

# LabVIEW™ 업그레이드 노트

## 버전 8.2

이 업그레이드 노트는 Windows, Mac OS, Linux 에서 LabVIEW 를 버전 8.2 로 업그레이드하는 과정 및 업그레이드할 때 유의사항, 새로운 기능들을 설명합니다 .

LabVIEW 7.1 또는 이전 버전에서 LabVIEW 8.2 로 업그레이드하는 경우, LabVIEW 8.0 의 개선 및 변경 내역과 추가된 기능에 대한 정보를 *LabVIEW 8.0 업그레이드 노트*에서 확인하십시오 . National Instruments 는 LabVIEW 7.1 또는 이전 버전에서 업그레이드하는 모든 사용자에게 본 업그레이드 노트와 더불어서 *LabVIEW 8.0 업그레이드 노트의 LabVIEW 8.0 특징과 변경 내역*을 읽어보시기를 권장합니다 . National Instruments 웹 사이트 ([ni.com/info](http://ni.com/info)) 를 방문하여 정보 코드 upnote8 을 입력하면 *LabVIEW 8.0 업그레이드 노트*를 확인할 수 있습니다 .

LabVIEW 프로그래밍 개념, LabVIEW 사용에 대한 단계별 설명, LabVIEW VI, 함수, 팔레트, 메뉴, 도구, 프로퍼티, 메소드, 이벤트, 대화 상자 등에 대한 참조 정보와 LabVIEW 8.2 의 특징에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오 . 또한, *LabVIEW 도움말*은 National Instruments 가 제공하는 LabVIEW 문서 리소스를 나열합니다 . **도움말 » LabVIEW 도움말 검색**을 선택해서 *LabVIEW 도움말*에 접근합니다 .

## 목록

---

LabVIEW 8.2 로 업그레이드하기 .....	2
VI 변환하기 .....	2
툴킷, 인스트루먼트 드라이버, 애드온을 업그레이드하기 .....	3
추가적인 National Instruments 소프트웨어 업그레이드하기 .....	4
이전 버전의 LabVIEW 에서 업그레이드하기 .....	4
업그레이드와 호환성 문제 .....	5
LabVIEW 8.0 에서 업그레이드하기 .....	5
LabVIEW 7.x 에서 업그레이드하기 .....	11
LabVIEW 6.x 에서 업그레이드하기 .....	28
LabVIEW 5.x 또는 이전 버전에서 업그레이드하기 .....	33
LabVIEW 8.2 의 특징과 변경 내역 .....	33
LabVIEW 문서 .....	33
새 예제 VI .....	33
시작 시간 단축 .....	33
블록다이어그램의 개선사항 .....	33
프런트패널 개선사항 .....	35

환경 개선사항	38
새로운 또는 변경된 VI, 함수, 노드 개선사항	42
새로운 프로퍼티, 메소드, 이벤트	50
LabVIEW MathScript 개선사항	51
3D 그림 컨트롤	53
LabVIEW 객체 지향 프로그래밍	54
LabVIEW 프로젝트 개선사항	55
여러 클라이언트로부터 원격으로 VI 컨트롤하기	60
공유 라이브러리 파일에서 함수 반입하기	60
인스트루먼트 드라이버 템플릿	61
.NET 과 ActiveX 개선사항 (Windows)	61
NI 예제 탐색기의 개선사항	62
소스 컨트롤 개선사항	62
TDM 개선사항	63
웹 서비스 반입하기 (Windows)	64
외부 코드 함수 변경 내역	64

## LabVIEW 8.2 로 업그레이드하기

LabVIEW 의 이전 버전에서 업그레이드하는 경우, 우선 이 *LabVIEW 8.2 로 업그레이드하기* 섹션과 이 문서의 *업그레이드와 호환성 문제* 섹션에서 *LabVIEW x.x 에서 업그레이드하기* 섹션을 참조하십시오. 이 때 x.x 는 업그레이드할 LabVIEW 의 버전을 지칭합니다.

### VI 변환하기

LabVIEW 4.0 또는 이후 버전에서 저장된 VI 를 열면 LabVIEW 8.2 는 자동으로 VI 를 변환하고 컴파일합니다. 이 VI 를 반드시 LabVIEW 8.2 으로 저장해야 합니다. 그렇지 않으면 VI 에 접근할 때마다 추가적인 메모리 리소스를 사용하는 변환 프로세스가 수행됩니다.

또한, 리컴파일과 같은 저장되지 않은 변경사항을 가지고 있는 VI 는 실행 성능을 크게 떨어뜨릴 수도 있습니다. 성능과 메모리에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.



#### 노트

LabVIEW 8.2 에서 저장한 VI 는 이전 버전의 LabVIEW 에서 로드되지 않습니다. VI 가 LabVIEW 8.0 에서 실행되도록 하려면 **파일 » 이전 버전으로 저장**을 선택하여 VI 를 저장하십시오. LabVIEW 8.0 또는 이전 버전에서 다시 사용할 VI 는 LabVIEW 8.2 로 저장하기 전에 백업 복사본을 남겨두십시오.

컴퓨터에 메모리가 부족하여 한번에 모든 VI 들을 변환하기 어려울 경우, 단계별로 나누어 VI 들을 변환하십시오. 변환하고자 하는 VI 의 계층구조를 확인하고 가장 낮은 계층구조의 subVI 부터 로딩한 후 저장을 시작합니다. 점차 상위 계층구조의 VI 까지 진행하십시오. 마지막으로 최상위 레벨 VI 를 열고 변환합니다. 또한, 디렉토리 내에 위치한 VI 를 변환하려면 **도구 » 고급 » 매스 컴파일**을 선택할 수 있습니다. 그러나 매스 컴파일은 디렉토리 내의 VI 또는 LLB 를 알파벳 순서로 변환합니다. 변환 프로세스가 상위 레벨 VI 에 먼

저 접근하게 되면, 매스 컴파일은 사용자가 상위 레벨 VI 를 먼저 열었을 때와 거의 동일한 양의 메모리를 필요로 합니다.

현재 사용하고 있는 메모리 양의 요약을 디스플레이하려면 **도움말 » LabVIEW 정보**를 선택하여 메모리 사용을 모니터링합니다.

## 툴킷, 인스트루먼트 드라이버, 애드온을 업그레이드하기

LabVIEW 8.2 를 설치한 후, 호환 가능한 툴킷과 애드온인지 확인한 다음 LabVIEW 8.2 디렉토리에 툴킷과 애드온을 다시 설치하십시오. 먼저 LabVIEW 이전 버전의 툴킷을 제거해야 할 수도 있습니다. 설치에 대한 추가적인 정보는 LabVIEW 툴킷 또는 애드온 문서를 참조하십시오.



### 노트

LabVIEW 모듈 버전이 LabVIEW 버전과 일치해야 합니다. 예를 들어, LabVIEW Real-Time 모듈 8.2 는 LabVIEW 8.2 와 함께 사용해야 합니다.

또한, 기존 툴킷, 인스트루먼트 드라이버, 애드온 VI 를 LabVIEW 8.2 에서 사용하려면 반드시 매스 컴파일을 해야 합니다. 매스 컴파일 VI 에 대한 추가적인 정보는 이 문서의 [VI 변환하기](#) 섹션을 참조하십시오.

다음의 툴킷, 인스트루먼트 드라이버, 애드온의 경우, LabVIEW 8.2 에서 사용하려면 업그레이드 또는 다운로드를 해야 합니다:

- LabVIEW 어플리케이션 빌더를 가지고 있을 경우, 반드시 LabVIEW 어플리케이션 빌더 8.2 로 업그레이드해야 합니다. LabVIEW 8.2 Professional Development System 에는 어플리케이션 빌더 8.2 가 포함되어 있습니다. LabVIEW 어플리케이션 빌더 설치에 대한 추가적인 정보는 **(Windows)** 의 labview\readme 디렉토리 또는 **(Mac OS 와 Linux)** 의 labview 디렉토리에 있는 *LabVIEW 어플리케이션 빌더 Readme* 를 참조하십시오.
- LabVIEW 8.x 에서는 VI Analyzer 1.1 을 사용해야 합니다. Upgrade Advisor 에 접근하고 VI Analyzer 1.1 을 구매하려면, National Instruments 웹 사이트 [ni.com/info](http://ni.com/info) 에서 정보 코드 exd8yy 를 입력하십시오.
- LabVIEW 8.x 와 함께 Internet Toolkit 6.0 을 사용하려면 추가적인 VI 를 다운로드해야 합니다. 필요한 VI 를 다운로드하려면 National Instruments 웹 사이트 [ni.com/info](http://ni.com/info) 에서 정보 코드 itkver6 을 입력하십시오.
- HP/Agilent 34401A Digital Multimeter (DMM) 의 인스트루먼트 드라이버는 이제 National Instruments DMM 템플릿 드라이버와 더욱 유사해졌습니다. 이 드라이버는 LabVIEW 7.x 및 이전 버전이 사용하는 HP34401A 드라이버와 호환되지 않습니다. LabVIEW 7.x HP34401A 드라이버와의 호환이 필요한 경우, National Instruments 의 인스트루먼트 드라이버 네트워크인 [ni.com/idnet](http://ni.com/idnet) 에서 해당 드라이버를 다운로드하십시오.

# 추가적인 National Instruments 소프트웨어 업그레이드하기

LabVIEW 8.x 에서는 NI TestStand 3.5 또는 이후 버전을 사용해야 합니다. Upgrade Advisor 에 접근하고 NI TestStand 3.5 또는 이후 버전을 구매하려면, National Instruments 웹 사이트 [ni.com/info](http://ni.com/info) 에서 정보 코드 exd8yy 를 입력하십시오.

## 이전 버전의 LabVIEW 에서 업그레이드하기

새 버전의 LabVIEW 는 다른 디렉토리에 설치되므로, 새 버전의 LabVIEW 를 업그레이드하더라도 컴퓨터에 설치된 이전 버전의 LabVIEW 에 영향을 주지 않습니다. LabVIEW 5.x 와 이전 버전은 labview 디렉토리에 설치됩니다. LabVIEW 6.0 과 이후 버전은 labview x.x 디렉토리에 설치되며, x.x 는 버전 번호를 나타냅니다. LabVIEW 의 이전 버전을 제거하지 않고 LabVIEW 8.2 를 설치할 수 있습니다.

## LabVIEW 기존 버전 대체하기

기존의 LabVIEW 버전을 대체하려면, LabVIEW 의 기존 버전의 설치를 제거하고 LabVIEW 8.2 설치 프로그램을 실행한 후 설치 디렉토리를 이전 LabVIEW 버전이 설치된 동일한 labview 디렉토리로 설정하십시오.

**(Windows)** 또한 프로그램 제어판의 프로그램 추가 / 제거에서 현재의 LabVIEW 버전을 제거한 후 LabVIEW 8.2 를 설치할 수도 있습니다. 설치 제거 프로그램은 사용자가 labview 디렉토리에 생성한 파일을 삭제하지 않습니다.



### 노트

LabVIEW 를 제거하거나 다시 설치할 때, LabVIEW 는 .lib 파일에 저장한 모든 VI 와 컨트롤을 포함하여 vi.lib 디렉토리의 .lib 파일을 제거합니다. 사용자 컨트롤과 VI 를 **컨트롤**과 **함수** 팔레트에 추가하려면 user.lib 디렉토리에 사용자 컨트롤과 VI 를 저장하십시오.

## LabVIEW 이전 버전에서 환경 셋팅 복사하기

이전 버전의 LabVIEW 의 환경 셋팅을 사용하려면 이전 버전이 설치된 labview 디렉토리에서 LabVIEW 환경 파일을 복사하십시오.



### 주의

LabVIEW 8.2 의 환경 파일을 이전 버전의 환경 파일로 변경하였다면, 이전 버전 이후에 LabVIEW 에 추가된 환경 설정을 덮어쓸 것입니다.

LabVIEW 8.2 의 설치가 완료되면 LabVIEW 환경 파일을 LabVIEW 8.2 디렉토리로 복사합니다.

**(Windows)** LabVIEW 는 환경을 labview.ini 파일에 저장합니다.

**(Mac OS)** LabVIEW 는 환경을 기본 디렉토리 Library:Preferences 폴더의 LabVIEW Preferences 파일에 저장합니다.

(Linux) LabVIEW 는 환경을 홈 디렉토리의 .labviewrc 파일에 저장합니다 .

## LabVIEW 이전 버전에서 user.lib 파일 복사하기

LabVIEW 이전 버전의 user.lib 디렉토리에 있는 파일을 사용하려면 , 이전 버전이 설치된 labview 디렉토리에서 파일을 복사하십시오 . LabVIEW 8.2 설치가 완료되면 , 이 파일들을 LabVIEW 8.2 디렉토리의 user.lib 디렉토리에 복사합니다 .

## 업그레이드와 호환성 문제

---

특정 LabVIEW 버전에 대한 업그레이드와 호환성에 관련된 사항은 다음 섹션을 참조하십시오 .

유의사항 , 추가적인 호환성 문제 , LabVIEW 8.2 의 최신 추가 기능에 대한 자세한 내용은 labview 디렉토리의 readme.html 파일을 참조하십시오 .

## LabVIEW 8.0 에서 업그레이드하기

LabVIEW 8.0 에서 LabVIEW 8.2 로 업그레이드할 때 다음과 같은 호환성 문제가 발생할 수 있습니다 .

### 지원하는 플랫폼

LabVIEW 8.2 가 지원하는 플랫폼과 관련하여 다음과 같은 변경사항이 있습니다 :

- LabVIEW 8.2 는 Windows XP x64 를 지원하지 않습니다 .
- LabVIEW 8.2 는 Mac OS X 10.3.8 또는 이전 버전을 지원하지 않습니다 .
- LabVIEW 8.2 는 Intel 프로세서가 내장된 Macintosh 컴퓨터를 일부 지원합니다 . National Instruments 웹 사이트 ([ni.com/info](http://ni.com/info)) 에서 정보 코드 macintel 을 입력하면 Macintosh 지원에 대한 더 많은 정보를 얻을 수 있습니다 .

### 시스템 사양

(Mac OS) LabVIEW 8.2 는 LabVIEW 최소 설치시 최소 500 MB 의 빈 디스크 공간이 필요하며 , LabVIEW 전체 설치시에는 700 MB 의 빈 디스크 공간이 필요합니다 .

(Linux) LabVIEW 8.2 는 LabVIEW 최소 설치시 최소 430 MB 의 빈 디스크 공간이 필요하며 , LabVIEW 전체 설치시에는 620 MB 의 빈 디스크 공간이 필요합니다 .

## 인쇄된 문서

LabVIEW 8.2 에서 다음의 문서는 변경되지 않았습니다. 따라서 LabVIEW 8.2 에 적용된 변경사항이 문서 내용에 반영되지 않을 수도 있습니다.

- *LabVIEW 도움 카드*
- *LabVIEW 기본 매뉴얼-LabVIEW 기본 매뉴얼은 LabVIEW 도움말에서 기본* 모음의 일부입니다. 따라서, 업데이트된 내용에 대한 정보는 *LabVIEW 도움말 목차* 탭의 **기본** 모음을 참조하십시오.

## VI 와 함수 동작 변경 내역

다음 VI 와 함수의 동작이 LabVIEW 8.2 에서 변경되었습니다.

### 어플리케이션 인스턴스 사이에서 통신하기

LabVIEW 8.2 에서는 { 큐 얻기 }, { 알림자 얻기 }, { 사용자 이벤트 생성 }, { 세마포어 생성 }, { 랑데부 생성 } 함수를 사용하여 LabVIEW 어플리케이션 인스턴스 사이에서 통신할 수 없습니다. 하나의 어플리케이션 인스턴스에서 큐, 알림자, 사용자 이벤트, 세마포어, 또는 랑데부 참조를 구하거나 생성하면, 다른 어플리케이션 인스턴스에서 해당 참조를 사용할 수 없습니다.

### 원래의 고유벡터로 변환 VI

{ 원래의 고유벡터로 변환 } VI 의 **작업**, **낮은 인덱스**, **높은 인덱스** 및 **스케일** 입력은 필수 입력입니다.

### DataSocket 쓰기 함수

LabVIEW 8.0.1 에서 { DataSocket 쓰기 } 함수의 기본 동작은 비동기형으로 변경되었습니다. LabVIEW 8.0 및 LabVIEW 8.2 가 컴퓨터에 설치되어 있는 경우, `labview\examples\Shared Variable` 디렉토리의 DataSocket API Client VI 예제는 사용자가 해당 VI 를 정지시킬 때 에러를 반환합니다. LabVIEW 8.2 에서 이 예제를 사용하려면 LabVIEW 8.0 을 LabVIEW 8.0.1 로 업데이트해야 합니다.

### 파일 I/O VI

{ 스프레드시트 파일에 쓰기 } VI 와 { 스프레드시트 파일로부터 읽기 } VI 는 다형성 VI 입니다. { 스프레드시트 파일에 쓰기 } VI 는 사용자가 **포맷** 입력에 연결한 값에 맞춥니다. { 스프레드시트 파일로부터 읽기 } VI 는 다음의 인스턴스를 포함합니다: DBL, I64, 문자열.

### GPIO 상태 함수

LabVIEW 8.0 에서 { GPIO 상태 } 함수는 **에러 입력**이 에러를 수신하는 경우 실행되지 않았습니다. LabVIEW 8.2 의 경우, { GPIO 상태 } 함수는 **에러 입력**이 에러를 수신하더라도 항상 실행됩니다.

## 히스토그램 VI

( 히스토그램 ) VI 의 기본 **간격** 입력이 10 으로 변경되었습니다 .

## VI 참조 열기 함수

(VI 참조 열기 ) 함수의 **옵션** 입력의 기본 동작은 사용자에게 참조된 VI 에서 찾을 수 없는 subVI 를 검색하도록 입력 요청을 하는 것입니다 . 새 값인 0x20 은 **찾기** 대화 상자를 디스플레이하지 않거나 또는 사용자가 참조된 VI 의 찾을 수 없는 subVI 를 검색하도록 입력 요청하지 않게 지정합니다 .

## 다항식 근 VI

**P(x)** 가 제로가 아닌 상수일 때 , ( 다항식 근 ) VI 는 에러를 반환하지 않습니다 . 그러나 , **P(x)** 가 0 이면 , ( 다항식 근 ) VI 는 에러 -20111 을 반환합니다 . 이 VI 의 입력 다항식 계수는 모두 제로가 될 수 없습니다 .

## 램프 패턴 VI

( 램프 패턴 ) VI 에서 **샘플**이 1 이고 **끝 제외?** 가 참인 경우 , VI 는 **시작**의 한 원소를 가진 배열을 에러 없이 반환합니다 . LabVIEW 8.0 에서 VI 는 이러한 조건에서 에러를 반환합니다 .

## 레지스트리 값 읽기 VI

LabVIEW 8.0 은 VI 가 패턴화된 문자열 배열에 사용하는 REG\_MULTI\_SZ 문자열 포맷팅을 올바르게 않게 처리했습니다 . 이 문제 때문에 사용자는 ( 레지스트리 값 읽기 ) VI 에서 이 데이터 타입을 처리하기 위해 분석기를 작성해야 했습니다 . LabVIEW 8.2 에서 ( 레지스트리 값 읽기 ) VI 가 ( 레지스트리 값 쓰기 ) VI 에서 사용되는 포맷과 같은 포맷으로 이 데이터 타입을 반환합니다 . 이제 사용자는 직접 분석기를 추가할 필요가 없습니다 . LabVIEW 8.2 에서 이러한 VI 와 사용자 분석기를 함께 사용하면 ( 레지스트리 값 읽기 ) VI 가 잘못된 데이터를 반환하는 원인이 됩니다 .

## 웨이브폼 리샘플 ( 한번 ) VI

( 웨이브폼 리샘플 ( 한번 ) ) VI 의 **열린 간격?** 입력의 기본값이 참에서 거짓으로 변경되었으며 , 기본으로 닫힌 간격이 선택됩니다 . 이에 따라 기존의 코드를 업데이트하지 않으면 , VI 는 예상되는 결과를 반환하지 않을 수도 있습니다 .

## 사운드 VI

( 사운드 입력 읽기 ) 와 ( 단순한 사운드 파일 읽기 ) VI 에서 , **데이터** 출력의 **10** 구성요소는 읽은 첫번째 샘플의 타임스탬프를 반환합니다 . LabVIEW 는 첫번째 샘플을 읽는 최초 시간의 근사치를 구합니다 .

이제 연속 사운드 태스크에서 사운드를 멈추기 위해서 ( 사운드 출력 정지 ) VI 를 호출할 필요가 없습니다 .

(사운드 출력 기다림) VI는 **연속 샘플** 모드와 **유한 샘플** 모드에서 작동합니다.

## 웨이브폼 VI

LabVIEW 8.2는 다음 웨이브폼 VI에 대한 변경사항을 포함합니다:

- 기본 레벨 트리거 검출 VI—이 VI의 양쪽 인스턴스에서, **기울기** 입력이 **트리거 기울기**로 변경되었습니다.
- 웨이브폼 부분 얻기 VI—다음의 인스턴스를 포함합니다: WDT 웨이브폼 부분 얻기 DBL, WDT 웨이브폼 부분 얻기 CDB, WDT 웨이브폼 부분 얻기 EXT, WDT 웨이브폼 부분 얻기 I16, WDT 웨이브폼 부분 얻기 I32, WDT 웨이브폼 부분 얻기 I8, WDT 웨이브폼 부분 얻기 SGL. **시작 / 지속 기간 포맷** 입력에는 **절대 시간** 옵션이 더 이상 포함되어 있지 않습니다. **시작** 입력이 **시작 샘플 / 시간**으로 변경되었으며, **실제 시작** 출력은 **실제 시작 샘플 / 시간**으로 변경되었습니다.
- 웨이브폼 시간 배열 얻기 VI—**X 배열** 출력은 배정도 부동소수 데이터 타입에서 타임스탬프 데이터 타입으로 변경되었습니다.
- Y 값 얻기 VI—이 VI와 그에 대응하는 다형성 인스턴스는 XY 값 얻기로 변경되었습니다. (XY 값 얻기) VI에는 이제 **X 값** 출력이 포함되며, **데이터 값** 출력은 **Y 값**으로 변경되었습니다.
- 웨이브폼 샘플 개수 VI—이 VI는 다음의 인스턴스가 있는 다형성 VI입니다: WDT 웨이브폼 샘플 개수 DBL, WDT 웨이브폼 샘플 개수 CDB, WDT 웨이브폼 샘플 개수 EXT, WDT 웨이브폼 샘플 개수 I16, WDT 웨이브폼 샘플 개수 I32, WDT 웨이브폼 샘플 개수 I8, WDT 웨이브폼 샘플 개수 SGL.
- 파일로부터 웨이브폼 읽기 VI—에러가 파일 끝 에러인 경우, **에러 출력** 출력의 에러 상태를 참으로 반환합니다.
- 부분 대체 VI—**시작** 입력이 **시작 샘플 / 시간**으로 변경되었으며, **실제 시작 값** 출력은 **실제 시작 샘플 / 시간**으로 변경되었습니다.
- 디지털 패턴 찾기 VI—**시작** 입력은 **시작 인덱스 / 시간**으로 변경되었습니다.
- 웨이브폼 찾기 VI—**최적 피팅 시간** 및 **피팅 시간** 출력은 배정도 부동소수 데이터 타입에서 타임스탬프 데이터 타입으로 변경되었습니다.
- 웨이브폼 최소 최대 VI—**최소 시간** 및 **최대 시간** 출력은 배정도 부동소수 데이터 타입에서 타임스탬프 데이터 타입으로 변경되었습니다.
- 웨이브폼을 XY 쌍으로 VI—**XY** 쌍 출력의 **x** 원소는 배정도 부동소수 데이터 타입에서 타임스탬프 데이터 타입으로 변경되었습니다.

## 프로퍼티, 메소드, 이벤트 동작 변경 내역

다음 프로퍼티, 메소드, 이벤트의 동작이 LabVIEW 8.2에서 변경되었습니다:

- ActiveX VI 참조 얻기 메소드의 **옵션** 입력의 기본 동작은 사용자에게 참조된 VI의 찾을 수 없는 subVI를 검색하도록 입력 요청을 하는 것입니다.

다. 새 값인 0x20 은 **찾기** 대화 상자를 디스플레이하지 않거나 또는 사용자가 참조된 VI 의 찾을 수 없는 subVI 를 검색하도록 입력 요청하지 않게 지정합니다.

