

# Hinweise zum Upgrade von LabVIEW™

## Version 8.2

In dieser Broschüre wird beschrieben, wie LabVIEW unter Windows, Mac OS und Linux auf Version 8.2 aktualisiert wird, welche Probleme beim Upgrade auftreten können und um welche Funktionen die neue LabVIEW-Version erweitert wurde.

Wenn Sie von LabVIEW 7.1 oder einer früheren Version auf LabVIEW 8.2 aktualisieren, lesen Sie die *Hinweise zum Upgrade auf LabVIEW 8.0*. Dort finden Sie Hinweise zu Verbesserungen, Änderungen und neuen Funktionen in LabVIEW 8.0. Generell wird empfohlen, dass alle Nutzer, die bislang mit einer LabVIEW-Version bis einschließlich 7.1 gearbeitet haben, zusätzlich zur vorliegenden Broschüre den Abschnitt *Änderungen und Neuerungen in LabVIEW 8.0* der *Hinweise zum Upgrade auf LabVIEW 8.0* zu Rate ziehen. Zum Downloaden der *Hinweise zum Upgrade auf LabVIEW 8.0* besuchen Sie bitte unsere Website [ni.com/info](http://ni.com/info) und geben den Infocode `rdg81v` ein.

In der *LabVIEW-Hilfe* finden Sie weitere Informationen zu neuen Funktionen in LabVIEW 8.2 und zu Methoden der LabVIEW-Programmierung, Schritt-für-Schritt-Anleitungen sowie Hinweise zu den LabVIEW-VIs, -Funktionen, -Paletten, -Menüs, -Werkzeugen, -Eigenschaften, -Methoden, -Ereignissen, -Dialogfeldern usw. In der *LabVIEW-Hilfe* wird auch auf weitere LabVIEW-Dokumentation von National Instruments verwiesen. Zum Öffnen der *LabVIEW-Hilfe* klicken Sie auf **Hilfe»LabVIEW-Hilfe durchsuchen**.

## Inhaltsverzeichnis

---

Upgrade auf LabVIEW 8.2 .....	2
Konvertieren von VIs.....	2
Upgrade von Toolkits, Gerätetreibern und Erweiterungspaketen (Add-Ons) .....	3
Upgrade zusätzlicher Software von National Instruments .....	4
Upgrade älterer LabVIEW-Versionen .....	4

Hinweise zur Kompatibilität beim Upgrade .....	5
Upgrade von LabVIEW 8.0.....	6
Upgrade von LabVIEW 7.x.....	13
Upgrade von LabVIEW 6.x.....	33
Upgrade von LabVIEW 5.x oder älteren Versionen .....	39
Änderungen und Neuerungen in LabVIEW 8.2 .....	39
LabVIEW-Dokumentation .....	39
Neue Beispiel-VIs .....	39
Verbesserter Startvorgang .....	40
Verbesserungen des Blockdiagramms.....	40
Verbesserungen am Frontpanel .....	41
Verbesserungen an der Umgebung.....	45
Verbesserungen an neuen und geänderten VIs, Funktionen und Knoten .....	51
Neue Eigenschaften, Methoden und Ereignisse .....	60
Verbesserungen an LabVIEW-MathScript.....	61
3D-Bildelement .....	64
Objektorientierte Programmierung in LabVIEW .....	65
Verbesserungen an LabVIEW-Projekten .....	67
Steuerung von VIs von mehreren Clients aus .....	72
Importieren von Funktionen aus einer DLL.....	73
Vorlagen für Gerätetreiber .....	73
Verbesserungen an .NET und ActiveX (Windows) .....	73
Neuerungen an der Beispielsuchmaschine .....	74
Verbesserungen an der Versionsverwaltung .....	74
TDM-Verbesserungen .....	76
Importieren von Webdiensten (Windows) .....	77
Änderungen an externen Code-Funktionen.....	77

## Upgrade auf LabVIEW 8.2

---

Wenn Sie Ihre bestehende LabVIEW-Version aktualisieren möchten, lesen Sie zunächst die Abschnitte *Upgrade auf LabVIEW 8.2* und *Upgrade von LabVIEW x.x* im Abschnitt *Hinweise zur Kompatibilität beim Upgrade* dieses Dokuments, wobei *x.x* für Ihre Version von LabVIEW steht.

### Konvertieren von VIs

Wenn Sie ein VI öffnen, das mit einer LabVIEW-Version ab 4.0 gespeichert wurde, wird es in LabVIEW 8.2 automatisch konvertiert und kompiliert. Um zu verhindern, dass das VI bei jedem Öffnen umgewandelt und neu kompiliert wird – was zusätzlichen Arbeitsspeicher kostet – sollte das VI am besten in LabVIEW 8.2 gespeichert werden.

