어플리케이션에서 데이터를 사용하기 위해서 종종 데이터를 문자열로 바꾸고 문자열을 VI, 함수, 또는 어플리케이션에서 읽을 수 있는 포맷으로 바꾸어야 할 때가 있습니다. 예를 들어, Microsoft Excel 은 문자열에 탭, 콤마, 또는 스페이스와 같은 구분 문자가 있어야 됩니다. Excel 은 이 구분 문자를 사용하여 숫자나 단어를 셀로 구분합니다.

예를 들어, 숫자값의 1 차원 배열을 (2 진 파일 쓰기) 함수를 사용하여 스프레드시트에 쓰려면, 배열을 문자열 포맷으로 바꾸고 탭과 같은 구분 문자로 각 숫자를 분리해 주어야 됩니다. (스프레드시트 파일에 쓰기) VI 를 이용하여 숫자값 배열을 스프레드시트에 쓰려면, ( 배열을 스프레드시트 문자열로 ) 함수를 이용하여 배열을 포맷하고 포맷과 구분 문자도 지정해 주어야 됩니다.

문자열 함수를 이용하여 다음과 같은 태스크를 수행합니다 :

- 문자열로부터 문자열의 부분을 빼냅니다.
- 데이터를 문자열로 바꿉니다.
- 워드 프로세싱 어플리케이션 또는 스프레드시트 어플리케이션에 사용할 수 있도록 문자열을 포맷합니다.

파일 I/O VI 와 함수를 사용하여 문자열을 텍스트나 스프레드시트 파일로 저장합니다.

### 포맷 지정자

대부분의 경우, 문자열의 포맷을 위하여 ( 문자열 ) 함수의 **포맷 문자열** 파라미터에 하나 또는 그 이상의 포맷 지정자를 입력해야 합니다. 포맷 지정자는 어떻게 숫자 데이터에서 문자열로 혹은 그 반대로 바꿀 것인가를 지정하는 코드입니다. LabVIEW 는 변환 코드를 이용하여 파라미터의 문자 포맷을 지정합니다. 예를 들어, 포맷 지정자 %x 는 16 진수 정수를 문자열로 혹은 그 반대로 바꿉니다.

## 배열과 클러스터를 이용하여 데이터 그룹화하기

배열과 클러스터 컨트롤 그리고 함수를 이용하여 데이터를 그룹화합니다. 배열은 같은 타입의 데이터 원소를 그룹화합니다. 클러스터는 혼합된 타입의 데이터 원소를 그룹화합니다.

### 배열

배열은 원소와 차원으로 구성됩니다. 원소는 배열을 구성하는 데이터입니다. 차원은 배열의 길이, 높이, 또는 폭입니다. 배열은 하나 또는 그 이상의

차원을 가질 수 있으며 메모리가 허용하는 한 차원마다 ( $2^{31}$ ) - 1 원소를 가집니다.

숫자, 불리언, 경로, 문자열, 웨이브폼, 클러스터 데이터 타입의 배열을 만들 수 있습니다. 비슷한 데이터의 모음으로 작업을 하거나 반복된 연산을 수행할 때는 배열을 사용하는 것을 고려합니다. 배열은 웨이브폼에서 수집한 데이터나 루프에서 생성된 데이터를 저장하는데 이상적입니다. 이 때 루프의 각 반복은 배열의 한 원소를 생성합니다.

## 제약

배열의 배열은 만들 수 없습니다. 그렇지만, 여러 차원의 배열을 사용하거나 각각의 클러스터가 하나 또는 그 이상의 배열을 포함하는 클러스터의 배열을 생성할 수 있습니다. 또한, 서브패널 컨트롤, 탭 컨트롤, .NET 컨트롤, ActiveX 컨트롤, 차트, 여러 플롯 XY 그래프의 배열도 생성할 수 없습니다.

클러스터에 대한 자세한 내용은 본 장의 *클러스터* 섹션을 참조하십시오.

## 인덱스

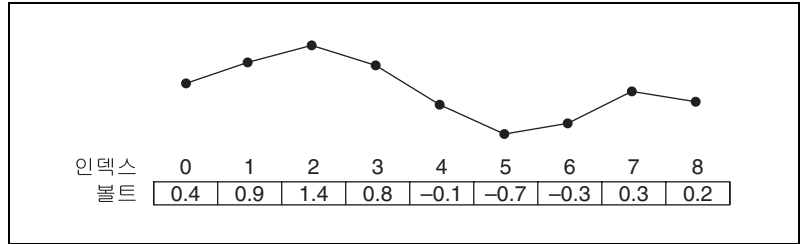
특정 원소를 배열에 위치시키기 위해서는 차원당 한 개의 인덱스가 필요합니다. LabVIEW 에서 인덱스는 배열을 탐색하고 블록 다이어그램의 배열에서 원소나, 행, 열, 그리고 페이지를 불러오기 위해 사용됩니다.

## 배열의 예

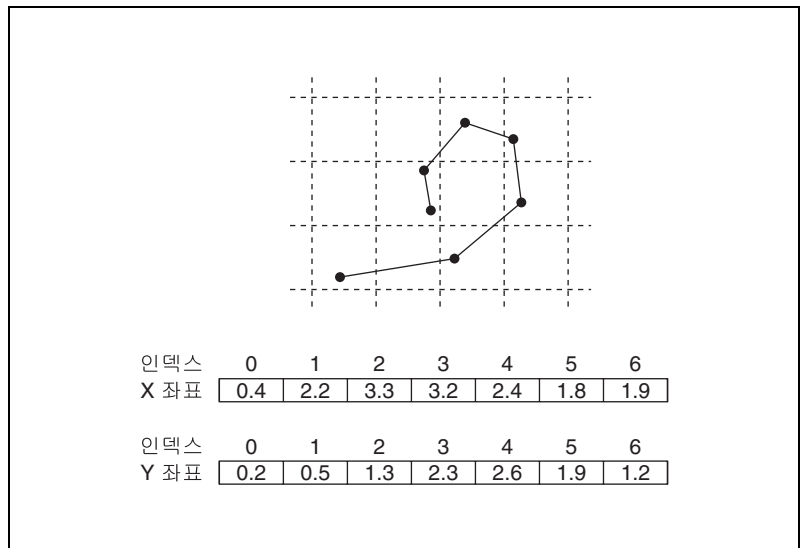
태양계의 9 개 행성을 나열한 문자 배열이 간단한 배열의 예입니다. LabVIEW 는 아홉 개의 원소를 가진 1D 문자열 배열로 이것을 나타냅니다.

배열의 원소에는 순서가 있습니다. 배열은 사용자가 특정 원소에 접근할 수 있게 하기 위하여 인덱스를 사용합니다. 인덱스는 0 에서 시작합니다. 이것은  $n$  개의 원소를 가진 배열에 대하여 0 에서  $n-1$  로 인덱스됨을 의미합니다. 예를 들어,  $n=9$  인 아홉 개의 행성에 대하여 인덱스는 0 에서 8 이 됩니다. 지구는 세번째 행성이므로 인덱스는 2 입니다.

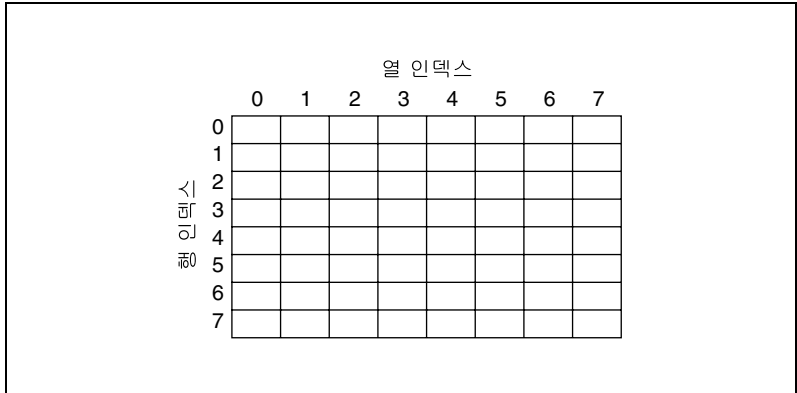
배열에 대한 또 다른 예는 다음 그림과 같이 연속적인 시간 간격에서 각 연속되는 원소가 연속되는 시간 간격에서의 볼트값인 숫자 배열로 표현되는 웨이브폼입니다.



좀더 복잡한 배열에 대한 예제는 다음 그림과 같이 포인트 배열로 나타내는 그래프입니다. 여기서 각 포인트는 X와 Y 좌표를 나타내는 숫자값의 쌍을 포함하는 클러스터입니다.

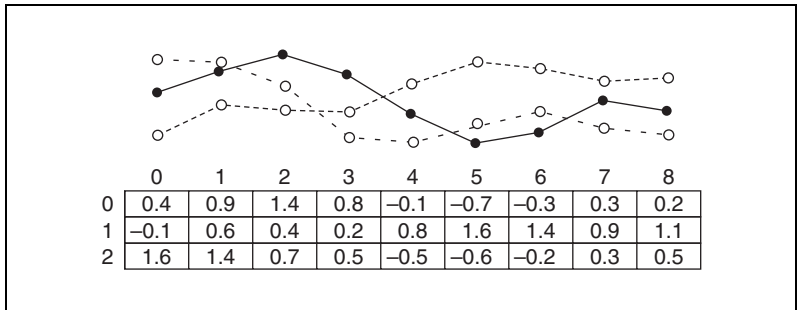


이전 예제는 1D 배열을 사용합니다. 2D 배열은 원소를 눈금으로 저장합니다. 원소를 찾으려면 행 인덱스와 열 인덱스가 필요하며, 이 인덱스들은 0을 기준으로 시작합니다. 다음 그림은  $8 \times 8 = 64$  원소를 가진 2D 배열의 8행과 8열을 보여줍니다.



예를 들어, 체스판은 8행 8열이며 총 64개 자리가 있습니다. 각 자리는 비어있거나 한 개의 체스말이 위치할 수 있습니다. 체스판은 2D 문자열 배열로 나타낼 수 있습니다. 각 문자열은 체스판의 대응하는 위치에 있는 체스말의 이름이며, 빈 곳은 빈 문자열로 나타냅니다.

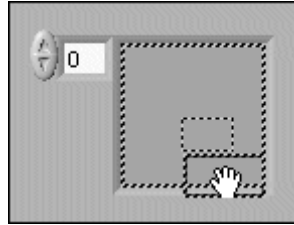
배열에 행을 추가하여 이전 1D 배열 예제를 2차원으로 일반화할 수 있습니다. 다음 그림은 2차원 숫자값 배열로 표현되는 웨이브폼들의 집합을 보여줍니다. 행 인덱스는 웨이브폼을 선택하고, 열 인덱스는 그 웨이브폼의 포인트를 선택합니다.



배열을 사용한 예제는 labview\examples\general\arrays.llb를 참조하십시오.

## 배열 컨트롤, 인디케이터, 상수 생성하기

다음 그림과 같이 프런트패널에 배열 셀을 놓고 숫자, 불리언, 문자열, 경로, 참조 번호 또는 클러스터 컨트롤 등의 데이터 객체 또는 원소를 배열 셀에 끌어다 놓아서 프런트패널에 배열 컨트롤 또는 인디케이터를 생성합니다.



배열 셀은 새 객체에 맞도록 자동으로 크기가 조절됩니다.

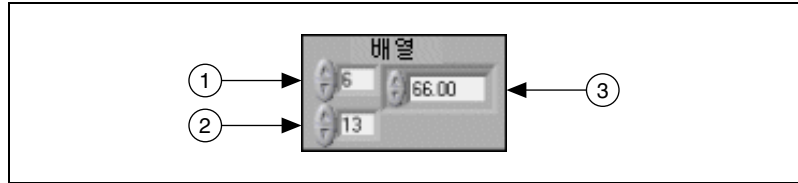
블록다이어그램에서 배열 상수를 생성하려면, **함수** 팔레트에서 배열 상수를 선택하고, 블록다이어그램에 배열 셀을 놓은 후, 문자열 상수, 숫자 상수, 또는 클러스터 상수를 배열 셀 안에 놓습니다. 배열 상수는 다른 배열과 비교를 하기 위한 기준으로 또는 상수 데이터를 저장하기 위해 사용할 수 있습니다.

## 여러 차원 배열 생성하기

프런트패널에서 여러 차원 배열을 생성하기 위해서, 인덱스 디스플레이를 마우스 오른쪽 버튼으로 클릭한 후 바로 가기 메뉴에서 **차원 추가**를 선택합니다. 또한, 인덱스 디스플레이를 크기 조정하여 원하는 여러 차원을 만들 수 있습니다. 한 번에 한 개의 차원을 삭제하려면, 인덱스 디스플레이를 마우스 오른쪽 버튼으로 클릭한 후 바로 가기 메뉴에서 **차원 제거**를 선택합니다. 또한, 인덱스 디스플레이를 크기 조정하여 차원을 제거할 수도 있습니다.

특정 원소를 프런트패널에 디스플레이하기 위하여, 인덱스 디스플레이에 인덱스 숫자를 입력하거나 또는 인덱스 디스플레이의 화살표를 이용하여 해당 숫자를 탐색합니다.

예를 들면 2D 배열은 행과 열을 가지고 있습니다. 다음 그림에서 보면, 왼쪽에 있는 두 개 박스의 위쪽 디스플레이는 행 인덱스이고 아래쪽 디스플레이는 열 인덱스입니다. 행과 열의 표시를 합하여 지정된 위치의 값을 보여줍니다. 다음 그림은 6 행, 13 열 값이 66임을 보여줍니다.



1 행 인덱스                      2 열 인덱스                      3 행과 열 위치에서의 값

행과 열은 0을 기준으로 하며, 첫번째 행이 0번 행, 두번째 행이 1번 행이 됩니다. 다음 배열에서 인덱스 디스플레이를 1행, 2열로 변경하면 6의 값이 나타납니다.

0	1	2	3
4	5	6	7
8	9	10	11

배열 차원의 범위를 벗어난 영역의 행과 열을 보이려고 하면, 그곳에는 정의된 값이 없다는 것을 나타내기 위하여, 배열 컨트롤이 회색화되어 나타나며 LabVIEW는 해당하는 데이터 타입의 기본값을 보여줍니다. 데이터 타입의 기본값은 배열의 데이터 타입에 의존합니다.

위치 도구를 사용하여 한번에 하나 이상의 행과 열이 보이도록 배열의 크기 조정을 합니다.

## 배열 함수

배열 함수를 사용하여 배열을 생성하고 조작합니다. 예를 들어, 다음 그림과 같은 태스크를 수행할 수 있습니다:

- 배열에서 개별 데이터 원소를 빼냅니다.
- 배열에서 데이터 원소를 삽입, 삭제, 또는 대체합니다.
- 배열을 분리합니다.

( 배열 만들기 ) 함수를 사용하여 배열을 프로그램적으로 만듭니다. 또한, 배열을 만들기 위해 루프를 사용할 수 있습니다.

배열을 만들기 위한 루프 사용에 대한 추가적인 정보는 8 장, *루프와 구조의 루프를 이용한 배열 만들기* 섹션을 참조합니다.

루프 안에서 배열 함수를 사용할 때 메모리 사용을 최소화하는데 대한 추가적인 정보는 *LabVIEW 도움말*의 *LabVIEW Style Checklist*를 참조하십시오.

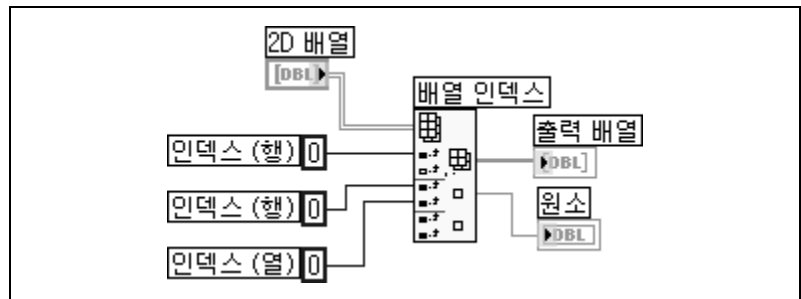
## 배열 함수의 자동 크기 조절

( 배열 인덱스 ), ( 배열 부분 대체 ), ( 배열에 삽입 ), ( 배열로부터 삭제 ), ( 배열 부분 ) 함수는 자동적으로 연결한 입력 배열의 차원에 맞도록 크기 조정합니다. 예를 들면, 1D 배열을 이들 함수에 연결하면 함수는 단일 인덱스 입력만 나타냅니다. 만약 2D 배열을 같은 함수에 연결하면 두 개의 인덱스 입력을 보여줍니다. 하나는 행, 다른 하나는 열에 대한 입력을 나타냅니다.

위치 도구를 사용하여 함수의 크기를 수동으로 조절하여 이들 함수에서 하나 이상의 원소나 부분 배열 ( 행, 열, 또는 페이지 ) 에 접근할 수 있습니다. 이들 함수 중 하나를 확장할 때, 함수에 연결된 배열의 차원에 따라 늘어나는 폭이 결정됩니다. 1D 배열을 이들 함수에 연결한 경우, 함수는 한 개의 인덱스 입력씩 확장할 수 있습니다. 만약 2D 배열을 같은 함수에 연결한 경우, 함수는 행과 열에서 각각 하나씩인 두 개의 인덱스 입력을 확장할 수 있습니다.

연결한 인덱스 입력이 접근하거나 수정하려고 하는 부분 배열의 형태를 결정합니다. 예를 들면, ( 배열 인덱스 ) 함수의 입력이 2D 배열이고 행 입력에만 연결했을 경우, 배열에서 1D 행을 추출합니다. 열 입력에만 연결했을 경우, 배열에서 1D 열을 추출합니다. 행 입력과 열 입력에 모두 연결했을 경우, 배열에서 한 개의 원소를 추출합니다. 각 입력 그룹은 독립적이며 배열의 모든 차원의 모든 부분에 접근할 수 있습니다.

다음 그림의 블록다이어그램은 ( 배열 인덱스 ) 함수를 이용하여 2 차원 배열에서 한 개의 행과 한 개의 원소를 추출합니다.



배열에서 여러 연속되는 값에 접근하기 위하여 ( 배열 인덱스 ) 함수를 확장합니다. 그러나 각 증분의 인덱스 입력에 값을 연결하지 않습니다. 예를 들어, 2D 배열에서 첫번째, 두번째, 그리고 세번째 행을 뽑아내기 위해서, ( 배열 인덱스 ) 함수를 3 개로 확장한 다음 1D 배열 인디케이터를 각 부분 배열 출력에 연결합니다.

## 배열의 기본 데이터

배열의 범위를 벗어나는 인덱스 선택은 배열 원소 파라미터에 기본값을 생성합니다. ( 배열 크기 ) 함수를 사용하여 배열의 크기를 결정할 수 있습니다.

While 루프를 사용하여 마지막 원소를 넘어 배열을 인덱스하거나, ( 배열 인덱스 ) 함수의 **인덱스** 입력에 너무 큰 값을 입력하거나, ( 배열 인덱스 ) 함수에 빈 배열을 공급하면, 의도치 않게 배열의 경계를 넘어 인덱스할 수 있습니다.

인덱싱에 대한 추가적인 정보는 8 장, *루프와 구조의 오토인덱싱 루프* 섹션을 참조합니다. 데이터 타입의 기본값에 대한 추가적인 정보는 *LabVIEW 도움말 카드*를 참조하십시오.

## 클러스터

클러스터는 혼합된 타입의 데이터 원소를 그룹화합니다. 클러스터의 예는 불리언 값, 숫자값, 문자열이 결합된 LabVIEW 에러 클러스터를 들 수 있습니다. 클러스터는 텍스트 기반의 프로그래밍 언어의 레코드나 구조체와 유사합니다.

에러 클러스터 사용에 대한 추가적인 정보는 6 장, *VI 실행하고 디버깅하기*의 *에러 클러스터* 섹션을 참조하십시오.

여러 데이터 원소를 클러스터로 묶으면, 블록 다이어그램에서 와이어의 복잡한 연결을 피할 수 있으며 SubVI 에 필요한 커넥터 팬 터미널의 수를 감소시킬 수 있습니다. 커넥터 팬은 최대 28 개의 터미널을 가질 수 있습니다. 프런트패널에 다른 VI 로 전달하려는 컨트롤과 인디케이터가 28 개 이상 있는 경우, 이중 일부를 하나의 클러스터로 그룹화하고, 이 클러스터를 커넥터 팬의 한 터미널에 할당합니다.

블록 다이어그램에서 대부분의 클러스터는 핑크색의 와이어 패턴과 데이터 타입 터미널을 가집니다. 숫자값의 클러스터가 때때로 데이터 포인트를 나타내는 경우, 갈색의 와이어 패턴과 데이터 타입 터미널을 가집니다. 갈색의 숫자 클러스터를 숫자 함수, 예를 들어 ( 더하기 ) 나 ( 제공근 ) 에 연결하여 클러스터 원소 모두에 같은 연산을 동시에 수행하게 할 수 있습니다.

## 클러스터 원소의 순서

클러스터와 배열 원소가 둘 다 요청되었다면, 반드시 한번에 모든 클러스터 원소를 풀거나 ( 이름으로 풀기 ) 함수를 이용하여 특정 클러스터 원소에 접근합니다. 또한, 클러스터는 고정된 크기를 가진다는 점에서 배열과 차이가 납니다. 배열과 마찬가지로 클러스터는 컨트롤이거나 인디케이터입니다. 클러스터는 컨트롤과 인디케이터를 함께 가질 수는 없습니다.