- 프로젝트에서 열려 있지 않은 라이브러리에 공유 변수를 추가하려는 경우, 프로젝트 아이템 클래스의 아이템 추가 메소드가 에러를 반환합니다.
- VI 실행 메소드의 **참조 자동 삭제** 입력이 참이고 메소드가 에러를 반환하는 경우, LabVIEW 는 참조를 삭제하지 않습니다.
- 어플리케이션 : 언어 프로퍼티의 유효한 값에 zh-cn 이 포함되어 있습니다. 이는 중국어 간체가 LabVIEW 환경 언어임을 나타냅니다.
- LabVIEW 8.0 에서, 참조에 의해 배열 데이터 타입을 전달하는 .NET 메소드는 모든 데이터를 참조 번호 데이터 타입으로 전달합니다. 참조에 의해 배열 데이터 타입을 전달하는 .NET 메소드는 데이터를 실제 데이터 타입으로 전달합니다.
- 디지털 테이블, 여러 열 리스트박스, 테이블, 트리 컨트롤 클래스의 편집 위치 프로퍼티는 (-2, -2) 의 값을 반환하여 사용자가 컨트롤 텍스트를 편집하는 중이 아님을 나타냅니다. 리스트박스 클래스의 행 편집 프로퍼티는 -2 의 값을 반환하여 사용자가 컨트롤 텍스트를 편집하는 중이 아님을 나타냅니다.
- LabVIEW 8.0 에서 패널 업데이트 연기 프로퍼티는 서브패널에 있는 컨트롤패널의 업데이트를 연기하지 않았습니다. LabVIEW 8.2 에서는 패널 업데이트 연기 프로퍼티가 서브패널에도 작동합니다.
- 어플리케이션 인스턴스 닫기 및 어플리케이션 인스턴스 닫기? 이벤트는 어플리케이션 종료 및 어플리케이션 종료? 이벤트를 대체합니다. LabVIEW 프로젝트 밖에서 실행 중인 VI 에서 어플리케이션 인스턴스 닫기 이벤트를 사용하는 경우, 사용자 인터페이스를 통해서 또는 프로그램적으로 LabVIEW 를 종료할 때 LabVIEW 는 이벤트를 생성합니다. LabVIEW 는 사용자 인터페이스를 통해 LabVIEW 를 종료할 때 어플리케이션 인스턴스 닫기? 이벤트를 생성합니다. 사용자가 LabVIEW 프로젝트 내에서 실행 중인 VI 를 위해 어플리케이션 인스턴스 닫기 및 어플리케이션 인스턴스 닫기? 이벤트를 등록하면, LabVIEW 는 어플리케이션 인스턴스를 닫거나 LabVIEW 를 종료할 때 이벤트를 생성합니다.

## 삭제된 프로퍼티, 메소드, 이벤트

LabVIEW 8.2 는 다음의 프로퍼티, 메소드, 이벤트를 지원하지 않습니다.

- LabVIEW 8.2 는 커넥터 팬 프로퍼티를 지원하지 않습니다.
- LabVIEW 8.x 는 변수 클래스에서 데이터 타입 프로퍼티를 지원하지 않습니다. 대신에 변수 클래스의 데이터 타입 (배리언트) 프로퍼티를 사용합니다.

## 이름이 변경된 프로퍼티, 메소드, 이벤트

LabVIEW 8.2 에서 다음의 프로퍼티, 메소드, 이벤트의 이름이 변경되었습니다 :

클래스	LabVIEW 8.0 이름	LabVIEW 8.2 이름	타입
어플리케이션	슬레이브 연결 끊기	LVRT: 슬레이브 연결 끊기	메소드
어플리케이션	어플리케이션 종료	어플리케이션 인스턴스 닫기	이벤트
어플리케이션	어플리케이션 종료 ?	어플리케이션 인스턴스 닫기 ?	이벤트
강도 그래프, 혼합 신호 그래프, 웨이브폼 그래프	커서 팔레트 보이기	커서 범례 보이기	프로퍼티
라이브러리	라이브러리 태그 삭제	라이브러리 태그 : 삭제	메소드
라이브러리	아이콘 얻기	아이콘 : 얻기	메소드
라이브러리	라이브러리 태그 얻기	라이브러리 태그 : 얻기	메소드
라이브러리	라이브러리 태그 이름 얻기	라이브러리 태그 : 이름 얻기	메소드
라이브러리	잠금 상태 얻기	잠금 상태 : 얻기	메소드
라이브러리	소스 영역 얻기	소스 영역 : 얻기	메소드
라이브러리	저장	저장 : 라이브러리	메소드
라이브러리	복사본 저장	저장 : 복사본	메소드
라이브러리	아이콘 설정	아이콘 : 설정	메소드
라이브러리	라이브러리 태그 설정	라이브러리 태그 : 설정	메소드
라이브러리	잠금 상태 설정	잠금 상태 : 설정	메소드
라이브러리	소스 영역 설정	소스 영역 : 설정	메소드
리스트박스, 여러 열 리스트박스, 트리 컨트롤	끌기 / 놓기 : 아이템 끌기 허용	끌기 / 놓기 : 끌기 허용	프로퍼티
프로젝트 아이템	태그 삭제	태그 : 삭제	프로퍼티
프로젝트 아이템	태그 얻기	태그 : 태그 얻기	프로퍼티
프로젝트 아이템	태그 이름 얻기	태그 : 이름 얻기	프로퍼티
프로젝트 아이템	XML 태그 얻기	태그 : XML 태그 얻기	프로퍼티
프로젝트 아이템	태그 설정	태그 : 태그 설정	프로퍼티
프로젝트 아이템	XML 태그 설정	태그 : XML 태그 설정	프로퍼티

클래스	LabVIEW 8.0 이름	LabVIEW 8.2 이름	타입
프로젝트 아이템	라이브러리 아이템 타입 문자열	라이브러리 아이템 타입 : 문자열	프로퍼티
프로젝트 아이템	라이브러리 아이템 타입	라이브러리 아이템 : 타입	프로퍼티

## LabVIEW 7.x 에서 업그레이드하기

LabVIEW 7.x 에서 LabVIEW 8.2 로 업그레이드할 때 다음과 같은 호환성 문제가 발생할 수 있습니다 . 업그레이드시 발생할 수 있는 다른 문제점에 대해서는 이 문서의 [LabVIEW 8.0 에서 업그레이드하기](#) 섹션을 참조하십시오 .

각 버전의 새로운 기능 및 변경 내역에 대한 추가적인 정보는 [ni.com/manuals](http://ni.com/manuals) 에서 LabVIEW 버전 7.x 부터 8.0 까지 각각의 버전에 대한 [LabVIEW 업그레이드 노트](#)를 참조하십시오 .



### 노트

*LabVIEW 도움 카드*와 *LabVIEW 기본 매뉴얼*은 LabVIEW 8.2 에서 변경되지 않았습니다 . 해당 문서의 PDF 버전은 labview\manuals 디렉토리에 저장할 수 있습니다 . 관련 문서에 대한 추가적인 정보는 이 문서의 [LabVIEW 8.0 에서 업그레이드하기](#) 섹션을 참조하십시오 .

## 지원하는 플랫폼

LabVIEW 8.x 가 지원하는 플랫폼과 관련하여 다음과 같은 변경사항이 있습니다 :

- LabVIEW 7.1 과 이후 버전은 Windows Me/98/95 를 지원하지 않습니다 . LabVIEW 8.x 는 Windows NT 를 지원하지 않습니다 .
- LabVIEW 8.x 는 Mac OS X 10.2 또는 이전 버전을 지원하지 않습니다 .
- LabVIEW 8.x 는 Sun Solaris 를 지원하지 않습니다 .

## 시스템 사양

LabVIEW 7.x 의 최소 RAM 사양은 128MB 이지만 , National Instruments 는 256MB 의 RAM 을 권장합니다 . LabVIEW 8.x 의 최소 RAM 사양은 256MB 이지만 , National Instruments 는 512MB 의 RAM 을 권장합니다 .

LabVIEW 7.x 는 800 × 600 픽셀의 화면 해상도를 요구하지만 , National Instruments 는 1,024 × 768 픽셀의 화면 해상도를 권장합니다 .

LabVIEW 8.x 는 1,024 × 768 픽셀의 화면 해상도를 요구합니다 .

## Windows

LabVIEW 7.x 는 최소 Pentium III 또는 그 이상 , Celeron 600 MHz 또는 동등한 프로세서를 요구하지만 , National Instruments 는 Pentium 4 또는 이와 동등한 프로세서를 권장합니다 . LabVIEW 8.x 는 최소 Pentium III, Celeron 866 MHz, 또는 동등한 프로세서를 요구하지만 , National Instruments 는 Pentium 4/M 또는 동등한 프로세서를 권장합니다 .

LabVIEW 7.x 는 LabVIEW 최소 설치시 최소 130 MB 의 빈 디스크 공간이 필요하며 , LabVIEW 전체 설치시 550 MB 의 빈 디스크 공간이 필요합니다 .  
LabVIEW 8.x 는 전체 설치시 1.2 GB 의 빈 디스크 공간이 필요합니다 .

## Mac OS

LabVIEW 7.x 는 LabVIEW 최소 설치시 최소 280 MB 의 빈 디스크 공간이 필요하며 , LabVIEW 전체 설치시 350 MB 의 빈 디스크 공간이 필요합니다 .  
LabVIEW 8.2 는 LabVIEW 최소 설치시 최소 500 MB 의 빈 디스크 공간이 필요하며 , LabVIEW 전체 설치시 700 MB 의 빈 디스크 공간이 필요합니다 .

## Linux

LabVIEW 7.x 는 최소 Pentium III 또는 그 이상 , Celeron 600 MHz 또는 동등한 프로세서를 요구하지만 , National Instruments 는 Pentium 4 또는 이와 동등한 프로세서를 권장합니다 . LabVIEW 8.x 는 최소 Pentium III, Celeron 866 MHz, 또는 동등한 프로세서를 요구하지만 , National Instruments 는 Pentium 4/M 또는 동등한 프로세서를 권장합니다 .

LabVIEW 7.x 는 LabVIEW 최소 설치시 최소 200 MB 의 빈 디스크 공간이 필요하며 , LabVIEW 전체 설치시 300 MB 의 빈 디스크 공간이 필요합니다 .  
LabVIEW 8.2 는 LabVIEW 최소 설치시 최소 430 MB 의 빈 디스크 공간이 필요하며 , LabVIEW 전체 설치시 620 MB 의 빈 디스크 공간이 필요합니다 .

LabVIEW 7.x 는 GNU C 라이브러리 (glibc) 2.1.3 또는 이후 버전을 요구하지만 , National Instruments 는 GNU C 라이브러리 2.2.4 또는 이후 버전을 권장합니다 . LabVIEW 8.x 는 GNU C 라이브러리 2.2.4 또는 이후 버전을 요구합니다 .

LabVIEW 7.x 는 Red Hat Linux 7.0 또는 이후 버전 , Mandrake Linux 8.0 또는 이후 버전 , SuSE Linux 7.1 또는 이후 버전 , Debian Linux 3.0 또는 이후 버전에서 동작합니다 . LabVIEW 8.x 는 Red Hat Enterprise Linux WS 3 또는 이후 버전 , MandrakeLinux/Mandriva 10.0 또는 이후 버전 , SuSE Linux 9.1 또는 이후 버전에서 동작합니다 .

## 사용자 팔레트 보기

LabVIEW 8.x 는 사용자 팔레트 보기를 지원하지 않습니다 . 사용자 팔레트 보기를 사용하지 않고도 팔레트 세트를 편집할 수 있습니다 . LabVIEW 8.0 에서 팔레트 변경 내역에 대한 추가적인 정보는 National Instruments 웹 사이트 [ni.com/info](http://ni.com/info) 에서 정보 코드 1v8palette 를 입력합니다 .

# VI 와 함수 동작 변경 내역

LabVIEW 7.1 또는 8.0 에서 다음 VI 와 함수의 동작이 변경되었습니다 .

## .NET VI 와 어플리케이션

LabVIEW 8.x 에서 .NET 함수와 어플리케이션을 사용하려면 .NET Framework 1.1 Service Pack 1 또는 이후 버전을 가지고 있어야 합니다 . .NET Framework 1.1 Service Pack 1 을 설치하기 전에 반드시 Microsoft .NET Framework 1.1 Hotfix KB886904 를 제거해야 합니다 .

LabVIEW 7.x 에서 마지막으로 저장한 .NET VI 를 로드하는 경우 , LabVIEW 8.x 는 어셈블리 파일이 VI 와 동일한 디렉토리에 있거나 LabVIEW 7.x 에서 **Tools»Advanced»NET Assembly References** 를 선택하여 어셈블리를 등록한 경우에도 어셈블리를 찾도록 입력 요청할 수 있습니다 .

## 분석 VI 알고리즘

LabVIEW 7.1 과 이후 버전에서 분석 VI 는 BLAS/LAPACK 알고리즘을 사용합니다 . 이 VI 는 더욱 정확한 결과를 생성합니다 . LabVIEW 8.x 에서 이 VI 는 **수학**과 **신호 처리** 팔레트에 있습니다 .

## 신호 추가 익스프레스 VI

LabVIEW 7.x 에서 , Append Signals VI 의 **Input Signal A** 가 비어있거나 연결되지 않은 상태에서 단일 신호 또는 결합된 신호를 **Input Signal B** 에 연결하는 경우 , **Appended Signals** 출력은 비어 있습니다 . LabVIEW 8.x 에서 , **입력 신호 A** 가 비어있거나 연결되지 않은 상태에서 단일 신호를 **입력 신호 B** 에 연결한 경우 , 익스프레스 VI 는 **입력 신호 B** 를 반환합니다 . 결합된 신호만을 **입력 신호 B** 에 연결하는 경우 , 결합된 신호의 각 신호는 다음 신호를 추가하여 결과로 하나의 신호를 생성합니다 .

## 비교 함수

LabVIEW 7.x 나 그 이전 버전에서는 배리언트 데이터를 비교하기 위해 비교 함수를 사용할 때 , LabVIEW 는 먼저 두 배리언트의 전체 길이를 비교하고 그 배리언트를 비트 단위로 비교합니다 . LabVIEW 8.x 는 타입 코드를 가지고 배리언트 데이터의 비교를 시작하는데 , 이는 배리언트의 실제 타입 정보를 인코딩한 후 다른 타입 특성의 속성을 비교합니다 .

## 내적 VI

LabVIEW 7.0 에서 Dot Product VI 는 다음 방정식을 사용하여 입력 벡터 X 와 Y 의 내적을 계산합니다 :

$$X*Y = \sum_{i=0}^{n-1} x_i y_i$$

LabVIEW 7.1 과 이후 버전에서 ( 내적 ) VI 는 다음 방정식을 사용하여 복소수 입력의 내적을 계산합니다 :

$$X * Y = \sum_{i=0}^{n-1} X_i Y_i^*$$

여기서 ,  $y_i^*$  는  $y_i$  에 대한 켈레 복소수입니다 .

## Easy Text Report VI (Mac OS 와 Linux)

Easy Text Report VI 의 커백트 팬이 변경되었습니다 . LabVIEW 8.x 에서 , LabVIEW 7.x 또는 이전 버전에서 마지막으로 저장되었던 Easy Text Report VI 를 사용한 VI 를 열 때 , subVI 에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **Relink to SubVI** 를 선택합니다 .

## 문자열로 포맷 함수

LabVIEW 7.x 에서 , %o 또는 %x 포맷 지정자 구문 원소를 Format Into String 함수와 함께 사용하면 입력을 문자열로 변환하기 전에 부동소수 입력을 32 비트 정수로 반올림합니다 .

LabVIEW 8.x 에서 , 이 포맷 지정자 구문 원소는 ( 문자열로 포맷 ) 함수가 입력을 문자열로 변환하기 전에 부동소수 입력을 64 비트 정수로 반올림하도록 합니다 .

## 숫자 결합 함수

LabVIEW 7.x 와 이전 버전에서 , Join Numbers 함수는 32 비트 정수 입력을 16 비트 정수로 강제 변환하여 하나의 32 비트 정수를 생성합니다 .

LabVIEW 8.x 의 ( 숫자 결합 ) 함수는 32 비트 정수 입력을 결합하여 하나의 64 비트 정수를 생성합니다 .



### 노트

LabVIEW 7.x VI 를 LabVIEW 8.x 에서 여는 경우 , LabVIEW 는 32 비트 정수 입력을 16 비트 정수로 강제 변환합니다 .

## 수학 VI 와 행렬

LabVIEW 8.x 에서 , **수학 VI** 는 행렬 데이터 타입을 지원합니다 .

LabVIEW 8.x 에서 LabVIEW 7.x 의 VI 를 로드하고 해당 VI 에 행렬 데이터 타입을 사용할 수 있는 함수에 연결된 수학 VI 가 포함된 경우 , 함수에 붉은 7.x 문양이 나타나 함수가 LabVIEW 7.x 의 동작을 사용함을 나타냅니다 .

## 숫자를 문자열로 변환 함수

LabVIEW 7.x 에서 , Number to Hexadecimal String, Number to Octal String, Number to Decimal String 함수는 입력을 문자열로 변환하기 전에 부동소수 입력을 32 비트 정수로 반올림합니다 .

LabVIEW 8.x 에서 , 이 함수는 입력을 문자열로 변환하기 전에 부동소수 입력을 64 비트 정수로 반올림합니다 . 그러나 LabVIEW 7.x VI 를 LabVIEW 8.x 에서 여는 경우 , LabVIEW 는 부동소수 입력을 32 비트 정수로 반올림하여 호환성과 기능을 유지합니다 .

## VI 참조 열기 함수

LabVIEW 7.x 에서 , Open VI Reference 함수의 **vi path** 입력이 경로이고 메모리에 같은 이름의 VI 가 존재하는 경우 , 메모리 상의 VI 경로가 지정한 경로와 일치하지 않더라도 LabVIEW 는 메모리 상의 VI 참조를 반환합니다 .

LabVIEW 8.x 에서 (VI 참조 열기 ) 의 **vi 경로** 입력이 문자열이면 , LabVIEW 는 **vi 경로**가 타겟 메모리 상에 있는 VI 의 유효한 파일 이름과 일치할 때만 VI 를 엽니다 . **vi 경로**가 경로인 경우 , LabVIEW 는 동일한 타겟의 같은 경로를 가진 VI 를 메모리에서 검색합니다 . LabVIEW 가 일치하는 경로를 가진 VI 를 찾지 못하는 경우 , LabVIEW 는 디스크의 지정된 경로에서 VI 를 로드합니다 . LabVIEW 가 파일을 찾을 수 없거나 파일이 메모리와 타겟 상의 다른 VI 와 충돌한 경우에는 에러가 발생합니다 .

## 빠른 스케일 VI

LabVIEW 7.1 과 그 이전 버전에서 , Quick Scale 1D VI 또는 Quick Scale 2D VI 의 **X** 입력이 제로의 배열인 경우 , 이 VI 는 **max|X|** 를 **0** 으로 , **Y(i)=X(i)/Max|X|** 또는 **Yij=Xij/Max|X|** 를 NaN 의 배열로 반환합니다 . LabVIEW 8.x 에서 , ( 빠른 스케일 ) VI 의 **X** 입력이 제로의 배열인 경우 , 이 VI 는 **max|X|** 를 **0** 으로 , **Y(i)=X(i)/Max|X|** 또는 **Yij=Xij/Max|X|** 를 제로의 배열로 반환합니다 .

## 키 읽기 VI

LabVIEW 7.x 및 이전 버전에서는 , Read Key VI 를 사용하여 Shift-JIS 에서 인코딩된 멀티바이트 문자의 문자열을 읽습니다 . **multibyte encoding** 입력에 1 또는 <Shift-JIS> 를 연결해야 합니다 . LabVIEW 8.x 에서 , OS 로 캄을 적절한 인코딩으로 설정해 놓으면 ( 키 읽기 ) VI 는 기본적으로 멀티바이트 문자의 인코딩된 문자열을 읽습니다 .

## 스케일 VI

LabVIEW 7.1 과 그 이전 버전에서 , Scale 1D VI 또는 Scale 2D VI 의 **X** 입력이 제로의 배열인 경우 , 이 VI 는 **scale** 을 **0** 으로 , **offset** 을 **0** 으로 , **Y=(X-offset)/scale** 을 NaN 의 배열로 반환합니다 . LabVIEW 8.x 에서 , ( 스케일 ) VI 의 **X** 입력이 제로의 배열인 경우 , 이 VI 는 **스케일** 을 **1** 로 , **오프셋** 을 **0** 으로 , **Y=(X-오프셋)/스케일** 을 제로의 배열로 반환합니다 .

## 세마포어 VI

LabVIEW 7.x 에서 , Release Semaphore VI 와 Acquire Semaphore VI 는 **error input** 입력이 에러를 받을 때 실행을 시도하지 않습니다 . LabVIEW 8.x 의 경우 이 VI 는 **에러 입력** 입력이 에러를 받더라도 실행을 시

도합니다. 그러나, LabVIEW 7.x VI 를 LabVIEW 8.x 에서 열 경우, LabVIEW 는 LabVIEW 7.x 의 특징을 유지합니다.

## SMTP E- 메일 VI

LabVIEW 7.x 및 이전 버전에서는, 값을 SMTP Email VI 의 **character set** 입력에 연결하여 문자 세트를 지정할 수 있습니다. LabVIEW 8.x 에서는, (SMTP E- 메일 ) VI 는 메시지가 시스템 문자 세트에 있다고 가정합니다. 이 VI 는 E- 메일을 보내기 전에 메시지를 UTF-8 포맷으로 인코딩합니다. (SMTP E- 메일 ) VI 에는 더 이상 **문자 세트** 또는 **문자 변환** 파라미터가 없습니다.

## 복소수 정렬 VI

LabVIEW 7.x 와 이전 버전에서 Sort Complex Numbers VI 의 **method** 입력을 **Magnitude** 로 설정할 경우 LabVIEW 는 동일한 크기를 가진 원소들의 순서를 바꾸지 않습니다. LabVIEW 8.x 에서 **메소드**를 **크기**로 설정하면, LabVIEW 는 동일한 크기를 가진 원소를 처음에는 실수 부분에 대하여 정렬하고 다음으로 허수 부분에 대하여 정렬합니다.

## 단위 벡터 VI

LabVIEW 7.x 와 이전 버전에서, Unit Vector VI 는 다음 방정식을 사용하여 입력 벡터의 노름을 계산합니다 :

$$\|X\| = \sqrt{x_0^2 + x_1^2 + \dots + x_{n-1}^2}$$

LabVIEW 8.x 에서, ( 단위 벡터 ) VI 는 다음 방정식을 사용하여 입력 벡터의 노름을 계산합니다 :

$$\|X\| = \left( |x_0|^y + |x_1|^y + \dots + |x_{n-1}|^y \right)^{\frac{1}{y}}$$

이 때 X 는 입력 벡터이고  $\|X\|$  는 노름, y 는 노름 타입입니다.

## 사용자 VI

labview\help, labview\project, labview\wizard 디렉토리에 놓여지는 VI 는 각각 **도움말**, **도구**, **파일** 메뉴에 나타납니다. LabVIEW 7.x 또는 이전 버전에서 이 디렉토리에 놓여지는 VI 는 LabVIEW 8.x 에서 의도대로 작동하지 않을 수 있습니다. 왜냐하면 LabVIEW 8.0 및 이후 버전은 이 VI 를 프라이빗 어플리케이션 인스턴스에서 열기 때문입니다.

labview\vi.lib\Utility\allVIsInMemory.llb 의 VIMemory Get VIs in Memory VI 를 사용하여 모든 어플리케이션 인스턴스에서 메모리의 모든 사용자 VI 리스트를 생성합니다. labview\vi.lib\Utility\allVIsInMemory.llb 의 Get User Application Reference VI 를 사용하여 현재 어플리케이션 인스턴스에 대한 참조를 생성합니다. 어플리케이션 인스턴스에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.