VIs mit ungespeicherten Änderungen – dazu zählt auch eine Neukompilierung – können merklich langsamer sein als gespeicherte VIs. Weitere Informationen zu diesem Thema und zum Speichern von VIs finden Sie in der *LabVIEW-Hilfe*.



**Hinweis** In LabVIEW 8.2 gespeicherte VIs lassen sich nicht in älteren Versionen öffnen. Um VIs in LabVIEW 8.0 ausführen zu können, wählen Sie im Menü **Datei»Für vorige Version speichern**. Vor dem Speichern in LabVIEW 8.2 empfiehlt es sich, eine Sicherungskopie der VIs anzulegen, die für die Verwendung in einer älteren Version vorgesehen sind.

Wenn Ihr Rechner nicht genug Speicherplatz hat, um alle VIs auf einmal zu konvertieren, gehen Sie bei der Umwandlung etappenweise vor. Werfen Sie unter Umständen auch einen Blick auf die Hierarchie der VIs, die konvertiert werden sollen. Laden und speichern Sie zunächst die SubVIs der unteren Hierarchiestufen. Fahren Sie in der Hierarchie aufwärts fort. Das Haupt-VI sollte zum Schluss geöffnet und konvertiert werden. Um VIs verzeichnisweise umzuwandeln, wählen Sie **Werkzeuge»Fortgeschritten»Massenkompilierung**. VIs eines Verzeichnisses oder einer LLB werden in alphabetischer Reihenfolge kompiliert. Wird während des Umwandlungsprozesses zuerst ein Haupt-VI gefunden, wird für die Massenkompilierung in etwa genau so viel Arbeitsspeicher benötigt, als ob dieses VI zuerst geöffnet worden wäre.

Die Speicherauslastung können Sie sich unter **Hilfe»Über LabVIEW** anzeigen lassen.

## Upgrade von Toolkits, Gerätetreibern und Erweiterungspaketen (Add-Ons)

Installieren Sie nach Abschluss der Installation von LabVIEW 8.2 alle Toolkits und Erweiterungspakete für das Programm in das Programmverzeichnis von LabVIEW 8.2. Eventuell müssen Sie zunächst das Toolkit früherer LabVIEW-Versionen deinstallieren. Weitere Informationen zur Installation finden Sie in der Dokumentation des LabVIEW-Toolkits oder Add-Ons.



**Hinweis** Die Version der LabVIEW-Module muss mit der LabVIEW-Version übereinstimmen. So gilt zum Beispiel das LabVIEW Real-Time Modul 8.2 nur für LabVIEW 8.2.

Toolkits, Gerätetreiber und Zusatz-VIs aus älteren Versionen müssen für LabVIEW 8.2 ebenfalls massenkompiliert werden. Weitere Informationen zur Massenkompilierung von VIs finden Sie im Abschnitt [Konvertieren von VIs](#) in diesem Dokument.

Folgende Toolkits, Gerätetreiber und Erweiterungspakete müssen für LabVIEW 8.2 entweder aktualisiert oder heruntergeladen werden:

- Für den LabVIEW Application Builder ist ein separates Upgrade auf den LabVIEW Application Builder 8.2 erforderlich. Im LabVIEW 8.2 Professional Development System ist der Application Builder 8.2 bereits enthalten. Weitere Informationen zur Installation des Programms finden Sie in der *LabVIEW Application Builder Readme* im Verzeichnis `labview\readme` (**Windows**) oder `labview` (**Mac OS und Linux**).
- Für LabVIEW 8.x wird VI Analyzer 1.1 benötigt. Dieses Programm finden Sie zusammen mit Hinweisen zum Upgrade auf unserer Website [ni.com/info](http://ni.com/info) nach Eingabe des Infocodes `exd8yy`.
- Für die Verwendung des Internet Toolkits 6.0 mit LabVIEW 8.x müssen zusätzliche VIs heruntergeladen werden. Zum Herunterladen der VIs besuchen Sie bitte unsere Website [ni.com/info](http://ni.com/info) und geben Sie den Infocode `itkver6` ein.
- Der Treiber für das digitale Multimeter 34401A von HP (jetzt: Agilent) ist der Treibervorlage für Digitalmultimeter von National Instruments ähnlicher geworden. Dieser Treiber ist daher nicht mit dem Treiber HP34401A kompatibel, der in früheren LabVIEW-Versionen enthalten war. Wenn Sie den Treiber von LabVIEW 7.x benötigen, laden Sie ihn von der Website [ni.com/idnet](http://ni.com/idnet) herunter.

## Upgrade zusätzlicher Software von National Instruments

Für LabVIEW 8.x wird NI TestStand ab 3.5 benötigt. Dieses Programm finden Sie zusammen mit Hinweisen zum Upgrade auf der Website [ni.com/info](http://ni.com/info) nach Eingabe des Infocodes `exd8yy`.