클러스터 원소는 셀에서의 그 위치에 관계 없이 논리적인 순서를 가지고 있습니다. 클러스터에 놓는 첫번째 객체는 원소 0 이고 두번째는 원소 1 이 되며, 이와 같은 형태로 진행됩니다. 원소를 삭제하면 순서가 자동으로 조절됩니다. 클러스터 순서는 블록 다이어그램의 (육기) 및 (폴기) 함수에서 원소가 터미널로 나타나는 순서를 결정합니다. 클러스터 경계에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **클러스터 내의 컨트롤 순서 재설정**을 선택하여 클러스터 순서를 보고 수정할 수 있습니다.

두 개의 클러스터를 연결하려면 반드시 양쪽이 같은 수의 원소를 가지고 있어야 합니다. 클러스터 순서로 결정되는 대응 원소는 반드시 호환되는 데이터 타입을 가져야 합니다. 예를 들어, 한 클러스터의 배열도 부동소수점이 다른 클러스터의 문자열에 대응되면, 블록 다이어그램의 와이어는 깨지고 VI 는 실행되지 않습니다. 숫자값이 다른 형을 가진다면 LabVIEW 는 같은 형으로 강제 변환합니다.

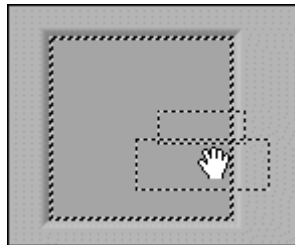
## 클러스터 함수

클러스터 함수를 이용하여 클러스터를 만들고 조작합니다. 예를 들어, 다음 그림과 같은 태스크를 수행할 수 있습니다:

- 클러스터에서 개별 데이터 원소를 빼냅니다.
- 클러스터에 개별 데이터 원소를 추가합니다.
- 클러스터를 개별 데이터 원소로 나눕니다.

## 클러스터 컨트롤, 인디케이터, 상수 생성하기

다음 그림과 같이 프런트패널에 클러스터 셀을 놓고 숫자, 불리언, 문자열, 경로, 참조 번호, 배열, 클러스터 컨트롤 또는 인디케이터와 같은 데이터 객체 또는 원소를 클러스터 셀 안에 끌어다 놓아 클러스터 컨트롤 또는 인디케이터를 생성합니다.



블록 다이어그램에서 클러스터 상수를 만들기 위하여, **함수** 팔레트에서 클러스터 상수를 선택하고, 블록 다이어그램에 클러스터 셀을 위치시킨 후, 문자열 상수, 숫자 상수, 또는 클러스터 상수를 클러스터 셀 안에 놓습니다. 클러스터 상수는 다른 클러스터와 비교를 하기 위한 기준으로 또는 상수 데이터를 저장하기 위해 사용할 수 있습니다.

## 그래프와 차트

데이터를 수집 또는 생성한 후에 그래픽 방식으로 데이터를 디스플레이하기 위해 그래프 또는 차트를 사용합니다.

그래프와 차트는 데이터를 디스플레이하고 업데이트하는 방법에서 차이가 납니다. 그래프를 가진 VI는 일반적으로 배열 안에 데이터를 모으고 그래프에 데이터를 플롯합니다. 이 과정은 데이터를 먼저 저장한 후 데이터의 플롯을 생성하는 스프레드시트와 비슷합니다. 데이터가 플롯되면, 그래프는 이전 데이터를 버리고 새로운 데이터만 디스플레이합니다. 그래프는 일반적으로 연속적으로 데이터를 수집하는 빠른 과정에 사용합니다.

그와 대조적으로, 차트는 새 데이터 포인트를 이미 그려진 데이터 뒤에 덧붙여서 디스플레이하고 히스토리를 생성합니다. 차트에서는, 현재의 읽기 또는 측정값 이전에 수집된 데이터와 함께 볼 수 있습니다. 차트에서 디스플레이할 수 있는 것보다 더 많은 데이터 포인트가 추가되면, 차트는 스크롤하여 차트의 왼쪽으로 이전 포인트가 이동하는 동안 차트의 오른쪽에 새 포인트를 추가합니다. 일반적으로 차트는 초당 아주 적은 수의 데이터 포인트만이 플롯에 추가되는 느린 과정에 사용합니다.

## 그래프와 차트의 종류

LabVIEW는 다음 타입의 그래프와 차트를 가집니다:

- **웨이브폼 차트와 그래프** — 일정한 속도로 수집된 데이터를 디스플레이합니다.
- **XY 그래프** — 일정하지 않은 속도로 얻은 데이터와 여러 값을 가진 함수를 디스플레이합니다.
- **강도 차트와 그래프** — 3 차원 값을 디스플레이하기 위해 색을 이용하여 2D 플롯에서 3D 데이터를 디스플레이합니다.
- **디지털 웨이브폼 그래프** — 펄스나 디지털 라인의 그룹으로 데이터를 디스플레이합니다.
- **(Windows) 3 차원 그래프** — 프런트패널 ActiveX 객체의 3 차원 플롯에 데이터를 디스플레이합니다.

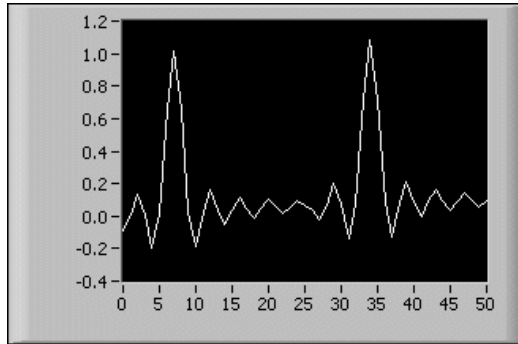
그래프와 차트에 대한 예제는 `labview\examples\general\graphs` 디렉토리를 참조하십시오.

## 웨이브폼 그래프와 차트

LabVIEW 는 일정한 속도로 수집되는 데이터를 디스플레이하기 위한 웨이브폼 그래프와 차트를 가집니다 .

### 웨이브폼 그래프

웨이브폼 그래프는 균일하게 샘플된 측정의 하나 또는 여러 플롯을 디스플레이합니다 . 웨이브폼 그래프는  $y = f(x)$  와 같이 x 축을 따라 포인트가 균일하게 분포하는 단일 값 함수만을 플롯합니다 . 다음 그림은 웨이브폼 그래프의 예제를 보여줍니다 .



웨이브폼 그래프는 임의 개수의 포인트를 가진 플롯을 디스플레이 할 수 있습니다 . 그래프는 데이터 호환성의 불편을 최소화하도록 여러 데이터 타입을 수용합니다 .



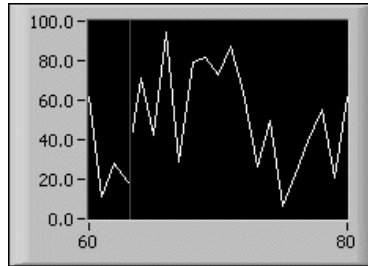
#### 노트

디지털 데이터를 디스플레이하기 위하여 디지털 웨이브폼 그래프를 사용합니다 . 디지털 웨이브폼 그래프와 그래프가 수용할 수 있는 데이터 타입에 대한 추가적인 정보는 본 장의 *디지털 웨이브폼 그래프* 섹션을 참조합니다 .

웨이브폼 그래프가 받는 데이터 타입의 예제는 `labview\examples\general\graphs\gengraph.llb` 에서 **Waveform Graph VI** 를 참조하십시오 .

## 웨이브폼 차트

웨이브폼 차트는 일반적으로 같은 속도로 수집된 하나 또는 그 이상의 플롯의 데이터를 디스플레이하는 숫자 인디케이터의 특별한 타입입니다. 다음 그림은 웨이브폼 차트의 예제를 보여줍니다.



웨이브폼 차트는 이전 업데이트에서 데이터 히스토리, 또는 버퍼를 유지합니다. 웨이브폼 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **차트 히스토리 길이**를 선택하여 버퍼를 설정합니다. 웨이브폼 차트의 기본 차트 히스토리 길이는 1,024 데이터 포인트입니다. 차트에 데이터를 보내는 주기가 차트를 얼마나 자주 업데이트하는지를 결정합니다.

웨이브폼 차트에 대한 예제는 `labview\examples\general\graphs\charts.llb`를 참조하십시오.

## 웨이브폼 데이터 타입

웨이브폼 데이터 타입은 웨이브폼의 데이터, 시작 시간, 델타  $f$ 를 전달합니다. 웨이브폼은 (웨이브폼 만들기) 함수로 생성할 수 있습니다. 웨이브폼을 수집 또는 분석하는 대부분의 VI와 함수는 기본적으로 웨이브폼 데이터를 받고 반환합니다. 웨이브폼 데이터를 웨이브폼 그래프 또는 웨이브폼 차트에 연결하면, 그래프나 차트는 자동적으로 웨이브폼의 데이터, 시작 시간, 델타  $x$ 를 기반으로 웨이브폼을 플롯합니다. 웨이브폼 데이터의 배열을 웨이브폼 그래프나 차트에 연결하면, 그래프와 차트는 자동으로 모든 웨이브폼을 플롯합니다.

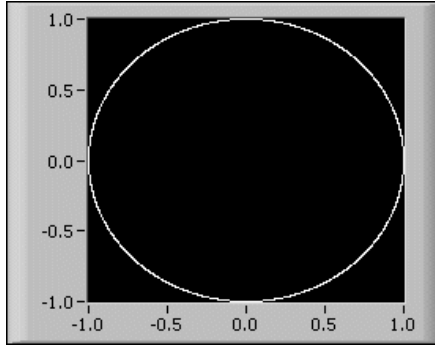
디지털 웨이브폼 데이터 타입에 대한 추가적인 정보는 본 장의 *디지털 웨이브폼 데이터 타입* 섹션을 참조합니다.

## XY 그래프

XY 그래프는 구형이나 가변 타임 베이스를 갖는 웨이브폼과 같은, 여러값의 함수를 플롯하는 일반적인 목적을 가진 직각좌표 그래픽 객체입니다. XY 그래프는 샘플링 속도의 균일도에 상관없이 모든 포인트 세트를 표시합니다.

또한, XY 그래프에서 나이퀴스트 평면, 니콜스 평면, S 평면, Z 평면을 디스플레이할 수 있습니다. 이들 평면의 라인과 라벨은 직각좌표 라인과 같은 색이며 선도 라벨 폰트를 수정할 수 없습니다.

다음 그림은 XY 그래프의 예제를 보여줍니다.

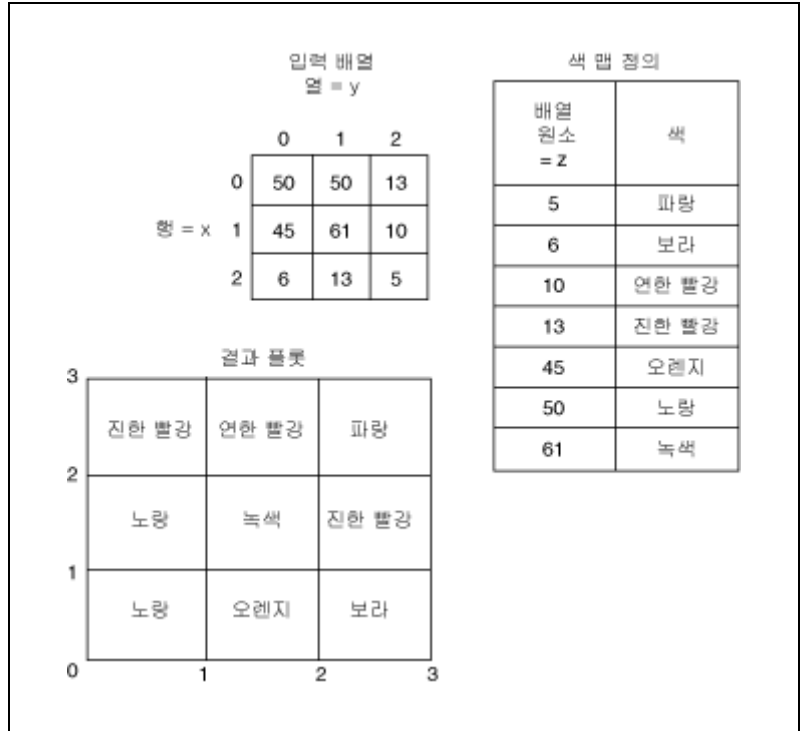


XY 그래프는 임의 개수의 포인트를 가진 플롯을 디스플레이할 수 있습니다. XY 그래프는 데이터 호환성의 불편을 최소화하도록 여러 데이터 타입을 수용합니다.

XY 그래프에 대한 예제는 `labview\examples\general\graphs\gengraph.llb`의 XY Graph VI를 참조하십시오.

## 강도 그래프와 차트

강도 그래프와 차트를 사용하여 직각 평면에 색 블록을 놓아서 3D 데이터를 2D 플롯에서 디스플레이합니다. 예를 들어, 강도 그래프와 차트를 이용하여, 크기가 고도를 나타내는 온도 패턴과 지형과 같은 패턴된 데이터를 디스플레이할 수 있습니다. 강도 그래프와 차트는 숫자의 3D 배열을 수용합니다. 배열에서 각 숫자는 특정 색을 나타냅니다. 2D 배열 원소의 인덱스는 색에서 플롯 위치를 설정합니다. 다음 그림은 강도 차트의 수행 개념을 나타냅니다.



데이터의 행은 그래프 또는 차트의 새로운 열로써 디스플레이에 전달됩니다. 행을 디스플레이 상에서 행으로 나타내려면 그래프 또는 차트에 2D 배열 데이터 타입을 연결하고, 그래프 또는 차트를 마우스 오른쪽 버튼으로 클릭한 후 바로 가기 메뉴에서 **배열 전치**를 선택합니다.

배열의 인덱스는 색 블록의 왼쪽 밑 꼭지점에 대응합니다. 색 블록은 배열 인덱스로 정의되는 두 포인트 사이의 영역인 단위 영역을 가지고 있습니다. 강도 그래프 또는 차트는 최대 256 개의 비연속적인 색을 디스플레이할 수 있습니다.

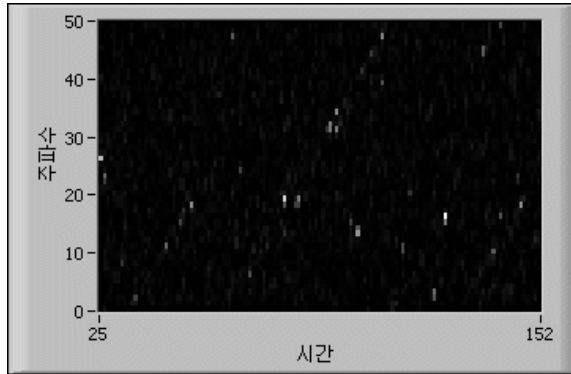
강도 그래프와 차트에 대한 예제는 `labview\examples\general\graphs\intgraph.llb` 를 참조하십시오.

## 강도 차트

강도 차트에 데이터 블록을 플롯한 후에, 직각 평면의 원점은 마지막 데이터 블록의 오른쪽으로 이동합니다. 차트가 새 데이터를 처리할 때, 새 데이터 값은 이전 데이터 값의 오른쪽에 나타납니다. 차트 디스플레이가 꺾 차면, 가장 오래된 데이터 값은 차트의 왼쪽으로 스크롤되어 사라집니다. 이런 동작 특성은 스트립 차트와 유사합니다.

스트림 차트에 대한 더 많은 정보는 이 장의 *차트 업데이트 모드 설정하기* 섹션을 참조합니다.

다음 그림은 강도 차트의 예제를 보여줍니다.



강도 차트는 스케일 범례와 차트를 마우스 오른쪽 버튼으로 클릭한 후 바로 가기 메뉴에서 **보이는 아이템**을 선택하여 보이거나 감출 수 있는 그래프 팔레트 등 웨이브폼 차트와 옵션에서 많은 부분을 공유합니다. 추가적으로, 강도 차트는 3 차원의 색을 가지고 있기 때문에, 컬러 램프 컨트롤과 유사한 스케일은 색 값의 맵핑과 범위를 지정합니다.

색 맵핑에 대한 더 많은 정보는 본 장의 *강도 그래프와 차트에서 색 맵핑 사용하기* 섹션을 참조합니다.

웨이브폼 차트와 마찬가지로, 강도 차트는 이전 업데이트에서 데이터 히스토리, 혹은 버퍼를 보존합니다. 웨이브폼 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **차트 히스토리 길이**를 선택하여 버퍼를 설정합니다. 강도 차트에서 히스토리 버퍼 크기의 기본 설정은 128 데이터 포인트입니다. 강도 차트를 디스플레이하기 위해서 많은 메모리가 필요할 수 있습니다.

## 강도 그래프

강도 그래프는 이전 데이터 값을 유지하지 않고 업데이트 모드를 포함하지 않는 부분을 제외하면, 강도 차트와 동일하게 작동합니다. 강도 그래프로 새 데이터 값이 전달될 때마다 새 데이터는 이전 데이터 값을 대체합니다. 다른 그래프와 같이 강도 그래프에는 커서가 있습니다. 각 커서는 그래프에서 특정 포인트의  $x, y, z$  값을 디스플레이합니다.

커서에 대한 추가적인 정보는 이 장의 *그래프 커서 사용하기* 섹션을 참조하십시오.

## 강도 그래프와 차트에서 색 맵핑 사용하기

강도 그래프 또는 강도 차트는 색을 사용하여 2D 플롯에서 3D 데이터를 디스플레이합니다. 강도 그래프와 차트에서 색 맵핑을 설정하려면, 그래프 또는 차트의 색 스케일을 설정해야 합니다. 색 스케일은 각각의 숫자값과 대응하는 디스플레이 색을 가지는 최소 두 개의 마커로 구성되어 있습니다. 강도 그래프와 차트에서 디스플레이되는 색은 지정된 색과 연관있는 숫자값으로 대응됩니다. 색 맵핑은 임계점 값을 넘는 플롯 데이터 등에서 데이터 범위를 시각적으로 지정하는데 유용합니다.

색 램프 숫자 컨트롤에서 색을 정의하는 것과 같은 방법으로 강도 그래프와 차트에서 색 맵핑을 대화식으로 설정할 수 있습니다.



### 노트

강도 그래프와 차트에서 디스플레이할 수 있는 색은 비디오 카드에서 디스플레이할 수 있는 정확한 색과 색의 수로 제한됩니다. 또한 모니터가 제공하는 색상의 수로 제한됩니다.

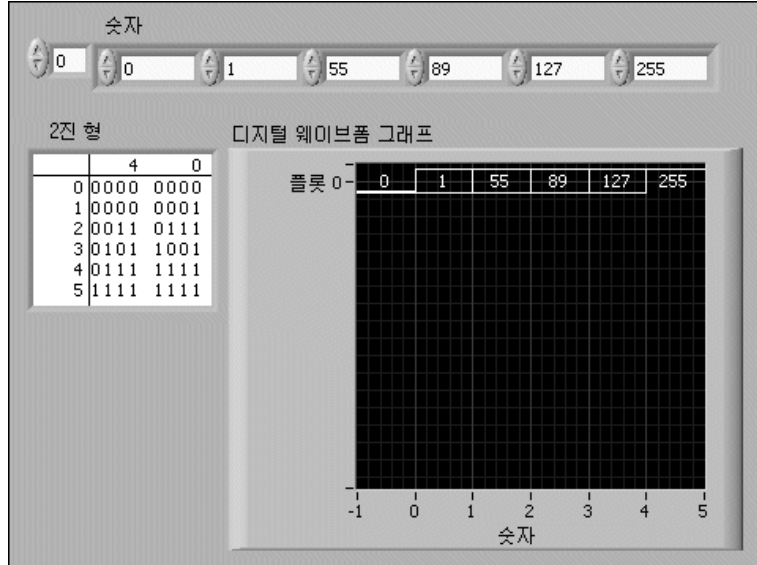
색 맵핑에 대한 예제는 `labview\examples\general\graphs\intgraph.llb`에 있는 `Create IntGraph Color Table VI`를 참조하십시오.

## 디지털 웨이브폼 그래프

디지털 웨이브폼 그래프를 사용하여 디지털 데이터를 디스플레이합니다. 특히 타이밍 다이어그램이나 논리 분석기와 함께 사용할 수 있습니다.

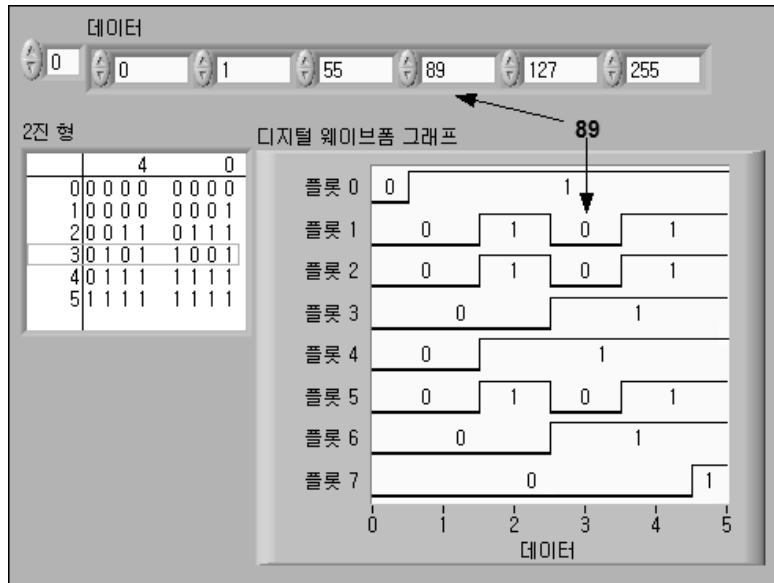
디지털 웨이브폼 그래프는 디지털 웨이브폼 데이터 타임, 디지털 데이터 타임, 이 데이터 타임들의 배열을 입력으로 받을 수 있습니다. 기본으로, 디지털 웨이브폼 그래프는 디지털 버스들을 붕괴시켜 그래프가 디지털 데이터를 단일 플롯으로 플롯하도록 합니다. 디지털 데이터 배열을 연결하면, 디지털 웨이브폼 그래프는 배열의 각 원소를 배열의 순서대로 다른 플롯에 플롯합니다.