## 삭제된 VI 와 함수

LabVIEW 8.x 는 다음의 VI 와 함수를 지원하지 않습니다 :

- LabVIEW 7.1 과 그 이후 버전은 Polynomial Real Zero Counter VI 를 설치하지 않습니다 . 대신 ( 실수 다항식의 제로 카운터 ) VI 를 사용하십시오 .
- **(Mac OS)** LabVIEW 7.1 과 그 이후 버전은 PPC VI 를 설치하지 않습니다 . 대신 TCP VI 를 사용하십시오 .
- LabVIEW 8.x 는 (QR 인수분해 ) VI 를 지원하지 않습니다 . 대신 (QR 분해 ) VI 를 사용하십시오 .
- LabVIEW 8.x 는 더이상 ( 레벤버그 - 마르카토 ) VI 또는 ( 비선형 레벤버그 - 마르카토 피팅 ) VI 를 지원하지 않습니다 . 대신에 ( 비선형 커브 피팅 ) VI 를 사용하십시오 .
- LabVIEW 8.x 에서는 **함수** 팔레트에 VISA Status Description 함수가 없습니다 . 대신 ( 단순 에러 핸들러 ) 또는 ( 일반 에러 핸들러 ) VI 를 사용하십시오 .
- LabVIEW 8.x 는 ( 카이 제공 분포 ), ( F 분포 ), ( 정규분포 ) 및 ( T 분포 ) VI 를 지원하지 않습니다 . 대신에 ( 연속적인 누적 분포 함수 ) VI 에서 카이제공 , F, 정규 , 스튜던트 t 인스턴스를 사용합니다 .
- LabVIEW 8.x 는 ( 역 카이제공 분포 ), ( 역 F 분포 ), ( 역 정규분포 ), ( 역 T 분포 ) VI 를 지원하지 않습니다 . 대신에 ( 연속적인 역 누적 분포 함수 ) VI 에서 카이제공 , F, 정규 , 스튜던트 t 인스턴스를 사용합니다 .
- LabVIEW 8.x 에서 ( 1D 선형식 계산 ) VI 와 ( 2D 선형식 계산 ) VI 는 더 이상 **함수** 팔레트에 있지 않습니다 . 대신에 ( 선형 계산 ) VI 를 사용하십시오 .
- LabVIEW 8.x 에서 ( 1D 다항식 계산 ) VI 와 ( 2D 다항식 계산 ) VI 는 더 이상 **함수** 팔레트에 있지 않습니다 . 대신에 ( 다항식 계산 ) VI 를 사용하십시오 .
- LabVIEW 8.x 에서 ( 1D 직각좌표를 극좌표로 ) 와 ( 2D 직각좌표를 극좌표로 ) VI 는 더 이상 **함수** 팔레트에 있지 않습니다 . 대신에 ( 실수 / 허수를 극좌표로 ) 함수와 ( 극좌표를 실수 / 허수로 ) 함수를 사용합니다 .
- LabVIEW 8.x 에서 , ( 하모닉 분석기 ) VI 는 더 이상 **함수** 팔레트에 있지 않습니다 . THD 나 **구성요소** 출력을 측정할 때는 ( 하모닉 왜곡 분석기 ) VI 를 사용하고 , SINAD 나 THD **더하기 노이즈** 출력을 측정할 때는 ( SINAD 분석기 ) VI 를 사용합니다 .
- LabVIEW 8.x 에서 , ( 네트워크 함수 ( 평균 )) VI 는 더 이상 **함수** 팔레트에 있지 않습니다 . 대신에 ( 주파수 응답 함수 ( 크기 - 위상 )) , ( 주파수 응답 함수 ( 실수 - 허수 )) , ( 크로스 스펙트럼 ( 크기 - 위상 )) , 또는 ( 크로스 스펙트럼 ( 실수 - 허수 )) VI 를 사용합니다 .
- LabVIEW 8.x 에서 , ( 펄스 파라미터 ) VI 는 더 이상 **함수** 팔레트에 있지 않습니다 . **슬루 속도** , **지속기간** , **오버샷** , **프리샷** 출력을 측정하려면 ( 변이 측정 ) VI 를 사용하고 , **주기** , **펄스 지속기간** , **주기 점유율** 출력을 측정하려면 펄스 측정 VI 를 사용하고 , **진폭** , **높은 상태 레벨** , **낮은 상태 레벨** 출력을 측정하려면 진폭과 레벨 VI 를 사용합니다 .

- LabVIEW 8.x 에서 ( 전달 함수 ) VI 는 **함수** 팔레트에 있지 않습니다 . 대신에 ( 주파수 응답 함수 ( 크기 - 위상 )) 또는 ( 주파수 응답 함수 ( 실수 - 허수 )) VI 를 사용합니다 .
- LabVIEW 8.x 에서 , NI DIAdem Report Wizard Express VI 는 **함수** 팔레트에 있지 않습니다 . 대신 ( DIAdem 리포트 ) 익스프레스 VI 를 사용합니다 .
- LabVIEW 8.x 에서 VISA 리소스 이름 상수와 IVI 논리적 이름 상수는 더 이상 **함수** 팔레트에 있지 않습니다 . VISA 리소스 이름을 지정하기 위해 VISA VI 의 **VISA 리소스 이름**을 사용합니다 . IVI 논리적 이름을 지정하기 위해 인스트루먼트를 초기화하는 적절한 드라이버 VI 의 적절한 입력을 사용합니다 .
- LabVIEW 8.x 에서 ( 에러 링 상수 ) 는 **함수** 팔레트에 있지 않습니다 . 원하는 에러 코드를 입력하는 대신에 32 비트 부호있는 정수를 사용하십시오 .
- **(Windows 와 Linux)** LabVIEW 8.x 에서는 , LabVIEW 7.x 의 **Sound** 팔레트에 있던 Sound VI 가 **함수** 팔레트에 있지 않습니다 . 대신에 LabVIEW 8.x 에서 사운드 VI 를 사용합니다 . LabVIEW 7.x 와 함께 출시된 예제는 LabVIEW 8.x 와 함께 출시되지 않습니다 .
- **(Mac OS)** LabVIEW 7.1 에서 제공된 Sound VI 는 LabVIEW 8.x 에서도 계속 함께 출시됩니다 . LabVIEW 7.x 와 함께 출시된 예제는 LabVIEW 8.x 와 함께 출시되지 않습니다 .

## 파일 I/O VI 와 함수

LabVIEW 8.x 에서 ( 파일로부터 문자 읽기 ) VI 는 **함수** 팔레트에 있지 않습니다 . 대신에 ( 텍스트 파일에서 읽기 ) 함수를 사용하십시오 .

LabVIEW 8.x 에서 ( 파일 열기 / 생성 / 대체 ) VI 는 **함수** 팔레트에 있지 않습니다 . 대신에 ( 파일 열기 / 생성 / 대체 ) 함수를 사용하십시오 . 다음 함수는 LabVIEW 7.x 및 이전 버전의 Open/Create/Replace File VI 의 일부 기능을 포함합니다 :

- ( 파일 크기 얻기 ) 함수를 사용하여 파일의 크기를 결정합니다 .
- ( 파일 대화 상자 ) 익스프레스 VI 를 사용하여 파일 대화 상자를 위한 파일이나 디렉토리의 시작 경로 , 파일 패턴 , 기본 이름을 지정합니다 .
- ( 참조 번호를 경로로 ) 함수를 사용하여 참조를 경로로 변환합니다 .
- ( 2 진 파일에 쓰기 ) 함수를 사용하여 플랫폼에 독립적인 텍스트 파일이나 다른 타입의 2 진 파일을 생성하고 , ( 2 진 파일에서 읽기 ) 함수를 사용하여 2 진 파일의 결과값을 읽습니다 .

LabVIEW 8.x 에서 Read File 와 Write File 함수는 **함수** 팔레트에 있지 않습니다 . 대신에 ( 2 진 파일에서 읽기 ) 와 ( 2 진 파일에 쓰기 ) 함수를 사용하십시오 .

LabVIEW 8.x 에서 Write Characters to File VI 는 **함수** 팔레트에 있지 않습니다 . 대신에 ( 텍스트 파일에 쓰기 ) 함수를 사용하십시오 .

LabVIEW 8.x에서는 **함수** 팔레트에 Access Rights 함수가 없습니다. 대신에 ( 권한 얻기 ) 와 ( 권한 설정 ) 함수를 사용합니다.

LabVIEW 8.x에서, EOF 함수는 **함수** 팔레트에 있지 않습니다. 대신에 ( 파일 크기 얻기 ) 와 ( 파일 크기 설정 ) 함수를 사용합니다.

LabVIEW 8.x에서는 **함수** 팔레트에 List Directory 함수가 없습니다. 대신에 ( 폴더 열거 ) 함수를 사용하십시오.

LabVIEW 8.x에서는 **함수** 팔레트에 Lock Range 함수가 없습니다. 대신에 ( 접근 거부 ) 함수를 사용하십시오.

블록다이어그램에서 새 디렉토리 함수를 포함한 LabVIEW 7.x에서 만들어진 VI를 열 경우, LabVIEW 8.x는 해당 함수를 ( 폴더 생성 ) 함수로 대체합니다. **경로** 입력에 지정된 폴더가 존재하지 않으면, 폴더 생성 함수는 새 디렉토리 함수가 했던 것처럼 에러를 반환하기보다는 디렉토리를 생성합니다.

LabVIEW 8.x에서는 **함수** 팔레트에 Seek 함수가 없습니다. 대신에 ( 파일 위치 얻기 ) 와 ( 파일 위치 설정 ) 함수를 사용합니다.

LabVIEW 8.x에서는 **함수** 팔레트에 Type and Creator 함수가 없습니다. 대신에 ( 타입과 생성자 얻기 ) 와 ( 타입과 생성자 설정 ) 함수를 사용합니다.

LabVIEW 8.x에서는 **함수** 팔레트에 Volume Info 함수가 없습니다. 대신에 ( 볼륨 정보 얻기 ) 함수를 사용합니다.

LabVIEW 8.x에서는 **함수** 팔레트에 Open File and New File 함수가 없습니다. ( 파일로부터 라인 읽기 ) VI는 **함수** 팔레트에 없지만 호환성을 위해 LabVIEW와 함께 출시됩니다.

LabVIEW 8.x에서 (116 파일로부터 읽기 ), (SGL 파일로부터 읽기 ), (116 파일에 쓰기 ) 와 (SGL 파일에 쓰기 ) VI는 **함수** 팔레트에 있지 않습니다. 대신에 (2진 파일에서 읽기 ) 와 (2진 파일에 쓰기 ) VI를 사용하십시오.

## 프로퍼티, 메소드, 이벤트 동작 변경 내역

LabVIEW 7.1 또는 8.0에서 다음 프로퍼티, 메소드, 이벤트의 동작이 변경되었습니다.

### 어플리케이션 프로퍼티와 메소드

LabVIEW 8.x에서 어플리케이션 프로퍼티와 메소드의 동작은 소속되어 있는 어플리케이션 인스턴스에 따라 다릅니다. 예를 들어, 어플리케이션 : 메모리상의 모든 VI 프로퍼티는 사용자가 사용하는 어플리케이션 인스턴스에 따라 달라집니다. 이 프로퍼티는 해당 프로퍼티와 같은 어플리케이션 인스턴스에서 메모리에 있는 모든 VI 리스트를 반환합니다. 그러나 어플리케이션의 동작 : 디렉토리 경로 프로퍼티는 사용자가 사용하는 어플리케이션 인스턴스에 달려 있지 않습니다. 이 프로퍼티는 어플리케이션이 위치해 있는 디렉토리에 대한 절대 경로를 반환합니다. 이 정보는 각 어플리케이션 인스턴스에서 변하지 않습니다.

어플리케이션 인스턴스에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.

## 프런트패널 : 열기 메소드

LabVIEW 7.0 Open FP 메소드는 LabVIEW 7.1 에서 Old Open FP 로 이름이 바뀌었습니다. LabVIEW 7.1 에는 프런트패널이 이미 열려있는 경우 에러를 반환하지 않는 다른 Open FP 메소드가 포함되어 있습니다.

LabVIEW 7.1 Open FP 메소드는 LabVIEW 8.x 에서 프런트 패널 : 열기로 이름이 바뀌었습니다. LabVIEW 7.0 의 Old Open FP 메소드를 사용하는 VI 를 가지고 있는 경우, 프런트패널 : 열기 메소드로 대체하십시오.

## VI 실행 메소드

LabVIEW 7.1 에서 Run VI 메소드의 **Auto Dispose Ref** 입력를 참으로 설정하면, VI 가 실행을 멈춘 후에도 LabVIEW 는 자동으로 참조를 삭제합니다. LabVIEW 8.x 및 이후 버전에서는, 메소드가 에러를 반환한 경우에도 LabVIEW 가 참조를 즉시 삭제합니다. 블록 다이어그램의 일부가 이 참조에 의존할 경우 실행 중에 이 동작에 의해 VI 가 깨질 수도 있습니다.

## 키 다운과 키 반복 이벤트

키 다운, 키 다운?, 키 반복, 키 반복? VI 이벤트와 컨트롤의 **V 키** 데이터 필드는 현재 키보드의 문자와 숫자 섹션에서 <Return> 키와 숫자 키패드에서 <Enter> 키에 대해 별개의 값을 가지고 있습니다. LabVIEW 7.x 와 이전 버전에서, <Enter> 키 또는 <Return> 키가 이 이벤트 중 하나를 생성할 때, LabVIEW 는 **VKey** 데이터 필드에 <Enter> 를 반환합니다. LabVIEW 8.x 에서, <Enter> 키 또는 <Return> 키가 이 이벤트 중 하나를 생성할 때, LabVIEW 는 **V 키** 데이터 필드에 각각 <Enter> 또는 <Return> 을 반환합니다.

(Mac OS) LabVIEW 8.x 는 바로 가기 메뉴에 대해 <Control>- 클릭만 받으며 <Command>- 클릭 키 조합은 받지 않습니다. 이벤트 구조를 사용하여 이 동작을 따라하려면, 새 동작에 맞게 VI 를 수정해야 합니다.

## 리스트박스 프로퍼티

LabVIEW 7.x 와 이전 버전에서, 리스트박스의 최상위 행 프로퍼티를 리스트박스의 제일 아래쪽 아이템 밑의 행으로 설정하면, LabVIEW 는 마지막으로 보이는 아이템으로 행을 고정합니다. LabVIEW 8.x 에서는, 리스트박스에 보이는 아이템의 숫자가 이 프로퍼티에 연결할 수 있는 행의 숫자를 제한하지 않습니다.

LabVIEW 8.x 는 단일 열 리스트박스에서 더블 클릭 프로퍼티를 지원하지 않습니다. 대신 더블 클릭된 열 열기 메소드를 사용합니다.

## 객체소유 VI 프로퍼티

LabVIEW 7.x 와 이전 버전에서는 Owning VI 프로퍼티는 객체가 속한 VI 에 대한 참조를 반환합니다 . 이 참조는 VI 를 메모리에 유지합니다 .  
LabVIEW 8.x 에서는 객체소유 VI 프로퍼티가 반환하는 참조는 메모리의 VI 를 유지하지 않습니다 . 객체소유 VI 가 메모리에서 삭제될 경우 , 이 참조는 유효하지 않게 됩니다 . (VI 참조 열기 ) 함수를 사용하여 참조를 명확하게 닫기 전까지 메모리에 있는 VI 에 대한 참조를 얻습니다 .

## 텍스트 프로퍼티

LabVIEW 7.x 와 이전 버전에서는 Text 프로퍼티가 문자열을 일반 디스플레이로 반환합니다 . LabVIEW 8.x 에서 , 텍스트 프로퍼티는 프런트패널 객체에서 디스플레이되는 동일한 텍스트로 문자열을 반환합니다 . 예를 들어 , 문자열 컨트롤을 양호 디스플레이로 디스플레이한 경우 , 텍스트 프로퍼티는 문자열을 양호 디스플레이로 반환합니다 .

## 트리 컨트롤 프로퍼티

LabVIEW 7.x 와 이전 버전에서 , Active Cell Properties:Cell Size:Height 와 Active Cell Properties:Cell Size:Width 프로퍼티는 트리 컨트롤에서 각 라인에 대해 17 픽셀을 반환합니다 . LabVIEW 8.x 에서 , 활성 셀 : 셀 크기 : 높이와 활성 셀 : 셀 크기 : 폭 프로퍼티는 트리 컨트롤에서 각 라인에 대해 16 픽셀을 반환합니다 .

## VI 문자열 메소드

VI 문자열 반출 메소드를 사용하여 LabVIEW 의 이전 버전에서 반출하는 문자열은 VI 문자열 : 반입 메소드를 사용할 때 LabVIEW 8.x 에서 적절하게 반입하지 못할 수 있습니다 .

## 삭제된 프로퍼티 , 메소드 , 이벤트

LabVIEW 8.x 는 다음의 프로퍼티 , 메소드 , 이벤트를 지원하지 않습니다 :

### 커서 프로퍼티

LabVIEW 8.x 는 Cursor Lock Style 프로퍼티를 지원하지 않습니다 . 대신 커서 모드 프로퍼티를 사용하십시오 .

### 리스트박스 , 테이블 , 디지털 테이블 , 트리 컨트롤 프로퍼티와 이벤트

LabVIEW 8.x 는 여러 열 리스트박스에서 셀 전경색 프로퍼티를 지원하지 않습니다 . 활성 셀 : 셀 폰트 : 색 프로퍼티를 대신 사용하십시오 .

LabVIEW 8.x 는 테이블 또는 디지털 테이블에서 셀 전경색 프로퍼티를 지원하지 않습니다 . 대신 테이블과 디지털 테이블에 활성 셀 : 셀 폰트 : 색 프로퍼티를 사용하십시오 .

LabVIEW 8.x 는 트리 컨트롤에서 활성 셀 : 전경색 프로퍼티를 지원하지 않습니다 . 활성 셀 : 셀 폰트 : 색 프로퍼티를 대신 사용하십시오 .

LabVIEW 8.x 는 트리 컨트롤 클래스에서 끌기 , 끌기 ? , 놓기 , 놓기 ? 이벤트를 지원하지 않습니다 . 대신에 컨트롤 클래스에서 끌기 종료 , 끌기 입력 , 끌기 이탈 , 끌기 통과 , 끌기 소스 업데이트 , 끌기 시작 , 끌기 시작 ? , 끌기 이벤트를 사용하십시오 .

## 이름있는 숫자 프로퍼티

LabVIEW 8.x 는 이름있는 숫자 객체의 이름있는 숫자 색 , 이름있는 숫자 색 : 배경색 , 이름있는 숫자 색 : 텍스트 색을 더 이상 지원하지 않습니다 . 텍스트 색 , 텍스트 색 : 배경색 , 텍스트 색 : 텍스트 색 프로퍼티를 각각 대신해서 사용하십시오 .

## 패널 프로퍼티

LabVIEW 8.x 는 구획 클래스의 색 프로퍼티를 지원하지 않습니다 . LabVIEW 8.x 에서 이 프로퍼티를 사용하면 , 프로퍼티는 왼쪽 상단의 구획에만 적용됩니다 . 대신에 구획 클래스의 구획 색 프로퍼티를 사용하십시오 .

## 서브패널 프로퍼티

LabVIEW 8.x 에서 서브패널의 subVI 팬을 사용하여 서브패널 컨트롤 스크롤 막대의 보이기를 설정하고 서브패널 컨트롤에서 프런트패널을 스케일합니다 .

LabVIEW 8.x 는 서브패널 컨트롤의 X 스크롤 막대 보이기 프로퍼티를 지원하지 않습니다 . 대신에 팬에 수평 스크롤 막대 보이기 프로퍼티를 사용하십시오 .

LabVIEW 8.x 는 서브패널 컨트롤의 Y 스크롤 막대 보이기 프로퍼티를 지원하지 않습니다 . 대신에 팬에 수직 스크롤 막대 보이기 프로퍼티를 사용하십시오 .

LabVIEW 8.x 는 서브패널 컨트롤의 패널 스케일 프로퍼티를 지원하지 않습니다 . 대신에 팬에 스케일링 모드 설정 메소드를 사용하십시오 .

## VI 프로퍼티 , 메소드 , 이벤트

LabVIEW 8.x 는 프런트패널 윈도우 : 자동 중심 설정 프로퍼티를 지원하지 않습니다 . 대신 프런트패널 : 중심 메소드를 사용하십시오 .

LabVIEW 8.x 는 프런트패널 윈도우 : 스크린에 맞춤 프로퍼티를 지원하지 않습니다 . 대신 프런트패널 윈도우 : 상태 프로퍼티를 사용하십시오 .

LabVIEW 8.x 는 VI 클래스에서 프런트패널 윈도우 : 원점 프로퍼티를 지원하지 않습니다 . LabVIEW 8.x 에서 이 프로퍼티를 사용하면 , 프로퍼티는 왼쪽 상단의 구획에만 적용됩니다 . 대신에 구획 클래스의 원형 프로퍼티를 사용하십시오 .

LabVIEW 8.x는 VI 클래스에서 프런트패널 윈도우; 스크롤 막대 표시 프로퍼티를 지원하지 않습니다. LabVIEW 8.x에서 이 프로퍼티를 사용하면, 프로퍼티는 왼쪽 상단의 구획에만 적용됩니다. 대신에 구획 클래스에서 수평 스크롤 막대 보이기와 수직 스크롤 막대 프로퍼티를 사용하십시오.

LabVIEW 8.x는 VI 클래스에서 프런트패널 스케일링 모드 열기나 프런트패널 스케일링 모드 설정 메소드를 지원하지 않습니다. LabVIEW 8.x에서 이 메소드를 사용하면, 메소드는 왼쪽 상단의 구획에만 적용됩니다. 대신에 구획 클래스에서 스케일링 모드 열기와 스케일링 모드 설정 메소드를 사용합니다.

LabVIEW 8.x는 VI 클래스에서 마우스 다운, 마우스 다운?, 마우스 입력, 마우스 커서 이탈, 마우스 이동, 마우스 업 이벤트를 지원하지 않습니다. 대신에 구획 클래스에서 각각 마우스 다운, 마우스 다운?, 마우스 입력, 마우스 커서 이탈, 마우스 이동, 마우스 업 이벤트를 사용하십시오.

## 어플리케이션 아이템 태그

다음의 어플리케이션 아이템 태그는 LabVIEW 8.x에 존재하지 않습니다:

- APP\_BUILD\_STANDALONE\_APP
- APP\_DN\_ASSEMBLY\_REFS
- APP\_EDIT\_VI\_LIBRARY
- APP\_SAVE\_WITH\_OPTIONS
- APP\_SHOW\_CLIPBOARD
- APP\_SRC\_CODE\_CTRL
- APP\_SWITCH\_EXEC\_TARGET
- APP\_UPDATE\_VXI
- APP\_VIEW\_PRINTED\_MANUALS

LabVIEW의 이전 버전으로 저장되어 있는 런타임 메뉴(.rtm) 파일을 사용하고 그 파일에 삭제된 태그가 있는 경우, LabVIEW 8.x는 **메뉴 편집기** 대화 상자에서 그 파일을 저장할 때 .rtm 파일에서 자동적으로 그 태그를 삭제합니다. 삭제된 어플리케이션 아이템 태그는 LabVIEW에 의해 지정되었고 사용자 태그로 사용될 수 없습니다.

## HiQ 지원

National Instruments는 8.x에서 HiQ 기능을 지원하지 않습니다. 어플리케이션이 HiQ VI를 사용하는 경우, 이를 수학과 신호 처리 VI로 대체하는 것을 고려하십시오.

## 에러 리스트 윈도우

LabVIEW 7.x와 이전 버전에서, **Error list** 윈도우의 **VI List** 섹션은 메모리의 모든 VI의 에러를 보여줍니다. LabVIEW 8.x에서, **에러 리스트** 윈도우의 **에러가 있는 아이템** 섹션은 VI 및 라이브러리와 같은 메모리 상의 모든

아이템의 에러를 보여줍니다. 둘 이상의 아이템이 같은 이름을 가지고 있는 경우, 이 섹션은 각 모호한 아이템에 대한 특정 어플리케이션 인스턴스를 보여줍니다. 어플리케이션 인스턴스에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.

## VI 문자열 파일 구문

LabVIEW 8.x는 **도구 » 고급 » 문자열 반입**을 선택하거나 VI 문자열 : 반입 메소드를 사용하여 VI 문자열 파일을 반입할 때, 새 태그의 세트인 <GROUPER></GROUPER>를 검색합니다. 이 태그 세트는 함께 그룹화된 프런트패널 객체를 나타냅니다. 그러므로 LabVIEW 8.x에서는 LabVIEW의 이전 버전에서 저장된 VI 문자열 파일을 반입할 수 없습니다.

LabVIEW 7.1과 이전 버전은 리스트박스 문자열을 프라이빗 데이터의 <ITEMS> 섹션에 나열합니다. LabVIEW 8.x는 리스트박스 문자열을 프라이빗 데이터의 <STRINGS> 섹션에 나열합니다. 또한, LabVIEW 7.1과 이전 버전에서 리스트박스는 한가지 폰트만 사용할 수 있습니다. LabVIEW는 이를 프라이빗 데이터의 <LBLABEL> 섹션에 나열합니다. LabVIEW 8.x의 리스트박스는 여러가지 폰트를 사용할 수 있습니다. LabVIEW는 이를 프라이빗 데이터의 <CELL\_FONTS> 섹션에 나열합니다.

LabVIEW 7.1과 이전 버전은 기본 데이터에 여러 열 리스트박스 문자열을 나열합니다. 그러나, 여러 열 리스트박스의 기본 데이터는 정수이거나 정수의 배열입니다. LabVIEW 8.x는 프라이빗 데이터에 여러 열 리스트박스 문자열을 보여줍니다.

LabVIEW 7.1과 이전 버전은 트리 컨트롤에 문자열이나 폰트를 반출하지 않습니다. LabVIEW 8.x는 트리 컨트롤 문자열과 폰트를 모두 반출하는데, 리스트박스 및 여러 열 리스트박스와 동일한 포맷으로 문자열과 폰트를 반출합니다.

LabVIEW 8.x에서 반환된 파일의 각 라인은 적어도 프라이빗 혹은 기본 데이터에 해당하는 두 개의 태그를 포함합니다. 또한, LabVIEW 8.x는 각 레벨의 아이템을 한번에 한단계 아래로 이동시킬 수 있습니다.