## Upgrade älterer LabVIEW-Versionen

Ein Upgrade von LabVIEW hat keine Auswirkungen auf die vorhandene Version, da die neue Version in ein anderes Verzeichnis installiert wird. So lautet das Programmverzeichnis von LabVIEW-Versionen bis 5.x beispielsweise `labview`. Alle LabVIEW-Versionen ab 6.0 werden im Verzeichnis `labview x.x` gespeichert (`x.x` steht für die Version). Sie können LabVIEW 8.2 ohne vorherige Deinstallation älterer Versionen installieren.

## Ersetzen einer bestehenden LabVIEW-Version

Deinstallieren Sie die bestehende Version von LabVIEW, starten Sie das Installationsprogramm von LabVIEW 8.2 und wählen Sie das `labview`-Verzeichnis der letzten Version als Installationsverzeichnis.

**(Windows)** Die bisherige LabVIEW-Version kann auch in der Systemsteuerung unter "Software" deinstalliert und durch LabVIEW 8.2 ersetzt werden. Dateien, die im `labview`-Verzeichnis erstellt wurden, bleiben vom Deinstallationsprogramm unberührt.



**Hinweis** Bei der Deinstallation oder Neuinstallation von LabVIEW werden alle LLB-Dateien im Verzeichnis `vi.lib` einschließlich der darin befindlichen VIs und Elemente entfernt. Speichern Sie Ihre VIs und Elemente daher im Verzeichnis `user.lib`. So werden die VIs auch den Paletten **Funktionen** und **Elemente** hinzugefügt.

## Kopieren der Einstellungen einer älteren LabVIEW-Version

Um die Einstellungen zur LabVIEW-Oberfläche von der letzten Version zu übernehmen, kopieren Sie die Datei mit den LabVIEW-Voreinstellungen (`*.ini`) aus dem `labview`-Verzeichnis.



**Vorsicht!** Alle bis dato vorgenommenen Änderungen an den Grundeinstellungen werden damit in der neuen Version überschrieben.

Kopieren Sie die Datei mit den Einstellungen nach der Installation in das Programmverzeichnis von LabVIEW 8.2.

**(Windows)** Die Einstellungen sind in der Datei `labview.ini` gespeichert.

**(Mac OS)** Die Datei mit den Einstellungen lautet `LabVIEW Preferences` und befindet sich im Stammverzeichnis unter `Library:Preferences`.

**(Linux)** LabVIEW speichert die Einstellungen unter dem Namen `.labviewrc` im Stammverzeichnis.

## Kopieren der user.lib-Dateien der bisherigen Version

Um Dateien aus dem Ordner `user.lib` von der bisherigen Version zu übernehmen, kopieren Sie die Dateien aus dem `labview`-Verzeichnis der vorigen LabVIEW-Version. Kopieren Sie die Dateien nach der Installation von LabVIEW 8.2 in den Ordner `user.lib` des neuen LabVIEW-Verzeichnisses.

# Hinweise zur Kompatibilität beim Upgrade

---

In den nachfolgenden Abschnitten werden Probleme beim Upgrade bezüglich der Kompatibilität verschiedener LabVIEW-Versionen besprochen.

Weitere Informationen zu bekannten Problemen, anderen Kompatibilitätsproblemen und Informationen zu zuletzt hinzugefügten Funktionen in LabVIEW 8.2 finden Sie in der Datei `readme.html` im `labview`-Verzeichnis.

## Upgrade von LabVIEW 8.0

Beim Upgrade von LabVIEW 8.0 auf LabVIEW 8.2 können folgende Kompatibilitätsprobleme auftreten:

### Unterstützte Plattformen

Im Hinblick auf die Plattformkompatibilität gibt es bei LabVIEW 8.2 folgende Änderungen:

- LabVIEW 8.2 arbeitet nicht mit Windows XP x64.
- LabVIEW 8.2 arbeitet weder mit Mac OS X 10.3.8 noch mit Vorgängerversionen davon.
- LabVIEW 8.2 arbeitet mit Macintosh-Rechnern mit Intel-Prozessor. Weitere Informationen zur Unterstützung von Macintosh-Rechnern finden Sie auf der Website von National Instruments [ni.com/info](http://ni.com/info) nach Eingabe des Infocodes `macintel`.

### Systemvoraussetzungen

**(Mac OS)** Für die LabVIEW-Minimalversion sind mindestens 500 MB Festplattenspeicher und für die vollständige Installation 700 MB erforderlich.

**(Linux)** Für die Minimalversion sind mindestens 430 MB Festplattenspeicher und für die vollständige Installation 620 MB erforderlich.