다음 프런트패널의 디지털 웨이브폼 그래프는 단일 플롯에 디지털 데이터를 플롯합니다. VI는 **숫자** 배열의 숫자를 디지털 데이터로 변환하고 숫자의 2진 표현을 디지털 데이터 인디케이터의 **2진 형**에 디스플레이합니다. 디지털 그래프에서 숫자 0은 모든 비트 값이 0이므로 상위 라인 없이 나타납니다. 숫자 255는 모든 비트 값이 1이므로 아래쪽 선 없이 나타납니다.



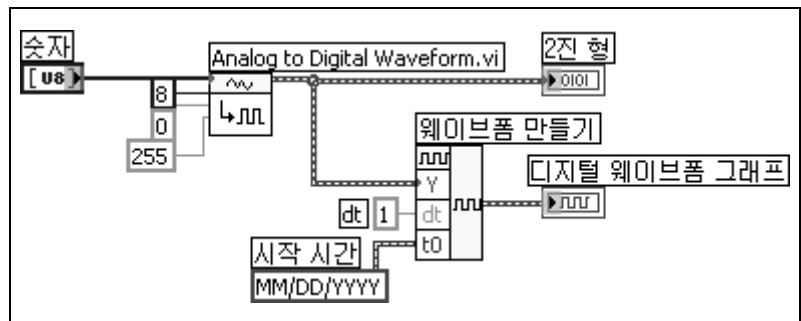
디지털 데이터의 각 샘플을 플롯하려면 y 스케일을 마우스 오른쪽 버튼으로 클릭한 후 바로 가기 메뉴에서 **디지털 버스 확장**을 선택합니다. 각 플롯은 디지털 패턴에서 다른 비트를 나타냅니다.

다음 프런트패널에서 디지털 웨이브폼 그래프는 **숫자** 배열에 있는 여섯 개의 숫자를 디스플레이합니다.



**2진형** 디지털 인디케이터는 숫자를 2진 형으로 디스플레이합니다. 테이블의 각 열은 하나의 비트를 나타냅니다. 예를 들어, 숫자 89는 7비트 메모리(7열의 0은 사용되지 않은 비트를 나타냅니다)가 요구됩니다. 디지털 웨이브폼 그래프의 포인트 3은 번호 89를 나타내기 위한 7비트를 플롯하며 플롯 7의 0은 사용되지 않는 8번째 비트를 나타냅니다.

다음 VI는 숫자 배열을 디지털 데이터로 변환하고 웨이브폼 만들기 함수를 이용하여 시작 시간, 델타 t 그리고 디지털 데이터 컨트롤에 입력한 숫자를 모으고 디지털 데이터를 디스플레이합니다.



디지털 데이터 컨트롤에 대한 추가적인 정보는 4 장, *프런트패널 만들기*의 *디지털 데이터 컨트롤* 섹션을 참조하십시오.

디지털 웨이브폼 그래프에 대한 예제는 `labview\examples\general\graphs\DWDT Graphs.llb` 를 참조하십시오.

## 디지털 웨이브폼 데이터 타입

디지털 웨이브폼 데이터 타입은 시작 시간, 델타  $x$  데이터, 그리고 디지털 웨이브폼의 속성으로 구성됩니다. 디지털 웨이브폼은 (웨이브폼 만들기) 함수로 생성할 수 있습니다. 디지털 웨이브폼 데이터를 디지털 웨이브폼 그래프에 연결하면, 그래프는 디지털 웨이브폼의 시간 정보와 데이터에 기반한 웨이브폼을 자동적으로 플롯합니다. 디지털 웨이브폼 데이터를 디지털 데이터 인디케이터에 연결하여 디지털 웨이브폼의 샘플과 신호를 봅니다.

웨이브폼 데이터 타입에 대한 추가적인 정보는 본 장의 *웨이브폼 데이터 타입* 섹션을 참조합니다.

## 3D 그래프

표면의 온도 분포, 조인트 시간 - 주파수 분석, 비행기의 움직임과 같은 대부분의 실제 데이터 세트는 3 차원으로 데이터를 시각화할 필요가 있습니다. 3D 그래프는 3 차원 데이터를 시각화할 수 있으며, 3D 그래프 프로퍼티를 수정하여 데이터를 나타내는 방법을 바꿀 수 있습니다.



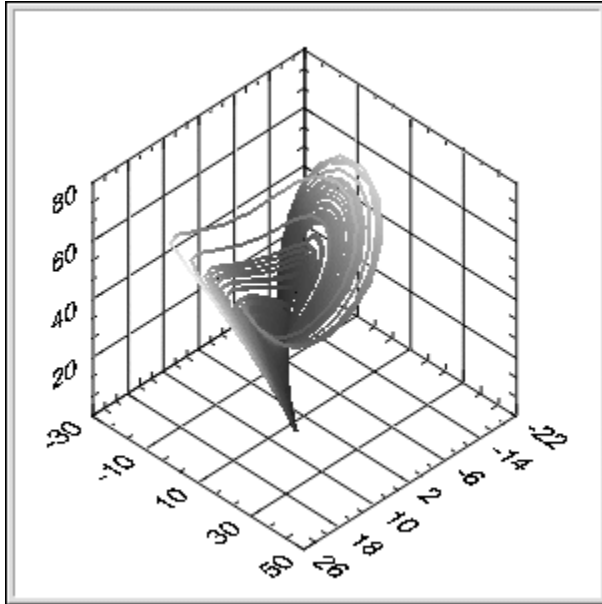
### 노트

3D 그래프 컨트롤은 Windows 의 LabVIEW Full 과 Professional Development Systems 에서만 사용 가능합니다.

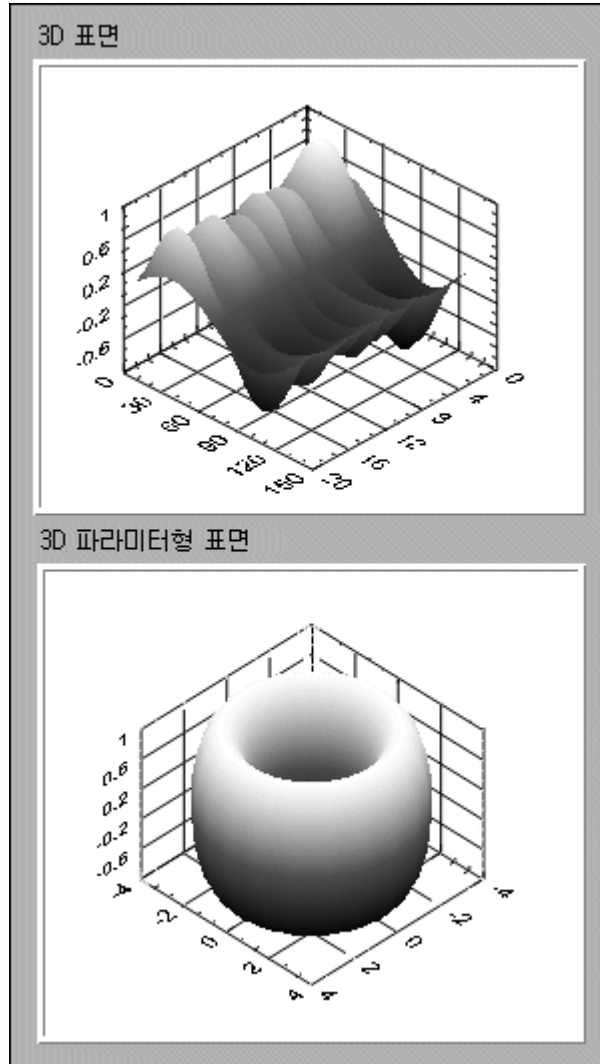
LabVIEW 는 다음과 같은 타입의 3D 그래프를 가집니다 :

- **3D 표면 그래프** — 3 차원 공간 위에 표면을 그립니다.
- **3D 파라미터형 표면 그래프** — 3 차원 공간 위에 파라미터형 표면을 그립니다.
- **3D 커브 그래프** — 3 차원 공간 위에 라인을 그립니다.

커브와 표면을 플롯하기 위해 3D 그래프를 3D 그래프 VI 와 함께 사용합니다. 커브는 그래프의 각 포인트들을 가집니다. 각 포인트는  $x, y, z$  좌표를 가지고 있습니다. VI 는 이 포인트들을 하나의 선으로 연결합니다. 커브는 비행기의 비행 경로와 같이 움직이는 객체의 경로를 시각화하는데 이상적입니다. 다음 그림은 3D 커브 그래프의 예제를 보여줍니다.



표면 플롯은  $x$ ,  $y$ ,  $z$  데이터를 사용하여 그래프에서 포인트를 플롯합니다. 표면 플롯은 이러한 포인트들을 연결하여, 데이터의 3 차원 표면 보기를 형성합니다. 예를 들어, 지형 맵핑을 위해 표면 플롯을 사용할 수 있습니다. 다음 그림은 3D 표면 그래프와 3D 파라미터형 표면 그래프의 예제를 보여줍니다.



3 차원 그래프는 ActiveX 기술과 3D 형태를 다루는 VI 를 사용합니다 . 3 차원 그래프를 선택하면 , LabVIEW 는 ActiveX 컨테이너를 3D 그래프 컨트롤이 내장된 프런트패널 위에 위치시킵니다 . 또한 LabVIEW 는 블록다이어그램에 3 차원 그래프 컨트롤의 참조를 위치시킵니다 . LabVIEW 는 이 참조를 3 개의 3D 그래프 VI 중 하나에 연결합니다 .

# 그래프와 차트 사용자 정의

각 그래프와 차트는 모양의 사용자 정의, 부가 정보 전달, 데이터 하이лай트를 사용할 수 있도록 많은 옵션을 가집니다. 그래프와 차트는 다른 방법으로 데이터를 그리지만, 바로 가기 메뉴에서 접근할 수 있도록 여러가지 공통된 옵션이 제공됩니다. 그러나, 어떤 옵션은 차트 또는 그래프의 특정한 타입에서만 사용 가능합니다.

그래프 또는 차트에만 사용 가능한 옵션에 대한 추가적인 정보는 이 장의 *그래프 사용자 정의하기*와 *차트 사용자 정의하기* 섹션을 참조합니다.

## 여러 X, Y 스케일 사용

모든 그래프는 여러 x, y 스케일을 지원하며, 모든 차트는 여러 y 스케일을 지원합니다. 공통적인 y 또는 y 스케일을 공유하지 않는 여러 플롯을 디스플레이하려면 그래프나 차트의 여러 스케일을 사용합니다. 그래프나 차트에 여러 스케일을 추가하기 위해서는 마우스 오른쪽 버튼으로 클릭한 후 바로 가기 메뉴에서 **스케일 복사**를 선택합니다.

## 오토 스케일

그래프와 차트는 자동적으로 연결한 데이터에 맞도록 수평과 수직 스케일을 조정할 수 있습니다. 이 동작을 오토스케일이라고 부릅니다. 그래프 또는 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **X 스케일»오토스케일 X** 또는 **Y 스케일»오토스케일 Y**를 선택하여 오토스케일을 키거나 끌 수 있습니다. 기본적으로, 그래프와 차트의 오토스케일은 활성화되어 있습니다. 하지만, 오토스케일은 실행을 느리게 할 수 있습니다.

수행 도구나 라벨링 도구를 사용하여 직접 수직 또는 수평 스케일을 바꿉니다.

## 여러 X, Y 스케일 포맷하기

그래프와 차트에서 x 축과 y 축의 스케일이 어떻게 나타나는지 지정하기 위해 **프로퍼티** 대화 상자의 **포맷과 정밀도** 페이지를 사용합니다.

기본적으로, x 스케일은 시간의 라벨을 가지며 부동소수 표기를 사용하도록 설정되었으며, y 스케일은 진폭의 라벨을 가지며 자동 포맷을 사용하도록 설정되었습니다. 그래프 또는 차트의 스케일을 설정하려면, 그래프 또는 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **프로퍼티**를 선택하여 **그래프 프로퍼티** 대화 상자 또는 **차트 프로퍼티** 대화 상자를 디스플레이합니다.

**프로퍼티** 대화 상자에서 **포맷과 정밀도** 페이지를 사용하여 그래프 또는 차트의 스케일을 위해 숫자 포맷을 지정합니다. **스케일** 탭을 클릭하여 스케일

의 이름을 바꾸고 축의 스케일 모양을 포맷합니다. 기본적으로, 그래프 또는 차트의 스케일이 6 자리 이상의 수를 표시하면 자동으로 지수 표기로 변환됩니다.

**포맷과 정밀도** 페이지에서, **고급 편집 모드**를 선택하여 포맷 문자열을 직접 입력하게 하는 텍스트 옵션을 디스플레이합니다. 포맷 문자열을 입력하여 스케일의 모양과 숫자 정밀도를 사용자 정의합니다.

## 그래프 팔레트 사용하기

다음과 같이 그래프 팔레트를 사용하여 VI가 실행되는 동안 그래프 또는 차트와 상호 동작합니다.



그래프 팔레트에서 커서 이동, 디스플레이 줌, 디스플레이 팬을 할 수 있습니다. 그래프 또는 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **보이는 아이템** > **그래프 팔레트**를 선택하여 그래프 팔레트를 디스플레이합니다. 그래프 팔레트에는 다음 버튼이 왼쪽에서 오른쪽의 순서로 나타납니다:

- **커서 이동 도구** (그래프에서만) — 디스플레이에서 커서를 움직입니다.
- **줌** — 디스플레이를 줌 인, 줌 아웃합니다.
- **팬 도구** — 플롯을 선택하여 플롯을 디스플레이에서 움직입니다.

그래프 팔레트에서 버튼을 클릭하여 커서 움직이기, 디스플레이 줌하기, 디스플레이 팬하기를 활성화합니다. 각 버튼은 활성화시 녹색 LED로 디스플레이됩니다.

## 그래프와 차트의 모양 사용자 정의하기

보이기 또는 숨기기 옵션으로 그래프 또는 차트의 모양을 사용자 정의합니다. 그래프 또는 차트를 마우스 오른쪽 버튼으로 클릭한 후 바로 가기 메뉴에서 **보이는 아이템**를 선택하여 다음 옵션들을 숨기거나 보입니다:

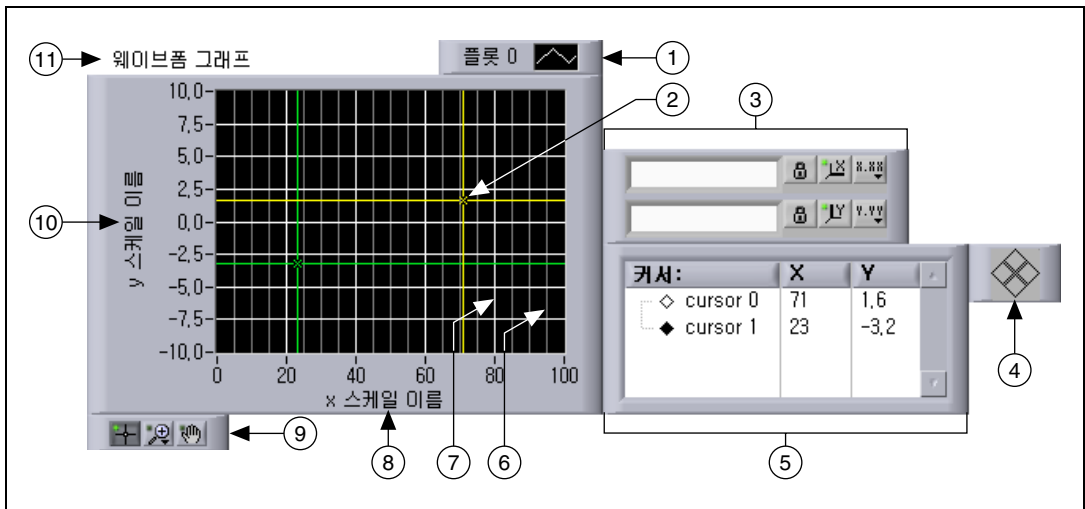
- **플롯 범례** — 플롯의 색과 스타일을 정의합니다. 여러 플롯을 표시하려면 플롯 범례의 크기를 조정합니다.
- **스케일 범례** — 스케일의 라벨을 정의하고 스케일 프로퍼티를 설정합니다.
- **그래프 팔레트** — VI가 실행되는 동안 그래프 또는 차트에서 커서를 움직이고 줌하고 팬할 수 있도록 해줍니다.
- **X 스케일과 Y 스케일** — x와 y 스케일을 포맷합니다.

이 장의 *여러 X, Y 스케일 포맷하기* 섹션을 참고하면 스케일 포맷에 대한 더 많은 정보를 얻을 수 있습니다.

- **커서 범례** (그래프에서만) — 정의된 포인트 좌표에 마커를 디스플레이합니다. 그래프에 여러 커서를 디스플레이할 수 있습니다.
- **X 스크롤바** — 그래프 또는 차트의 데이터를 스크롤합니다. 스크롤 막대를 사용하여 현재 보이지 않는 그래프나 차트의 데이터를 관찰합니다.
- **디지털 디스플레이** (웨이브폼 차트에서만) — 차트의 숫자값을 디스플레이합니다.

## 그래프 사용자 정의하기

각 그래프는 사용자의 데이터 디스플레이 요건에 일치하기 위해 그래프 사용자 정의를 할 수 있는 옵션을 포함합니다. 예를 들면 그래프 커서의 동작과 모양 또는 그래프 스케일 설정의 수정을 할 수 있습니다. 다음 그림은 그래프의 원소를 보여줍니다.



1 플롯 범례	4 커서 이동자	7 눈금 마크	10 Y 스케일
2 커서	5 커서 범례	8 X 스케일	11 라벨
3 스케일 범례	6 보조 눈금 마크	9 그래프 팔레트	

위 범례에 나열된 대부분의 아이템을 추가하기 위해서 그래프에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **보이는 아이템**을 선택하고 적절한 항목을 선택합니다. 그래프에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 옵션을 선택하여 그래프 옵션을 설정합니다.

## 그래프 커서 사용하기

그래프에서 커서를 사용하여 플롯 또는 플롯 영역 포인트의 정확한 값을 읽습니다. 커서 값은 커서 범례에 디스플레이됩니다.

그래프에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **보이는 아이템** > **커서 범례**를 선택합니다. 커서 범례에서 어디든 마우스를 놓고 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **커서 생성**을 선택하고 커서 모드를 선택하여 그래프에 커서를 추가합니다.

커서 위치는 커서 모드에서 정의합니다. 커서는 다음 모드를 가집니다:

- **독립** — 플롯 위치에 구애받지 않고 플롯 영역에서 자유롭게 커서를 움직입니다.
- **단일 플롯** — 커서와 연관되어 있는 플롯에만 커서를 위치시킵니다. 커서를 대응되는 플롯에 따라 움직일 수 있습니다. 커서 범례에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **고정**을 선택하여 하나 또는 모든 플롯을 커서와 연관시킵니다.
- **여러 플롯** — 플롯 영역의 특정 데이터 포인트에서만 커서가 위치합니다. 여러 플롯 커서는 커서가 연관된 모든 플롯의 지정된 x 값에서 값을 리포트합니다. 플롯 영역의 어느 플롯에나 커서를 놓을 수 있습니다. 커서 범례에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **고정**을 선택하여 하나 또는 모든 플롯을 커서와 연관시킵니다. 이 모드는 복합 신호 그래프에서만 사용 가능합니다.



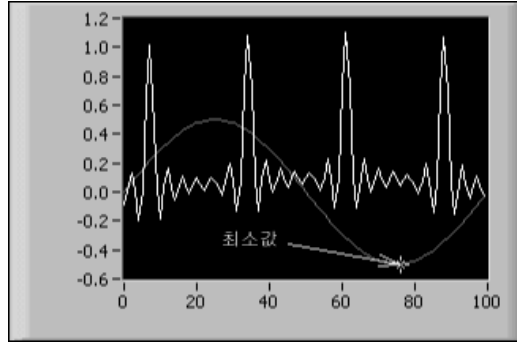
### 노트

커서의 모드를 생성한 후 변경할 수 없습니다. 커서를 삭제하고 다른 커서를 생성해야 합니다.

여러가지 방법으로 커서의 모양을 사용자 정의할 수 있습니다. 플롯에서 커서에 라벨링하고, 커서의 색을 지정하고, 라인, 포인트, 커서 스타일을 지정할 수 있습니다. 커서 범례 열에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 아이템을 선택하여 커서를 사용자 정의합니다.

## 그래프 주석 사용하기

그래프에 주석을 사용하여 플롯 영역의 데이터 포인트를 하이라이트합니다. 주석은 주석과 데이터 포인트를 식별하는 라벨과 화살표를 포함합니다. 그래프는 여러 개의 주석을 가질 수 있습니다. 다음 그림은 주석을 사용한 그래프의 예제를 보여줍니다.



그래프에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **데이터 처리** > **주식 생성**을 선택하여 **주식 생성** 대화 상자를 디스플레이합니다. **주식 생성** 대화 상자를 사용하여 주식 이름을 지정하고 어떻게 주식이 플롯 영역의 플롯에 고정되는지 지정합니다.

**주식 생성** 대화 상자에서 **고정 스타일** 폴다운 메뉴를 사용하여 어떻게 주식이 플롯 영역의 플롯에 고정되는지 지정합니다. **고정 스타일** 구성 요소에는 다음 옵션이 있습니다 :

- **독립** — 주석을 플롯 영역의 어디에나 이동할 수 있습니다. LabVIEW는 주석을 플롯 영역의 어느 플롯에도 고정시키지 않습니다.
- **모든 플롯에 고정** — 플롯 영역의 임의의 플롯에 따라 주석을 가장 가까운 데이터 포인트로 이동시킬 수 있습니다.
- **한 플롯에 고정** — 지정된 플롯을 따라서만 주석을 이동시킬 수 있습니다.