다음 단계를 따라 VI 문자열 파일을 LabVIEW 8.x 포맷으로 변환합니다.

1. LabVIEW의 이전 버전에서 VI 문자열 파일을 반입하기
2. VI를 저장합니다.
3. LabVIEW 8.x에서 VI 로드
4. **도구 » 고급 » 문자열 반출**을 선택하여 VI 문자열 파일을 LabVIEW 8.x 포맷으로 저장합니다.

## LabVIEW 7.x 로 ( 혹은 로부터 ) 타입 설명자 데이터 변환하기

LabVIEW 가 타입 설명자를 저장하는 포맷이 LabVIEW 8.x 에서 변경되었습니다 . LabVIEW 7.x 는 16 비트의 패턴 형태로 타입 설명자를 저장합니다 . LabVIEW 8.x 는 32 비트의 패턴 형태로 타입 설명자를 저장합니다 . 이 변화때문에 타입 설명자의 64 KB 크기 제한이 없어졌습니다 .

LabVIEW 8.x 는 LabVIEW 7.x 로 작성된 타입 설명자를 읽고 LabVIEW 7.x 가 읽을 수 있는 타입 설명자를 쓰는 메커니즘을 제공합니다 . ( 패턴화된 문자열로 ) 함수는 **7.x 데이터 변환** 바로 가기 메뉴 아이템을 가지고 있습니다 . 이 함수에서 마우스 오른쪽 버튼을 클릭하고 이 메뉴 항목을 선택할 경우 , 함수는 입력 데이터를 LabVIEW 7.x 에서 쓰여진 것과 같이 처리합니다 . **7.x 데이터 변환** 바로 가기 메뉴 아이템을 선택하고 **데이터 문자열** 출력이 연결되면 , LabVIEW 8.x 는 함수에 붉은색 7.x 문양을 표시하여 데이터가 LabVIEW 7.x 포맷으로 변환되거나 해당 포맷으로부터 변환된 것을 나타냅니다 . 데이터의 변환을 방지하려면 , 바로 가기 메뉴에서 **7.x 데이터 변환**을 다시 선택하여 확인 표시를 제거하십시오 .

LabVIEW 8.x 에서 LabVIEW 7.x 또는 이전 버전에서 마지막으로 저장한 VI 를 로드하면 , LabVIEW 8.x 는 자동으로 ( 패턴화된 문자열로 ) 함수에 **7.x 데이터 변환** 속성을 설정합니다 . 이 함수는 계속해서 LabVIEW 7.x 와 이전 버전에서처럼 동작합니다 . VI 가 LabVIEW 8.x 타입 설명자 포맷을 사용하려는 경우 , ( 패턴화된 문자열로 ) 함수에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **7.x 데이터 변환**을 선택하여 확인 표시를 제거합니다 . VI 가 LabVIEW 7.x 또는 이전 버전에서 쓰여진 데이터를 포함하는 파일을 수정할 필요가 없거나 LabVIEW 7.x 또는 이전 버전에서 실행되는 VI 로 데이터를 제공하거나 받지 않는 경우 , LabVIEW 8.x 의 타입 설명자 포맷을 사용해야 합니다 . 앞으로 나올 LabVIEW 버전에서는 이전의 타입 설명자 포맷의 지원이 중단될 수도 있습니다 .

## LabVIEW 내장 소스 컨트롤 제공자에서 타사 소스 컨트롤 제공자로 파일 이동하기

LabVIEW 7.x 와 이전 버전의 내장 소스 컨트롤 제공자는 LabVIEW 8.x 에서 사용할 수 없습니다 . LabVIEW 에서 소스 컨트롤을 사용하려면 반드시 타사 소스 컨트롤 제공자를 선택해야 합니다 . 이전 버전에서 내장 제공자를 사용한 경우 , LabVIEW 에서 소스 컨트롤을 사용하려면 파일을 다른 제공자로 이동해야 합니다 . LabVIEW 에서 지원하는 타사 소스 컨트롤 제공자의 가장 최근 리스트를 찾으려면 내셔널인스트루먼트 웹 사이트 [ni.com/info](http://ni.com/info) 에서 정보 코드 exgucn 을 입력합니다 .

파일을 새 소스 컨트롤 제공자로 이동할 때 , 내장되어 있는 제공자에 저장된 개정 히스토리를 잃게 됩니다 . 파일의 이전 버전을 새 제공자에 이동할 수 없습니다 .

다음 단계를 따라 내장 소스 컨트롤을 제공자에서 타사 소스 컨트롤을 제공자로 파일을 이동합니다.

1. LabVIEW의 이전 버전에서, 모든 사용자가 LabVIEW 내장 소스 컨트롤을 제공자에 포함된 파일을 체크 인했는지 확인합니다.
2. 파일을 새 소스 컨트롤을 제공자에 추가하려는 컴퓨터에서, 내장 제공자를 사용하여 모든 파일의 최신 버전을 얻습니다.
3. 내장 제공자를 사용하여 소스 컨트롤에서 파일을 체크 아웃합니다.
4. 타사 소스 컨트롤을 제공자에서, 새 소스 컨트롤을 프로젝트에 사용하려는 셋팅을 설정합니다.
5. LabVIEW를 타사 소스 컨트롤을 제공자와 함께 작동하도록 설정합니다. LabVIEW가 타사 소스 컨트롤을 제공자와 함께 작동하도록 설정하는 것에 대한 정보는 *LabVIEW 도움말* **목차** 탭의 **기본 » 프로젝트 구성하기 와 관리하기 » 사용법 » LabVIEW에서 소스 컨트롤을 사용하기** 모음을 참조하십시오.
6. LabVIEW 프로젝트를 생성합니다. 내장 제공자에 포함된 파일을 프로젝트에 추가합니다. LabVIEW가 입력 요청을 하면 파일을 소스 컨트롤에 추가합니다. 또한, 타사 제공자로부터 파일을 직접 추가할 수 있습니다. LabVIEW 프로젝트 생성에 대한 정보는 *LabVIEW 도움말* **목차** 탭의 **기본 » 프로젝트 구성하기와 관리하기 » 사용법 » LabVIEW 프로젝트 생성하기** 모음을 참조하십시오.

## NaN 문자열을 정수 타입으로 변환하기 (Windows)

LabVIEW 7.x에서 명백하게 또는 함축적으로 NaN을 정수로 변환할 때, 값은 해당 정수 데이터 타입의 가장 작은 값이 됩니다. 예를 들어, NaN을 16 비트 부호있는 정수로 변환하면 16 비트 부호있는 정수의 가장 작은 값인 -32,768을 생성합니다.

LabVIEW 8.x에서 명백하게 또는 함축적으로 NaN을 정수로 변환할 때, 값은 해당 정수 데이터 타입의 가장 큰 값이 됩니다. 예를 들어, NaN을 16 비트 부호있는 정수로 변환하면 16 비트 부호있는 정수의 가장 큰 값인 32,767을 생성합니다.

## 케이스 구조에 연결된 상수

LabVIEW 7.x과 이전 버전에서는, 케이스 구조에 상수를 연결하고 실행되지 않는 케이스에 subVI를 놓아 메모리에 subVI를 보존할 수 있습니다. 예를 들어, 케이스 구조에 참 상수를 연결하고 케이스 구조의 거짓 케이스에 subVI를 놓은 경우, LabVIEW는 VI 호출과 함께 subVI를 로드합니다. LabVIEW 8.x는 실행되지 않는 모든 코드를 제거합니다. 그러므로 LabVIEW의 이전 버전으로 저장된 VI를 케이스 구조에 연결된 상수와 함께 LabVIEW 8.x에서 로드하는 경우, LabVIEW는 상수를 숨겨진 컨트롤로 변경하여 LabVIEW 이전 버전의 동작을 유지합니다.

## OS 메시지 지연

LabVIEW 7.x 에서 , LabVIEW 는 .NET 과 ActiveX 이벤트를 다루기 위해 콜백 VI 를 실행하는 동안 OS 메시지를 처리합니다 . LabVIEW 8.x 에서 LabVIEW 는 콜백 VI 가 실행을 멈추거나 모달 대화 상자를 로드할 때까지 OS 메시지의 처리를 지연시킵니다 . 이 지연에 의해 콜백 VI 는 간섭없이 실행할 수 있고 , LabVIEW 는 다른 이벤트 내에서 이벤트가 실행되어 교착 상태에 빠질 수 있는 것을 막을 수 있습니다 .

콜백 VI 로부터 모달이 아닌 대화 상자에 대해 동기화된 호출을 만들 수 없습니다 . 모달이 아닌 대화 상자에서 VI 실행 메소드를 실행하고 이 메소드의 **완료까지 기다림** 입력에 거짓 불리언 상수를 연결하여 콜백 VI 로부터 비동기적으로 모달이 아닌 대화 상자를 호출해야 합니다 .

LabVIEW 7.x 에서 , LabVIEW 는 DLL 이나 공유 라이브러리 함수를 실행하는 동안 OS 메시지를 처리합니다 . LabVIEW 8.x 에서 LabVIEW 는 DLL 함수에 대한 호출이 끝날 때까지 또는 DLL 로부터 모달 대화 상자를 로드할 때까지 OS 메시지의 처리를 지연시킵니다 . 이 지연에 의해 DLL 함수는 간섭 없이 실행할 수 있고 , LabVIEW 는 DLL 함수가 실행되고 있는 동안 같은 DLL 을 호출하는 것을 막을 수 있습니다 .

이 기본 동작을 사용하면 DLL 이 실행되는 동안 모달이 아닌 대화 상자에 대해 동기화된 호출을 만들 수 없습니다 . 모달이 아닌 대화 상자에서 VI 실행 메소드를 실행하고 이 메소드의 **완료까지 기다림** 입력에 거짓 불리언 상수를 연결하여 DLL 로부터 비동기적으로 이 대화 상자를 호출해야 합니다 .

생성한 DLL 에서 OS 메시지를 지연시킬지를 선택할 수 있습니다 . **프로젝트 탐색기** 윈도우의 DLL 에서 마우스 오른쪽 버튼을 클릭하고 , 바로 가기 메뉴에서 **프로퍼티**를 선택하고 , **항목** 리스트에서 **고급**을 선택하고 , **공유 라이브러리에서 OS 메시지 지연** 확인란에서 확인 표시를 제거하여 DLL 함수가 실행되고 있는 동안 OS 메시지를 처리합니다 .

## 리소스 관리자 (Mac OS)

LabVIEW 7.x 와 이전 버전은 Macintosh 리소스 파일을 읽고 쓸 수 있는 문서화되지 않은 기능을 제공합니다 . LabVIEW 8.x 에서는 이 메소드가 더 이상 존재하지 않습니다 . 이 문서화되지 않은 기능을 사용하는 유틸리티가 더 이상 작동하지 않으므로 VI 에서 Macintosh 리소스 파일을 읽거나 쓸 수 없습니다 .

## 단일 및 두 개 버튼 대화 상자

LabVIEW 7.x 와 이전 버전에서 , 단일 버튼 대화 상자 또는 두 개 버튼 대화 상자를 디스플레이하는 VI 를 프로그램적으로 강제 종료할 수 없습니다 . LabVIEW 8.x 에서는 강제 종료 VI 메소드를 사용하여 이러한 대화 상자를 디스플레이하는 VI 를 프로그램적으로 강제 종료할 수 있습니다 .

## 프로퍼티와 인보크 노드

LabVIEW 7.x 의 커서 범례로부터 내부적으로 링크된 프로퍼티 노드나 인보크 노드를 생성하면 , LabVIEW 는 LabVIEW 8.x 에서 VI 를 열 때 노드를 삭제합니다 .

## 공유 라이브러리 업데이트하기

labview.lib 에 링크되어 있는 LabVIEW 7.x 나 이전 버전에서 공유 라이브러리 (DLL) 를 만들면 , LabVIEW 8.x 에서는 그 공유 라이브러리를 labviewv.lib 에 연결합니다 . 공유 라이브러리를 labviewv.lib 에 링크하는 것에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오 .

## 인쇄를 위한 여백 값

LabVIEW 7.x 와 이전 버전에서 **Options** 대화 상자의 **Printing** 페이지에서 지정한 **Margins** 옵션은 여백 값에 대해 센티미터를 사용합니다 . LabVIEW 8.x 에서 **여백** 옵션은 여백 값에 대해 밀리미터를 사용합니다 .

## LabVIEW 6.x 에서 업그레이드하기

LabVIEW 6.x 에서 LabVIEW 8.2 로 업그레이드할 때 다음과 같은 호환성 문제가 발생할 수 있습니다 . 업그레이드할 때 발생할 수 있는 문제에 대한 정보는 이 문서의 [LabVIEW 7.x 에서 업그레이드하기](#) 와 [LabVIEW 8.0 에서 업그레이드하기](#) 섹션을 참조하십시오 .

각 버전의 새로운 기능 및 변동 사항에 대한 추가적인 정보는 [ni.com/manuals](http://ni.com/manuals) 에서 LabVIEW 버전 6.x 부터 8.0 까지 각각의 버전에 대한 *LabVIEW 업그레이드 노트*를 참조하십시오 .

## 웨이브폼 데이터 타입의 변경

LabVIEW 7.0 에서는 웨이브폼 데이터 타입이 10 구성요소를 위해서 배정도 부동소수가 아닌 타임스탬프 데이터 타입을 사용합니다 . LabVIEW 6.x 에서 데이터 형태에 대한 정보를 포함하지 않고 웨이브폼 데이터를 파일로 저장했다면 , LabVIEW 7.x 와 이후 버전에서 데이터를 가져올 때 에러가 발생할 수 있습니다 .

LabVIEW 7.x 와 이후 버전에서 , **Read Waveform from File VI** 는 파일의 예전 웨이브폼 데이터 타입 포맷을 새로운 웨이브폼 데이터 타입 포맷으로 변환해 줍니다 . 이 VI 는 변환을 수락할 것인지에 대해 입력 요청하는 대화 상자를 디스플레이합니다 . LabVIEW 런타임 엔진에서 ( 파일로부터 웨이브폼 읽기 ) VI 는 변환을 수행하지 못하고 대신 에러를 반환합니다 . LabVIEW 6.x 에서 LabVIEW 7.x 와 이후 버전으로 웨이브폼 데이터 이동에 대한 추가적인 정보는 National Instruments 웹 페이지 [ni.com/info](http://ni.com/info) 에서 정보 코드 exd9zq 를 입력하여 참조하십시오 .

## 시리얼 호환 VI

LabVIEW 7.x 와 이전 버전에서 시리얼 호환 VI 는 **함수** 팔레트에 나타나지 않습니다 . VXI 장비와 통신하는 VI 를 작성하려면 VISA VI 와 함수를 사용하십시오 .

LabVIEW 7.x 와 이전 버전에서 LabVIEW 는 OS 의 시리얼 드라이버와 통신하기 위해 serpdrv 드라이버를 사용하지 않습니다 . LabVIEW 는 VISA 를 기반으로 하는 호환 VI 를 포함합니다 . 새 어플리케이션인 경우 VISA, 시리얼 VI, 함수를 사용하여 시리얼 장비를 컨트롤하십시오 . 시리얼 VI 를 포함한 이전 LabVIEW 버전에서 작성한 모든 VI 는 LabVIEW 7.1 과 이후 버전에서도 동작합니다 .

포트 번호의 매핑을 포트로 다시 설정한 경우, 매핑을 이 포트로 반드시 지정해 주어야 합니다 . 시리얼 포트 매핑을 지정하기 위해서 labview\vi.lib\Instr\\_sersup.llb 의 Set Serial Alias Ports VI 를 사용하십시오 . 이 VI 의 **VISA 가명** 입력에 문자열 배열을 연결하고 사용할 포트의 명칭을 입력 배열에 입력합니다 . 배열의 각 원소는 하나의 포트에 대응되어야 합니다 . 예를 들어, 포트 0 을 VISA 가명인 내 시리얼 포트에 매핑하려고 설정한 경우, **VISA 가명** 입력 배열의 첫 원소에 내 시리얼 포트를 기입합니다 . 이런 경우, (VISA 시리얼 포트 설정 ) VI 를 호출하기 전에 반드시 set serial alias ports VI 를 먼저 호출해야 합니다 .

VISA VI 및 함수를 사용하여 시리얼 인스트루먼트를 컨트롤하는 예제는 labview\examples\instr\smp1ser1.llb 를 참고하십시오 .

## 루프의 기본 데이터

LabVIEW 6.0 과 이전 버전에서는 For 루프가 실행되지 않을 경우 정의되지 않은 데이터를 발생시켰습니다 . LabVIEW 6.1 과 이후 버전의 For 루프는 카운터 터미널에 0 을 와이어링하거나 빈 배열을 For 루프에 연결하여 오토인덱싱을 활성화하면 기본값을 만듭니다 . 루프는 실행되지 않고 오토인덱싱이 비활성화된 모든 출력 터미널에는 이 터미널 데이터 타입의 기본값이 발생합니다 .

## 리모트 프론트패널 라이선스

LabVIEW Full Development System 과 어플리케이션 빌더는 한 명의 클라이언트가 원격으로 프론트패널을 모니터링하고 컨트롤할 수 있는 리모트 프론트패널 라이선스를 포함합니다 . LabVIEW Professional Development System 은 다섯 명의 클라이언트가 원격으로 프론트패널을 보고 컨트롤할 수 있도록 하는 리모트 프론트패널 라이선스를 포함합니다 .

더 많은 수의 클라이언트가 원격으로 리모트 프론트패널을 사용할 수 있도록 업그레이드할 수 있습니다 .

## 멀티스레드 할당

LabVIEW 7.1 과 이후 버전은 VI 를 실행하는데 있어서 LabVIEW 7.1 이전 버전보다 더 많은 스레드를 할당합니다. 이러한 변경 때문에, 호출한 DLL 이 실제 재호출이 아닌 경우 라이브러리 함수 호출 노드를 재호출로 잘못 지정하면 멀티스레드 에러가 발생합니다. 라이브러리 함수 호출 노드와 재호출에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.

LabVIEW 가 스레드를 할당하는 방법을 변경하려면 labview\vi.lib\Utility\sysinfo.llb 의 threadconfig VI 를 사용하십시오. 또한, **파일 » VI 프로퍼티**를 선택하고, **항목** 폴다운 메뉴에서 **실행**을 선택한 후, **재호출 실행** 확인란에서 확인 표시를 제거하여 VI 에 대한 재호출을 비활성화시킬 수 있습니다.

스레드 할당에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오.

## 인스트루먼트 드라이버

LabVIEW 7.x 나 이전 버전의 LabVIEW 패키지에는 인스트루먼트 드라이버가 포함된 LabVIEW 인스트루먼트 드라이버 라이브러리 CD 가 더 이상 포함되지 않습니다. National Instruments 의 인스트루먼트 드라이버 네트워크인 [ni.com/idnet](http://ni.com/idnet) 에서 인스트루먼트 드라이버를 다운로드하십시오. National Instruments 디바이스 드라이버 CD는 NI-DAQ, NI-VISA, 그리고 기타 National Instruments 드라이버를 포함합니다.

## 단위와 변환율

LabVIEW 7.x 와 이전 버전에서 Compound Arithmetic 함수를 사용하게 되면 여분의 단위를 제거하기 위해 Convert Unit 함수를 더 이상 사용할 필요가 없습니다.

LabVIEW 7.1 과 이후 버전의 변환율은 National Institute for Standards and Technology (NIST) 가 *국제단위계 (SI) 사용법에 대한 가이드*에서 제시한 가이드 라인을 더욱 정확하게 준수합니다. 또한, 칼로리 단위는 이제 칼로리 ( 열 ) 로, 마력은 마력 ( 전기 ) 로 사용됩니다. 이들 단위의 약자는 변경되지 않았습니다. 다음 테이블은 LabVIEW 6.1 과 7.x 및 이후 버전 사이에서 변경된 단위 변환율을 설명합니다.

단위	6.1 의 정의	7.x 및 이후 버전의 정의
천문학 단위 (AU)	149,498,845,000 m	149,597,900,000 m
영국 온도 단위 (평균)	1055.79 J	1055.87 J
전자 볼트 (eV)	1.602e-19 J	1.60217642e-19 J
푸트캔들 (조명도)	10.764 lx	10.7639 lx
마력 vs. 마력 (전기)	745.7 W	746 W. 새 변환값이 정확합니다.
영국 갤런	4.54596 l	4.54609 l

단위	6.1 의 정의	7.x 및 이후 버전의 정의
광년	9.4605 Pm	9.46073 Pm
파운드	4.448 N	4.448222 N
로드 ( 길이 )	502.92 cm	5.029210 m
슬러그 ( 질량 )	32.174 lb	14.59390 kg
합쳐진 원자 질량 (u)	1.66057e-27 kg	1.66053873e-27 kg

## 패널 업데이트 연기 프로퍼티

LabVIEW 6.1 과 이전 버전은 수정이 끝나지 않은 프론트패널 객체를 새로 그리기 위해서 패널 업데이트 연기 프로퍼티가 거짓으로 되기까지 기다려야 했습니다 . LabVIEW 7.0 과 이후 버전에서 , 이 프로퍼티에 참을 연결하면 LabVIEW 는 변경을 기다리는 모든 패널 객체를 다시 그리고 프론트패널 업데이트의 새로운 요청을 모두 연기합니다 . 일부 경우에 , 이 변경은 프론트패널의 수정된 객체를 필요 이상으로 다시 그리도록 유발할 수 있습니다 .

## 숫자 컨트롤의 데이터 범위

LabVIEW 6.1 과 이전 버전에서 , 특정 숫자 컨트롤은 기본 최소값이 0.00, 최대값이 0.00, 증가값이 0.00, 그리고 범위 이탈 작업이 **무시**입니다 . LabVIEW 7.x 및 이후 버전에서 숫자 컨트롤은 데이터 타입의 기본 데이터 범위값을 이용합니다 .

## 강제 변환점과 타입 정의

LabVIEW 6.1 과 이후 버전의 와이어는 타입 정의에 대한 정보를 포함하고 있기 때문에 , 블록 다이어그램에서 더 많은 강제 변환점을 확인할 수 있습니다 . 타입 정의된 요소를 타입 정의되지 않은 VI 또는 함수의 터미널에 연결할 경우 , 강제 변환점이 나타납니다 . 또한 , 타입 정의된 출력 터미널을 타입 정의되지 않은 인디케이터에 연결해도 강제 변환점이 나타납니다 . 이러한 강제 변환점은 VI 에서 일관되게 타입 정의를 사용하지 않은 부분을 나타냅니다 . 이 경우에 강제 변환점은 런타임 성능에 영향을 주지 않습니다 .

패턴화된 타입 정의를 위한 ( 패턴화된 문자열로 ) 함수 사용에 대한 정보는 *LabVIEW 도움말*을 참조하십시오 .

## 파일 대화 상자의 버튼 라벨

LabVIEW 6.1 과 이전 버전에서 , File Dialog 함수가 디스플레이하는 파일 대화 상자에는 사용자가 새 파일 이름을 입력할 수 있을 때 **저장** 버튼 라벨이 나타납니다 . 그렇지 않으면 , 버튼 라벨은 **열기**입니다 . LabVIEW 8.x 에서는 , ( 파일 대화 상자 ) 익스프레스 VI 가 디스플레이하는 파일 대화 상자의 버튼 라벨은 라벨을 변경한 경우를 제외하고 모든 경우에 **확인**입니다 . ( 파일 대화 상자 ) 익스프레스 VI 의 **버튼 라벨** 입력을 사용하여 버튼의 라벨을 변경합니다 . 기존의 VI 에서 ( 파일 대화 상자 ) 익스프레스 VI 를 사용

할 경우, VI의 동작을 다시 검토하여 **확인**으로 표시되는 기본 라벨이 VI의 기능에 적절한지를 확인해 주십시오.

## 온라인 도움말 컨트롤 함수

( 온라인 도움말 컨트롤 ) 함수의 **도움말 파일의 경로** 입력은 이제 필수입니다. 컴파일된 도움말 파일 이름 (.chm 또는 .hlp) 또는 컴파일된 도움말 파일의 전체 경로를 입력에 연결할 수 있습니다. 컴파일된 도움말 파일 이름만 연결할 경우, LabVIEW는 이 파일을 labview\help 디렉토리에서 검색합니다.

## 로드시 프론트패널 디스플레이

LabVIEW 7.x 및 이후 버전에서, LabVIEW가 VI를 로드할 때 VI의 프론트패널이 디스플레이되도록 설정하고 VI 서버를 사용하여 VI를 로드하면, LabVIEW는 프론트패널을 디스플레이하지 않습니다. 프로그램적으로 프론트패널을 디스플레이하려면 프론트패널 : 열기 메소드를 반드시 사용해야 합니다.

## VI 참조 열기 함수

LabVIEW 6.1 과 이전 버전에서, Open VI Reference 함수의 **options** 파라미터에 값을 연결하지 않은 경우, 템플릿이 메모리에 없다면 LabVIEW는 템플릿에서 VI를 예로 들어 설명합니다. 템플릿이 메모리에 있는 경우 LabVIEW는 템플릿의 참조를 엽니다. LabVIEW 7.0 과 7.1 에서, Open VI Reference 함수를 사용하여 이미 메모리에 있는 템플릿의 참조를 생성하는 경우, **options** 파라미터에 0x02 를 지정해 주지 않으면 이 함수는 에러를 반환합니다. LabVIEW 8.0 과 이후 버전에서, (VI 참조 열기) 함수를 사용하여 템플릿의 참조를 생성하는 경우, LabVIEW는 템플릿이 이미 메모리에 있더라도 템플릿에서 VI를 예로 들어 설명합니다.

## 지수형

LabVIEW 6.0 과 이전 버전에서, ^ 연산자는 수식 노드에서 지수를 나타냅니다. LabVIEW 6.1 과 이후 버전에서 지수 연산자는 \*\* 입니다—예를 들어, x\*\*y. ^ 연산자는 배타적 비트 또는 (XOR) 연산을 나타냅니다.

## IVI 설정 저장 파일

현재 IVI 설정 저장 파일 포맷의 모든 명칭은 대소문자를 구분합니다. 어플리케이션에서 논리 이름, 드라이버 세션 이름, 또는 가상 이름을 사용한 경우, 사용한 이름이 대소문자에 아무 변동없이 IVI 설정 저장 파일에서 정의한 이름과 정확하게 일치하는지 확인하십시오.

## 기술 지원 양식

LabVIEW 7.x 와 이전 버전에서 LabVIEW 설치 프로그램은 techsup.11b 를 설치하지 않습니다. 설치, 설정, 어플리케이션에 문제 및 질문이 생기면 National Instruments 의 웹 사이트인 [ni.com/support](http://ni.com/support) 를 방문하십시오.

## LabVIEW 5.x 또는 이전 버전에서 업그레이드하기

LabVIEW 5.x 또는 이전 버전에서 LabVIEW 8.2 로 업그레이드하는 것에 대한 추가적인 정보는 National Instruments 웹 사이트 [ni.com/info](http://ni.com/info) 에서 정보 코드 ext8h9 를 입력하십시오.

## LabVIEW 8.2 의 특징과 변경 내역

---

프로그래밍 개념과 단계별 해설, 참조 정보 등 LabVIEW 8.2 기능에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오. **도움말 » LabVIEW 도움말 검색**을 선택해서 *LabVIEW 도움말*에 접근합니다.

유의사항, 추가적인 호환성 문제, LabVIEW 8.2 의 최신 추가 기능에 대한 자세한 내용은 labview 디렉토리의 readme.html 를 참조하십시오.

## LabVIEW 문서

LabVIEW 8.2 에서는 다음과 같이 문서가 개선되었습니다:

- 모든 *LabVIEW 도움말* 항목 맨 아래에 **Submit feedback on this topic** 링크가 나타납니다. 도움말 항목에 대한 피드백을 보내려면 이 링크를 클릭한 후 **Documentation Suggestion Details** 형식을 작성하십시오. 이 링크는 영어 버전의 *LabVIEW 도움말*에만 나타납니다.
- 개념 항목의 경우 도움말 항목의 오른쪽 위 코너에 탐색 테이블이 나타납니다. 테이블의 링크를 클릭하면 관련된 세부항목으로 넘어갑니다.

## 새 예제 VI

LabVIEW 8.x 에 추가된 예제 VI 들의 설명을 보고 실행하기 위해서는 NI 예제 탐색기의 **탐색** 탭에서 **LabVIEW 8.x 의 새로운 예제** 폴더를 참고하십시오.

## 시작 시간 단축

LabVIEW 8.2 는 성능 최적화 기능 덕분에 LabVIEW 8.0 보다 로드 시간이 단축되었습니다.

## 블록다이어그램의 개선사항

LabVIEW 8.2 에서는 블록다이어그램의 다음과 같은 사항과 관련된 기능이 개선되었습니다.

## 기본색 변경 내역

다음 블록다이어그램 구성요소의 기본색이 보다 잘 보이도록 하기 위해 변경되었습니다 .

- 에러 클러스터의 와이어와 터미널이 블록다이어그램에서 핑크색이 아니라 짙은 노란색으로 나타납니다 .
- 강제 변환점은 기본적으로 회색이 아니라 빨간색으로 나타납니다 . 강제 변환점의 색을 변경하기 위해서는 **도구 » 옵션**을 선택하고 **항목** 리스트에서 **색**을 선택합니다 . **기본 색 사용** 확인란에서 확인 표시를 제거하고 **강제 변환점** 색 상자를 클릭하여 다른 색을 선택합니다 .

## VI 계층구조에서 브레이크포인트 제거하기

VI 계층구조에서 모든 브레이크포인트를 제거하려면 , VI 에서 **편집 » 계층구조에서 브레이크포인트 제거** 옵션을 선택합니다 . 다이내믹하게 호출된 VI 혹은 ( 정적 VI 참조 ) 함수를 사용하여 참조된 VI 의 브레이크포인트는 수동으로 제거해야 합니다 .

## 상시 폴딩 사용으로 성능 최적화

LabVIEW 는 상시 폴딩을 사용하여 VI 의 성능을 최적화합니다 . 상시 폴딩을 사용하여 , LabVIEW 는 VI 를 컴파일할 때 실행 중 상수값을 계산하는 대신 이를 저장합니다 . 구조에 연결된 상수의 경우 , LabVIEW 는 VI 를 컴파일할 때 구조의 출력값을 계산한 후 저장하여 실행 중 사용할 수 있도록 합니다 .

**도구 » 옵션**을 선택한 후 , **항목** 리스트에서 **블록다이어그램**을 선택하고 , **와이어의 상시 폴딩 보이기** 및 **구조의 상시 폴딩 보이기** 확인란에 확인 표시를 하여 블록다이어그램에 상시 폴딩 해쉬 마크를 디스플레이할 수 있습니다 . **와이어의 상시 폴딩 보이기** 확인란에 확인 표시를 하면 상시 폴딩된 상수에 연결된 와이어에 회색 해쉬 마크가 나타납니다 . **구조의 상시 폴딩 보이기** 확인란에 확인 표시를 하면 상수에 연결된 구조에 회색의 해쉬 마크가 나타납니다 . VI 를 실행하거나 저장한 후에는 VI 에 해쉬 마크가 나타나지 않을 수도 있습니다 .

LabVIEW 8.2 는 케이스 구조 선택자 터미널과 While 루프 조건 터미널에 연결한 계산된 상수를 폴딩합니다 .

## 기타 블록다이어그램의 개선사항

LabVIEW 8.2 에서는 블록다이어그램의 다음과 같은 기타 사항이 개선되었습니다 .

- 프로퍼티 또는 인보크 노드의 **참조 출력**에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **생성**을 선택하면 **생성** 메뉴가 프로퍼티 또는 인보크 노드와 같은 클래스의 프로퍼티 또는 메소드를 디스플레이합니다 .

- 보이는 아이템 : 계층구조 라인 보이기 프로퍼티는 다음 값을 받습니다 : **보이기**인 경우 , LabVIEW 는 항상 아이템의 계층구조를 나타내는 라인을 트리 컨트롤의 아이템 왼쪽에 수직과 수평 라인으로 디스플레이합니다 . **보이지 않기** 상태라면 , 계층구조 라인은 항상 보이지 않습니다 . **OS 기본값**이면 , OS 의 트리가 계층구조 라인을 보이는 경우 LabVIEW 는 계층구조 라인을 디스플레이합니다 .

## 프런트패널 개선사항

LabVIEW 8.2 에서는 프런트패널의 다음과 같은 사항과 관련된 기능이 개선되었습니다 .

### 구획의 배경 이미지 설정하기

구획의 배경 이미지를 설정하고 반입할 수 있습니다 . 구획의 스크롤 막대에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **프로퍼티**를 선택합니다 . **구획 프로퍼티** 대화 상자의 **배경** 리스트에서 이미지를 선택합니다 .

**배경** 리스트에 나타나지 않는 이미지를 선택하려면 **탐색** 버튼을 클릭합니다 . LabVIEW 는 배경 이미지로 BMP, JPEG, PNG 그래픽 포맷을 지원합니다 . 또한 배경 이미지 프로퍼티를 사용하여 프로그램적으로 구획 배경을 설정할 수 있습니다 .



**노트** **배경** 리스트에 나타나지 않는 이미지를 선택하는 경우 , LabVIEW 는 해당 이미지를 **배경** 리스트에 영구적으로 추가하지 않습니다 . 이미지를 리스트에 영구적으로 추가하려면 labview\resource\backgrounds 디렉토리에 이미지를 저장해야 합니다 .

배경 이미지가 있는 구획이 포함된 VI 를 저장하면 LabVIEW 는 구획 배경 이미지를 VI 와 함께 저장합니다 .

### 노브와 다이얼 최소와 최대에서 잠그기

기본으로 , 노브와 다이얼은 최소 및 최대값 이상으로 회전되지 않습니다 .

이 잠금 동작을 비활성화하려면 , 노브나 다이얼에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **프로퍼티**를 선택한 후 **최대와 최소에서 잠금** 확인란에서 확인 표시를 제거합니다 . 잠금하면 노브나 다이얼이 최소에서 최대 또는 최대에서 최소로 뛰는 것을 막아줍니다 . 이 동작을 비활성화하면 예상치 못한 값을 얻을 수 있습니다 .

LabVIEW 8.2 에서는 LabVIEW 8.0 또는 이전 버전에서 마지막으로 저장했던 VI 를 열면 잠금이 비활성화됩니다 . 잠금을 활성화하려면 **최대와 최소에서 잠금** 확인란에 확인 표시를 하십시오 .

## 트리 컨트롤과 리스트박스에 여러 아이템 끌어오기

트리 컨트롤과 리스트박스 안팎으로 여러 아이템을 끌어서 놓을 수 있습니다. 트리 컨트롤 또는 리스트박스에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **선택 모드 » 0 또는 그 이상의 아이템**이나 **선택 모드 » 1 또는 그 이상의 아이템**을 선택하여 여러 아이템 끌기와 놓기를 활성화합니다. 선택된 여러 아이템을 끌기 시작하면 모든 선택된 아이템이 움직입니다.

## 디지털 웨이브폼 그래프의 개선사항

디지털 웨이브폼 그래프에 대한 추가적인 정보는 *LabVIEW 도움말* **목차** 탭의 **기본 » 그래프와 차트** 모음을 참조하십시오.

LabVIEW 8.2에서는 디지털 웨이브폼 그래프의 다음과 같은 사항이 개선되었습니다.

### 라인 굵기 설정하기

디지털 웨이브폼 그래프 플롯 범례의 바로 가기 메뉴에서 **두꺼운 라인 위치**가 **라인 스타일**로 대체되었습니다. 플롯 범례의 플롯에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **라인 스타일**을 선택하여 라인 굵기를 설정합니다. LabVIEW 8.2에는 전체 라인을 굵게 하기 위한 **라인 스타일** 옵션이 포함되었습니다.

### 비교 데이터 어둡게하기

디지털 웨이브폼 그래프에 드라이브 및 비교 로직 상태 모두의 디지털 데이터가 포함되어 있는 경우, 기본으로 비교 데이터는 플롯에서 드라이브 데이터보다 어둡게 나타납니다. 비교 데이터가 어둡게 나타나지 않게 하려면, 플롯에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **고급 » 비교 데이터 어둡게**를 선택하고 확인 표시를 제거합니다. 또한 비교 데이터 어둡게 프로퍼티를 사용하여 비교 데이터를 프로그램적으로 어둡게할 수도 있습니다.



#### 노트

이 기능은 주로 디지털 I/O 신호를 생성하는 사용자에게 적용됩니다. 비교 데이터에 대한 추가적인 정보는 *LabVIEW 도움말* **목차** 탭의 **기본 » 그래프와 차트 » 개념 » 그래프와 차트 사용자 정의** 모음을 참조하십시오.

## 기타 프런트패널 개선사항

LabVIEW 8.2에서는 프런트패널의 다음과 같은 기타 사항이 개선되었습니다.

- LabVIEW 8.0에서는 처음으로 프런트패널 객체의 캡션을 디스플레이할 때 LabVIEW가 라벨을 옆으로 이동시킵니다. LabVIEW 8.2에서는 LabVIEW가 라벨을 숨기고 캡션만을 디스플레이합니다.
- **XY 그래프 프로퍼티** 대화 상자에서, **평면 옵션 보기** 폴다운 메뉴와 선택한 평면을 설정하는 옵션들이 **스케일** 페이지에서 **모양** 페이지로 이동되었습니다.

- 색칠 도구를 사용하여 시스템 테이블의 배경색을 변경할 수 있습니다.
- 시스템 여러 열 리스트박스, 리스트박스, 테이블, 또는 트리 컨트롤의 헤더와 셀의 색을 변경할 수 있습니다.
- **기본값으로 다시 초기화, 데이터 잘라내기, 데이터 붙여넣기** 옵션은 VI가 실행 모드일 때 바로 가기 메뉴에서 사용할 수 없습니다. 이러한 바로 가기 메뉴 옵션은 실행 모드에서 컨트롤에서만 사용할 수 있습니다.
- 탭 컨트롤의 크기를 내용에 맞게 조정하려면, 탭 컨트롤에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **고급 » 맞도록 크기 조정**을 선택합니다.
- 열거형 타입 컨트롤, 링 컨트롤, 또는 콤보 박스 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **아이템 편집**을 선택하면, 셀을 더블 클릭해야 **아이템** 또는 **값** 열의 아이템을 편집할 수 있습니다. 슬라이드 컨트롤이나 노브의 **텍스트 라벨**에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **아이템 편집**을 선택할 때에도 이 변경 내역이 적용됩니다.
- LabVIEW 8.0에서는 컨트롤 편집기 윈도우에서 **파일 » 변경된 사항 적용**을 선택하면 LabVIEW가 원래 컨트롤의 라벨, 캡션, 값을 보존합니다. LabVIEW 8.2에서는 LabVIEW가 컨트롤 편집기 윈도우에서 변경된 사항을 모두 반영하며 원래 컨트롤의 정보를 보존하지 않습니다. 이 동작은 사용자 컨트롤에만 적용되며 타입 정의에는 적용되지 않습니다.
- LabVIEW는 사용자가 그래프나 차트의 축을 오토스케일할 때 숨겨진 플롯을 포함시키지 않습니다. 오토스케일할 때 숨겨진 플롯을 포함시키려는 경우, 플롯을 숨기는 대신 투명하게 하십시오. 플롯 범례에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **색**을 선택하여 플롯의 색을 변경합니다.
- **(Windows)** LabVIEW의 라디오 버튼과 확인란은 Windows의 라디오 버튼 및 확인란 동작과 같습니다. 스페이스바만을 사용해서 라디오 버튼과 확인란을 토글할 수 있습니다. 알파벳과 숫자 키보드의 <Enter> 키, 숫자 키패드의 <Enter> 키, 또는 스페이스바로 대화 상자 버튼을 토글할 수 있습니다. LabVIEW 키보드 바로 가기 키 <T> 와 <F> 는 대화 상자 버튼을 토글하지 않습니다.
- LabVIEW 8.2에는 타임스탬프 컨트롤의 고급 편집을 위한 새로운 토큰이 포함되어 있습니다. 컨트롤에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **프로퍼티**를 선택하여 **타임스탬프 프로퍼티** 대화 상자를 디스플레이합니다. **포맷과 정밀도** 페이지에서, **고급 편집 모드** 옵션을 선택하여 **절대 시간 포맷 코드** 리스트를 디스플레이합니다. LabVIEW 8.2에는 다음 새 포맷 코드가 포함되어 있습니다.

포맷 코드	값
<%^<>T>	세계시 컨테이너
<%z>	현지 시간과 세계시의 차이

# 환경 개선사항

LabVIEW 8.2에서는 LabVIEW 환경의 다음과 같은 사항이 개선되었습니다.

## 복구를 위한 자동 저장

비정상적으로 종료되거나 시스템 실패의 경우, LabVIEW는 종료 또는 실패 시 열려있던 모든 변경된 VI (.vi), VI 템플릿 (.vit), 컨트롤 (.ctl), 또는 컨트롤 템플릿 (.ctt)을 임시 장소에 백업합니다. 그러나 프로젝트 (.lvproj), 프로젝트 라이브러리 (.lvlib), X 컨트롤 (.xctl), 또는 LabVIEW 클래스 (.lvclass)는 백업하지 않습니다.

LabVIEW가 비정상적인 종료나 시스템 실패 전에 자동으로 파일을 저장하는 경우, LabVIEW를 다시 시작할 때 **복구할 파일 선택** 윈도우가 나타납니다. 복구하려는 파일을 선택한 후 **복구** 버튼을 클릭합니다. 파일을 복구하지 않으려는 경우, 모든 파일의 선택을 해제하고 **버림** 버튼을 클릭합니다. **취소** 버튼을 클릭하면 모든 선택된 파일이 기본 데이터 디렉토리의 LVAutoSave\archives 서브디렉토리로 이동됩니다.

**도구 » 옵션**을 선택하고 **항목** 리스트에서 **환경**을 선택하여 복구를 위한 저장을 활성화 또는 비활성화할 수 있고, LabVIEW가 얼마나 자주 파일을 백업하는지 지정할 수 있습니다.

## 대화 상자 개선사항

LabVIEW 8.2에서는 대화 상자의 다음과 같은 사항이 개선되었습니다.

### 옵션 대화 상자 개선사항과 변경 내역

LabVIEW 8.2에서는 **옵션** 대화 상자의 다음과 같은 사항이 개선되었습니다. 삭제된 옵션과 처리 방법에 대한 정보는 National Instruments 웹 사이트 ([ni.com/info](http://ni.com/info))에서 정보 코드 ex6rkc를 입력하십시오.

- **성과와 디스크** 페이지 및 **시작할 때 가능한 디스크 용량 검사**와 **여러 스레드 실행** 옵션은 더 이상 존재하지 않습니다.
- **프론트패널** 페이지에서 **시스템 기본 기능 키 셋팅 덮어쓰기**와 **그리는 동안 자연스러운 업데이트 사용** 옵션은 더 이상 존재하지 않습니다.
- **블록다이어그램** 페이지에서 **선 연결 안내 보기** 옵션은 더 이상 존재하지 않습니다.
- **경로** 페이지에서 **라이브러리 디렉토리** 옵션은 더 이상 존재하지 않습니다.
- **환경 옵션** 페이지에서 **축소된 메뉴 사용** 옵션은 더 이상 존재하지 않습니다.
- **환경** 페이지에서 **저스트 인 타임 어드바이스 활성화** 확인란에 더 이상 기본으로 확인 표시가 되어있지 않습니다.
- **블록다이어그램 옵션** 페이지에서 **놓여질 때 subVI 이름 보기** 옵션은 글로벌 변수에도 적용됩니다. 블록다이어그램에 글로벌 변수를 놓을 때

라벨이 디스플레이되도록 하려면 이 확인란에 확인 표시가 되어있는지 확인하십시오.

- **웹 서버: VI 보이기** 페이지에서 **현재 선택된 VI** 옵션이 **VI 보이기** 옵션으로 변경되었습니다.

## LLB 관리자 개선사항

LabVIEW 8.2에서는 **LLB 관리자** 윈도우의 다음과 같은 사항이 개선되었습니다.

- **디렉토리** 영역의 오른쪽에 **탐색** 버튼이 나타납니다. 이 버튼을 클릭하여 수정하려는 LLB를 탐색합니다.
- 도구 모음에 **삭제** 버튼이 나타납니다. **파일** 리스트에서 파일을 선택하고 이 버튼을 클릭하여 선택된 파일을 LLB에서 제거합니다.
- 열에서 마우스 오른쪽 버튼을 클릭하여 **파일** 리스트를 열로 정렬할 수 있습니다.

## 데이터 연결 페이지

LabVIEW 8.2에서는 모든 프런트패널 컨트롤의 **프로퍼티** 대화 상자에 있는 **데이터 연결** 페이지의 다음과 같은 사항이 개선되었습니다.

- **현재 네트워크 아이템이 선택되었습니다**와 **현재 프로젝트 아이템이 선택되었습니다** 텍스트 박스는 **경로** 텍스트 박스로 변경되었습니다.
- **모드** 섹션은 **접근 타입** 풀다운 메뉴로 변경되었습니다.
- **알람이 작동할 때 점멸** 확인란은 없어졌습니다.
- **데이터 연결 선택** 풀다운 메뉴에서 **공유 변수 엔진 (NI-PSP)**을 선택하면 **탐색** 버튼이 나타납니다. 이 버튼을 누르면 **네트워크 아이템 선택** 대화 상자가 아닌 **소스 아이템 선택** 대화 상자가 시작됩니다. **네트워크 아이템 선택** 대화 상자는 없어졌습니다.
- **네트워크 소스** 풀다운 메뉴는 **데이터 연결** 페이지가 아닌 **소스 아이템 선택** 대화 상자에 나타납니다.

## 기타 대화 상자 개선사항

LabVIEW 8.2에서는 대화 상자의 다음의 같은 부수적인 사항이 변경되었습니다:

- **VI 매트릭스** 윈도우의 **통계 매트릭스** 테이블에는 열의 모든 값의 평균을 디스플레이하는 행이 포함됩니다. 또한 **통계로부터 이 폴더의 파일을 제외합니다** 확인란에 확인 표시를 하여 특정한 폴더에 위치한 파일들을 제외할 수도 있습니다.
- **VI 프로퍼티** 대화 상자에서 **보안** 페이지가 **보호** 페이지로 변경되었습니다.
- **다형성 VI** 윈도우에는 **라이선스 경고 보이기** 버튼이 포함되어 있습니다. 이 버튼은 다형성 VI가 평가판이거나 라이선스가 유효하지 않은 프로젝트 라이브러리에 속해있는 경우 나타납니다. 이 버튼을 클릭하

여 경고 메시지를 디스플레이합니다. 라이선스 상태에 대한 추가적인 정보는 경고 메시지에서 **도움말** 버튼을 클릭하십시오.

- **포맷 문자열 편집** 대화 상자와 **스캔 문자열 편집** 대화 상자의 **선택된 작업** 풀다운 메뉴에서 SI 표기법을 변환 타입으로 선택할 수 있습니다.
- **단순화된 이미지 반출** 대화 상자에서 **클립보드에 저장** 옵션이 **클립보드에 반출**로 변경되었습니다.
- **인스트루먼트 드라이버 VI 생성** 대화 상자에서, **컨트롤 설정** 페이지의 **출력 데이터 타입** 풀다운 메뉴에서 **없음** 옵션이 없어졌습니다.
- **에러 코드 파일 편집기** 윈도우에서 **현재 에러 코드와 설명** 옵션이 **에러 코드** 및 **에러 텍스트** 옵션으로 변경되었습니다.

## 숨겨진 컨트롤과 인디케이터 디스플레이하기

**편집 » 숨겨진 컨트롤과 인디케이터 보이기**를 선택하여 사용자 컨트롤과 글로벌 변수의 프런트패널에 있는 모든 숨겨진 컨트롤과 인디케이터를 디스플레이할 수 있습니다. 이 옵션은 사용자 컨트롤 또는 글로벌 변수의 **편집** 메뉴에서만 사용 가능합니다.

또한 `labview\project\_ShowHidden` 디렉토리의 `ShowHidden Core VI` 를 실행하여 사용자 컨트롤이나 글로벌 변수가 아닌 VI의 컨트롤과 인디케이터도 디스플레이할 수 있습니다.

## 에러 리스트 윈도우 개선사항

**에러 리스트** 윈도우의 **에러가 있는 아이템** 리스트에서 깨진 아이템의 이름 옆에는 빨간 x 모양이 나타납니다. LabVIEW는 이러한 아이템을 **에러가 있는 아이템** 리스트의 맨 위로 정렬합니다. 편집 중이기 때문에 다른 아이템에 에러를 발생시키는 아이템은 아이템 이름 옆에 연필 아이콘이 나타납니다. 아이템 이름 옆에 아이콘이 없는 아이템은 상위 아이템에 에러가 있기 때문에 에러가 발생하는 아이템입니다.

## 찾고 대체하기 개선사항

LabVIEW 8.2에서는 찾고 대체하기 작업의 다음과 같은 사항과 관련된 기능이 개선되었습니다.

- **텍스트 검색** 옵션에서 **복제 무시** 확인란은 VI에서 텍스트를 검색할 때 프런트패널 복제 또는 재호출 프런트패널을 모두 무시합니다.
- **찾기** 대화 상자에서 **객체 선택** 버튼은 객체의 파일 이름이 아닌 사용자가 선택한 객체의 제목과 함께 나타납니다.
- **검색 결과 윈도우**는 LabVIEW가 검색 중에 하나 이상의 객체를 찾은 경우에만 나타납니다. LabVIEW가 하나의 객체만을 찾는 경우, LabVIEW는 프런트패널 윈도우 또는 블록 다이어그램 윈도우의 객체를 하이라이트합니다.
- **찾기** 대화 상자는 (정규식 일치) 함수 및 변수와 같은 익스프레스 VI와 노드도 찾아서 대체합니다.