여러가지 방법으로 주석의 모양을 사용자 정의할 수 있습니다. 플롯 영역에 주석 이름이나 화살표를 숨기거나 보여주고, 주석의 색을 지정하고, 라인, 포인트, 주석 스타일을 지정할 수 있습니다. 주석에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 옵션을 선택하여 주석을 사용자 정의합니다.

주석을 삭제하려면, 주석에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **주석 삭제**를 선택합니다. 그래프에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **데이터 처리** > **모든 주석 삭제**를 선택하여 플롯 영역의 모든 주석을 삭제합니다.

## 3D 그래프 사용자 정의하기

3D 그래프는 3D 플롯 스타일, 스케일 포맷, 눈금, 플롯 투영 등 사용자 정의를 하는 많은 옵션을 가집니다. 3D 그래프는 ActiveX 기술과 3D 형태를 다루는 VI를 사용하기 때문에 다른 그래프의 설정하는 옵션과 다릅니다. 어플리케이션을 생성하는 동안 ActiveX 프로퍼티 브라우저를 사용하여 3D 그

래프의 프로퍼티를 설정합니다. 3D 그래프에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 ActiveX 프로퍼티 브라우저를 디스플레이하기 위해 **프로퍼티 브라우저**를 선택합니다.

사용자에게 실행시 공통 프로퍼티 변경을 허용하거나 프로그램적으로 프로퍼티를 설정할 필요가 있다면 3D 그래프 프로퍼티 VI를 사용합니다.

## 차트 사용자 정의하기

저장된 데이터를 새 데이터로 덮어쓰고 디스플레이하는 그래프와 달리, 차트는 이전에 저장된 데이터의 히스토리를 보존하고 주기적으로 업데이트합니다.

사용자의 데이터 디스플레이 요구에 맞도록 차트를 사용자 정의할 수 있습니다. 모든 차트에서 스크롤 막대, 스케일 범례, 그래프 팔레트, 디지털 디스플레이, 시간에 대한 스케일 형의 옵션이 사용 가능합니다. 또한 차트 히스토리 길이, 업데이트 모드, 그리고 플롯 디스플레이의 차트 동작을 수정할 수 있습니다.

## 차트 히스토리 길이 설정하기

LabVIEW는 버퍼, 또는 차트 히스토리에 이미 추가된 데이터 포인트를 저장합니다. 차트 히스토리 버퍼의 기본 크기는 1,024 데이터 포인트입니다. 웨이브폼 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **차트 히스토리 길이**를 선택하여 히스토리 버퍼를 설정합니다. 차트의 스크롤 막대를 이용해 이전에 수집된 데이터를 관찰할 수 있습니다. 스크롤 막대를 나타내려면 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **보이는 아이템** > **X 스크롤 막대**를 선택합니다.



**노트** 커다란 차트 히스토리 값은 많은 메모리가 필요할 수 있습니다.

## 차트 업데이트 모드 설정하기

새로운 데이터를 디스플레이하기 위해 차트를 업데이트 하는 방법을 설정합니다. 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **고급** > **업데이트 모드**를 선택하여 차트의 업데이트 모드를 설정합니다. 차트는 데이터를 디스플레이하기 위해 다음 모드를 사용합니다:

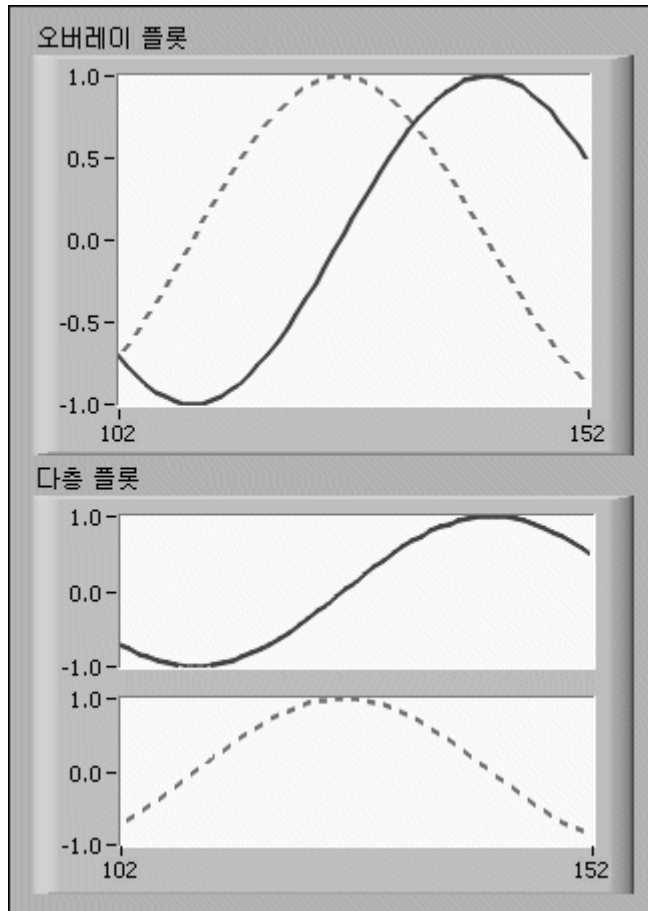
- **스트립 차트** — 차트의 이전 데이터는 왼쪽 그리고 새 데이터는 오른쪽에 연속적으로 스크롤되면서 데이터를 보여줍니다. 스트립 차트는 종이 테이프 스트립 차트 기록기와 비슷합니다. **스트립 차트**는 기본 업데이트 모드입니다.
- **스쿼프 차트** — 펄스 또는 웨이브와 같은 데이터의 한 아이템을 왼쪽에서 오른쪽으로 스크롤하는 도중에 보여줍니다. 각 새로운 값에 대해, 차트는 최근값의 오른쪽에 값을 플롯합니다. 플롯이 그리는 영역의 오른

쪽 경계에 도달하면, LabVIEW 는 플롯을 지우고 왼쪽 경계부터 다시 그리기 시작합니다. 스코프 차트의 다시 그리는 디스플레이는 오실로스코프와 비슷합니다.

- **스왑 차트** — 수직선을 기준으로 오른쪽에는 이전 데이터가 유지되고 왼쪽에는 새 데이터가 그려짐을 제외하면 스코프 차트와 유사합니다. 플롯이 그리는 영역의 오른쪽 경계에 도달하면 LabVIEW 는 스왑 차트에서 플롯을 지우지 않습니다. 스왑 차트는 EKG 디스플레이와 비슷합니다.

## 오버레이와 다층 플롯 사용하기

하나의 수직 스케일인 오버레이 플롯이나 여러 수직 스케일인 다층 플롯을 사용하여 차트에 여러 플롯을 디스플레이할 수 있습니다. 다음 그림은 오버레이 플롯과 다층 플롯의 예입니다.



차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **다중 플롯**을 선택하여 여러 개의 수직 스케일로 플롯하는 차트를 디스플레이할 수 있습니다. 차트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **오버레이 플롯**을 선택하면 하나의 수직 스케일로 플롯하는 차트를 디스플레이할 수 있습니다.

다른 종류의 차트와 그들이 받는 데이터 타입의 예제는 `labview\examples\general\graphs\charts.llb`에 있는 **Charts VI**를 참조하십시오.

## 파일 I/O

파일 I/O 는 파일 안팎으로 데이터를 전달합니다. **파일 I/O** 팔레트의 파일 I/O VI 와 함수를 사용하여 다음을 포함한 모든 파일 I/O 기능을 다룹니다 :

- 데이터 파일의 열기와 닫기 .
- 파일에서 데이터 읽기와 데이터 쓰기 .
- 스프레드시트 형식 파일 읽기와 쓰기 .
- 파일 및 디렉토리 이동하기와 이름 바꾸기 .
- 파일 특성 변경하기 .
- 설정 파일 만들기 , 수정하기 , 읽기 .

하나의 VI 또는 함수를 사용하여 파일을 열거나 , 읽거나 , 쓰거나 , 닫을 수 있습니다 . 또한 , 함수를 사용하여 프로세스의 각 단계를 컨트롤할 수 있습니다 . 또한 , ( 측정 파일로부터 읽기 ) 익스프레스 VI 와 ( 측정 파일에 쓰기 ) 익스프레스 VI 를 사용하여 .lvn 또는 .tdm 파일에서 데이터를 읽고 데이터를 쓸 수 있습니다 .

.tdm 파일에 대한 추가적인 정보는 이 장의 *스토리지 VI 사용하기* 섹션을 참조하십시오 .

## 파일 I/O 의 기초

일반적인 파일 I/O 는 다음과 같은 과정으로 진행됩니다 .

1. 파일을 만들거나 엽니다 . 새 파일 생성 경로를 지정하거나 기존 파일을 선택하려면 파일 위치 경로를 직접 입력하거나 대화 상자의 지시에 따라 선택할 수 있습니다 . 파일이 열린 후에는 참조 번호가 파일을 표시합니다 .  
참조 번호에 대한 추가적인 정보는 4 장 , *프린트패널 만들기의 객체 또는 어플리케이션의 참조* 섹션을 참조하십시오 .
2. 파일을 읽거나 씁니다 .
3. 파일을 닫습니다 .

파일 I/O VI 와 ( 텍스트 파일에서 읽기 ) 및 ( 텍스트 파일 쓰기 ) 함수와 같은 일부 파일 I/O 함수는 일반적인 파일 I/O 작업을 위한 세 단계 모두를 수행할 수 있습니다 . 여러 작업을 위해 디자인된 VI 와 함수는 개별 작업을 위해 설정되거나 디자인된 함수보다 효율성이 떨어질 수도 있습니다 .

대부분의 파일 I/O VI 와 함수들은 참조 번호 또는 경로와 같이 대응하는 입력 파라미터와 같은 값을 반환하는 흐름 파라미터를 포함합니다 .

흐름 파라미터에 대한 추가적인 정보는 5 장 , *블록 다이어그램 만들기의 흐름 파라미터* 섹션을 참조하십시오 .

## 파일 I/O 포맷 선택하기

**파일 I/O** 팔레트의 VI 들은 파일의 포맷에 따라 사용합니다 . 텍스트 , 2 진 , 데이터로그의 세 포맷으로 파일 읽기 또는 쓰기가 가능합니다 . 파일 포맷은 수집 또는 생성되는 데이터나 해당 데이터에 접근하는 어플리케이션에 따라 사용됩니다 .

어떤 데이터 포맷을 사용할 것인지 결정하기 위해서 다음 기본 지침을 참고합니다 :

- Microsoft Excel 과 같은 다른 어플리케이션에서 사용할 수 있는 데이터를 만들려면 , 가장 일반적이고 통용 가능한 텍스트 파일을 사용합니다 .
- 무작위 접근 파일 읽기나 쓰기를 수행하거나 , 수행 속도와 컴팩트한 디스크 공간이 중요하다면 , 텍스트 파일보다 효율적인 2 진 파일을 선택합니다 .
- 복잡한 데이터 레코드 또는 서로 다른 데이터 타입을 LabVIEW 에서 이용하려면 , 데이터로그 파일 형식을 사용합니다 . 데이터로그 파일은 LabVIEW 에서만 데이터에 접근하고 복잡한 데이터 구조를 저장해야 할 때 데이터를 저장하는 가장 좋은 방법이기 때문입니다 .

그래프나 차트 데이터처럼 원래의 형식이 텍스트가 아닐 때 , 텍스트 파일은 2 진이나 데이터로그 파일보다 더 큰 메모리를 차지합니다 . 데이터를 ASCII 형으로 나타내면 일반적으로 데이터 자체보다 더 크기 때문입니다 . 예를 들어 ,  $-123.4567$  는 단정도 부동소수로 4 바이트에 저장할 수 있습니다 . 그러나 , 이것의 ASCII 표현은 각 문자마다 하나의 바이트 즉 , 9 바이트를 차지합니다 .

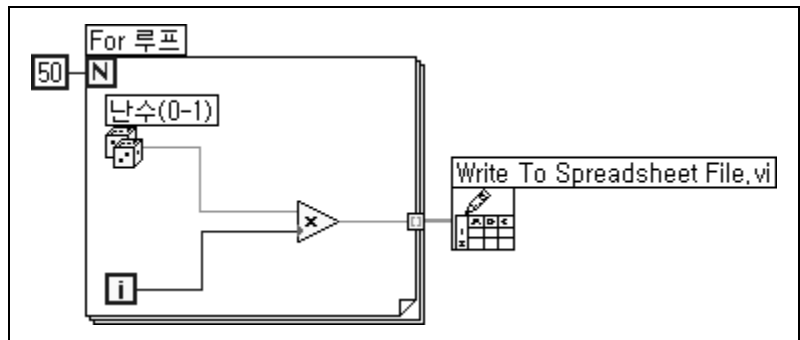
게다가 , 텍스트 파일에서는 숫자 데이터에 무작위 접근하기가 힘듭니다 . 문자열에서 각각의 문자는 정확히 1 바이트씩 차지하지만 , 숫자를 텍스트로 표현하기 위해 필요한 공간은 고정적이지 않습니다 . 텍스트 파일에서 9 번째 숫자를 찾으려면 , LabVIEW 는 앞의 8 개 숫자를 먼저 읽고 변환해야 합니다 .

# 일반적인 파일 I/O 작업을 위한 VI 와 함수 사용 하기

**파일 I/O** 팔레트에는 다음 타입의 데이터를 쓰거나 읽는 등 일반적인 파일 I/O 작업을 위해 디자인된 VI 와 함수가 포함되어 있습니다 .

- 스프레드시트 텍스트 파일에서 숫자값
- 텍스트 파일에서 문자
- 텍스트 파일에서 라인
- 2 진 파일에서 데이터

다음 그림의 블록다이어그램은 ( 스프레드시트 파일에 쓰기 ) VI 를 사용하여 탭으로 자리를 구분한 스프레드시트 파일에 숫자를 보내는 방법을 보여줍니다 . VI 를 실행시키면 LabVIEW 는 새 파일을 생성하거나 이미 존재하는 파일에 데이터를 기록하도록 입력요청합니다 .



열기 , 읽기 , 쓰기 함수는 파일 경로 입력을 기대합니다 . 파일 경로를 지정하지 않으면 , 읽거나 쓸 파일을 지정하도록 대화 상자가 나타납니다 .

**파일 I/O** 팔레트에는 개별적으로 각 파일 I/O 작업을 컨트롤하는 함수가 포함되어 있습니다 . 이 함수를 사용하여 파일을 생성하거나 열고 , 파일로부터 데이터를 읽거나 파일에 데이터를 쓰고 , 파일을 닫습니다 . 이런 함수를 사용하여 다음 태스크도 수행할 수 있습니다 :

- 디렉토리 생성하기 .
- 파일 이동하기 , 복사하기 , 또는 삭제하기 .
- 디렉토리 내용 리스트하기 .
- 파일 특성 변경하기 .
- 경로 변경하기 .

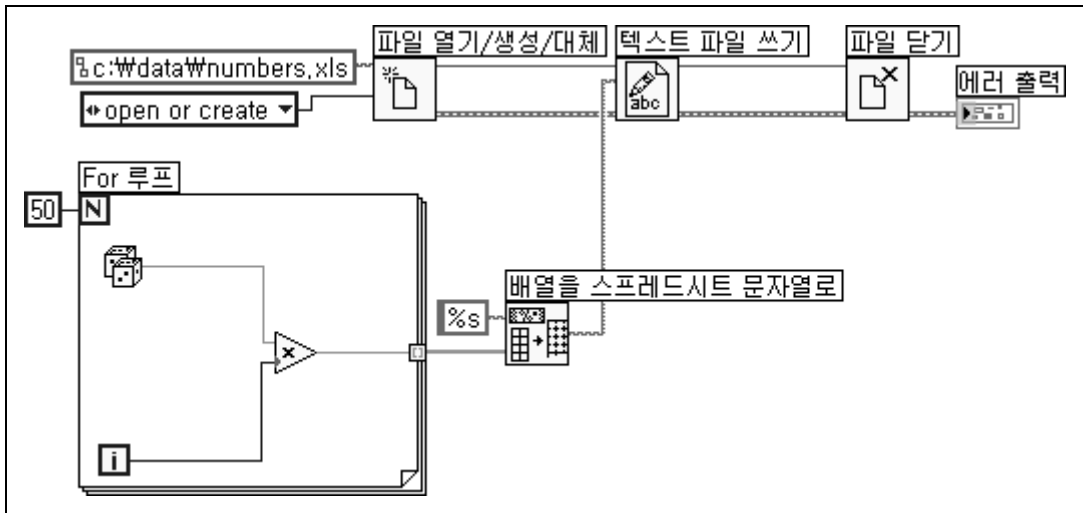
경로는 다음 그림과 같이 , 디스크에 파일의 위치를 지정하는 LabVIEW 데이터 타입입니다 .



경로는 파일의 크기 , 파일 시스템 최상위 레벨부터 현재 파일까지 디렉토리 , 그리고 파일명을 나타냅니다 . 현재 플랫폼의 표준 구문으로 경로 컨트롤 또는 인디케이터에 파일 경로를 입력하거나 디스플레이합니다 .

제 4 장 , *프론트패널 만들기의 경로 컨트롤과 인디케이터* 섹션을 참고하면 경로 컨트롤과 인디케이터에 대한 더 많은 정보를 얻을 수 있습니다 .

다음 블록다이어그램은 파일 I/O 함수를 사용하여 탭으로 자리를 구분한 스프레드시트 파일로 숫자를 보내는 방법을 보여줍니다 . VI 를 실행할 때 , ( 파일 열기 / 생성 / 대체 ) 함수는 numbers.xls 파일을 엽니다 . ( 텍스트 파일 쓰기 ) 함수는 숫자 문자열을 파일에 씁니다 . ( 파일 닫기 ) 함수는 파일을 닫습니다 . 파일을 닫지 않으면 , 파일은 메모리에 남게되며 이를 다른 어플리케이션이나 다른 사용자가 접근할 수 없습니다 .



이전의 블록다이어그램을 같은 작업을 완료하는 ( 스프레드시트에 쓰기 ) VI 와 비교합니다 . 이전 블록다이어그램은 각 파일 작업에 개별 함수를 사용합니다 . 예를 들어 숫자 배열을 문자열로 포맷하기 위해 ( 배열을 스프레드시트 문자열로 ) 함수를 사용합니다 . ( 스프레드시트 파일에 쓰기 ) VI 는 파일을 열고 , 숫자 배열을 문자열로 전환하고 , 파일을 닫는 등의 여러 파일 작업을 완료합니다 .

고급 작업을 위해 파일 I/O VI 와 함수를 사용하는 예제는 labview\examples\file\dataalog.llb 의 Write Datalog File Example VI 를 참조하십시오 .

또한 , 디스크 스트리밍 작업에도 파일 I/O 함수를 사용할 수 있습니다 . 디스크 스트리밍 작업은 함수가 파일을 열고 닫기 위해 OS 와 상호작용하는 횟수를 줄여 메모리 리소스를 절약합니다 . 디스크 스트리밍은 루프 내에서 파일을 닫지 않고 연 상태를 유지하면서 여러가지 데이터 저장 작업을 수행하는 기술입니다 . 경로 컨트롤 또는 함수를 ( 텍스트 파일 쓰기 ) 함수에 연결하면 ( 2 진 파일에 쓰기 ) 함수 또는 ( 스프레드시트 파일에 쓰기 ) VI 는 함수나 VI 가 실행할 때마다 파일을 열고 닫는 오버헤드를 추가합니다 . 동일한 파일을 자주 열고 닫는 것을 피하면 VI 의 효율성을 더 높일 수 있습니다 .

일반적인 디스크 스트리밍 작업을 하려면 , ( 파일 열기 / 생성 / 대체 ) 함수를 루프 앞에 놓고 , 루프 안에 읽기 또는 쓰기 함수를 놓고 , ( 파일 닫기 ) 함수를 루프 뒤에 놓아 지속적인 파일 쓰기 작업이 각 반복마다 파일을 열고 닫는 것에 관련된 오버헤드 없이 루프 내에서 발생할 수 있도록 합니다 .

디스크 스트리밍은 속도에 민감한 장시간 데이터 수집 수행시 이상적입니다 . 데이터를 수집하면서 한편으로는 파일에 데이터를 연속적으로 기록할 수 있습니다 . 최상의 결과를 위해 , 데이터 수집이 완료될 때까지 분석 VI 와 함수같은 다른 VI 와 함수를 함께 실행하지 않습니다 .

## 스토리지 VI 사용하기

**스토리지** 팔레트의 스토리지 VI 를 사용하여 웨이브폼과 웨이브폼 프로퍼티를 2 진 측정 파일 (.tdm) 에서 읽고 씁니다 . LabVIEW 와 DIAdem 과 같이 NI 소프트웨어 간의 데이터 교환은 .tdm 파일을 사용하십시오 .



**노트** 스토리지 VI 는 오직 Windows 에서만 사용 가능합니다 .

스토리지 VI 는 채널을 구성하기 위해서 웨이브폼과 웨이브폼 속성을 결합합니다 . 채널 그룹은 채널 세트를 구성합니다 . 파일은 채널 그룹의 세트를 포함합니다 . 이름에 따라 채널을 저장한다면 , 빠르게 기존의 채널에 데이터를 추가하거나 기존의 채널로부터 데이터를 얻을 수 있습니다 . 숫자값과 더불어 , 스토리지 VI 는 문자열 배열과 타임스탬프 배열을 지원합니다 . 참조 번호는 블록 다이어그램 상의 파일 , 채널 그룹 그리고 채널을 나타냅니다 .