## 열린 윈도우 관리하기

**윈도우** 메뉴는 최대 10 개까지 열린 윈도우를 디스플레이합니다. **윈도우 » 모든 윈도우**를 선택하거나 <Ctrl-Shift-W> 를 눌러 **모든 윈도우** 대화 상자를 디스플레이하여 모든 열린 윈도우를 관리할 수 있습니다. 대화 상자의 오른쪽에서 대응하는 버튼을 클릭하여 윈도우를 보거나 닫고 또는 윈도우에 대응하는 아이템을 저장할 수 있습니다.

**제목** 열에서, 마지막으로 저장된 이후 변경사항이 있는 윈도우 아이템의 이름 끝에는 별표가 붙어 있습니다.

## 팔레트 개선사항

LabVIEW 8.2 에서는 다음과 같이 팔레트가 개선되었습니다:

- **즐거찾기 구성** 대화 상자를 사용하여 **즐거찾기** 팔레트 항목의 아이템 순서를 변경할 수 있습니다.
- **메뉴 문서** 대화 상자는 **팔레트 문서** 대화 상자로 변경되었습니다. **메뉴 설명** 필드는 **팔레트 설명**으로 변경되었습니다.
- 팔레트가 프로젝트 라이브러리에 속해있는 경우, 프로젝트 라이브러리의 경로를 볼 수 있습니다. **도구 » 고급 » 팔레트 세트 편집**을 선택하고 팔레트에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **팔레트 파일로의 경로를 디스플레이**를 선택합니다. LabVIEW 는 팔레트의 실제 경로를 디스플레이하며, 팔레트가 라이브러리에 속해있는 경우 라이브러리의 경로도 디스플레이합니다.
- **컨트롤과 함수 팔레트 세트 편집** 대화 상자에는 **저장하기 전에 변경사항 미리보기** 확인란이 포함되어 있습니다. **저장하기 전에 변경사항 미리보기** 확인란에 확인 표시를 한 후 **변경 저장** 버튼을 클릭하여 **팔레트 변경사항 미리보기** 대화 상자를 디스플레이합니다.
- **VI 놓기 (Drop VI)** 바로 가기 메뉴는 **VI 놓기 (Place VI)** 로 변경되었습니다. 이 바로 가기 메뉴는 **함수** 팔레트의 VI 에서 마우스 오른쪽 버튼을 클릭할 때 나타납니다.
- LabVIEW 8.2 에는 애드온을 설치하기 전에는 아무것도 없는 팔레트가 있습니다. 예를 들어 **컨트롤** 팔레트의 **컨트롤 디자인과 시뮬레이션** 서브팔레트는 LabVIEW Control Design & Simulation 제품 중 하나를 설치하기 전에는 비어있습니다.

## 계층구조 복제 저장하기

소스 배포를 생성하지 않고도 VI 와 해당 subVI 를 계층구조 복제로 저장할 수 있습니다. **파일 » 다른 이름으로 저장**을 선택하고 VI 를 저장하고 **새 위치에 계층구조 복제** 옵션을 선택하여 VI 와 해당 계층구조를 새로운 위치에 저장합니다.

## 기타 환경 개선사항

LabVIEW 8.2 에서는 프런트패널의 다음과 같은 기타 사항이 개선되었습니다 :

- LabVIEW 평가판 또는 학생판에서 , LabVIEW 는 이제 VI 서브패널의 subVI 에 워터마크를 디스플레이하지 않습니다 . 또한 LabVIEW 는 디버그 개발 버전에서 더 이상 워터마크를 디스플레이하지 않습니다 .
- 구매한 LabVIEW 의 버전에서 특정한 유효한 라이선스가 없는 경우 , 다형성 VI 를 편집하지 못할 수도 있습니다 .
- 컨트롤에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **생성 » 참조**를 선택하여 컨트롤 참조를 엄격한 타입 정의로 생성할 수 있습니다 .
- **보기** 메뉴에는 **프로젝트 탐색기** 윈도우에서 현재 VI 가 속한 프로젝트 라이브러리 , X 컨트롤 , 또는 LabVIEW 클래스를 하이라이트하는 **이 VI 의 라이브러리** , **이 VI 의 X 컨트롤** , 또는 **이 VI 의 클래스** 아이템이 포함되어 있습니다 . 메뉴 아이템은 VI 를 소유하는 라이브러리 타입에 따라 바뀝니다 . 라이브러리 , X 컨트롤 , 또는 클래스가 LabVIEW 프로젝트에 없는 경우 , LabVIEW 는 현재 VI 가 속한 라이브러리 , X 컨트롤 , 또는 클래스만을 포함하는 새로운 윈도우를 엽니다 .
- LabVIEW 8.0 에서 **도움말 » 여러 설명** 메뉴 아이템은 **시작하기** 윈도우에서 사용할 수 없었습니다 . LabVIEW 8.2 에서는 이 메뉴 아이템을 **시작하기** 윈도우에서 사용할 수 있습니다 .
- **(Mac OS) Context Help** 윈도우를 디스플레이하거나 숨기려면 , **Help»Show Context Help** 를 선택하거나 <Command-Shift-H> 키를 누릅니다 . <Command-H> 바로 가기 키는 열려 있는 LabVIEW 어플리케이션을 숨깁니다 . 키보드 바로 가기 키를 사용자 정의하는 것에 대한 추가적인 정보는 *LabVIEW 도움말*을 참조하십시오 .
- **도구 » 인스트루멘테이션 » 고급 개발**을 선택하여 다음과 같은 인스트루먼트 드라이버를 사용한 고급 개발 옵션에 접근합니다 .
  - **드라이버 가이드라인 보이기**는 웹 브라우저에 National Instrument 드라이버 네트워크 [ni.com/idnet](http://ni.com/idnet) 의 인스트루먼트 드라이버 네트워크를 디스플레이합니다 .
  - **Icon Art Glossary 보이기**는 웹 브라우저에 National Instrument 드라이버 네트워크 [ni.com/idnet](http://ni.com/idnet) 의 Icon Art Glossary 를 디스플레이합니다 .
  - **기타 리소스**는 웹 브라우저에 National Instrument 드라이버 네트워크 [ni.com/idnet](http://ni.com/idnet) 의 개발 도구와 리소스를 디스플레이합니다 .

## 새로운 또는 변경된 VI, 함수, 노드 개선사항

LabVIEW 8.2 에는 다음과 같은 새로운 또는 변경된 VI 와 함수가 포함되어 있습니다 : VI, 함수, 노드에 대한 추가적인 정보는 *LabVIEW 도움말* **독자** 탭의 **VI 와 함수 참조** 모음을 참조하십시오 .

## 새로운 VI 와 함수

LabVIEW 8.2 에는 다음과 같은 새로운 VI 와 함수가 포함되어 있습니다 .

### 고급 파일 VI

**고급 파일 함수** 팔레트는 다음과 같은 새 VI 가 포함되어 있습니다 :

- 파일 또는 폴더가 존재하는지 확인 VI
- 두 경로 비교 VI
- 임시 파일 경로 생성 VI
- 파일 확장자 얻기 VI
- MD5 체크섬 파일 VI
- 순환적 파일 리스트 VI

### 디지털 웨이브폼 함수

**디지털 웨이브폼** 팔레트에는 다음과 같은 새 함수가 포함되어 있습니다 :

- 웨이브폼 만들기 함수
- 디지털 데이터 만들기 함수
- 웨이브폼 구성요소 얻기 함수
- 디지털 데이터 구성요소 얻기 함수

### 수학 VI

**수학** 팔레트에는 LabVIEW Full 과 Professional Development Systems 에서만 제공되는 다음과 같은 새 VI 가 포함되어 있습니다 :

- 크로네커 곱 VI
- 리아푸노프 방정식 VI

### .NET VI

**.NET** 팔레트에는 다음과 같은 새 VI 가 포함되어 있습니다 :

- .NET 객체로 VI
- .NET 객체를 배리언트로 VI

### 스케일링 VI

**스케일링** 팔레트에는 다음과 같은 새 VI 가 포함되어 있습니다 :

- RTD 읽기 변환 VI
- 스트레인 게이지 읽기 변환 VI
- 써미스터 읽기 변환 VI
- 열전쌍 읽기 변환 VI

## 신호 처리 VI

**신호 처리** 팔레트에는 LabVIEW Full 과 Professional Development Systems 에서만 제공되는 다음과 같은 새 VI 가 포함되어 있습니다 :

- 보먼 윈도우 VI
- 부분제거 ( 연속 ) VI
- 부분제거 ( 단일 ) VI
- FIR 필터 VI
- FIR 초기 조건 지정 필터 VI
- 가우스 변조된 사인 패턴 VI
- 가우스 모노펄스 VI
- 역 처프 (Chirp) Z 변환 VI
- 수정된 바틀렛 - 해닝 윈도우 VI
- 파젠 윈도우 VI
- 주기적 싱크 (Sinc) 패턴 VI
- 펄스 열 VI
- 유리수 리샘플 VI
- 사비츠키 고레이 필터 VI
- 사비츠키 고레이 필터 계수 VI
- 삼각 패턴 VI
- 업샘플 VI
- 웰치 윈도우 VI

## TDM 스트리밍 VI 와 함수

**TDM 스트리밍** 팔레트에는 다음과 같은 새 VI 와 함수가 포함되어 있습니다 :

- TDMS 파일 보기 VI
- TDMS 닫기 함수
- TDMS 세분화 함수
- TDMS 비우기 함수
- TDMS 프로퍼티 얻기 함수
- TDMS 내용 리스트 함수
- TDMS 열기 함수
- TDMS 읽기 함수
- TDMS 프로퍼티 설정 함수
- TDMS 쓰기 함수

**스토리지** 팔레트에는 다음과 같은 새 VI 가 포함되어 있습니다 :

- TDM 을 TDMS 로 변환 VI
- TDMS 를 TDM 으로 변환 VI

TDM 스트리밍 파일 포맷에 대한 정보는 이 문서의 [TDM 스트리밍 파일 포맷](#) 섹션을 참조하십시오.

## 변경된 VI, 함수, 노드

LabVIEW 8.2 에서 다음과 같은 VI, 함수, 노드가 변경되었습니다.

### DataSocket 함수

DataSocket 팔레트에는 다음과 같은 변경된 VI 가 포함되어 있습니다 :

- (DataSocket 쓰기 ) 함수를 사용할 때 NI-PSP 데이터 아이템의 동기화 알람을 활성화할 수 있습니다 . 동기화 알람을 활성화할 때 `?sync="true"` 를 psp URL 의 끝에 추가하여 제로가 아닌 **ms 타임아웃** 값을 지정할 수 있습니다 . 함수는 작업이 완료되거나 타임아웃이 만료될 때까지 기다립니다 . 동기화 알람을 활성화하면 특히 RT 타겟에서 성능이 느려질 수 있습니다 .
- (DataSocket 읽기 ) 함수는 PSP 서버 또는 FieldPoint 컨트롤러로부터의 경고 또는 에러를 보고하는 **상태** 출력을 가지고 있습니다 .

### 수학 VI

LabVIEW Base Package 에서 (A x B) VI 는 벡터 x A 와 복소수 벡터 x A 라는 새 인스턴스를 포함합니다 .

수학 팔레트에는 LabVIEW Full 과 Professional Development Systems 에서만 제공되는 다음과 같은 변경된 VI 가 포함되어 있습니다 .

- ( 지수 피팅 ) VI 에는 측정값 (**X, Y**) 의 가중치의 배열을 지정하는 새 **가중치** 입력이 포함되어 있습니다 . 신호 노이즈의 결과로 생기는 음수 값때문에 더 이상 지수 피팅 실패가 발생하지 않습니다
- (p(x) 와 q(x) 의 최대공약수 ) VI 에는 VI 가 다항식의 최대공약수를 계산하는데 사용하는 알고리즘을 지정하는 새 **알고리즘** 입력이 포함되어 있습니다 .
- ( 일반 다항식 피팅 ) VI 에는 측정값 (**X, Y**) 의 가중치의 배열을 지정하는 새 **가중치** 입력이 포함되어 있습니다 .
- ( 히스토그램 ), ( 히스토그램 PtByPt ), ( 일반 히스토그램 ) 과 ( 일반 히스토그램 PtByPt ) VI 에는 입력 시퀀스 **X** 의 히스토그램의 막대 그래프를 디스플레이하는 **히스토그램 그래프** 출력이 포함되어 있습니다 . y 축은 히스토그램 카운트 , x 축은 히스토그램의 간격 ( 계급 ) 의 히스토그램 중앙값입니다 .
- (p(x) 와 q(x) 의 최소공배수 ) VI 에는 VI 가 다항식의 최소공배수를 계산하는데 사용하는 알고리즘을 지정하는 새 **알고리즘** 입력이 포함되어 있습니다 .
- ( 부분 분수 만들기 ) VI 에는 VI 가 **분자**와 **분모**의 공통 인자를 어떻게 처리하는지 지정하는 새 **옵션** 입력이 포함되어 있습니다 .

- ( 실베스터 방정식 ) VI 에는 입력 **A** 와 **B** 의 타입을 지정하는 새 **행렬 타입** 입력이 포함되어 있습니다 . 이를 사용하면 **X** 의 계산이 빨라집니다 .
- (3D 직각좌표 회전 ( 방향 )) VI 의 배열 인스턴스의 **방향 코사인** 입력이 **회전 행렬**로 변경되었습니다 .

## 숫자 함수

**숫자형** 팔레트에는 다음과 같은 변경된 함수가 포함되어 있습니다 :

- ( 몫 & 나머지 ) 함수는 큰 음수 64 비트 제수를 사용할 때 정확한 답을 생성합니다 .
- ( 바이트 교환 ) 과 ( 워드 교환 ) 함수의 **입력의 데이터** 입력이 **데이터** 입력으로 변경되었습니다 . 문자열 , 태그 , 경로 , 불리언 , 비정수 숫자 , 열거형 타입 , 에러 클러스터 , 그림 , 행렬 및 이러한 타입의 배열과 클러스터 데이터를 연결할 수 있으며 데이터를 변경하지 않고 통과시킬 수 있습니다 . 입력에 연결된 모든 16, 32, 64 비트 정수의 경우 , 함수는 각 워드 모든 바이트 또는 각 롱워드의 모든 워드 쌍을 교환합니다 . 어커런스 , 모든 참조 번호 타입 , 또는 배리언트는 입력에 연결할 수 없습니다 .
- x86 기반 플랫폼에서 (2의 거듭제곱으로 스케일 ) 함수는 **x** 와 **n** 입력 모두가 부동소수일 때 올바른 답을 생성합니다 .

## 프로토콜 VI 와 함수

**프로토콜** 팔레트에는 다음과 같은 변경된 VI 와 함수가 포함되어 있습니다 :

- (UDP 열기 ) 함수와 (UDP 멀티캐스트 열기 ) VI 에는 사용된 함수의 포트 번호를 반환하는 새 **포트** 출력이 포함되어 있습니다 .
- (UDP 열기 ) 함수와 (UDP 멀티캐스트 열기 ) VI 에는 새 **넷주소** 입력이 포함되어 있으며 , 네트워크 주소는 이 주소에서 리슨하게 됩니다 .
- (TCP 연결 열기 ) 함수의 **주소** 입력이 옵션에서 권장으로 변경되었습니다 .
- (TCP 리슨 ) VI 에는 원격 주소에 (IP 를 문자열로 ) 함수를 호출할지 여부를 나타내는 **원격 주소 변경** 입력이 포함되어 있습니다 . 기본은 참입니다 .

## 2 차 프로그래밍 VI

(2 차 프로그래밍 ) VI 에서는 다음과 같은 사항이 변경되었습니다 :

- (2 차 프로그래밍 ) VI 는 다음 인스턴스를 가진 다형성 VI 입니다 : 2 차 프로그래밍 IP 와 2 차 프로그래밍 AS . LabVIEW 8.0 에서 2 차 프로그래밍 IP 인스턴스는 2 차 프로그래밍 VI 와 같은 기능을 가지고 있었습니다 . (2 차 프로그래밍 ) AS 인스턴스는 활성 세트 알고리즘을 사용하여 최소를 찾습니다 .
- (2 차 프로그래밍 ) VI 의 두 인스턴스 모두 최적화 과정이 시작되는  $n$  차원의 포인트를 지정하는 **시작** 입력을 가지고 있습니다 .

- 2 차 프로그래밍 VI 의 두 인스턴스 모두 LabVIEW 가 최적화 과정의 시작과 끝에서 허용하는 최대 시간을 지정하는 **최대 시간 ( 초 )** 입력을 가지고 있습니다 .
- ( 2 차 프로그래밍 ) VI 의 2 차 프로그래밍 AS 인스턴스에는 최적화의 원시작을 허용하는지 여부를 지정하는 **원시작 ?** 입력이 포함되어 있습니다 .

## 신호 처리 VI

**신호 처리** 팔레트에는 LabVIEW Full 과 Professional Development Systems 에서만 제공되는 다음과 같은 변경된 VI 가 포함되어 있습니다 .

- ( 중앙값 필터 ) VI 에는 LabVIEW 8.0 의 **계수** 입력을 대체하는 새 **왼쪽 계수**와 **오른쪽 계수** 입력이 포함되어 있습니다 . 이 VI 는 계수의 중앙값 필터를 입력 시퀀스 **X** 에 적용합니다 . 여기서 **오른쪽 계수**가 제로보다 크면 계수는 **오른쪽 계수**이며 , **오른쪽 계수**가 제로보다 작으면 계수는 **왼쪽 계수**입니다 .



**노트** **계수** 입력에 값이 연결되어 있는 ( 중앙값 필터 ) VI 를 포함하는 LabVIEW 8.0 또는 이전 버전의 VI 를 여는 경우 , LabVIEW 는 해당 값을 LabVIEW 8.2 의 **왼쪽 계수**에 사용합니다 .

- ( 사인파 ) , ( 사인파 PtByPt ) , ( 삼각파 ) , ( 삼각파 PtByPt ) , ( 톱니파 ) , ( 톱니파 PtByPt ) , ( 사각파 ) , ( 사각파 PtByPt ) 와 ( 임의의 웨이브 ) VI 의 **f** 입력이 **주파수**로 변경되었습니다 .
- ( 사비츠키 고레이 필터 PtByPt ) VI 의 **사이트 포인트 계수** 입력은 **사이드 포인트**로 변경되었습니다 .
- ( 포인트 단위 ) VI 의 아이콘이 업데이트되었습니다 .
- ( 대칭 윈도우 ) 와 ( 스케일된 시간 영역 윈도우 ) VI 의 **윈도우** 입력에는 다음과 같은 새 값이 포함되어 있습니다 : 수정된 바틀렛 - 해닝 , 보먼 , 파젠 , 웰치 .

## Sound VI (Linux)

**Sound VI** 를 사용하려면 Open Sound System (OSS) 드라이버를 가지고 있어야 합니다 . 적용 가능한 OSS 저작권에 대한 정보는 **LabVIEW 도움말** **목차** 탭의 **중요한 정보 » 저작권** 모음을 참조하십시오 .



**노트** LabVIEW 는 /dev/dsp 또는 /dev/dspX 이름을 가진 파일을 찾아 디바이스를 프로브합니다 . 여기서 X 는 0 과 16 사이의 정수입니다 . LabVIEW 는 입력과 출력을 위해 각 디바이스 열기를 시도합니다 . LabVIEW 가 사운드 카드를 감지하지 못하는 경우 , 이름이 /dev/dsp 또는 /dev/dspX 인 디바이스 파일이 로컬 시스템에 있는지와 해당 디바이스에 읽고 쓰는 권한이 있는지 확인하십시오 . 이 디바이스를 기본 이외의 다른 위치로 이동한 경우 , LabVIEW 는 기호 링크와 함께 작동할 수 있습니다 .

Linux 의 Sound VI 에서는 다음과 같은 사항이 변경되었습니다 :

- VI 가 단일톤과 스테레오 사운드를 지원합니다 .
- 웨이브폼은 사운드 데이터를 나타냅니다 . 부호없는 8 비트 , 부호있는 16 비트 , 부호있는 32 비트 정수 또는 단정도 , 배정도 데이터 타입을 가진 원소를 사용하여 Y 배열 데이터를 나타낼 수 있습니다 . 각 웨이브폼은 하나의 채널을 정의합니다 .
- 사운드 데이터의 포맷은 Pulse Code Modulated (PCM) 입니다 .
- 이 VI 는 연속적인 사운드 출력을 생성할 수 있습니다 .
- 이 VI 는 웨이브 파일의 스트리밍 보기를 허용합니다 .
- 이 VI 에는 에러 검사에 대한 개선사항이 있습니다 .

## 문자열 함수

**문자열** 팔레트에서는 다음과 같은 함수가 변경되었습니다 :

- ( 문자열로부터 스캔 ) 함수의 **포맷 문자열** 입력이 System International (SI) 로 표기된 값을 받습니다 .
- ( 스프레드시트 문자열을 배열로 ) 함수는 **배열 타입** 입력이 복소수일 때 올바르게 작동합니다 .

## VISA 함수

특정 VISA 함수의 터미널에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **생성 » 상수** , **생성 » 컨트롤**이나 **생성 » 인디케이터**를 선택하면 링 객체가 나타납니다 . 링 객체는 사용자가 돌아가면서 선택할 수 있는 풀다운 메뉴로 나타납니다 . LabVIEW 가 지원하는 모든 값이 풀다운 메뉴에서 나타납니다 .

이같은 변화는 다음의 함수와 터미널에 영향을 끼칩니다 .

함수	터미널
VISA 인터럽트 신호 지정	모드
VISA 트리거 지정	프로토콜
VISA 유틸리티 신호 지정	버스 신호
VISA 이벤트 비활성화	이벤트 타입
VISA 이벤트 버리기	이벤트 타입
VISA 이벤트 활성화	이벤트 타입
VISA 리소스 찾기	검색 모드
VISA GPIB ATN 컨트롤	모드
VISA GPIB REN 컨트롤	모드
VISA 트리거 맵핑	트리거 소스 , 트리거 목적지

함수	터미널
VISA 열기	접근 모드
VISA 트리거 맵핑 해제	트리거 소스, 트리거 목적지
VISA VXI 명령 또는 쿼리	모드
VISA 이벤트 기다림	이벤트 타입 입력, 이벤트 타입 출력

## 라이브러리 함수 호출 노드

( 라이브러리 함수 호출 노드 )에는 여러 터미널이 포함되어 있습니다. **라이브러리 함수 호출** 대화 상자에서도 다음과 같은 사항이 변경되었습니다:

- 대화 상자에는 파라미터를 보다 쉽게 설정할 수 있는 여러 페이지가 포함되어 있습니다.
- **함수** 페이지에서, **다이어그램에 경로 지정** 확인란에 확인 표시를 하면 프로그램적으로 라이브러리 경로를 지정하도록 노드를 설정할 수 있습니다.
- **파라미터** 페이지에서, 문자열이나 배열 파라미터를 설정할 때 **최소 크기** 텍스트 박스를 사용하여 메모리를 적절하게 할당할 수 있습니다.
- **콜백** 페이지에서 사용자 정의 콜백을 지정할 수 있습니다.

## 기타 VI, 함수, 노드 변경 내역

LabVIEW 8.2에서는 다음과 같은 VI, 함수, 노드가 부수적으로 변경되었습니다:

- ( 시간 영역 연산 ) 익스프레스 VI의 설정 대화 상자에서 **수학 연산** 옵션이 **미분**에서 **도함수 (dX/dt)**로, **차이**에서 **차분 (dX)**으로, **적분**에서 **적분 (합 (XdT))**으로, **시그마**에서 **시그마 (합 (X))**로 변경되었습니다.
- **(Windows)**( 사운드 파일 정보 ) VI의 사운드 파일 정보 ( 참조 번호 ) 인스턴스의 **참조 번호 입력** 입력이 **사운드 파일 참조 번호**로 변경되었습니다.
- **(Mac OS, Linux)** Set Report Footer Text VI는 LabVIEW 이전 버전에서와는 다소 다른 크기의 바닥글을 생성합니다. Set Report Footer Text VI에는 **HTML footer size** 입력도 포함되어 있습니다.
- (Timed 구조 시작 동기화) VI에는 그룹에 지정하는 Timed 구조를 추가하기 전에 모든 Timed 구조를 제거하고 전체 그룹을 삭제하는 **지우기** 입력이 포함되어 있습니다. 이 입력을 사용하여 Timed 루프에 대응하지 않는 모든 Timed 구조를 제거합니다. 또한 이 VI에는 LabVIEW가 이름을 지정한 동기화 그룹에 추가한 후 그룹의 모든 Timed 구조 이름을 반환하는 **Timed 구조 이름 출력** 출력이 포함되어 있습니다.
- **(Windows)** LabVIEW 8.0에서, ( 폴더 열기 ) 함수는 **패턴**이 이미 확장자에 포함되지 않은 경우 .\*를 **패턴** 입력에 추가합니다. LabVIEW 8.2에서는 파일 이름에 확장자가 없으나 **패턴**에 확장자가 있는 경우 함수가 **패턴** 입력을 변경하지 않고 .을 파일 이름에 추가합니다.