또한 , 스토리지 VI 를 사용하여 지정된 조건에 부합하는 채널 그룹 또는 채널을 얻기 위해서 파일을 쿼리할 수 있습니다 .

개발 중에 시스템 사양이 변경되고 파일에 데이터를 추가해야 하는 경우 , 파일을 사용 불가능으로 만들지 않고도 파일의 포맷을 변경할 수 있습니다 .

스토리지 VI 를 사용한 예제는 `labview\examples\file\storage.llb` 를 참조하십시오.

또한, ( 측정 파일로부터 읽기 ) 익스프레스 VI 와 ( 측정 파일에 쓰기 ) 익스프레스 VI 를 사용하여 .tdm 측정 파일로부터 데이터를 읽고 데이터를 쓸 수 있습니다.

## 텍스트와 스프레드시트 파일 생성하기

텍스트 파일에 데이터를 쓰려면, 반드시 데이터를 문자열로 변환해야 합니다. 스프레드시트 파일에 데이터를 쓰려면 탭과 같은 구분자를 포함하는 문자열인 스프레드시트 문자열로 포맷해야 합니다.

9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 문자열 포맷과 분석* 섹션을 참조하면 포맷 문자열에 대한 더 많은 정보를 얻을 수 있습니다.

대부분의 워드 프로세싱 어플리케이션들은 텍스트를 읽기 위해 포맷된 텍스트를 요구하지 않기 때문에 포맷하지 않고 텍스트 파일에 텍스트를 씁니다. 텍스트 파일에 텍스트 문자열을 쓰려면, 파일을 자동으로 열고 닫는 ( 텍스트 파일 쓰기 ) 함수를 사용합니다.

( 2 진 파일 쓰기 ) 함수를 사용하여 플랫폼에 독립적인 텍스트 파일을 생성합니다. ( 2 진 파일에서 읽기 ) 함수를 사용하여 플랫폼에 독립적인 텍스트 파일을 읽습니다.

2 진 파일에 대한 추가적인 정보는 *2 진 파일 생성하기* 섹션을 참조하십시오.

그래프나 차트에 나타난 숫자 세트나 획득한 데이터를 스프레드시트 문자열로 변환하기 위해서는 ( 스프레드시트 파일에 쓰기 ) VI 또는 ( 배열을 스프레드시트 문자열로 ) 함수를 사용합니다.

워드 프로세싱 어플리케이션에서 생성한 텍스트를 읽다가 에러가 발생하는 경우는 파일 I/O VI 가 처리할 수 없는 폰트, 색상, 스타일, 그리고 사이즈로 텍스트가 포맷되었기 때문입니다.

숫자와 문자를 스프레드시트 또는 워드 프로세싱 어플리케이션에 쓰고자 할 경우, 문자열 함수 또는 배열 함수로 데이터를 포맷하고 문자열끼리 조합합니다. 그 후 파일에 데이터를 기록합니다.

9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 문자열 편집, 포맷, 분석하기와 배열 함수* 섹션을 참고하면 데이터 포맷과 조합에 대한 더 많은 정보를 얻을 수 있습니다.

## 데이터를 파일에 기록하기와 포맷하기

( 파일로 포맷 ) 함수를 사용하여 문자열, 숫자, 경로 및 불리언 데이터를 텍스트로 포맷하고 포맷된 텍스트를 파일에 씁니다. ( 파일로 포맷 ) 함수로 문자열을 포맷하고 ( 텍스트 파일 쓰기 ) 함수로 결과 문자열을 쓰는 대신, 이 함수를 사용할 수 있습니다.

9 장, *문자열, 배열, 클러스터를 이용한 데이터의 그룹화의 문자열 포맷과 분석* 섹션을 참조하면 포맷 문자열에 대한 더 많은 정보를 얻을 수 있습니다.

## 파일에서 데이터를 스캔하기

파일에서 문자열, 숫자, 경로, 그리고 불리언 값 등의 원하는 텍스트를 찾으려면 ( 파일로부터 스캔 ) 함수를 사용하고 텍스트를 데이터 타입으로 변환합니다. ( 2 진 파일에서 읽기 ) 함수 또는 ( 텍스트 파일에서 읽기 ) 함수로 파일에서 데이터를 읽고 ( 문자열로부터 스캔 ) 함수로 결과 문자열을 스캔하는 대신에 이 함수를 사용할 수 있습니다.

## 2 진 파일 생성하기

( 2 진 파일 쓰기 ) 함수를 사용하여 2 진 파일을 생성합니다. 어떤 데이터 타입도 ( 2 진 파일에 쓰기 ) 함수의 **데이터** 입력에 연결할 수 있습니다. ( 2 진 파일에서 읽기 ) 함수를 사용하여 읽으려는 파일 데이터의 데이터 타입을 지정합니다. 해당 데이터 타입의 컨트롤 또는 상수를 ( 2 진 파일에서 읽기 ) 함수의 **데이터 타입** 입력에 연결합니다. ( 2 진 파일 쓰기 ) 와 ( 2 진 파일에서 읽기 ) 함수를 사용하여 다른 OS 에서 생성된 텍스트 파일을 읽고 쓸 수 있습니다.

## 데이터로그 파일 생성하기

**데이터로그** 팔레트의 ( 데이터로그 ) 함수를 사용하여 데이터를 수집하고 데이터를 함수에 씌으로써 데이터 로그 파일을 생성하고 읽을 수 있습니다.

데이터로그 파일에서는 데이터를 포맷할 필요가 없습니다. 그러나 데이터로그 파일을 읽거나 쓸 때, 반드시 데이터 타입을 지정해야 합니다. 예를 들어, 시간과 날짜를 온도 데이터와 함께 기록하고자 한다면, 데이터로그 파일에 데이터를 쓰고 한 개의 숫자와 두 개의 문자열을 가진 클러스터로 이 데이터를 지정합니다. 데이터로그 파일에 데이터를 쓰는 예제는 `labview\examples\file\datalog.11b` 의 Simple Temp Datalogger VI 를 참조하십시오.

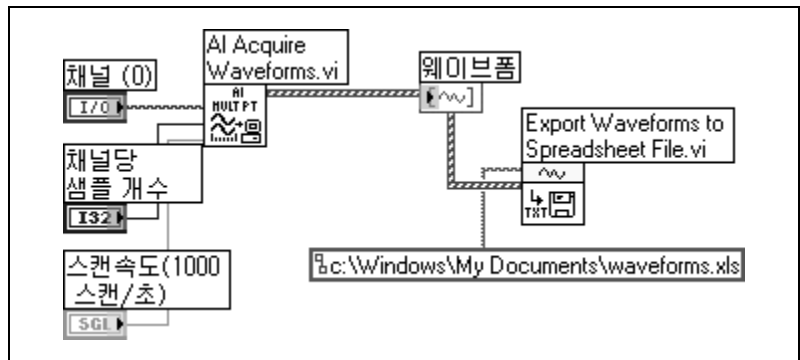
온도, 시간, 날짜가 함께 존재하는 파일을 읽는다면, 하나의 숫자와 두 개의 문자열이 있는 클러스터로 데이터 타입을 지정합니다. 데이터로그 파일을 읽는 예제는 labview\examples\file\data\log.11b 의 Simple Temp Datalog Reader VI 를 참조하십시오.

## 파일에 웨이브폼 쓰기

파일에 웨이브폼을 보내기 위해서는 ( 웨이브폼을 파일에 쓰기 ) 와 ( 웨이브폼을 스프레드시트 파일로 반출 ) VI 를 사용합니다. 웨이브폼은 스프레드시트, 텍스트, 또는 데이터로그 파일에 쓸 수 있습니다.

생성한 웨이브폼을 VI 에서만 사용한다면, 데이터로그 파일 (.log) 로 웨이브폼을 저장합니다.

다음 VI 는 여러 개의 웨이브폼을 수집하고, 그래프에 디스플레이하며, 스프레드시트 파일에 씁니다.

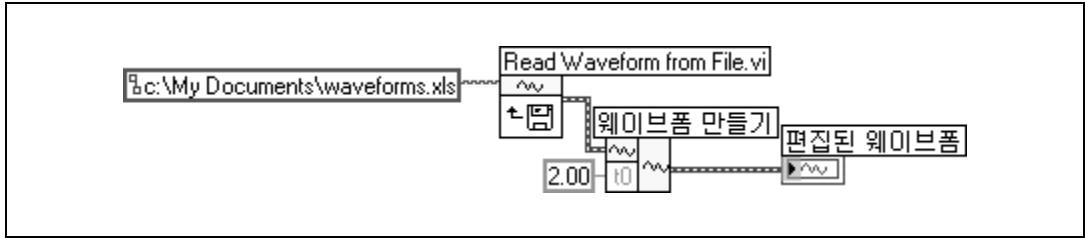


또한, **스토리지** 팔레트의 스토리지 VI 나 ( 측정 파일에 쓰기 ) 익스프레스 VI 를 사용하여 웨이브폼을 파일에 쓸 수 있습니다.

## 파일에서 웨이브폼 읽어오기

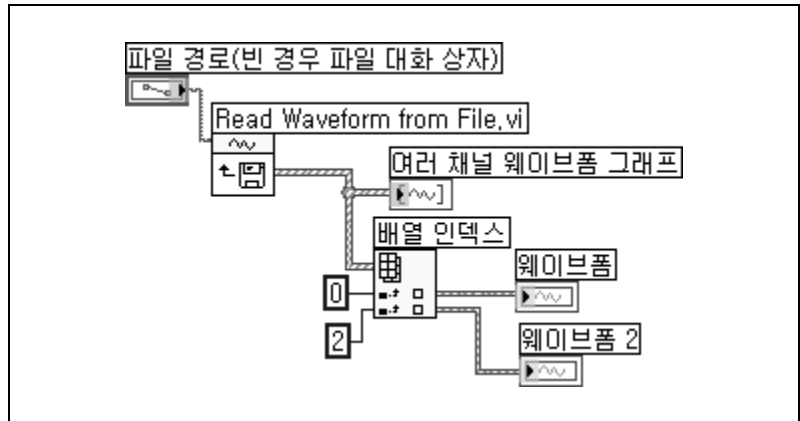
( 파일로부터 웨이브폼 읽기 ) VI 를 이용하여 파일에서 웨이브폼을 읽습니다. 하나의 웨이브폼을 읽은 후, ( 웨이브폼 만들기 ) 함수로 웨이브폼 데이터 타입의 구성요소를 추가 또는 편집하거나 ( 웨이브폼 속성 얻기 ) 함수로 웨이브폼의 구성요소를 추출할 수 있습니다.

다음 VI 는 파일에서 웨이브폼을 읽고, 웨이브폼의 **IO** 구성요소를 편집하며, 편집된 웨이브폼을 그래프에 플롯합니다.



( 파일로부터 웨이브폼 읽기 ) VI 는 파일에서 여러 개의 웨이브폼을 읽을 수 있습니다 . 이 VI 는 여러 플롯 그래프에서 디스플레이할 수 있는 웨이브폼 데이터의 배열을 반환합니다 . 파일에서 하나의 웨이브폼에 접근하려면 , 반드시 다음 블록다이어그램과 같이 웨이브폼 데이터의 배열을 인덱싱해야 합니다 . 이 VI 는 여러 개의 웨이브폼을 포함하는 파일에 접근할 수 있습니다 . ( 배열 인덱스 ) 함수는 파일에서 첫번째와 세번째 웨이브폼을 읽고 , 두 개의 웨이브폼 그래프에서 이를 각각 그립니다 .

배열 인덱싱에 대한 더 많은 정보는 9 장 , 문자열 , 배열 , 클러스터를 이용한 데이터의 그룹화의 배열 섹션을 참조하십시오 . 웨이브폼 그래프에 대한 더 많은 정보는 10 장 , 그래프와 차트의 웨이브폼 그래프 섹션을 참조하십시오 .



또한 , **스토리지** 팔레트의 스토리지 VI 나 ( 특정 파일로부터 읽기 ) 익스프레스 VI 를 사용하여 파일에서 웨이브폼을 읽을 수 있습니다 .

## VI 문서화 및 인쇄하기

LabVIEW 를 사용하여 VI 를 문서화하고 인쇄할 수 있습니다 .

VI 를 문서화해서 개발 단계에서의 블록다이어그램 그리고 / 또는 프런트패널에 대한 정보를 기록합니다 .

VI 인쇄에 대한 일부 옵션은 VI 에 대한 정보를 인쇄하는데 보다 적절하며 다른 옵션은 VI 가 생성하는 데이터와 결과를 리포트하는데 보다 적절합니다 . 수동 또는 프로그램적으로 인쇄할지 , 리포트 포맷에 몇가지 옵션이 필요한지 , 만드는 독립 어플리케이션에 이런 기능이 필요한지 , 어떤 플랫폼에서 VI 를 실행하는지를 포함하여 많은 요소들이 사용할 인쇄 방법에 영향을 미칩니다 .

### VI 문서화하기

LabVIEW 를 사용하여 완성된 VI 를 문서화하고 VI 사용자를 위한 설명을 생성할 수 있습니다 . 또한 LabVIEW 에서 문서를 보고 , 인쇄하고 , HTML, RTF, 또는 텍스트 파일로 저장할 수 있습니다 .

VI 에 효과적인 문서를 생성하기 위해서 , VI 와 객체 설명을 생성합니다 .

VI 및 컨트롤과 인디케이터 같은 VI 객체에 대한 설명을 생성하여 VI 또는 객체의 용도를 설명하고 사용자에게 VI 또는 객체를 사용하는데 대한 설명을 제공합니다 . LabVIEW 에서 설명을 보고 , 인쇄하거나 , HTML, RTF, 또는 텍스트 파일로 저장할 수 있습니다 .

**파일** > **VI 프로퍼티**를 선택하고 **항목** 풀다운 메뉴에서 **문서**를 선택하여 VI 설명을 생성하고 , 편집하고 , 봅니다 . 객체에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **설명과 팁**을 선택하여 객체 설명을 생성하고 , 편집하고 , 봅니다 . 팁 상자는 VI 가 실행되는 동안 커서를 객체 위로 움직일 때 나타나는 간략한 설명입니다 . **설명과 팁** 대화 상자에 팁을 입력하지 않은 경우 , 팁 상자는 나타나지 않습니다 . 커서를 VI 아이콘 또는 객체 위로 움직일 때 **기본 도움말** 윈도우에 각각 VI 또는 객체의 설명이 나타납니다 .



**노트** 함수 팔레트에 위치한 VI 또는 함수에 대한 설명은 입력할 수 없습니다 .

# VI 인쇄하기

다음의 기본적인 방법을 사용하여 VI 를 인쇄할 수 있습니다 :

- **파일>원도우 인쇄**를 선택하여 활성 윈도우의 내용을 인쇄합니다 .
- **파일>인쇄**를 선택하여 프런트패널 , 블록다이어그램 , subVI, 컨트롤 , VI 히스토리 등을 포함한 VI 에 대한 보다 포괄적인 정보를 인쇄합니다 .
- 프로그램적으로 VI 윈도우를 인쇄하거나 VI 문서 또는 VI 가 반환하는 데이터를 포함하는 리포트를 프로그램적으로 인쇄하거나 저장합니다 .

**파일>인쇄**를 선택하여 VI 문서를 인쇄하거나 HTML, RTF, 또는 텍스트 파일로 저장합니다 . 문서에 내장 포맷을 선택하거나 사용자 포맷을 생성할 수 있습니다 . 생성하는 문서는 다음 아이템을 포함할 수 있습니다 :

- 아이콘과 커넥터 팬
- 프런트패널과 블록다이어그램
- 컨트롤 , 인디케이터 , 데이터 타입 터미널
- 컨트롤과 인디케이터의 라벨과 캡션
- VI 와 객체 설명
- VI 계층구조
- SubVI 의 리스트
- 개정 히스토리



## 노트

특정 타입의 VI 에 대해서 생성하는 문서는 위의 모든 아이템을 포함할 수 없습니다 . 예를 들어 , 다형성 VI 에는 프런트패널 또는 블록다이어그램이 없기 때문에 다형성 VI 에 대해서 문서를 생성할 때에는 이러한 아이템을 포함할 수 없습니다 .

LabVIEW 가 생성하는 HTML 또는 RTF 파일을 사용하여 컴파일된 사용자 도움말 파일을 생성할 수 있습니다 . **(Windows)** LabVIEW 로 생성한 개별의 HTML 파일을 HTML 도움말 파일로 컴파일할 수 있습니다 .

**(Mac OS)** LabVIEW 가 생성하는 개별 HTML 파일을 Apple 도움말에 사용할 수 있습니다 .

LabVIEW 로 생성한 RTF 파일을 **(Windows)** WinHelp 또는 **(Linux)** HyperHelp 파일로 컴파일할 수 있습니다 .

# 기술 지원과 전문 서비스

기술 지원과 전문 서비스에 관해서는 National Instruments 웹 사이트 [ni.com](http://ni.com)의 다음 섹션을 방문하십시오 :

- **지원** — [ni.com/support](http://ni.com/support)의 온라인 기술 지원 리소스는 다음을 포함합니다 :
  - **셀프 도움말 리소스** — 소프트웨어 드라이버와 업데이트, 검색 가능한 KnowledgeBase, 제품 매뉴얼, 단계별 문제해결 마법사, 수많은 예제 프로그램, 길라잡이, 어플리케이션 노트, 인스트루먼트 드라이버 등을 포함하는 National Instruments 웹 사이트를 통해 답변과 솔루션을 얻을 수 있습니다.
  - **무료 기술 지원** — 모든 등록된 사용자는 [ni.com/exchange](http://ni.com/exchange)의 NI Developer Exchange에서 전세계 수많은 어플리케이션 엔지니어의 지원을 포함하는 무료 기본 서비스를 받을 수 있습니다. 내셔널인스트루먼트 어플리케이션 엔지니어는 모든 질문이 답변을 얻을 수 있도록 합니다.

각 지역의 기술 지원 옵션에 대한 추가적인 정보는, [ni.com/services](http://ni.com/services)를 방문하거나 [ni.com/contact](http://ni.com/contact)의 가까운 내셔널인스트루먼트 사무소에 연락하십시오.
- **교육 및 인증** — 맞춤 교육, 원거리교육 가상 교실, 대화식 CDs, 인증 프로그램에 대한 정보는 [ni.com/training](http://ni.com/training)을 방문하십시오. 또한, 전세계 여러 곳에서 강사가 지도하는 실습 코스에 등록할 수 있습니다.
- **시스템 통합** — 시간의 제약, 제한된 사내 기술 리소스, 또는 다른 프로젝트상의 문제점이 있는 경우, National Instruments Alliance Partner 회원사가 도움을 드릴 수 있습니다. 추가적인 정보는, 가까운 NI 사무소에 연락하시거나 [ni.com/alliance](http://ni.com/alliance)를 방문하십시오.

[ni.com](http://ni.com)을 검색하여 원하는 답을 찾지 못한 경우, 가까운 현지 사무소나 NI 본사에 연락하십시오. 전세계 사무소의 전화번호는 이 매뉴얼의 앞에 실려 있습니다. 또한 [ni.com/niglobal](http://ni.com/niglobal)의 Worldwide Offices 섹션을 방문하여 최신 연락 정보, 지원 전화번호, e- 메일 주소 및 이벤트 정보를 제공하는 각 사무소의 웹 사이트에 접속할 수 있습니다.

# 용어집

---

기호	접두사	값
y	yocto	$10^{-24}$
z	zepto	$10^{-21}$
a	atto	$10^{-18}$
f	femto	$10^{-15}$
p	pico	$10^{-12}$
n	nano	$10^{-9}$
$\mu$	micro	$10^{-6}$
m	milli	$10^{-3}$
c	centi	$10^{-2}$
d	deci	$10^{-1}$
da	deka	$10^1$
h	hecto	$10^2$
k	kilo	$10^3$
M	mega	$10^6$
G	giga	$10^9$
T	tera	$10^{12}$
P	peta	$10^{15}$
E	exa	$10^{18}$
Z	zetta	$10^{21}$
Y	yotta	$10^{24}$

## 숫자 / 기호

$\infty$  무한대 .

$\Delta$  델타 ; 차이 .  $x$ 는  $x$ 가 하나의 인덱스에서 다음 인덱스까지 변한 값을 나타냅니다 .

$\pi$  원주율 .

- 1D 1 차원 .
- 2D 2 차원 .
- 3D 3 차원 .
- 1 차원 단지 하나의 행의 원소를 가지는 배열의 경우에서처럼 1 차원을 가집니다 .
- 2 차원 몇가지 행과 열을 가지는 배열의 경우에서처럼 2 차원을 가집니다 .

## A

- A 암페어 .
- ASCII American Standard Code for Information Interchange.

## D

- DAQ 데이터 수집 (DAQ) 과 NI-DAQ 을 참조하십시오 .
- DAQ 디바이스 데이터를 수집하거나 생성하는 디바이스이며 여러 개의 채널과 변환 디바이스를 가질 수 있습니다 . DAQ 디바이스에는 플러그인 드라이버 , PCMCIA 카드 , 컴퓨터 USB 나 1394 (FireWire®) 포트를 이용한 DAQPad 디바이스 등이 있습니다 . SCXI 모듈도 DAQ 디바이스에 포함됩니다 .
- DAQ Assistant 측정 태스크 , 채널 , 스케일을 설정하기 위한 그래픽 인터페이스 .

## F

- For 루프 서브다이어그램을 지정한 회수만큼 실행하는 반복적인 루프 구조 . 다음 텍스트 기반 코드와 같음 : `For i = 0 to n - 1, do...`

## G

- G LabVIEW 가 사용하는 그래픽 프로그래밍 언어 .
- General Purpose Interface Bus GPIB. HP-IB 와 비슷합니다 . 컴퓨터에서 전자적인 인스트루먼트를 컨트롤 하기 위해 사용하는 표준 버스입니다 . 이는 IEEE 488 버스라고도 불리는데 , 그 이유는 ANSI/IEEE Standards 488-1978, 488.1-1987, 488.2-1992 로 정의되기 때문입니다 .
- GPIB General Purpose Interface Bus 를 참조하십시오 .