- 블록다이어그램에 ( 정적 VI 참조 ) 함수를 놓은 후 , 함수를 더블 클릭하여 VI 를 선택할 수 있는 파일 대화 상자를 디스플레이합니다 . LabVIEW 8.0 에서는 이 함수를 더블 클릭하면 파일 없음 에러 대화 상자가 디스플레이되었습니다 .
- ( 파일로 포맷 ), ( 문자열로 포맷 ), ( 파일로부터 스캔 ) 과 ( 문자열로부터 스캔 ) 함수의 **포맷 문자열** 입력은 더이상 필수 입력이 아닙니다 .
- 프로젝트에서 공유 변수 노드가 포함된 VI 를 열었으나 공유 변수 노드가 **프로젝트 탐색기** 윈도우에서 관련된 공유 변수를 찾을 수 없는 경우 공유 변수 노드가 깨집니다 . 찾을 수 없는 공유 변수와 관련된 모든 프런트패널 컨트롤도 깨집니다 . 이 동작은 Windows 에 특정하며 VI 노드를 프로젝트에서 열었을 때에만 발생합니다 . VI 를 주요 어플리케이션 인스턴스에서 열었을 경우 , 공유 변수를 찾을 수 없다는 알림을 받지 않습니다 . LabVIEW 의 이전 버전은 공유 변수 노드가 프로젝트 탐색기 윈도우에서 관련된 변수를 찾을 수 없다는 것을 나타내지 않습니다 .
- LabVIEW 8.0.1 에서 , Timed 루프의 **휴면 해제 이유** 출력은 링입니다 . LabVIEW 8.2 에서는 출력이 열거형 타입이며 , 이는 LabVIEW 8.0 동작과 일관됩니다 .
- 다음 VI 와 함수의 복제 출력의 이름이 **복제 ( 출력 )** 에서 ( 출력 ) 출력으로 변경되었습니다 . 이 변경 사항은 VI 와 함수의 기능에는 영향을 미치지 않습니다 :
  - 참조에 의한 호출 노드
  - 퀘리식 실행
  - 파일 / 디렉토리 정보
  - 객체 정보 얻기
  - 프로퍼티 얻기
  - 인보크 노드
  - 폴더 열거
  - 프로퍼티 노드
  - 토큰 문자열 스캔
  - 참조 번호를 ID 로
  - 프로퍼티 설정

## 새로운 프로퍼티 , 메소드 , 이벤트

LabVIEW 8.2 에는 새로운 VI 서버 클래스 , 프로퍼티 , 메소드 , 이벤트가 포함되어 있습니다 . 새 클래스 , 프로퍼티 , 메소드 , 이벤트의 리스트는 **LabVIEW 도움말** 목차 탭의 **LabVIEW 8.2 의 특징과 변경 내역 » 새 VI 서버 클래스 , 프로퍼티 , 메소드 , 이벤트** 모음을 참조하십시오 .

또한 LabVIEW 8.2 에는 다음과 같은 새 VISA 프로퍼티가 포함되어 있습니다 : PXI/PCI Settings:Is PCI Express, PXI/PCI Settings:Maximum Link Width, PXI/PCI Settings:Actual Link Width, PXI/PCI Settings:Slot Link Width, PXI/PCI Settings:D-Star Bus Number, PXI/PCI Settings:D-Star Set.

# LabVIEW MathScript 개선사항

LabVIEW MathScript 에 대한 추가적인 정보는 *LabVIEW 도움말* **목차** 탭의 **기본 » 수식과 방정식** 모음을 참조하십시오 .

LabVIEW 8.2 는 다음과 같은 MathScript 개선사항과 변경사항을 포함합니다 :

## 새 MathScript 함수

LabVIEW 8.2 에는 다음과 같은 새로운 MathScript 함수가 포함되어 있습니다 . 이러한 함수를 **LabVIEW MathScript 윈도우**나 MathScript 노트에서 사용할 수 있습니다 .

- **plots** 클래스 : area, bar, bar3, bar3h, barh, contour, contour3, contourf, errorbar, ezcontour, ezcontourf, ezmesh, ezmeshc, ezplot, ezplot3, ezpolar, ezsurf, ezsurf, feather, fill, fplot, gplot, meshc, pie, plotmatrix, plotyy, polar, quiver, scatter, scatter3, shg, stem3, strips, surf, treepplot, waterfall.
- **dsp** 클래스 : ac2poly, ac2rc, arburg, arcov, armcov, aryule, barthannwin, bartlett, besslap, besself, bilinear, bitrevorder, blackman, blackmanharris, bohmanwin, buttap, cceps, cheblap, cheb2ap, chebwin, chirp, conv2, convmtx, corrmatrix, czt, dct, dftmtx, digitrevorder, diric, downsample, dst, ellipap, eqtflength, filternorm, filtic, firgauss, firrcos, flattopwin, freqspace, gausspuls, gausswin, gmonopuls, goertzel, hann, icceps, iczt, idct, idst, impinvar, intfilt, invfreqs, invfreqz, is2rc, kaiserord, lar2rc, latc2tf, levinson, lp2bp, lp2bs, lp2hp, lp2lp, lpc, lsf2poly, maxflat, medfilt1, nuttallwin, parzenwin, phasedelay, phasez, poly2ac, poly2lsf, poly2rc, polyscale, polystab, prony, pulstran, rc2ac, rc2is, rc2lar, rc2poly, rceps, rectpuls, rectwin, resample, residuez, rlevinson, schurrc, seqperiod, sgolay, sgolayfilt, sinc, sos2ss, sos2tf, sos2zp, sosfilt, spline, ss2sos, ss2tf, ss2zp, stepz, stmcb, tf2latc, tf2sos, tf2ss, tf2zp, tf2zpk, triang, tripuls, tukeywin, udecode, uencode, upfirdn, upsample, vco, xcorr, xcorr2, xcov, zerophase, zp2sos, zp2ss, zp2tf.
- **support** 클래스 : csvread, csvwrite, dlmread, dlmwrite, labviewroot, type, uiload, what.
- **string** 클래스 : eval.
- **libraries** 클래스 : loadlibrary, calllib, unloadlibrary, libisloaded, libfunctionsview. 이러한 함수를 사용하여 **LabVIEW MathScript 윈도우**나 MathScript 노트에서 공유 라이브러리를 호출할 수 있습니다 . MathScript 에서 공유 라이브러리를 호출하

는 예제는 labview\examples\MathScript\MathScript Shared Libraries 디렉토리의 MathScript Shared Libraries.lvproj 를 참조하십시오 .

## 부수적인 MathScript 개선사항과 변경 내역

LabVIEW 8.2 에서는 MathScript 의 다음과 같은 부수적인 사항이 변경되었습니다 :

- **LabVIEW MathScript 윈도우**의 전반적인 성능이 향상되었습니다 . MathScript 노드의 컴파일 시간이 개선되었습니다 .
- LabVIEW 런타임 엔진은 MathScript 를 지원합니다 . MathScript 노드를 어플리케이션 빌더로 만들려는 독립 어플리케이션과 공유 라이브러리에 포함시킬 수 있습니다 . LabVIEW 런타임 엔진은 현재 일부 MathScript 함수를 지원하지 않습니다 . 스크립트에 이같은 지원되지 않는 함수가 포함된 경우, 어플리케이션이나 공유 라이브러리를 만들기 전에 해당 스크립트를 수정해야 할 수도 있습니다 . 지원되지 않는 MathScript 함수 리스트는 *LabVIEW 도움말*에서 *LabVIEW 런타임 엔진에서 지원되지 않는 MathScript 함수* 항목을 참조하십시오 .
- MathScript help 명령문을 호출하면 LabVIEW 는 HTML 도움말 윈도우에 도움말을 디스플레이합니다 . **LabVIEW MathScript 윈도우의 출력 윈도우**에 도움말을 디스플레이하려면, **파일 » MathScript 환경을 선택하고 HTML 도움말 디스플레이?** 확인란에서 확인 표시를 제거합니다 . 사용자가 함수를 정의하면, LabVIEW 는 항상 **출력 윈도우**에 정의한 함수의 도움말을 디스플레이합니다 .
- 입력 터미널에 지원되지 않는 데이터 타입을 연결하면 MathScript 노드는 다른 스크립트 노드와 다르게 작동합니다 . MathScript 노드에서, LabVIEW 는 데이터 타입을 지원되는 타입으로 변경하거나 깨진 와이어를 디스플레이합니다 . LabVIEW 가 데이터 타입을 변경하는 경우, 변환이 일어나는 터미널에 강제 변환점이 나타납니다 . MathScript 노드에 입력을 연결한 후 입력 터미널에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **데이터 타입 보이기**를 선택하여 입력의 데이터 타입을 봅니다 .
- MathScript 는 텍스트 문자열의 모든 유니코드가 아닌 문자를 지원하지 않지만 변수 이름에서는 지원하지 않습니다 . 변수 이름에서는 ASCII 문자만을 사용할 수 있습니다 . 예를 들어, 텍스트 문자열에서는 á 를 사용할 수 있지만 **LabVIEW MathScript 윈도우**나 MathScript 노드에서는 다음의 스크립트를 호출할 수 없습니다 :

```
á=rand(50, 1)
```

```
plot(á)
```

데이터를 유니코드가 아닌 문자를 포함하고 있는 경로에 저장할 수 있습니다 . 또한 LabVIEW 를 경로에 유니코드가 아닌 문자가 포함된 디렉토리에 설치했을 경우, MathScript 는 올바르게 기능합니다 .

- LabVIEW 는 MathScript 의 복합 논리적 식을 평가하는데 쇼트 서킷 평가 (short circuit evaluation) 를 사용합니다 . 예를 들어, 명령 `if 0 == 0 || foo(a) == 2` 를 실행하는 경우, LabVIEW 는 식의

첫번째 부분이 이미 참이기 때문에 `foo(a)` 를 실행하지 않습니다. 마찬가지로, 명령 `if 0 ~= 0 && foo(a) == 2` 를 실행하는 경우, LabVIEW 는 식의 첫번째 부분이 이미 거짓이기 때문에 `foo(a)` 를 실행하지 않습니다.

LabVIEW 8.0 은 식이 참인지 거짓인지에 상관없이 MathScript 의 모든 복합 논리적 식 부분을 평가하기 때문에 복합 논리적 식을 포함하는 LabVIEW 8.0 스크립트는 LabVIEW 8.2 에서 예상과 다르게 실행될 수도 있습니다. 예를 들어, LabVIEW 8.0 이 명령문 `if`

`0 == 0 || foo(a) == 2` 에서 `foo(a)` 를 실행하는 경우, `foo(a)` 의 결과를 사용하여 스크립트의 변수를 정의할 수 있습니다. 같은 스크립트가 LabVIEW 8.2 에서는 다르게 실행됩니다. LabVIEW 8.2 가 명령문 `if 0 == 0 || foo(a) == 2` 에서 `foo(a)` 를 실행하지 않기 때문에, `foo(a)` 의 결과를 스크립트에서 사용할 수 없습니다.

LabVIEW 가 쇼트 서킷 평가 (short circuit evaluation) 를 사용하지 않도록 하려면 기존 코드에서 복합 논리적 식을 제거합니다.

- MathScript 는 사용자 정의된 함수내에서 `nargin` 과 `nargout` 함수를 지원합니다.
- MathScript 는 `return` 키워드를 지원합니다. 또한 MathScript 는 행렬 인덱스에서 `end` 키워드도 지원합니다.
- `prod` 와 `sum` 함수에는 곱이나 합계를 계산하는 차원을 지정할 수 있는 **b** 입력이 포함되어 있습니다.

## 3D 그림 컨트롤

3D 그림 컨트롤에 대한 추가적인 정보는 *LabVIEW 도움말* **목차** 탭의 **기본 > 그래픽 & 사운드 VI** 모음을 참조하십시오.

**3D 그림 컨트롤 VI**, 프로퍼티, 메소드는 3D 객체를 사용자가 보고 수정할 수 있는 3D 장면으로 변환합니다. 여러 3D 객체를 생성하고 크기, 형태, 움직임, 모양, 장면에서 다른 객체와의 관계를 지정할 수 있습니다.

다음 VI, 프로퍼티, 메소드를 사용하여 3D 객체의 모양을 지정합니다:

- **객체 VI** 를 사용하여 3D 객체를 생성하거나 찾습니다. 또한 장면 객체 프로퍼티와 장면 객체 메소드를 사용하여 3D 객체를 상황에 놓고 프로그램적으로 객체 특징을 할당할 수 있습니다.
- **파일 로드 VI** 를 사용하여 기존 모델이나 장면 파일을 3D 상황에 추가합니다.
- **기하 VI** 를 장면 기하 프로퍼티 및 장면 기하 메소드와 함께 사용하여 3D 객체가 취하는 기하 형을 지정합니다.
- **변환 VI** 를 사용하여 객체를 3D 장면에 놓습니다.

**고급 그림 기능 VI** 를 사용하여 일반적인 3D 장면 작업을 수행합니다. 장면을 위한 별도의 윈도우를 설정하고, 장면 내에 새로운 절단 평면을 생성하고, 빛 소스를 추가하고, 3D 객체에 텍스처를 적용하고, LabVIEW 색 값을 변환하여 3D 그림 컨트롤에 나타나도록 할 수 있습니다.

또한 다음 프로퍼티와 메소드를 사용하여 프로그램적으로 3D 장면을 설정할 수 있습니다 .

- 장면 윈도우 프로퍼티를 사용하여 장면을 별도의 윈도우에서 제공하고 , 윈도우를 설정하고 , 카메라 컨트롤러와 장면의 상호작용을 설정합니다 .
- 장면 절단 평면 프로퍼티를 사용하여 객체가 나타나거나 절단된 장면에서 평면을 지정합니다 .
- 장면 빛 프로퍼티를 사용하여 장면의 빛 소스를 설정합니다 .
- 장면 텍스처 프로퍼티와 장면 텍스처 메소드를 사용하여 3D 객체에 텍스처를 적용합니다 .

3D 그림 컨트롤로 3D 장면을 생성하는 예제는 labview\examples\picture\3D Picture Control 디렉토리의 solarsystem VI 를 참조하십시오 .

## LabVIEW 객체 지향 프로그래밍

LabVIEW 객체 지향 프로그래밍에 대한 추가적인 정보는 *LabVIEW 도움말* 목차 탭의 **기본 »LabVIEW 객체 지향 프로그래밍** 모음을 참조하십시오 .

LabVIEW 객체 지향 프로그래밍은 캡슐화와 상속 등 C++ 및 Java 와 같은 다른 객체 지향 프로그래밍 언어의 개념을 사용합니다 . 이러한 개념을 사용하여 어플리케이션 내의 다른 코드 섹션에 영향을 미치지 않고도 관리하고 수정하기 쉬운 코드를 생성할 수 있습니다 . LabVIEW 의 객체 지향 프로그래밍을 사용하여 사용자 정의 데이터 타입을 생성할 수 있습니다 .

## LabVIEW 클래스 생성하기

LabVIEW 클래스를 생성하여 LabVIEW 에서 사용자 정의 데이터 타입을 생성할 수 있습니다 . LabVIEW 클래스는 객체와 관련된 데이터뿐만 아니라 데이터에 실행할 수 있는 동작을 정의하는 메소드를 정의합니다 .

LabVIEW 에서 , 클래스의 데이터는 프라이빗으로서 클래스의 멤버인 VI 만이 데이터에 접근할 수 있습니다 . 프라이빗 데이터 컨트롤에서 클래스의 데이터를 정의합니다 . LabVIEW 클래스를 생성하고 저장할 때 , LabVIEW 는 새로운 데이터 타입을 정의하는 클래스 라이브러리 파일 (.lvclass) 을 생성합니다 . 클래스 라이브러리 파일을 프라이빗 데이터 컨트롤과 VI 리스트 및 다양한 VI 프로퍼티 등의 사용자가 생성한 모든 멤버 VI 의 정보를 기록합니다 . 클래스 라이브러리는 프로젝트 라이브러리 (.lvlib) 와 유사합니다 . 그러나 , 클래스 라이브러리는 새로운 데이터 타입을 정의합니다 .

다음 방법 중 하나를 사용하여 클래스를 생성할 수 있습니다 :

- **프로젝트 탐색기** 윈도우의 **내 컴퓨터**에서 마우스 오른쪽 버튼을 클릭한 후 , 바로 가기 메뉴에서 **새로 만들기 » 클래스**를 선택합니다 .
- **파일 » 새로 만들기**를 선택하여 **새로 만들기** 대화 상자를 디스플레이한 후 **새로 만들기** 리스트에서 **기타 파일 » 클래스**를 선택합니다 .

## 프라이빗 데이터 컨트롤 정의하기

LabVIEW 는 사용자가 LabVIEW 클래스를 생성할 때 자동으로 클래스의 프라이빗 데이터 컨트롤을 생성합니다 .

컨트롤 편집기 윈도우를 사용하여 클래스의 프라이빗 데이터 컨트롤을 사용자 정의할 수 있습니다 . **프로젝트 탐색기** 윈도우 클래스의 프라이빗 데이터 컨트롤을 더블 클릭하면 LabVIEW 는 컨트롤 편집기 윈도우를 디스플레이 합니다 . **클래스 프라이빗 데이터의 클러스터** 안에 컨트롤과 인디케이터를 놓으면 LabVIEW 클래스의 프라이빗 데이터 타입을 정의할 수 있습니다 . **클래스 프라이빗 데이터의 클러스터** 안의 컨트롤에 설정한 기본값은 클래스의 기본값입니다 .

## 멤버 VI 생성하기

멤버 VI 는 LabVIEW 클래스에 생성한 메소드를 수행합니다 . 멤버 VI 를 생성하여 클래스의 프라이빗 데이터에 작업을 수행합니다 . 멤버 VI 는 멤버 VI 를 생성한 LabVIEW 클래스의 멤버로 , 클래스의 프라이빗 데이터 컨트롤 밑의 **프로젝트 탐색기** 윈도우에 나타납니다 . 한 클래스의 단일 멤버 VI 를 사용하여 대부분의 메소드를 정의할 수 있지만 , 일부 메소드는 클래스 계층구조 전반에 여러 멤버 VI 를 생성하면서 정의할 수 있습니다 .

여러 핸들링과 클래스 객체를 포함하는 VI 템플릿 , 새 VI , 혹은 상위 멤버 VI 에서 멤버 VI 를 생성할 수 있습니다 . 클래스에서 마우스 오른쪽 버튼을 클릭하고 다음의 바로가기 메뉴 아이템에서 선택하십시오 :

- **새로 만들기 »VI**—새 멤버 VI 를 엽니다 .
- **새로 만들기 »다이나믹 VI**—LabVIEW 는 **여러 입력**과 **여러 출력** 클러스터 , 여러 핸들링을 위한 케이스 구조 , 입력 LabVIEW 클래스 , 출력 LabVIEW 클래스를 통해 새 멤버 VI 를 채웁니다 .
- **새로 만들기 »덮어쓰기 VI**—상위 멤버 VI 를 덮어쓰는 멤버 VI 를 생성합니다 .

## 다른 개발자와 사용자에게 LabVIEW 클래스 배포하기

개발한 LabVIEW 클래스를 다른 LabVIEW 클래스 개발자와 LabVIEW 클래스 사용자에게 배포할 수 있습니다 . 클래스를 배포하는 방법에는 여러가지가 있으므로 가장 적합한 방법을 선택하도록 합니다 . 어플리케이션 빌더를 사용하여 압축 (Zip) 파일을 생성함으로써 하나 혹은 그 이상의 클래스를 배포할 수 있습니다 . 또한 배포하기 전에 LabVIEW 클래스 잠그기를 수행하여 프라이빗 데이터와 멤버 VI 에 대한 LabVIEW 클래스 사용자의 접근을 제한할 수도 있습니다 . 클래스를 잠그면 어플리케이션에 에러가 발생하는 것을 방지할 수 있습니다 .

## LabVIEW 프로젝트 개선사항

LabVIEW 8.2 에서는 LabVIEW 프로젝트와 관련된 기능의 다음과 같은 사항이 개선되었습니다 :

# 어플리케이션 빌더 개선사항

LabVIEW 8.2 LabVIEW Professional Development System 의 어플리케이션 빌더에서는 다음과 같은 사항이 개선되었습니다 :

- **제품 버전 자동 증가**—설치 프로그램을 여러번 만드는 경우 , LabVIEW 는 설치 프로그램을 새롭게 만들 때마다 자동으로 제품 버전 번호를 증가시킬 수 있습니다 . **제품 버전 자동 증가** 확인란은 **설치 프로그램 프로퍼티** 대화 상자의 **제품 정보** 페이지에 나타납니다 .
- **설치 프로그램을 저장하는 미디어 크기 지정하기**—설치 프로그램 구성요소를 저장할 때 미디어 크기를 사용자 지정할 수 있습니다 . **미디어 확장 활성화** 확인란은 **설치 프로그램 프로퍼티** 대화 상자의 **고급** 페이지에 나타납니다 . 미디어의 풀다운 메뉴에는 **사용자** 옵션이 포함되어 있습니다 . 이 옵션을 사용하여 **미디어 크기 (MB)** 텍스트 박스에 임의의 미디어 크기를 입력하십시오 .
- **Windows 2000 또는 이후 버전 필요**—설치 프로그램을 실행할 때 사용자가 반드시 Windows 2000 또는 이후 버전을 사용하도록 할 수 있습니다 . **Windows 2000 또는 이후 버전** 옵션은 **설치 프로그램** 대화 상자 **고급** 페이지의 **시스템 사양** 섹션에 나타납니다 .
- **설치 프로그램 구성요소 캐시하기**—설치 프로그램을 한번 이상 만들고 , 만든 설치 프로그램에 추가적인 설치 프로그램이나 구성요소가 포함된 경우 , 캐시하기를 사용하면 설치 프로그램을 만들 때마다 추가적인 구성요소의 위치를 지정하지 않아도 됩니다 . 처음으로 설치 프로그램을 만들면 **배포 찾기 ?** 대화 상자가 나타나 추가적인 구성요소를 포함하는 배포를 찾도록 입력요청합니다 . **이 배포로부터 캐쉬 구성요소** 확인란에 확인 표시를 하면 파일이 배포에서 로컬 시스템의 지정된 위치로 복사됩니다 . 향후에 구성요소를 포함하는 설치 프로그램을 만들게 되면 , 어플리케이션 빌더는 배포 CD 를 넣도록 입력요청하지 않고 자동으로 로컬 시스템에서 구성요소를 복사합니다 .
- **다이나믹 VI 와 의존성**—**소스 파일 셋팅 (어플리케이션)** 페이지에서 다이나믹 VI 로 지정된 VI 가 만들어진 어플리케이션 이외의 다른 대상으로 갈 경우 , 어플리케이션 빌더는 다이나믹 VI 의 모든 의존성을 만들어진 어플리케이션에 보관하지 않고 다이나믹 VI 가 있는 새로운 대상으로 이동합니다 . 둘 이상의 다이나믹 또는 최상위 레벨 VI 가 VI 를 호출하고 두 개의 다른 장소로 이동하고자 할 경우 , 어플리케이션 빌더는 VI 와 모든 SubVI 를 만들어진 어플리케이션으로 이동합니다 . **소스 파일 셋팅** 페이지에서 **참조될 경우 포함**으로 지정된 VI 를 새로운 위치로 이동하려면 VI 를 반드시 다이나믹 VI 로 지정해야 합니다 .
- **프로젝트 가명 파일을 공유 라이브러리와 함께 사용하기**—**공유 라이브러리 프로퍼티** 대화 상자의 **고급** 페이지에는 프로젝트 가명 파일을 공유 라이브러리에 연관시키는 **기본 프로젝트 가명 파일 사용** 확인란이 있습니다 . 확인란에서 확인 표시를 제거할 경우 , **프로젝트의 가명 파일** 텍스트 박스에서 사용할 가명 파일을 지정합니다 .
- **추가적인 LabVIEW 헤더 파일을 공유 라이브러리에 포함시키기**—**공유 라이브러리 프로퍼티** 대화 상자의 **고급** 페이지에는 빌더 과정에서 생성된 모든 추가적인 LabVIEW 헤더 파일 및 참조를 복사하는 **추가적인**

LabVIEW **헤더 파일 포함** 확인란이 있습니다. 추가적인 헤더 파일을 포함시키면 LabVIEW 가 빌드한 공유 라이브러리를 이 헤더 파일이 필요한 C 또는 다른 언어에서 사용할 수 있습니다.