## H

hex 16 진수 . 밑이 16 인 시스템 .

## I

I/O 입력 / 출력 . 통신 채널 , 연산자 입력 디바이스 , 데이터 수집 , 컨트롤 인터페이스와 관련된 컴퓨터 시스템으로 또는 컴퓨터 시스템으로부터의 데이터 전달 .

Inf 무한대의 부동소수 형을 위한 디지털 디스플레이 값 .

IVI Interchangeable Virtual Instruments 공통적인 테스트와 측정 인스트루먼트를 위한 공통 인터페이스 (API) 를 생성하는 표준 소프트웨어 .

## L

LabVIEW Laboratory Virtual Instrument Engineering Workbench. LabVIEW 는 프로그램을 생성하기 위해 텍스트의 라인 대신에 아이콘을 사용하는 그래픽 프로그래밍 언어입니다 .

LED 빛을 발산하는 다이오드 .

LLB 특정한 사용 목적을 위한 연관된 VI 모음을 포함하는 LabVIEW 파일 .

## M

Measurement & Automation Explorer Windows 를 위한 표준 National Instruments 하드웨어 설정과 진단 환경 .

## N

NaN < 숫자가 아님 > 의 부동소수 형을 위한 디지털 디스플레이 값 . 일반적으로  $\log(-1)$  와 같은 정의되지 않은 연산의 결과 .

NI-DAQ 모든 NI DAQ 디바이스와 신호 컨디셔닝 구성요소에 포함되는 드라이버 소프트웨어 . NI-DAQ 은 M Series multifunction I/O (MIO) DAQ 디바이스 , 신호 컨디셔닝 모듈 , 스위치 모듈과 같은 NI 측정 디바이스를 프로그래밍하기 위해 LabVIEW 와 같은 어플리케이션 개발 환경 (ADE) 로부터 호출할 수 있는 VI 와 함수의 광범위한 라이브러리입니다 .

**NI-DAQmx** 측정 디바이스를 컨트롤하기 위한 새로운 VI, 함수, 개발 도구를 가진 최신의 NI-DAQ 드라이버. NI-DAQ의 이전 버전에 대해 NI-DAQmx의 장점은 LabVIEW, LabWindows™/CVI™, Measurement Studio에서 사용할 측정 디바이스를 DAQ Assistant를 이용하여 채널과 측정 태스크를 설정할 수 있고; 더 빠른 단일 포인트 아날로그 I/O 처럼 성능이 향상되었고; 대부분의 지원되는 디바이스에서 하드웨어에 연결하지 않고도 어플리케이션을 테스트 및 수정할 수 있는 NI-DAQmx 시뮬레이션을 제공하고; NI-DAQ의 이전 버전보다 적은 함수와 VI로 DAQ 어플리케이션을 만들 수 있는 더 직관적인 API를 제공한다는 것입니다.

## P

**PXI** PCI eXtensions for Instrumentation. 컴퓨터 기반의 모듈형 인스트루멘테이션 플랫폼.

## S

**SubVI** 다른 VI의 블록 다이어그램에서 사용되는 VI. 서브루틴과 비교할 수 있습니다.

## V

**VI** 버추얼 인스트루먼트 (VI)를 참조하십시오.

**VI 계층구조** 윈도우 VI와 subVI의 계층구조를 그래프로 디스플레이하는 윈도우.

**Virtual Instrument Software Architecture** VISA, GPIB, VXI, RS-232와 다른 타입의 인스트루먼트를 컨트롤하는 단일 인터페이스 라이브러리.

**VISA** Virtual Instrument Software Architecture를 참조하십시오.

## W

**While 루프** 조건을 만족시킬 때까지 코드의 한 섹션을 반복하는 루프 구조.



강도 맵 / 플롯	색을 사용하여 2D 플롯에 3D 데이터를 디스플레이하는 방법 .
강제 변환	데이터 원소의 숫자 형을 변경하기 위해 LabVIEW 가 수행하는 자동 변환 .
강제 변환점	두 개의 다른 숫자 데이터 타입을 가진 데이터를 연결했다는 것을 경고하기 위해 블록 다이어그램 노드에 나타납니다 . 또한 임의의 데이터 타입을 배리 언트 데이터 타입에 연결할 때 나타납니다 .
객체	프런트패널이나 블록다이어그램에서 컨트롤 , 인디케이터 , 노드 , 와이어 , 반입된 그림을 포함하는 모든 아이템을 의미하는 일반적인 용어 .
<b>계층구조</b> 윈도우	VI 계층구조 윈도우를 참조하십시오 .
구문	명령문이 특별한 프로그래밍 언어와 일치해야 하는 일련의 규칙 .
구조	플랫 시퀀스 구조 , 다층 시퀀스 구조 , 케이스 구조 , For 루프 , 또는 While 루프와 같은 프로그램 컨트롤 원소 .
그래프	하나 또는 그 이상의 플롯의 2D 디스플레이 . 그래프는 블록으로 데이터를 받고 플롯합니다 .
그래프 컨트롤	직각좌표 평면에 데이터를 디스플레이하는 프런트패널 객체 .
그림	그림 인디케이터가 그림을 생성하기 위해 사용하는 일련의 그래픽 명령 .
기본	미리 설정된 값 . 값을 지정하지 않는 경우 많은 VI 입력은 기본값을 사용합니다 .
<b>기본 도움말</b> 윈도우	커서를 객체 위에서 움직일 때 , LabVIEW 객체에 대한 기본 정보를 디스플레이하는 윈도우 . 기본 도움말 정보를 가진 객체는 VI , 함수 , 상수 , 구조 , 팔레트 , 프로퍼티 , 메소드 , 이벤트 , 대화 상자 구성요소 , <b>프로젝트 탐색기</b> 윈도우의 아이템이 있습니다 .
깨진 <b>실행</b> 버튼	에러 때문에 VI 를 실행할 수 없을 때 <b>실행</b> 버튼을 대체하는 버튼 .
깨진 VI	에러가 있어서 실행할 수 없는 VI 입니다 ; 깨진 <b>실행</b> 버튼에서 깨진 화살표로 나타냅니다 .
끝기	스크린에서 커서를 사용하여 객체를 선택하거나 , 움직이거나 , 복사하거나 , 삭제합니다 .

## ㄴ

**노드** 프로그램 실행 요소. 노드는 텍스트 기반 프로그래밍 언어의 구문, 연산자, 함수, 서브루틴과 유사합니다. 블록 다이어그램에서 노드에는 함수, 구조, subVI 가 있습니다.

## ㄷ

**다층 시퀀스 구조** 서브다이어그램을 숫자 순서로 실행하는 프로그램 컨트롤 구조. 흐름 파라미터를 사용할 수 없는 경우, 이 구조를 사용하여 데이터에 의존적이지 않은 노드가 순서대로 실행되도록 합니다. 다층 시퀀스 구조는 각 프레임을 디스플레이해서 한번에 단지 하나의 프레임을 열고 마지막 프레임이 실행될 때까지 그 프레임을 순서대로 실행합니다.

**다형성** 다른 형, 타입, 구조의 데이터에 노드를 자동으로 맞추는 기능입니다.

**대화 상자** 어플리케이션이 명령을 수행하기 위해 더 많은 정보를 필요로 할 때 나타나는 윈도우.

**데이터 수집 (DAQ)**

1. 센서, 데이터 수집 변환기, 테스트 프로브, 또는 구조물로부터의 아날로그나 디지털 전기 신호를 수집하고 측정하기.
2. 아날로그나 디지털 전기 신호 생성하기.

**데이터 의존성** 노드가 다른 노드에서 데이터를 받기 전에는 노드가 실행할 수 없다는 데이터 흐름 프로그래밍 언어의 조건. 또한, 인위적인 데이터 의존성을 참조하십시오.

**데이터 타입** 정보에 대한 포맷. LabVIEW 에서 대부분의 VI 와 함수가 사용 가능한 데이터 타입은 숫자, 배열, 문자열, 불리언, 경로, 참조 번호, 열거형, 웨이브폼, 클러스터입니다.

**데이터 흐름** 모든 필수 입력 데이터를 받을 때만 실행되는 실행 가능한 노드로 구성된 프로그래밍 시스템. 이 노드를 실행하면 출력 데이터를 자동으로 생성합니다. LabVIEW 는 데이터 흐름 시스템입니다. 노드를 통한 데이터의 이동은 블록 다이어그램에서 VI 와 함수의 실행 순서를 결정합니다.

**데이터로그** 데이터를 열고 동시에 그것을 디스크 파일에 저장합니다. LabVIEW 파일 I/O VI 와 함수는 데이터를 로그할 수 있습니다.

**데이터로그 파일** 파일을 생성할 때 지정한 하나의 임의의 데이터 타입의 레코드 시퀀스로 데이터를 저장하는 파일. 데이터로그 파일의 모든 레코드는 하나의 타입이어야 하지만, 그 타입은 복잡할 수 있습니다. 예를 들면, 각 레코드가 문자열, 숫자, 배열을 가지고 있는 클러스터라고 지정할 수 있습니다.

**도구** 특정한 동작을 수행하는 특별한 커서.

**도구 모음** VI 를 실행하고 디버그하기 위한 명령 버튼을 가지고 있는 모음.

<b>도구 팔레트</b>	프런트패널과 블록다이어그램 객체를 편집하고 디버그하기 위해 사용할 수 있는 도구를 가진 팔레트 .
독립 라벨	다른 어떤 객체에도 속해 있지 않은 프런트패널이나 블록다이어그램에 있는 라벨 .
드라이버	DAQ 디바이스와 같이 특정한 하드웨어 디바이스를 컨트롤하는 소프트웨어 .
드라이브	콜론 (:) 과 함께 사용되는 a-z 의 문자입니다 . 논리 디스크 드라이브를 나타냅니다 .
디렉토리	편리한 그룹으로 파일을 조직할 수 있는 구조 . 디렉토리는 파일의 위치를 보여주는 주소와 같습니다 . 디렉토리는 파일이나 파일의 서브디렉토리를 가질 수 있습니다 .
디바이스	실제 I/O 포인트를 컨트롤하거나 모니터하는 하나의 객체로서 접근할 수 있는 인스트루먼트 또는 컨트롤러 . 디바이스는 종종 통신 네트워크의 일부 타입을 통해 호스트 컴퓨터에 연결할 수 있습니다 . 또한 , DAQ 디바이스와 측정 디바이스를 참조하십시오 .
디스플레이할 수 없는 문자	널 , 백스페이스 , 탭과 같이 디스플레이할 수 없는 ASCII 문자 .

## ㄹ

라벨	프런트패널이나 블록다이어그램에서 객체나 영역을 이름짓거나 설명하기 위해 사용되는 텍스트 객체 .
라벨링 도구	라벨을 생성하고 텍스트 윈도우에 텍스트를 입력하기 위한 도구 .
라이브러리	LLB 또는 프로젝트 라이브러리를 참조하십시오 .
리스트박스	명령에 사용할 수 있는 모든 선택을 나열한 대화 상자 안의 박스 . 예를 들면 , 디스크에서 파일 이름의 리스트 .
링 컨트롤	32 비트 정수와 관련한 특별한 숫자 컨트롤로 0 에서 시작하고 일련의 텍스트 라벨이나 그래픽과 함께 순차적으로 증가합니다 .

## ㅁ

마법사	정보를 입력함에 따라 앞과 뒤로 이동할 수 있는 페이지의 시퀀스로 되어있는 대화 상자 .
-----	---

## 용어집

메뉴 모음	어플리케이션의 주요 메뉴의 이름을 열거하는 수평 막대. 메뉴 모음은 윈도우의 제목 표시줄 아래에 나타납니다. 일부 메뉴와 명령은 많은 어플리케이션에서 공통적이지만 각 어플리케이션은 해당 어플리케이션에만 있는 메뉴 모음을 가지고 있습니다.
메소드	객체가 메시지를 받을 때 실행되는 절차. 메소드는 항상 클래스와 연관되어 있습니다.
묶기 함수	여러가지 타입의 원소로부터 클러스터를 생성하는 함수.
문양	작은 그림이나 아이콘.
문자열	텍스트로 값을 표현.
문자열 컨트롤과 인디케이터	텍스트를 다루고 디스플레이하는 프런트패널 객체.

## H

바로 가기 메뉴	객체에서 마우스 오른쪽 버튼을 클릭하여 접근할 수 있는 메뉴. 특히 해당 객체에 속한 메뉴 아이템.
반복 터미널	완료된 반복의 현재 숫자를 가지고 있는 For 루프나 While 루프의 터미널.
배열	같은 타입의 데이터 원소가 순서대로 인덱스되어 있는 리스트.
배열 셀	배열이 머무르는 프런트패널 객체. 배열 셀은 인덱스 디스플레이, 데이터 객체 윈도우, 옵션 라벨로 구성되어 있습니다. 그것은 여러가지 데이터 타입을 받습니다.
버추얼 인스트루먼트 (VI)	물리적인 인스트루먼트의 외형과 기능을 모델로 하는 LabVIEW의 프로그램.
버퍼	수집하거나 생성된 데이터의 임시 저장소.
범례	그래프나 차트에서 플롯의 스타일이나 이름을 디스플레이하기 위해 그래프나 차트가 소유하는 객체.
범위	값을 측정하고, 받고, 전달하는 한계 사이의 영역. 상하 범위 값으로 표현합니다.
변환	데이터 원소의 타입 변경.
볼리언 컨트롤과 인디케이터	볼리언 (참 또는 거짓) 데이터를 다루고 디스플레이하는 프런트패널 객체.
브레이크포인트	디버깅을 하기 위해 실행을 멈춥니다.

브레이크포인트 도구	VI, 노드, 와이어에 브레이크포인트를 설정하는 도구.
블록다이어그램	프로그램이나 알고리즘의 그림으로 나타낸 설명이나 표현. 블록다이어그램은 노드 사이에 데이터를 전달하는 노드와 와이어를 가진 실행 가능한 아이콘으로 구성되어 있습니다. 블록다이어그램은 VI의 소스 코드입니다. 블록다이어그램은 VI의 블록다이어그램 윈도우에 상주합니다.
빈 배열	원소가 없지만 정의된 데이터 타입을 가지는 배열. 예를 들면, 데이터 디스플레이 윈도우에 숫자 컨트롤을 가지지만 모든 원소에 대해 정의된 값을 가지고 있지 않은 배열은 비어 있는 숫자 배열입니다.
<b>人</b>	
사각형	4개의 16비트 정수를 가진 클러스터. 처음 두 개의 값은 왼쪽 위 코너의 수직과 수평 좌표를 나타냅니다. 마지막 두 개의 값은 오른쪽 아래 코너의 수직과 수평 좌표를 나타냅니다.
사용자	운영자를 참조하십시오.
사용자 정의된 상수	설정된 값을 나타내는 블록다이어그램 객체.
상수	블록다이어그램에 고정 데이터 값을 제공하는 블록다이어그램의 터미널. 또한, 일반적인 상수와 사용자 정의된 상수를 참조하십시오.
색칠 도구	전면색과 배경색을 설정하는 도구.
샘플	단일 아날로그나 디지털 입력 또는 출력 데이터 포인트.
서브다이어그램	구조 경계 안의 블록다이어그램.
선택 표시	선택된 객체를 둘러싸고 있는 움직이는 점선 경계 표시.
설정 유틸리티	Windows에서는 Measurement & Automation Explorer, Mac OS와 Linux에서는 인스트루먼트의 설정 유틸리티를 참조하십시오.
소스 컨트롤	의도치 않은 데이터 손실을 막기 위한 VI 공유와 접근 컨트롤 문제에 대한 솔루션. 소스 컨트롤 제공자를 사용하여 여러 개발자 사이에서 파일을 공유하고 보안과 품질을 향상시키며, 공유된 프로젝트의 변경 사항을 추적할 수 있습니다. 소스 코드 컨트롤이라고도 불립니다.
수행 도구	컨트롤에 데이터를 입력하거나 그것을 수행하기 위한 도구.
숫자 컨트롤과 인디케이터	숫자 데이터를 다루고 디스플레이하는 프런트패널 객체.
스텝 차트	오실로스코프의 동작을 모델로 한 숫자 인디케이터, 디스플레이를 가로지르는 라인 스텝이 오래된 데이터를 새 데이터와 분리시키는 점만 제외하면 스코프 차트와 비슷합니다.

## 용어집

스칼라	한 스케일의 한 포인트가 나타낼 수 있는 숫자, 배열과 반대로 단일 값, 스칼라 불리언 값과 클러스터는 각 데이터 타입의 특이형 인스턴스입니다.
스케일	측정 단위를 나타내며 알려진 간격의 연속적인 표시나 포인트를 가지고 있는 그래프, 차트, 일부 숫자 컨트롤과 인디케이터의 부분.
스코프 차트	오실로스코프의 동작을 모델로 한 숫자 인디케이터.
스트립 차트	종이 스트립 차트 기록기를 모방하여 모델한 숫자를 플롯하는 인디케이터로, 이것은 데이터를 플롯하면서 스크롤합니다.
슬라이더	슬라이드 컨트롤과 인디케이터의 움직일 수 있는 부분.
시퀀스 구조	플랫 시퀀스 구조나 다층 시퀀스 구조를 참조하십시오.
시프트 레지스터	루프의 한 반복에서 다음 반복으로 변수의 값을 전달하는 루프 구조에서 옵션적인 메커니즘. 시프트 레지스터는 텍스트 기반 프로그래밍 언어에서 정적 변수와 비슷합니다.
실행 모드	VI가 실행하고 있거나 실행하기로 되어 있을 때, 프런트패널 도구 모음에서 <b>실행</b> 이나 <b>연속 실행</b> 을 클릭하거나 블록다이어그램 도구 모음에서 단계별 실행 버튼을 클릭하거나 <b>수행&gt;실행 모드로 변경</b> 을 선택할 때, VI는 실행 모드에 들어섭니다. 실행 모드에서 모든 프런트패널 객체는 바로 가기 메뉴 아이템의 축소된 세트를 가지고 있습니다. VI가 실행 중일 때는 VI를 변경할 수 없습니다.

## O

아이콘	블록다이어그램에서 노드의 그래픽 표현.
어플리케이션	LabVIEW 개발 시스템을 사용하여 생성되었고 LabVIEW 런타임 시스템 환경에서 실행되는 어플리케이션.
어플리케이션 인스턴스	LabVIEW 프로젝트의 각 타겟에 생성된 LabVIEW의 인스턴스. <b>프로젝트 탐색기</b> 윈도우에서 VI를 열 때, VI는 타겟의 어플리케이션 인스턴스에서 열립니다. 또한, LabVIEW는 주요 어플리케이션 인스턴스를 생성합니다. 여기에는 프로젝트의 일부가 아닌 열린 VI와 프로젝트에서 열지 않은 VI가 포함됩니다. 또한 타겟을 참조하십시오.
에러 메시지	소프트웨어나 하드웨어의 작동 불능 또는 받아들일 수 없는 데이터 입력의 시도를 나타냅니다.
에러 입력	VI에 들어오는 에러 클러스터.
에러 출력	VI에서 나가는 에러 클러스터.

에러 클러스터	불리언 상태 인디케이터, 숫자 코드 인디케이터, 문자 소스 인디케이터로 구성되어 있습니다.
오토스케일링	플롯된 값의 범위에 맞게 스케일하는 기능. 그래프 스케일에서, 오토스케일링은 최대와 최소 스케일 값을 결정합니다.
오토인덱싱	루프 구조가 그 경계선에서 배열을 분해하고 함께 합치는 기능. 배열이 오토인덱싱이 활성화된 루프에 들어가면, 루프는 배열을 1 차원 배열에서 스칼라로, 2 차원 배열에서 1 차원 배열로 나누어서 자동적으로 분해합니다. 데이터 값이 루프에서 나올 때 루프는 데이터 값을 반대 순서로 병합합니다.
와이어	노드 사이의 데이터 경로.
와이어 분기	사이에 교차점이 없는 경우 교차점에서 교차점에, 터미널에서 교차점에, 터미널에서 터미널에 이르는 모든 와이어의 부분을 가지고 있는 와이어의 섹션.
와이어 선분	단일 수평, 수직의 와이어 조각.
와이어 표시자	와이어링 도구를 VI 나 함수 노드 위에서 움직일 때 연결되지 않은 터미널 옆에 나타나는 잘려나간 와이어.
와이어링 도구	터미널 사이의 데이터 경로를 정의하는 도구.
운영자	프로세스의 처리를 시작하고 모니터하는 사람.
원형	디자인이 잘 작동한다는 것을 잠정적으로 보여주기 위한 특정한 태스크의 간단하고 빠른 실행. 원형은 보통 일부 기능을 가지고 있지 않고 디자인상의 결점을 가지고 있을 수 있습니다. 일반적으로 원형은 없어져야 하고 그 기능은 최종 버전에서 다시 설치되어야 합니다.
원형이나 핸들의 크기 재조정	객체의 크기를 재조정할 수 있는 포인트를 나타내기 위해 객체의 경계에 나타나는 원형이나 핸들.
웨이브폼	특정한 샘플링 속도로 얻은 여러 전압값.
웨이브폼 차트	특정한 속도로 데이터 포인트를 플롯하는 인디케이터.
위치 도구	객체를 움직이고 크기를 조정하기 위한 도구.
이벤트	아날로그나 디지털 신호의 조건이나 상태.
이산	시간과 같은 독립 변수의 비연속적인 값을 가집니다.
익스프레스 VI	일반적인 측정 태스크를 수행하도록 디자인된 subVI 입니다. 설정 대화 상자를 이용하여 익스프레스 VI 를 설정할 수 있습니다.