- **소스 배포 대화 상자 배포 셋팅** 페이지의 **새 대상** 버튼은 **추가** 버튼으로 변경되었습니다. 또한 소스 배포에서 파일을 제외하는 것과 관련된 모든 옵션은 **추가적인 예외** 페이지로 이동되었습니다.
- **어플리케이션 프로퍼티** 대화 상자와 **공유 라이브러리 프로퍼티** 대화 상자의 **대상** 페이지에 있는 **새 대상** 버튼은 **추가** 버튼으로 변경되었습니다.
- VI 를 먼저 저장하지 않고도 어플리케이션이나 소스 배포를 만들 수 있습니다.
- 어플리케이션 빌더에서 생성한 압축 (Zip) 파일에 .zip 확장자를 추가하지 않는 경우, LabVIEW 는 자동으로 확장자를 추가합니다.

## 새로 만들기 대화 상자 페이지

LabVIEW 8.2 에서는 다음과 같은 새 **어플리케이션 빌더** 대화 상자 페이지가 생겼습니다:

- **설치 프로그램 프로퍼티** 대화 상자
  - **대화 상자 정보**—이 페이지를 사용하여 설치 프로그램의 사용자 인터페이스를 디자인합니다. 텍스트의 언어를 설정하고, 사용자 readme 와 라이선스 협약을 디스플레이하며, 환영 제목과 메시지를 설정할 수 있습니다. 이 페이지는 LabVIEW 8.0 의 **제품 정보** 페이지의 **대화 상자 정보** 섹션을 대체합니다.



**노트** **대화 상자 정보** 페이지의 **사용자 라이선스 협약 포함** 옵션은 **제품 정보** 페이지에 나타나던 **라이선스 파일** 옵션을 대체합니다.

- **하드웨어 설정**—이 페이지를 사용하여 설치 프로그램에 포함할 하드웨어 설정 정보의 소스를 지정합니다. 새 **반입 모드** 섹션에는 Measurement & Automation Explorer 로부터 하드웨어 설정 파일을 반입하기 위한 추가적인 옵션이 포함되어 있습니다. 이 페이지는 LabVIEW 8.0 **고급** 페이지의 **하드웨어 설정** 섹션을 대체합니다.
- **어플리케이션 프로퍼티, 공유 라이브러리 프로퍼티, 소스 배포 프로퍼티** 대화 상자
  - **추가적인 예외**—이 페이지를 사용하여 타입 정의, 사용하지 않는 다형성 VI 인스턴스, 프로젝트 라이브러리의 참조되지 않는 멤버를 제거하거나 연결 해제하기 위한 셋팅을 설정합니다. 이 셋팅을 사용하여 빌드의 크기를 줄입니다.

## 빌드 스펙 복제하고 재배포하기

**프로젝트 탐색기** 윈도우에 빌드 스펙을 복제할 수 있습니다. 복제할 빌드 스펙 아이템에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **복제**를 선택하여 **빌드 스펙** 아래에 빌드 스펙의 복사본을 생성합니다.

또한 빌드 스펙 아이템을 끌어서 놓아 같은 **빌드 스펙** 내의 빌드 순서를 다시 정렬할 수 있습니다.

## 어플리케이션 빌더 정품인증하기 (Windows)

LabVIEW Base Package 혹은 Full Development System 의 정품인증 버전을 가지고 있는 경우, **도움말** » **어플리케이션 빌더 정품인증**을 선택하여 어플리케이션 빌더를 정품인증합니다. LabVIEW 를 재실행할 때 라이선스가 적용됩니다.

## 프로그램적으로 프로젝트 라이브러리 생성하고 공유 변수 추가하기

labview\vi.lib\Utility\Variable 디렉토리의

CreateOrAddLibrary VI 를 사용하여 프로그램적으로 라이브러리를 프로젝트나 타겟, 폴더 또는 다른 라이브러리 같은 상위 아이템에 추가합니다.

또한 labview\vi.lib\Utility\Variable 디렉토리의

AddSharedVariableToLibrary VI 를 사용하여 프로그램적으로 공유 변수를 라이브러리에 추가할 수 있습니다.

## LabVIEW 프로젝트와 프로젝트 라이브러리를 이전 버전으로 저장하기

LabVIEW 8.0 에서 읽을 수 있도록 LabVIEW 프로젝트와 프로젝트 라이브러리를 저장할 수 있습니다. LabVIEW 프로젝트를 이전 버전으로 저장하려면, **프로젝트 탐색기** 윈도우에서 **파일** » **이전 버전으로 저장**을 선택합니다. 프로젝트 라이브러리를 이전 버전으로 저장하려면, **프로젝트 탐색기** 윈도우의 라이브러리 파일에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **이전 버전으로 저장**을 선택하거나 프로젝트 라이브러리를 열고 **파일** » **이전 버전으로 저장**을 선택합니다.

## 어플리케이션 인스턴스 선택하기

LabVIEW 는 LabVIEW 프로젝트에 각 타겟에 대한 어플리케이션 인스턴스를 생성합니다. **프로젝트 탐색기** 윈도우에서 VI 를 열 때, VI 는 타겟의 어플리케이션 인스턴스에서 열립니다. 또한, LabVIEW 는 주요 어플리케이션 인스턴스를 생성합니다. 여기에는 프로젝트의 일부가 아닌 열린 VI 와 프로젝트에서 열지 않은 VI 가 포함됩니다. 어플리케이션 인스턴스 바로 가기 메뉴를 사용하여 특정한 어플리케이션 인스턴스에서 VI 를 열 수 있습니다. 프린트 패널 윈도우 또는 블록 다이어그램 윈도우의 왼쪽 아래 코너에 있는 현재 인스턴스 이름에서 마우스 오른쪽 버튼을 클릭하여 바로 가기 메뉴를 디

스플레이하고 모든 어플리케이션 인스턴스 중에서 선택합니다. 새로운 어플리케이션 인스턴스를 선택하면 선택한 어플리케이션 인스턴스에서 VI 가 다시 열립니다. VI 는 원래 어플리케이션 인스턴스에서도 열려 있습니다.

## 공유 변수 개선사항

LabVIEW 8.2 에서는 다음과 같이 공유 변수가 개선되었습니다 :

- LabVIEW 는 네트워크 : 버퍼 크기 , 네트워크 : 원소 크기 , 네트워크 : 웨이브폼당 포인트 프로퍼티를 적절히 사용하여 네트워크 공유 변수의 네트워크 버퍼 크기를 계산합니다 .
  - 스칼라 네트워크 공유 변수의 경우 , LabVIEW 는 버퍼할 값의 개수를 나타내는 네트워크 : 버퍼 크기 프로퍼티를 사용합니다 .
  - 배열과 문자열 네트워크 공유 변수의 경우 , LabVIEW 는 네트워크 : 버퍼 크기와 네트워크 : 원소 크기 프로퍼티를 사용합니다 .
  - 웨이브폼 네트워크 공유 변수의 경우 , LabVIEW 는 네트워크 : 버퍼 크기와 네트워크 : 웨이브폼당 포인트 프로퍼티를 사용합니다 .
  - 웨이브폼 네트워크 공유 변수의 배열인 경우 , LabVIEW 는 네트워크 : 버퍼 크기와 네트워크 : 원소 크기 , 네트워크 : 웨이브폼당 포인트 프로퍼티를 사용합니다 .
- 공유 변수 노드의 변수 입력은 노드가 데이터를 쓰도록 설정된 경우에만 필요합니다 .
- **공유 변수 프로퍼티의 변수** 페이지에서는 다음과 같은 사항이 변경되었습니다 :
  - **데이터 타입** 폴다운 메뉴의 **사용자** 아이템은 **사용자 컨트롤로부터** 로 변경되었습니다 .
  - **데이터 타입** 폴다운 메뉴에는 새 **베리언트** 아이템이 포함되어 있습니다 .
  - 네트워크 버퍼의 원소 크기는 더 이상 디스플레이되지 않습니다 .
  - **접근 타입**을 **읽기 전용**이나 **쓰기 전용**으로 설정한 경우 , 각각 데이터 읽기 전용 또는 데이터 쓰기 전용으로 설정된 공유 변수를 생성할 수 있습니다 . 읽기 전용 또는 쓰기 전용인 소스에 연결된 공유 변수에서 마우스 오른쪽 버튼을 클릭하면 , LabVIEW 는 바로 가기 메뉴의 **쓰기로 변경** 및 **읽기로 변경** 옵션을 비활성화합니다 .
  - LabVIEW 8.0 에서는 공유 변수 노드에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **타임스탬프 보기**를 선택하여 단일 프로세스 공유 변수에 대한 타임스탬프 정보를 얻습니다 . LabVIEW 8.2 에서는 , 우선 프로젝트의 공유 변수에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **프로퍼티**를 선택하고 , **변수** 페이지의 **타임스탬프 활성화** 확인란에 확인 표시를 하여 단일 프로세스 공유 변수에 대한 타임스탬프 정보를 얻습니다 . 타임스탬프 정보를 보고 공유 변수 노드에 **타임스탬프** 출력을 추가하려면 , 공유 변수 노드에서 마우스 오른쪽 버튼을 클릭한 후 바로 가기 메뉴에서 **타임스탬프 보기**를 선택합니다 . LabVIEW 8.0 에서 생성한 단일

프로세스 공유 변수를 LabVIEW 8.2 에서 로드하는 경우 , 타임스탬프는 기본으로 활성화됩니다 .

- LabVIEW 8.0 에서 , LabVIEW 는 **변수 선택** 대화 상자의 리스트에서 선택할 수 없는 아이템을 비활성화합니다 . LabVIEW 8.2 에서는 사용자가 리스트에서 유효하지 않은 아이템을 선택하는 경우 LabVIEW 가 **확인** 버튼을 비활성화합니다 .

## 기타 프로젝트 개선사항

LabVIEW 8.2 에서는 다음과 같이 프로젝트의 부수적인 사항이 개선되었습니다 :

- LabVIEW 8.0 에서는 프로젝트 라이브러리에서 타입 정의를 생성하고 프라이빗으로 타입 정의의 접근 영역을 설정하는 경우 , 프로젝트 라이브러리 밖에서도 VI 의 프라이빗 타입 정의를 참조할 수 있습니다 . LabVIEW 8.2 에서는 라이브러리 밖에서 프라이빗 타입 정의를 참조할 수 없습니다 .
- X 컨트롤에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **새로 만들기 »VI** 를 선택하여 X 컨트롤 안에 VI 를 생성합니다 .
- LabVIEW 8.0 에서 , 디버깅할 때 브레이크포인트를 설정한 후에는 X 컨트롤을 기능 , 프로퍼티 , 또는 메소드 VI 를 저장해야 합니다 . LabVIEW 8.2 에서는 브레이크포인트를 설정한 후에 기능 , 프로퍼티 , 또는 메소드 VI 를 저장할 필요가 없습니다 .
- 구매한 LabVIEW 버전에 특정한 유효한 라이선스가 없는 LabVIEW 프로젝트 내의 라이브러리에 암호를 입력하면 아이템을 라이브러리 안이나 밖으로 끌어내 놓을 수 없습니다 .
- 프로젝트 라이브러리가 암호로 보호되어 있으며 해당 암호가 LabVIEW 암호 캐시에 없는 경우 , 프로젝트 라이브러리에서 마우스 오른쪽 버튼을 클릭하고 바로 가기 메뉴에서 **암호 입력** 을 선택하여 프로젝트 라이브러리를 잠금 해제할 수 있습니다 .

## 여러 클라이언트로부터 원격으로 VI 컨트롤하기

여러 클라이언트가 동시에 원격으로 어플리케이션이나 VI 를 컨트롤할 수 있습니다 . VI 의 동시 컨트롤을 허용하려면 VI 가 재호출되어야 합니다 . VI 를 재호출로 하려면 **파일 »VI 프로퍼티** 를 선택하고 **항목** 리스트에서 **실행** 을 선택한 후 , **재호출 실행** 확인란에 확인 표시를 합니다 . LabVIEW 는 리모트 프런트패널의 각 클라이언트 요청에 대한 재호출 VI 의 복제를 엽니다 . 웹 서버 :VI 접근 리스트 프로퍼티를 사용하여 프로그램적으로 리모트 프런트패널 연결시 이미 메모리에 있는 복제본에 대한 접근을 제한할 수 있습니다 .

## 공유 라이브러리 파일에서 함수 반입하기

Windows .dll 파일 , Mac OS .framework 파일 , Linux .so 파일에서 반입된 함수를 위한 VI 를 생성하고 업데이트할 수 있습니다 . 그 후 **도구 » 반입 » 공유 라이브러리** 를 선택하고 입력요청을 따라 공유 라이브러리 파일

을 위한 래퍼 VI 를 생성합니다 . 공유 라이브러리 파일의 이름과 헤더 .h 파일 을 제공해야 합니다 .

공유 라이브러리 파일을 위한 래퍼를 생성하는 것에 대한 추가적인 정보는 **LabVIEW 도움말 목차 탭의 기본 » 텍스트 기반 프로그래밍 언어에서 작성된 코드 호출하기 » 사용법 » 공유 라이브러리 파일에서 함수 반입하기**를 참조하십시오 .

## 인스트루먼트 드라이버 템플릿

LabVIEW 8.2 에는 새 인스트루먼트 드라이버 생성 마법사를 위한 다음과 같은 새로운 템플릿이 포함되어 있습니다 :

- **일반 목적 ( 레지스터 기반 )**—클래스에 특정한 템플릿이 없는 레지스터 기반 인스트루먼트에 사용합니다 . 일반적인 레지스터 기반 인스트루먼트는 다음을 포함합니다 : VXI 및 PXI.
- **스펙트럼 분석기**—주파수 범위 , 스위프 커플링 설정과 같은 기본 작업을 컨트롤합니다 . 이 템플릿은 마커 설정 및 쿼리하기와 같은 고급 기능도 포함합니다 .

## .NET 과 ActiveX 개선사항 (Windows)

.NET 컨트롤에는 사용자 인터페이스 스레드 의존성이 더 이상 존재하지 않습니다 . .NET 컨트롤 작업의 경우 , 사용자 인터페이스 스레드에서 실행하기 위해 VI 또는 subVI 를 설정할 필요가 없습니다 . 기본으로 , LabVIEW 는 모든 .NET 컨트롤이 올바른 스레드에서 실행되도록 설정하며 , 대부분의 경우 사용자 인터페이스 스레드입니다 .

LabVIEW 는 ActiveX 와 .NET 콜백 VI 에서 디버깅을 지원하지 않습니다 . ActiveX 와 .NET 콜백 VI 에서 브레이크포인트를 사용하는 경우 , LabVIEW 는 브레이크포인트에서 일시 정지하지 않습니다 .

LabVIEW 8.0 과 이전 버전에서 , .NET 참조 번호 프로브는 참조의 16 진수 값만 디스플레이합니다 . LabVIEW 8.2 의 경우 , .NET 참조 번호 프로브는 .NET 참조 번호 , 객체의 타입 , 객체의 해쉬 코드 , 참조의 16 진수 값으로 나타낸 객체에 ToString() 메소드의 호출 결과를 디스플레이합니다 . 값이 유효하지 않은 참조 번호인 경우 이 프로브를 사용하여 브레이크포인트를 설정할 수 있습니다 .

LabVIEW 8.2 에서는 다음과 같이 .NET 과 ActiveX 메뉴가 변경되었습니다 :

- **도구 » .NET & ActiveX** 메뉴 아이템인 **.NET 컨트롤을 팔레트에 추가**와 **ActiveX 컨트롤을 팔레트에 추가**는 **도구 » 반입 » .NET 컨트롤을 팔레트로**와 **도구 » 반입 » ActiveX 컨트롤을 팔레트로**로 각각 변경되었습니다 .
- **도구 » .NET & ActiveX » ActiveX 프로퍼티 탐색** 메뉴 아이템은 **보기 » ActiveX 프로퍼티 브라우저**로 변경되었습니다 .

## NI 예제 탐색기의 개선사항

**가장 최근** 예제 폴더는 기본으로 NI 예제 탐색기의 **탐색** 탭에 나타납니다 . NI 예제 탐색기의 **설정** 버튼을 클릭한 후 **일반** 탭을 클릭하여 대화 상자의 **가장 최근** 예제 폴더에서 디스플레이될 예제의 최대 개수를 설정할 수 있습니다 .

NI 예제 탐색기의 개선사항에 대한 추가적인 정보는 *NI 예제 탐색기 도움말* 을 참조하십시오 . NI 예제 탐색기에서 **도움말** 버튼을 클릭하여 *NI 예제 탐색기 도움말* 을 디스플레이합니다 .

## 소스 컨트롤 개선사항

LabVIEW 의 소스 컨트롤에 대한 추가적인 정보는 *LabVIEW 도움말* **목차** 탭의 **기본 » 프로젝트 구성하기와 관리하기** 모음을 참조하십시오 .

다음 타사 소스 컨트롤 제공자는 LabVIEW 8.2 Professional Development System 에서 통합 및 기본 기능의 테스트를 마쳤습니다 :

- Seapine Surround SCM
- Borland StarTeam
- Telelogic Synergy
- PushOK (CVS 및 SVN 플러그인 )
- ionForge Evolution



**노트** 현재 ionForge Evolution 2.8 및 이후 버전은 LabVIEW 에서 작동합니다 .

또한 LabVIEW 가 VI 의 그래픽 비교에서 사용하는 과정의 경우 , LabVIEW 8.0 호환보다 더 많은 타사 소스 컨트롤 제공자가 호환됩니다 .

## LabVIEW 프로젝트 폴더에서 소스 컨트롤 작동

LabVIEW 프로젝트 내의 아이템 폴더에서 소스 컨트롤 작업을 수행할 수 있습니다 . LabVIEW 는 계층구조 내의 모든 적절한 아이템에 작업을 수행합니다 . 예를 들어 파일을 소스 컨트롤에 추가하는 경우 , LabVIEW 는 폴더 내의 파일 중 사용자가 소스 컨트롤에 추가하지 않은 파일들만 추가합니다 . 아이템 폴더에서 소스 컨트롤 작업을 수행할 때 폴더 아이템에 적용되는 모든 소스 컨트롤 설정 옵션이 나타납니다 .



**노트** 폴더 내에 프로젝트 라이브러리가 있는 경우 , 수행하는 소스 컨트롤 작업은 프로젝트 라이브러리 내의 아이템까지 확장되지 않습니다 . 프로젝트 라이브러리 내의 아이템에 소스 컨트롤 작업을 수행하려면 수동으로 해당 아이템을 선택해야 합니다 . 프로젝트 라이브러리 내의 폴더에서 소스 컨트롤 작업을 수행할 수 있습니다 .

## 소스 컨트롤 작업의 저장되지 않은 파일

변경 내역이 저장되지 않은 파일에 체크인 작업을 시도하는 경우, **저장되지 않은 파일** 대화 상자가 나타나 변경 내역을 저장하거나 무시하도록 입력요청합니다. **소스 컨트롤 작업** 대화 상자에서 **확인** 버튼을 클릭하고 소스 컨트롤 작업을 포함한 LabVIEW 파일 타입에 저장하지 않은 변경사항이 있는 경우 이 대화 상자가 나타납니다. 또한 저장하지 않은 변경사항이 있는 라이브러리 또는 프로젝트 파일에서 **도구 » 소스 컨트롤 » 차이 보기**를 선택한 경우에도 이 대화 상자가 나타납니다.

## 기타 소스 컨트롤 개선사항

LabVIEW 8.2에서는 다음과 같이 소스 컨트롤의 부수적인 사항이 개선되었습니다:

- LabVIEW 8.0에서, **도구 » 소스 컨트롤** 메뉴의 **소스 컨트롤에 추가** 메뉴 아이템은 저장되지 않은 파일에서 사용할 수 있었습니다. LabVIEW 8.2의 경우, 파일을 저장하기 전에는 **소스 컨트롤에 추가** 메뉴 아이템을 사용할 수 없습니다.
- **도구 » 소스 컨트롤 » 소스 컨트롤 설정**을 선택하고 **vi.lib 제외**와 **instr.lib 제외** 확인란을 사용하여 파일을 소스 컨트롤에 추가할 때 **vi.lib**와 **instr.lib**의 파일을 제외합니다. 파일을 소스 컨트롤에 추가할 때 LabVIEW가 의존성을 포함하도록 설정하는 경우, 이 옵션은 작업에서 필요하지 않은 파일을 제외합니다.
- **Perforce 개정 히스토리** 대화 상자에는 **이 개정으로 동기화**와 **변경 리스트 설명하기** 옵션이 포함되어 있습니다. 이 옵션에 접근하려면 **히스토리** 리스트의 개정에서 마우스 오른쪽 버튼을 클릭합니다.
- **Perforce 개정 히스토리** 대화 상자에서 **변경 리스트에 동작** 영역은 현재 변경 리스트에서 취한 동작을 디스플레이합니다. 가능한 작업은 추가, 편집, 삭제, 분기, 통합 등이 있습니다. 또한 이 필드는 동작이 발생한 파일 타입도 디스플레이합니다.
- **(Windows) Perforce 코어 설치** 프로그램을 설치한 경우, **Perforce SCM**은 **옵션** 대화 상자 **소스 컨트롤** 페이지의 **소스 컨트롤 제공자 이름** 폴다운 메뉴에 나타납니다.

## TDM 개선사항

LabVIEW 8.2에서는 다음과 같이 TDM이 개선되었습니다.

## TDM 스트리밍 파일 포맷

LabVIEW 8.2에는 2진 파일을 저장하기 위한 TDM 스트리밍 (.tdms) 파일 포맷이 포함되어 있습니다. .tdms 파일 포맷은 LabVIEW 8.0 및 이전 버전에서 사용되는 .tdm 파일 포맷보다 더욱 빠른 쓰기 성능을 제공합니다. 또한 .tdms 파일 포맷은 프로퍼티를 정의할 때 더욱 단순한 인터페이스를 제공합니다.

TDMS VI 와 함수에 대한 정보는 이 문서의 [TDM 스트리밍 VI 와 함수](#) 섹션을 참조하십시오 .

## 사용자 정의 TDM 과 TDMS 프로퍼티

사용자 정의 프로퍼티를 생성하고 사용자 정의하고 2 진 측정 파일의 DAQmx 프로퍼티를 지정할 수 있습니다 . ( 측정 파일에 쓰기 ), ( 데이터 쓰기 ), ( 프로퍼티 설정 ), ( 프로퍼티 열기 ) 익스프레스 VI 를 사용하여 .tdm 또는 .tdms 파일의 사용자 정의 프로퍼티를 설정합니다 .

( 측정 파일에 쓰기 ) 익스프레스 VI 설정 페이지의 **고급** 버튼을 클릭하여 프로퍼티를 설정합니다 . ( 데이터 쓰기 ), ( 프로퍼티 설정 ), ( 프로퍼티 열기 ) 익스프레스 VI 의 설정 페이지를 디스플레이하여 프로퍼티를 설정합니다 .

## 웹 서비스 반입하기 (Windows)

웹 서비스의 모든 복잡한 설정을 관리하지 않고도 LabVIEW 8.2 에서 웹 서비스를 사용할 수 있습니다 . 웹 서비스를 VI 의 프로젝트 라이브러리로 변환한 후 로컬 컴퓨터에서 독립적으로 사용할 수 있는 웹 서비스처럼 프로그램할 수 있습니다 . 웹 서비스 설명 언어 (WSDL) 에 유효한 URL 을 제공해야 합니다 . WSDL 은 웹 서비스와 기능을 설명하는데 사용되는 XML 포맷의 언어입니다 . **도구 » 웹 서비스**를 선택하여 마법사를 시작합니다 . 이 마법사는 웹 서비스에 메소드를 반입하고 VI 라이브러리를 생성하는 과정을 도와줍니다 .



### 노트

웹 서비스 반입 마법사를 사용하려면 .NET Framework 1.1 또는 이후 버전이 설치되어 있어야 합니다 .

웹 서비스 반입에 대한 추가적인 정보는 [LabVIEW 도움말 목차](#) 탭의 **기본 » Windows 연결 » 개념 » 웹 서비스 반입하기** 모음을 참조하십시오 .

## 외부 코드 함수 변경 내역

외부 소스 코드가 사용하는 Memory Manager 인터페이스의 일부에서 데이터 타입 size\_t 가 타입 int32 를 대체합니다 . 이 변경사항은 기존 DLL 과 호환됩니다 .

Memory Manager 함수에 대한 추가적인 정보는 [LabVIEW 도움말에서 Memory Manager Functions](#) 항목을 참조하십시오 .

National Instruments, NI, ni.com 과 LabVIEW 는 National Instruments Corporation 의 상표입니다 . National Instruments 의 상표들에 관한 더 많은 정보를 원하신다면 ni.com/legal 에서 [Terms of Use](#) 란을 참조하십시오 . 이 문서에서 언급된 다른 제품과 회사의 이름들은 각각 해당 회사들의 상표이거나 상호입니다 . 적절한 위치에서 내셔널인스트루먼트의 특허권을 참조할 수 있습니다 : 소프트웨어의 [Help » Patents](#), CD 의 patents.txt 파일 , 또는 ni.com/patents . MATLAB® 은 The MathWorks, Inc. 의 등록된 상표입니다 .