## 용어집

인디케이터	그래프나 LED 같은 출력을 디스플레이하는 프런트패널 객체 .
인스트루먼트 드라이버	시스템에서 인스트루먼트 하드웨어를 컨트롤하고 통신하는 상위 레벨 함수의 세트 .
인위적인 데이터 의존성	데이터 값보다는 데이터의 도달이 노드의 실행을 트리거하는 데이터 흐름 프로그래밍 언어의 조건 .
일반적인 상수	예를 들어, $\pi$ 와 같이 , 특정한 ASCII 문자나 표준 숫자 상수를 제공하는 편집할 수 없는 블록 다이어그램의 객체입니다 .

## ㄷ

정수	임의의 자연수 , 그 음수 , 또는 제로
조건 터미널	VI 가 다음 반복을 수행할 것인지 결정하는 불리언 값을 가지고 있는 While 루프의 터미널 .
주파수	기본적인 속도 단위인 f 는 주파수 카운터나 스펙트럼 분석기를 이용하여 초당 이벤트나 진동을 측정합니다 . 주파수는 신호 주기의 역수입니다 .

## ㄷ

차원	배열의 크기와 구조 .
차트	디스플레이가 정의하는 최대값까지 이전 데이터의 히스토리를 유지하는 하나 이상의 플롯에 대한 2D 디스플레이 . 차트는 데이터를 받고 디스플레이를 목적으로 한 버퍼에 어느 정도 지나간 포인트를 유지하면서 , 디스플레이를 포인트마다 또는 배열마다 업데이트합니다 . 또한 , 스킵 차트 , 스트립 차트 , 스왑 차트를 참조하십시오 .
참조 번호	참조 번호 . LabVIEW 가 VI, 어플리케이션 , ActiveX, .NET 객체와 같은 , 객체에 대한 참조로서 사용하는 식별자 . 함수나 VI 를 위한 입력 파라미터로서 참조 번호를 사용하여 객체에서 동작을 수행합니다 .

채널	<p>1. 물리적—아날로그나 디지털 신호를 측정하거나 생성할 수 있는 터미널이나 핀을 의미합니다. 한 개의 물리적 채널은 여러 개의 터미널을 가질 수 있습니다. 차동 아날로그 입력 채널이나 한 개의 디지털 포트에 있는 8개 라인이 좋은 예입니다. 또한, 카운터가 디지털 신호를 측정하거나 생성할 수 있는 터미널에서 카운터 이름이 터미널의 이름이 아니라고 할지라도 카운터는 물리적 채널이 될 수 있습니다.</p> <p>2. 버추얼—속성 설정의 모음입니다. 이름, 물리적 채널, 입력 터미널 연결, 측정 또는 생성의 타입, 그리고 스케일 정보등이 여기에 포함됩니다. 외부 태스크(글로벌) 또는 내부 태스크(로컬)에 대한 NI-DAQmx 버추얼 채널을 정의할 수 있습니다. Traditional NI-DAQ(이전)이나 이전 버전에서 버추얼 채널을 설정할 수 있습니다. 그러나 모든 측정을 NI-DAQmx에서 통합할 수 있습니다. Traditional NI-DAQ(이전)에서는 MAX에서 버추얼 채널을 설정할 수 있습니다. NI-DAQmx에서는 MAX에서 또는 프로그램 안에서 버추얼 채널을 설정할 수 있고, 채널을 태스크의 일부나 따로 설정할 수 있습니다.</p> <p>3. 스위치—스위치 채널은 스위치의 연결 포인트를 나타냅니다. 이것은 스위치의 형태에 따라서 하나 또는 그 이상의 신호 와이어(일반적으로 한 개, 두 개, 또는 네 개)로 구성될 수 있습니다. 버추얼 채널은 스위치 채널로 만들 수 없습니다. 스위치 채널은 NI-DAQmx Switch 함수와 VI에서만 사용될 수 있습니다.</p>
최대값	Measurement & Automation Explorer를 참조하십시오.
최상위 VI	VI 계층구조의 상단에 있는 VI. 이 말이 VI를 그 subVI로부터 구별합니다.
측정 디바이스	E Series multifunction I/O (MIO) 디바이스, SCXI 신호 컨디셔닝 모듈, 그리고 스위치 모듈과 같은 DAQ 하드웨어를 의미합니다.
<b>ㄱ</b>	
카운트 터미널	For 루프가 그 서브다이어그램을 실행하는 횟수를 결정하는 For 루프의 터미널 값.
커벡터	입력과 출력 터미널을 가지고 있는 VI나 함수 노드의 부분. 커벡터를 통해 노드로 그리고 노드로부터 데이터 값을 전달합니다.
커벡터 팬	VI 터미널 패턴을 디스플레이하는 프런트패널이나 블록다이어그램 윈도우의 오른쪽 상단 코너의 영역. 그것은 VI에 연결할 수 있는 입력과 출력을 정의합니다.
컨트롤	노브, 밀기 버튼, 다이얼과 같은 데이터를 대화식으로 VI에 입력하거나 프로그래밍적으로 subVI에 입력하기 위한 프런트패널 객체.
<b>컨트롤</b> 팔레트	프런트패널 컨트롤, 인디케이터, 장식 객체를 가진 팔레트.

## 용어집

컨트롤 흐름	명령의 순차적인 순서가 실행 순서를 결정하는 프로그래밍 시스템. 텍스트 기반의 대부분의 프로그래밍 언어는 순서 제어 언어입니다.
컴파일	상위 레벨 코드를 컴퓨터가 실행 가능한 코드로 변환하는 프로세스. VI를 생성하고 편집한 후에 그 VI가 처음 실행되기 전에 LabVIEW는 그 VI를 자동적으로 컴파일합니다.
케이스	케이스 구조 중 하나의 서브다이어그램.
케이스 구조	케이스 구조에 대한 입력을 바탕으로 그 서브다이어그램 중 하나를 실행하는 조건적 분기 컨트롤 구조. 그것은 컨트롤 흐름 언어에서 IF, THEN, ELSE, CASE 구문들의 조합입니다.
클래스	프로퍼티, 메소드, 이벤트를 포함하는 항목. 클래스는 계층구조로 정렬되며 각 클래스가 상위 레벨 클래스와 관련된 프로퍼티와 메소드를 상속받습니다.
클러스터	숫자형, 불리언, 문자열, 배열, 클러스터를 포함하는 모든 데이터 타입의 정리되고 인덱스가 없는 데이터 원소의 집합. 원소는 모두 컨트롤이거나 모두 인디케이터여야 합니다.
클러스터 셸	클러스터의 원소를 가지고 있는 프런트패널 객체.

## ■

타겟	VI가 실행되는 디바이스 또는 컴퓨터. RT, FPGA, 또는 PDA 타겟으로 작업하려면 LabVIEW 프로젝트를 사용해야 합니다.
터널	구조에서 데이터의 입구나 출구 터미널.
터미널	데이터 값이 통과하는 노드의 객체나 범위.
트리거	데이터 수집의 어떤 형태를 시작하는 모든 이벤트.
팁 상자	터미널 이름을 확인할 수 있는 작은 노란색 텍스트 배너로 와이어 연결을 위해 터미널을 보다 쉽게 인식하게 해줍니다.

## ■

팔레트	프런트패널이나 블록다이어그램을 만들기 위해 사용할 수 있는 객체나 도구를 디스플레이합니다.
펄스	짧은 시간동안 제로가 아닌 진폭을 가진 신호.
편집 모드	VI를 변경할 수 있는 때.

포인트	수평과 수직 좌표를 나타내는 2 개의 16 비트 정수를 가지고 있는 클러스터 .
폴다운 메뉴	메뉴 모음에서 접근할 수 있는 메뉴 . 폴다운 메뉴 아이템은 매우 일반적입니다 .
프런트패널	VI 의 대화식 사용자 인터페이스 . 프런트패널의 외형은 오실로스코프나 멀티미터와 같은 물리적인 인스트루먼트와 비슷합니다 .
프레임	플랫 또는 다층 시퀀스 구조의 서브다이어그램 .
프로브	VI 에서 즉시 값을 검색하기 위한 디버깅 기능 .
프로브 도구	와이어에서 프로브를 생성하기 위한 도구 .
프로젝트	빌드 스펙을 생성하고 파일을 타겟에 배포하거나 다운로드하는데 사용할 수 있는 LabVIEW 파일과 비 LabVIEW 파일의 집합 .
프로젝트 라이브러리	VI, 타입 정의, 공유 라이브러리, 팔레트 메뉴 파일, 다른 프로젝트 라이브러리를 포함한 기타 파일의 집합 .
<b>프로젝트 탐색기</b> 윈도우	LabVIEW 프로젝트를 생성하고 편집할 수 있는 윈도우 .
플랫 시퀀스 구조	서브다이어그램을 숫자 순서로 실행하는 프로그램 컨트롤 구조 . 흐름 파라미터를 사용할 수 없는 경우, 이 구조를 사용하여 데이터에 의존적이지 않은 노드가 순서대로 실행되도록 합니다 . 플랫 시퀀스 구조는 모든 프레임을 한번에 디스플레이하고 마지막 프레임을 실행할 때까지 왼쪽부터 오른쪽으로 프레임들을 실행합니다 .
플롯	그래프나 차트에 나타난 데이터 배열의 그래픽 표현 .
픽셀	디지털화한 그림의 가장 작은 단위 .
<b>공</b>	
하이라이트 실행	VI 에서 데이터의 흐름을 설명하기 위해 VI 실행을 움직임으로 보여주는 디버깅 기술 .
함수	텍스트 기반의 프로그래밍 언어의 연산자, 함수, 구문에 비교되는 내장된 실행 원소 .
<b>함수</b> 팔레트	VI, 함수, 블록다이어그램 구조, 상수를 가진 팔레트 .
핸들	참조 배열과 문자열을 관리하는 한 메모리 블록에 대한 포인터의 포인터 . 문자열의 배열은 문자열의 핸들이 있는 한 메모리 블록에 대한 핸들입니다 .

## 용어집

행렬	선형 방정식에서 계수를 나타내는 숫자나 수학적인 기호의 사각형 배열 .
현재 VI	프런트패널 , 블록다이어그램 , 또는 아이콘 편집기가 활성 윈도우인 VI.
형	8, 16, 32 비트 부호가 있는 그리고 부호가 없는 정수 , 또한 단정도 , 배정도 , 확장된 부동소수가 있는 숫자 데이터 타입의 서브타입 .
확인란	선택하거나 선택 해제할 수 있는 대화 상자의 작은 사각형 상자 . 일반적으로 확인란은 설정할 수 있는 여러 옵션과 연관되어 있습니다 . 하나 이상의 확인란을 선택할 수 있습니다 .
활성 윈도우	사용자 입력을 받도록 현재 설정되어 있는 윈도우 , 일반적으로 맨 앞의 윈도우입니다 . 활성 윈도우의 제목 표시줄은 하이라이트되어 있습니다 . 윈도우를 클릭하거나 <b>윈도우</b> 메뉴에서 그것을 선택하여 윈도우를 활성화합니다 .

# 색인

---

## Numerics

2 진

파일 생성하기, 11-7

3D 그래프, 10-10

## D

DAQ

채널 이름 전달하기, 4-9

## F

For 루프

기본 데이터, 8-10

반복 터미널, 8-2

시프트 레지스터, 8-6

실행하기, 8-2

오도인덱싱, 8-5

카운트 터미널, 8-2

타이밍 컨트롤하기, 8-4

## G

GPIB

설정하기, 1-4

## H

HTML

도움말 파일, 12-2

## I

I/O

(또한 파일 I/O 를 참조하십시오), 11-1

에러, 6-6

이름 컨트롤과 인디케이터, 4-9

IVI

논리 이름 전달하기, 4-9

## K

KnowledgeBase, A-1

## L

LabVIEW

사용자 정의하기, 3-7

설치 제거하기, 1-2

설치하기, 1-2

소개, 1-1

옵션, 3-7

LabVIEW 설치 제거하기, 1-2

LabVIEW 소개, 1-1

## M

Measurement & Automation Explorer, 1-4

MRU 메뉴 아이템, 3-3

## N

National Instrument 지원 및 서비스, A-1

NI 지원 및 서비스, A-1

## R

Repeat-Until 루프

(While 루프를 참조하십시오), 8-3

## S

subVI, 7-1

VI 의 섹션에서 생성하기, 7-3

계층구조, 7-4

다형성 VI, 7-4

만들기, 7-1

생성하기, 7-1

**V**

## VI, 2-1

- 계층구조, 7-4
- 깨진, 6-2
- 다형성, 7-4
- 디버깅 기술, 6-3
- 모양과 특성 설정하기, 7-7
- 문서화, 12-1
- 설명 생성하기, 12-1
- 수정, 6-2
- 실행하기, 6-1
- 업그레이드하기, 7-6
- 에러 핸들링하기, 6-5
- 예제, 1-4
- 이름 붙이기, 7-6
- 인쇄하기, 12-2
- 템플릿, 7-1

## VI 계층구조 윈도우

- 디스플레이하기, 7-4
- 인쇄, 12-2

## VI 단계별로 실행하기

- VI 디버깅하기, 6-4

## VI 문서화하기

- VI 설명 생성하기, 12-1
- 객체 설명 생성하기, 12-1
- 도움말 파일, 12-2
- 인쇄, 12-2
- 팁 상자 생성하기, 12-1

## VI 실행하기, 6-1

## VI 업그레이드하기, 7-6

## VI 저장하기

- 이전 버전으로, 7-6

## VISA

- 리소스 이름 전달하기, 4-9

## VI 의 계층구조

- 보기, 7-4
- 인쇄, 12-2

**W**

## While 루프

- 기본 데이터, 8-10
- 무한, 8-4
- 반복 터미널, 8-4
- 시프트 레지스터, 8-6
- 실행하기, 8-3
- 에러 핸들링하기, 6-7

오토인덱싱, 8-5

조건 터미널, 8-3

타이밍 컨트롤하기, 8-4

**X**

XY 그래프, 10-3

x 스케일

여러, 10-13

**Y**

y 스케일

여러, 10-13

**ㄱ**

가장 최근에 사용된 메뉴 아이템, 3-3

강도 그래프, 10-4

색 맵핑, 10-7

옵션, 10-6

강도 그래프와 차트에 색 맵핑하기, 10-7

강도 차트, 10-4

색 맵핑, 10-7

옵션, 10-5

강제 변환점, 5-9

개정 히스토리

인쇄, 12-2

객체

간격을 균일하게 띄우기, 4-13

라벨 붙이기, 4-14

배포하기, 4-13

블록다이어그램, 5-1

블록다이어그램에서 수동으로 와이어 연결, 5-6

블록다이어그램에서 자동으로 와이어링하

기, 5-8

설명 생성하기, 12-1

설명 인쇄하기, 12-2

옵션 원소, 4-11

옵션 원소 디스플레이하기, 4-11

정렬하기, 4-13

컨트롤을 인디케이터로 또는 반대로 바꾸

기, 4-12

팁 상자 생성하기, 12-1

프런트패널과 블록다이어그램 터

미널, 5-1

- 프런트패널에서 검치기, 4-8
- 프런트패널에서 그룹화하기, 4-13
- 프런트패널에서 대체하기, 4-12
- 프런트패널에서 색칠하기, 4-12
- 프런트패널에서 숨기기, 4-11
- 프런트패널에서 잠그기, 4-13
- 프런트패널에서 크기 조정하기, 4-13
- 객체 간격을 균일하게 띄우기, 4-13
- 객체 정렬하기, 4-13
- 검색하기
  - 팔레트의 컨트롤, VI, 함수를 위하여, 3-2
- 게이지
  - (또한 숫자를 참조하십시오), 4-2
  - 프런트패널, 4-3
- 경고
  - 디스플레이하기, 6-2
- 경로
  - 옵션, 3-7
  - 컨트롤과 인디케이터, 4-6
  - 파일 I/O, 11-4
- 고유 라벨, 4-14
- 공간
  - 프런트패널 또는 블록다이어그램에 추가하기, 4-14
- 관련 문서, 1-1
  - (또한 문서를 참조하십시오), 1-1
- 교육 및 인증 (NI 리소스), A-1
- 구조, 8-1
  - For 루프, 8-2
  - While 루프, 8-3
  - 다중 시퀀스, 8-13
  - 블록다이어그램에서, 2-4
  - 이벤트, 8-14
  - 케이스, 8-10
  - 플랫 시퀀스, 8-13
- 그래프, 10-1
  - 3D, 10-10
  - 3D 사용자 정의하기, 10-17
  - XY, 10-3
  - 강도, 10-4
  - 데이터 포인트 주석 달기, 10-16
  - 동작을 사용자 정의하기, 10-15
  - 모양 사용자 정의하기, 10-14
  - 스케일 포맷하기, 10-13
  - 스케일링, 10-13
  - 스크롤하기, 10-14
  - 여러 스케일, 10-13
  - 옵션, 10-13
  - 웨이브폼, 10-2
  - 커서, 10-16
  - 타입, 10-1
  - 팔레트, 10-14
  - 그래프 팔레트, 10-14
  - 그룹 해제하기
    - 프런트패널 객체, 4-13
  - 그룹화하기
    - 문자열의 데이터, 9-1
    - 배열안의 데이터, 9-3
    - 클러스터 안의 데이터, 9-10
    - 프런트패널 객체, 4-13
  - 그림
    - 링 컨트롤, 4-7
  - 기본 데이터
    - 루프, 8-10
    - 배열, 9-10
  - 기본 도움말 윈도우, 3-5
    - VI 설명 생성하기, 12-1
    - 객체 설명 생성하기, 12-1
  - 기본 케이스, 8-11
  - 기본값
    - 데이터 타입, 5-2
  - 기술 지원, A-1
  - 기호형 숫자값, 5-3
  - 깨진 VI
    - 수정, 6-2
    - 에러 디스플레이하기, 6-2
    - 일반적인 원인, 6-3
  - 깨진 와이어, 5-8
- L**
- 노드, 2-3
  - 블록다이어그램, 5-3
  - 실행 흐름, 5-10
- 노브
  - (또한 숫자를 참조하십시오), 4-2
  - 프런트패널, 4-3
- 눈금, 4-13
  - 옵션, 3-7
  - 눈금에 붙이기, 4-13

**ㄷ**

- 다이얼
  - ( 또한 숫자를 참조하십시오 ), 4-2
  - 프런트패널, 4-3
- 다층 시퀀스 구조
  - 실행하기, 8-13
- 다층 플롯, 10-19
- 다형성
  - VI, 7-4
  - VI 만들기, 7-5
- 다형성 VI 의 인스턴스
  - ( 또한 다형성 VI 를 참조하십시오 ), 7-5
  - 수동으로 선택하기, 7-5
- 단계별 실행
  - VI 디버깅하기, 6-4
- 단순 메뉴, 3-3
- 대상 터미널
  - ( 인디케이터를 참조하십시오 ), 5-1
- 대체하기
  - 프런트패널 객체, 4-12
- 대화 상자
  - 라벨, 4-2
  - 링 컨트롤, 4-7
  - 설계하기, 4-16
  - 인디케이터, 4-2
  - 컨트롤, 4-2
  - 폰트, 4-15
- 데이터 의존성, 5-10
  - 인위적, 5-11
  - 읽기, 5-11
- 데이터 타입, 5-2
  - 기본값, 5-2
  - 웨이브폼, 10-3
  - 인쇄, 12-2
  - 컨트롤과 인디케이터, 5-2
  - 케이스 선택자 값, 8-11
- 데이터 흐름
  - ( 데이터 흐름을 참조하십시오 ), 6-3
  - 관찰하기, 6-3
- 데이터 흐름 프로그래밍 모델, 5-9
  - 메모리 관리, 5-12
- 데이터로그 파일
  - 생성하기, 11-7
  - 파일에서 읽기, 11-7
- 도구
  - 시작하기, 1-4
  - 팔레트, 3-3

- 도구 모음, 3-4
  - 프로젝트, 3-4
- 도움
  - 기술 지원, A-1
- 도움 카드, 1-2
- 도움말
  - ( 또한 상세 도움말 윈도우를 참조하십시오 ), 3-5
- 도움말 파일
  - HTML, 12-2
  - RTF, 12-2
  - 생성하기, 12-2
- 독립 라벨, 4-14
- 드라이버 (NI 리소스), A-1
- 디버깅
  - 기술, 6-3
  - 깨진 VI, 6-2
  - 단계별 실행, 6-4
  - 루프, 8-10
  - 브레이크포인트 도구 사용하기, 6-4
  - 실행 하이라이트 사용하기, 6-3
  - 에러 핸들링하기, 6-5
  - 옵션, 3-7
  - 자동 에러 핸들링하기, 6-5
  - 정의되지 않은 데이터, 5-3
- 디스크 공간
  - 옵션, 3-7
- 디스크 스트리밍, 11-5
- 디스플레이하기
  - 경고, 6-2
  - 에러, 6-2
  - 터미널, 5-1
  - 프런트패널 객체의 옵션 원소, 4-11
- 디지털 그래프, 10-7
- 디지털 데이터
  - 디지털 웨이브폼 데이터 타입, 10-10
- 디지털 웨이브폼 그래프
  - 디지털 데이터 디스플레이하기, 10-7
- 디지털 웨이브폼 데이터 타입, 10-10

**ㄹ**

- 라디오 버튼 컨트롤, 4-5
- 라벨
  - 대화 상자, 4-2
- 라벨 붙이기
  - 상수, 5-3

- 폰트 , 4-15
- 런타임
  - 바로 가기 메뉴 , 3-4
- 루프
  - For ( 개요 ), 8-2
  - While ( 개요 ), 8-3
  - 기본 데이터 , 8-10
  - 무한 , 8-4
  - 배열 만들기 , 8-6
  - 시프트 레지스터 , 8-6
  - 오토인덱싱 , 8-4
  - 타이밍 컨트롤하기 , 8-4
- 루프에 인덱싱하기
  - While 루프 , 8-5
- 리스트박스 , 4-7
  - 컨트롤 , 4-6
- 릴리즈 노트 , 1-2
- 링 컨트롤 , 4-7

## ㄴ

- 마법사 , 1-4
- 만들기
  - subVI , 7-1
  - 다형성 VI , 7-5
  - 블록다이어그램 , 5-1
  - 프런트패널 , 4-1
- 매뉴얼 . 문서를 참조하십시오 .
- 메뉴 , 3-3
  - 링 컨트롤 , 4-7
  - 바로 가기 , 3-4
  - 축소된 , 3-3
  - 콤보 박스 , 4-6
- 메뉴 모음
  - 숨기기 , 4-16
- 메모리
  - 강제 변환점 , 5-9
  - 데이터 흐름 프로그래밍 모델을 이용한 관  
리 , 5-12
- 명령문
  - ( 노트를 참조하십시오 ) , 5-3
- 무한 While 루프 , 8-4
- 무한대 부동소수 값 , 5-3
- 문서 , 1-1
  - ( 또한 관련된 문서를 참조하십시오 ) , 1-1
  - NI 리소스 , A-1
  - 다른 리소스와 함께 사용하기 , 1-1

- 안내 , 1-1
  - 이 매뉴얼 소개 , xi
  - 이 매뉴얼에서 사용된 규약 , xi
- 문자
  - 포맷하기 , 4-15
- 문자열 , 9-1
  - 디스플레이 타입 , 9-2
  - 인디케이터 , 4-5
  - 지정자 포맷하기 , 9-3
  - 컨트롤 , 4-5
  - 콤보 박스 , 4-6
  - 테이블 , 9-2
  - 포맷하기 , 9-3
  - 프로그램적으로 편집하기 , 9-2
- 문제 해결
  - ( 또한 디버깅을 참조하십시오 ) , 6-3
- 문제해결 (NI 리소스) , A-1
- 미터
  - ( 또한 숫자를 참조하십시오 ) , 4-2
  - 프런트패널 , 4-3

## ㄷ

- 바늘
  - 바로 가기 메뉴에서 접근하기 , 4-4
- 바로 가기 메뉴 , 3-3, 3-4
  - 실행 모드에서 , 3-4
- 반복
  - 코드의 블록 , 8-2
- 반복 터미널
  - For 루프 , 8-2
  - While 루프 , 8-4
- 배열
  - 1D 배열의 예제 , 9-4
  - 2D 배열의 예제 , 9-6
  - 기본 데이터 , 9-10
  - 다차원 배열에서 인덱스 , 9-7
  - 다차원 배열의 인덱스 , 9-4
  - 루프를 이용하여 만들기 , 8-6
  - 상수 생성하기 , 9-7
  - 오토인덱싱 루프 , 8-4
  - 의 크기 , 9-10
  - 제약 , 9-4
  - 차원 , 9-3
  - 컨트롤과 인디케이터 , 4-6
  - 컨트롤과 인디케이터 생성하기 , 9-7

## 색인

### 배포하기

프런트패널의 객체, 4-13

### 버전

VI 를 이전 버전으로 저장하기, 7-6

### 버튼

프런트패널, 4-4

### 부동소수

오버플로우와 언더플로우, 5-3

### 볼리언 컨트롤과 인디케이터, 4-4

### 브레이크포인트 도구

VI 디버깅하기, 6-4

### 블록 다이어그램, 2-2

강제 변환점, 5-9

객체, 5-1

구조, 8-1

노드, 5-3

데이터 타입, 5-2

데이터 흐름, 5-9

라벨, 4-14

상수, 5-3

설계하기, 5-12

수동으로 와이어 연결, 5-6

옵션, 3-7

자동으로 와이어링하기, 5-8

컨트롤과 인디케이터의 터미널, 5-1

터미널 디스플레이, 5-1

폰트, 4-15

항수, 5-4

항수에 터미널 추가하기, 5-5

항수에서 터미널 제거하기, 5-5

### 블록 다이어그램의 회색 점, 5-9

## 人

사용자 메뉴얼, 1-2

사용자 인터페이스

(프런트패널을 참조하십시오), 2-1

사용자 정의하기

VI 모양과 특성, 7-7

작업 환경, 3-6

팔레트, 3-6

삭제하기

깨진 와이어, 5-8

상수, 5-3

배열, 9-7

클러스터, 9-11

## 색

낮은 색 품질의 컨트롤과 인디케이터, 4-1, 4-2

높은 색 품질의 컨트롤과 인디케이터, 4-1

맵핑, 10-7

옵션, 3-7

## 색칠하기

프런트패널 객체, 4-12

## 생성하기

2진 파일, 11-7

subVI, 7-1

VI 설명, 12-1

VI의 섹션에서 subVI, 7-3

객체 설명, 12-1

데이터로그 파일, 11-7

배열, 9-7

사용자 정의 상수, 5-3

스프레드시트 파일, 11-6

아이콘, 7-2

클러스터, 9-11

텍스트 파일, 11-6

팁 상자, 12-1

## 서브루틴

(SubVI를 참조하십시오), 7-1

서브패널 컨트롤, 4-8

선택자 터미널 값, 8-11

선택하기

와이어, 5-8

설계하기

대화 상자, 4-16

블록 다이어그램, 5-12

사용자 인터페이스, 4-15

프런트패널, 4-15

설정하기

VI 모양과 특성, 7-7

프런트패널, 4-12

프런트패널 인디케이터, 4-11

프런트패널 컨트롤, 4-11

설치하기

LabVIEW, 1-2

성능

옵션, 3-7

셋팅

작업 환경 옵션, 3-7

소스 코드

(블록 다이어그램을 참조하십시오), 2-2

소스 터미널

(컨트롤을 참조하십시오), 5-1

소프트웨어 (NI 리소스), A-1

수정

- 깨진 VI, 6-2
- 깨진 와이어, 5-8
- 예상치 않은 데이터를 가진 VI, 6-3

수정하기

- VI, 6-2

수직 스크롤 막대, 4-3

수평 스크롤 막대, 4-3

숨기기

- 메뉴 모음, 4-16
- 스크롤 막대, 4-16
- 프런트패널 객체의 옵션 원소, 4-11

숫자

- 기호형 값, 5-3
- 오버플로우와 언더플로우, 5-3
- 컨트롤과 인디케이터, 4-2
- 포맷하기, 4-3

숫자 아님 부동소수 값, 5-3

숫자에서의 언더플로우, 5-3

숫자에서의 오버플로우, 5-3

스위치

- 프런트패널, 4-4

스텝 차트, 10-18

스케일링

- 그래프, 10-13

스코프 차트, 10-18

스크롤 막대

- 리스트박스, 4-7
- 숨기기, 4-16

스크롤 막대 컨트롤, 4-3

스크롤하기

- 그래프, 10-14
- 차트, 10-14

스트립 차트, 10-18

스프레드시트 파일

- 생성하기, 11-6

슬라이더

- 추가하기, 4-3

슬라이드 컨트롤과 인디케이터, 4-3

(또한 숫자를 참조하십시오), 4-2

시스템

- 컨트롤과 인디케이터, 4-2

시스템 도움말

- 관련 문서, 1-1

시스템 폰트, 4-15

시작하기, 1-2

- LabVIEW, 3-1

시퀀스 구조

- 실행 순서 제어하기, 5-10
- 지나친 사용, 8-13
- 플랫과 다층 비교하기, 8-12

시프트 레지스터, 8-6

실행

- VI 디버깅하기, 6-3
- 하이라이트, 6-3
- 흐름, 5-9

실행의 속도

- 컨트롤하기, 8-4

실행의 순서, 5-9

실행의 흐름, 5-9

쓰기

- 파일, 11-1

**○**

아이콘, 2-4

- 생성하기, 7-2
- 인쇄, 12-2
- 편집하기, 7-2

알려진 유의사항, 1-3

어플리케이션 빌더 (Application Builder)

- readme, 1-3

어플리케이션 폰트, 4-15

업그레이드 노트, 1-2

에러

- I/O, 6-6
- While 루프를 사용하여 핸들링하기, 6-7
- 깨진 VI, 6-2
- 디버깅 기술, 6-3
- 디스플레이하기, 6-2
- 리스트, 6-2
- 에러 처리 방법, 6-5
- 윈도우, 6-2
- 자동으로 핸들링하기, 6-5
- 찾기, 6-2
- 케이스 구조 사용하여 핸들링하기, 6-7
- 코드, 6-6
- 클러스터, 6-6
- 핸들링하기, 6-5
- 확인하기, 6-5

연속적으로 VI 실행하기, 6-1

열거형 타입 컨트롤, 4-8

예상치 못한 데이터, 5-3

예제, 1-4

- 예제 (NI 리소스), A-1
  - 오버레이 플롯, 10-19
  - 오토인덱싱
    - For 루프, 8-5
    - While 루프, 8-5
    - 기본 데이터, 8-10
  - 온도계
    - (또한 숫자를 참조하십시오), 4-2
    - 슬라이드 컨트롤과 인디케이터, 4-3
  - 옵션
    - 셋팅, 3-7
  - 와이어, 2-3
    - 깨진, 5-8
    - 선택하기, 5-8
  - 와이어링
    - 객체, 5-7
    - 수동, 5-7
    - 자동, 5-8
  - 요약 메뉴, 3-3
  - 웨이브폼
    - 그래프, 10-2
    - 데이터 타입, 10-3
    - 차트, 10-3
    - 파일에 쓰기, 11-8
    - 파일에서 읽기, 11-8
  - 웹 리소스, A-1
  - 이 매뉴얼에서 사용된 규약, xi
  - 이름 붙이기
    - VI, 7-6
  - 이전 버전
    - VI 저장하기, 7-6
  - 익스프레스 VI 와 함수
    - 개요, 5-5
  - 인덱스
    - 배열에 사용하기, 9-4
  - 인덱싱 루프, 8-4
    - For 루프, 8-5
  - 인디케이터, 4-1
    - I/O 이름, 4-9
    - 경로, 4-6
    - 그룹화하기, 4-13
    - 낮은 색 품질, 4-1
    - 높은 색 품질, 4-1
    - 대체, 4-12
    - 대화 상자, 4-2
    - 데이터 타입, 5-2
    - 데이터 타입 터미널, 5-1
    - 문자열, 4-5
    - 문자열 디스플레이 타입, 9-2
    - 배열, 4-6
    - 볼리언, 4-4
    - 블록다이어그램에서, 5-1
    - 사용자 인터페이스 디자인, 4-15
    - 색칠하기, 4-12
    - 숨기기, 4-11
    - 숫자, 4-2
    - 스크롤 막대, 4-3
    - 슬라이드, 4-3
    - 아이콘, 5-1
    - 옵션 원소, 4-11
    - 옵션 원소 디스플레이하기, 4-11
    - 인쇄, 12-2
    - 일반, 4-1
    - 장그기, 4-13
    - 컨트롤로 바꾸기, 4-12
    - 크기 조정하기, 4-13
    - 클래식, 4-2
    - 클러스터, 4-6
    - 타임스탬프, 4-4
    - 탭, 4-8
    - 터미널, 5-1
    - 프런트패널에서의 사용 지침, 4-15
    - 행렬, 4-6
    - 회전식, 4-3
  - 인쇄
    - VI 의 문서, 12-2
    - 옵션, 3-7
  - 인스트루먼트
    - 설정하기, 1-4
  - 인스트루먼트 드라이버 (NI 리소스), A-1
  - 인위적인 데이터 의존성, 5-11
  - 읽기
    - 파일, 11-1
- ㅈ**
- 자동 와이어링, 5-8
  - 작업 환경 옵션
    - 셋팅, 3-7
  - 장그기
    - 프런트패널 객체, 4-13
  - 잠금 해제
    - 프런트패널 객체, 4-13
  - 적합성 선언 (NI 리소스), A-1
  - 전체 메뉴, 3-3

## 점

- 강제 변환, 5-9
- 점진적으로 VI 실행하기, 6-4
- 정수
  - 오버플로우와 언더플로우, 5-3
- 정의되지 않은 데이터, 5-3
  - 무한대, 5-3
  - 배열, 9-10
  - 숫자 아님, 5-3
- 제거하기
  - 깨진 와이어, 5-8
  - 함수에서 터미널, 5-5
- 조건 터미널, 8-3
- 주석
  - (또한 라벨링을 참조하십시오), 4-14
  - 사용하기, 10-16
- 지원
  - 기술, A-1
- 진단 도구 (NI 리소스), A-1

## ㄹ

- 차원
  - 배열, 9-3
- 차트, 10-1
  - 강도, 10-4
  - 그래프 팔레트, 10-14
  - 다층 플롯, 10-19
  - 동작을 사용자 정의하기, 10-18
  - 모양 사용자 정의하기, 10-14
  - 스케일 포맷하기, 10-13
  - 스크롤하기, 10-14
  - 업데이트 모드, 10-18
  - 여러 스케일, 10-13
  - 오버레이 플롯, 10-19
  - 옵션, 10-13
  - 웨이브폼, 10-3
  - 타입, 10-1
  - 히스토리 길이, 10-18
- 참조 번호
  - 컨트롤, 4-10
  - 파일 I/O, 11-1
- 찾기
  - 에러, 6-2
  - 팔레트의 컨트롤, VI, 함수, 3-2
- 추가 문서, 1-1
  - (또한 관련된 문서를 참조하십시오), 1-1

## 추가하기

- 프런트패널에 공간, 4-14
- 함수에 터미널, 5-5

## ㅋ

- 카운트 터미널, 8-2
  - 오토인덱싱을 사용하여 설정하기, 8-5
- 커넥터 팬, 2-4
  - 만들기, 7-2
  - 인쇄, 12-2
- 커서
  - 그래프, 10-16
- 컨테이너, 4-8
  - 서브패널 컨트롤, 4-8
  - 탭 컨트롤, 4-8
- 컨트롤, 4-1
  - I/O 이름, 4-9
  - 검색하기, 3-2
  - 경로, 4-6
  - 그룹화하기, 4-13
  - 낮은 색 품질, 4-1
  - 높은 색 품질, 4-1
  - 대체, 4-12
  - 대화 상자, 4-2
  - 데이터 타입, 5-2
  - 데이터 타입 터미널, 5-1
  - 리스트박스, 4-6
  - 링, 4-7
  - 문자열, 4-5
  - 문자열 디스플레이 타입, 9-2
  - 배열, 4-6
  - 불리언, 4-4
  - 블록다이어그램에서, 5-1
  - 사용자 인터페이스 디자인, 4-15
  - 색칠하기, 4-12
  - 숨기기, 4-11
  - 숫자, 4-2
  - 스크롤 막대, 4-3
  - 슬라이드, 4-3
  - 아이콘, 5-1
  - 열거형 타입, 4-8
  - 옵션 원소, 4-11
  - 옵션 원소 디스플레이하기, 4-11
  - 인디케이터로 바꾸기, 4-12
  - 인쇄, 12-2
  - 일반, 4-1

잠그기, 4-13  
 참조 번호, 4-10  
 크기 조정하기, 4-13  
 클래식, 4-2  
 클러스터, 4-6  
 타임스탬프, 4-4  
 탐색하기, 3-2  
 탭, 4-8  
 터미널, 5-1  
 테이블, 문자열 입력, 9-2  
 팔레트, 3-1  
 프런트패널에서의 사용 지침, 4-15  
 행렬, 4-6  
 회전식, 4-3  
 컨트롤 팔레트, 3-1  
 검색하기, 3-2  
 탐색하기, 3-2  
 컴퓨터 기반 인스트루먼트  
 설정하기, 1-4  
 케이스 구조  
 기본 케이스 지정하기, 8-11  
 데이터 타입, 8-11  
 선택자 터미널, 8-11  
 실행하기, 8-10  
 에러 핸들링하기, 6-7  
 콤보 박스, 4-6  
 크기 조정하기  
 ( 크기 조정하기를 참조하십시오 ), 4-13  
 프런트패널 객체, 4-13  
 클래식 컨트롤과 인디케이터, 4-2  
 클러스터  
 상수, 9-11  
 생성하기, 9-11  
 에러, 6-6  
 와이어 패턴, 9-10  
 원소의 순서, 9-11  
 컨트롤과 인디케이터, 4-6  
 클러스터 원소의 순서, 9-11

■

타이밍  
 컨트롤하기, 8-4  
 타임스탬프  
 ( 또한 숫자를 참조하십시오 ), 4-2  
 컨트롤과 인디케이터, 4-4

다입 컨트롤  
 열거형, 4-8  
 탐색 윈도우  
 특징, 3-6  
 탐색하기  
 컨트롤과 함수 팔레트, 3-2  
 탭 컨트롤, 4-8  
 탱크  
 ( 또한 숫자를 참조하십시오 ), 4-2  
 슬라이드 컨트롤과 인디케이터, 4-3  
 터미널, 8-1  
 입력과 출력, 8-12  
 터미널, 2-3  
 For 루프에서 반복, 8-2  
 While 루프에서 반복, 8-4  
 강제 변환점, 5-9  
 디스플레이하기, 5-1  
 블록 다이어그램, 5-1  
 상수, 5-3  
 선택자, 8-11  
 오토인덱싱을 사용하여 카운트 설정하기,  
 8-5  
 와이어링, 5-6  
 인쇄, 12-2  
 조건의, 8-3  
 카운트, 8-2  
 컨트롤과 인디케이터, 5-1  
 패턴, 7-3  
 함수에 추가하기, 5-5  
 함수에서 제거하기, 5-5  
 터미널 연결하기, 5-6  
 테이블, 4-7  
 텍스트  
 링 컨트롤, 4-7  
 엔트리 박스, 4-5  
 포맷하기, 4-15  
 텍스트 파일  
 2 진 포맷, 11-7  
 생성하기, 11-6  
 여러 플랫폼에 생성하기, 11-7  
 템플릿  
 VI, 7-1  
 통신  
 파일 I/O, 11-1  
 트리 컨트롤, 4-7  
 팁 상자  
 생성하기, 12-1

**표**

## 파라미터

데이터 타입, 5-2

흐름, 5-11

## 파라미터 리스트

( 커넥터 팬을 참조하십시오 ), 7-2

## 파일 I/O, 11-1

2진 파일 생성하기, 11-7

경로, 11-4

고급 파일 함수, 11-3

기본적인 작업, 11-1

데이터로그 파일 생성하기, 11-7

데이터로그 파일 읽기, 11-7

디스크 스트리밍, 11-5

스토리지 VI 사용하기, 11-5

스프레드시트 파일, 11-6

스프레드시트 파일 생성하기, 11-6

웹이브폼 쓰기, 11-8

웹이브폼 읽기, 11-8

일반적인 작업을 위한 VI, 11-3

일반적인 작업을 위한 함수, 11-3

참조 번호, 11-1

텍스트 파일 생성하기, 11-6

포맷, 11-2

## 파일 I/O를 위한 포맷, 11-2

## 팔레트

도구, 3-3

사용자 정의하기, 3-6

옵션, 3-7

컨트롤, 3-1

컨트롤 사용자 정의하기, 3-6

탐색하기, 3-2

함수, 3-2

함수를 사용자 정의하기, 3-6

## 팝업 메뉴

( 바로 가기 메뉴를 참조하십시오 ), 3-4

## 패턴

터미널, 7-3

## 포맷 문자열 파라미터, 9-3

## 포맷하기

문자열, 9-3

문자열에서 지정자, 9-3

프런트패널의 텍스트, 4-15

## 폰트

대화 상자, 4-15

셋팅, 4-15

시스템, 4-15

어플리케이션, 4-15

옵션, 3-7

## 프런트패널, 2-1

객체 간격 조절하기, 4-13

객체 간격을 균일하게 띄우기, 4-13

객체 겹치기, 4-8

객체 그룹화하기, 4-13

객체 대체하기, 4-12

객체 색칠하기, 4-12

객체 잠그기, 4-13

객체 정렬하기, 4-13

객체 크기 조정하기, 4-13

라벨, 4-14

서브패널 컨트롤에 불러오기, 4-8

설계하기, 4-15

옵션, 3-7

옵션 객체 원소 디스플레이하기, 4-11

옵션 원소 숨기기, 4-11

인디케이터, 4-1

인디케이터를 컨트롤로 바꾸기, 4-12

컨트롤, 4-1

컨트롤을 인디케이터로 바꾸기, 4-12

크기 조정없이 공간 추가하기, 4-14

터미널, 5-1

텍스트 특성, 4-15

폰트, 4-15

## 프런트패널 객체 겹치기, 4-8

## 프런트패널의 빛, 4-4

## 프런트패널의 폴다운 메뉴, 4-7

## 프로그래밍 예제, 1-4

## 프로그래밍 예제 (NI 리소스), A-1

## 플랫 시퀀스 구조

실행하기, 8-13

## 플롯

다층의, 10-19

오버레이, 10-19

## 피드백 노드

선택하기, 8-9

시프트 레지스터로 대체하기, 8-10

초기화하기, 8-9

**중**

## 하드웨어

설정하기, 1-4

## 하이라이트 실행

VI 디버깅하기, 6-3

색인

함수 , 5-4

    검색하기 , 3-2

    탐색하기 , 3-2

    터미널 제거하기 , 5-5

    터미널 추가하기 , 5-5

함수 팔레트 , 3-2

    검색하기 , 3-2

    사용자 정의하기 , 3-6

    탐색하기 , 3-2

행렬

    컨트롤과 인디케이터 , 4-6

환경

    ( 옵션을 보십시오 ) , 3-7

회전식 컨트롤과 인디케이터 , 4-3

흐름 제어 프로그래밍 모델 , 5-9

히스토리

    옵션 , 3-7

    차트 , 10-18