### Gedruckte Dokumentation

Die nachfolgenden Dokumente wurden für LabVIEW 8.2 nicht geändert. Daher sind für LabVIEW 8.2 relevante Änderungen möglicherweise nicht darin enthalten.

- *LabVIEW-Schnellübersicht*
- *LabVIEW-Grundlagen*—Da die *LabVIEW-Grundlagen* Teil des Buchs **Grundlagen** in der *LabVIEW-Hilfe* sind, finden Sie aktuelle Informationen im Buch **Grundlagen** auf der Registerkarte **Inhalt** der *LabVIEW-Hilfe*.

### Änderungen im Verhalten von VIs und Funktionen

Das Verhalten der nachfolgenden VIs und Funktionen hat sich in LabVIEW 8.2 geändert.

## Kommunikation zwischen Applikationsinstanzen

In LabVIEW 8.2 können die Funktionen “Queue anfordern”, “Melder anfordern”, “Benutzerereignis erzeugen”, “Semaphor erstellen” und “Rendezvous erstellen” nicht mehr zwischen LabVIEW-Applikationsinstanzen kommunizieren. Wenn Sie eine Queue-, Melder-, Benutzerereignis-, Semaphor- oder Rendezvous-Referenz in einer Applikationsinstanz anfordern oder erstellen, kann diese Referenz nicht in einer anderen Applikationsinstanz verwendet werden.

## VI “Rücktransformation der Eigenvektoren”

An den Eingängen **Funktion**, **Unterer Index**, **Oberer Index** und **Skalierung** des VIs “Rücktransformation der Eigenvektoren” müssen Angaben gemacht werden, damit das VI funktioniert.

## Funktion “DataSocket: Schreiben”

In LabVIEW 8.0.1 hat sich das Standardverhalten der Funktion “DataSocket: Schreiben” in “asynchron” geändert. Wenn sowohl LabVIEW 8.0 als auch LabVIEW 8.2 installiert sind, gibt das Beispiel-VI “DataSocket API Client” im Verzeichnis `labview\examples\Shared Variable` bei Anhalten des VIs einen Fehler aus. Sie müssen daher LabVIEW 8.0 auf 8.0.1 upgraden, um das Beispiel in LabVIEW 8.2 ausführen zu können.

## Datei-I/O-VIs

Die VIs “In Tabellenkalkulationsdatei schreiben” und “Aus Tabellenkalkulationsdatei lesen” sind polymorph. Das VI “In Tabellenkalkulationsdatei schreiben” passt sich dem mit dem Eingang **Format** verbundenen Wert an. Das VI “Aus Tabellenkalkulationsdatei lesen” enthält folgende Instanzen: “DBL”, “I64” und “String”.

## Funktion “GPIB: Status”

In LabVIEW 8.0 wurde die Funktion “GPIB: Status” nicht ausgeführt, wenn **Fehler (Eingang)** einen Fehler empfangen hat. In LabVIEW 8.2 arbeitet die Funktion unabhängig von Eingangsfehlern.

## Histogramm-VI

Der Standardwert des Eingangs **Intervalle** des VIs “Histogramm” wurde auf 10 geändert.

## Funktion “VI-Referenz öffnen”

Die Voreinstellung für den Eingang **Optionen** der Funktion “VI-Referenz öffnen” ist jetzt, den Benutzer zum Suchen nach fehlenden SubVIs aufzufordern. Es gibt jedoch eine neue Einstellung,  $0 \times 20$ , bei der das Dialogfeld **Suchen** nicht angezeigt wird.

## VI “Nullstellen eines Polynoms”

Wenn  $P(x)$  ein konstanter Wert ungleich 0 ist, gibt das VI “Nullstellen eines Polynoms” keinen Fehler aus. Wenn  $P(x)$  aber gleich 0 ist, gibt das VI “Nullstellen eines Polynoms” den Fehler -20111 aus. Es können nicht alle Polynomkoeffizienten am Eingang dieses VIs Null sein.

## VI “Rampen-Muster”

Wenn der Eingang **Samples** im VI “Rampen-Muster” auf 1 eingestellt ist und **Ende ausschließen?** TRUE ist, wird ein Array mit einem **Start**-Element und kein Fehler ausgegeben. In LabVIEW 8.0 hat das VI unter diesen Bedingungen einen Fehler ausgegeben.

## VI “Registry: Wert lesen (einfach)”

In LabVIEW 8.0 traten bei der Formatierung des Strings `REG_MULTI_SZ`, den das VI für ein serialisiertes Array aus Strings benötigt, Fehler auf. Daher musste ein Parser erstellt werden, der diesen Datentyp für das VI “Registry: Wert lesen (einfach)” verarbeiten konnte. In LabVIEW 8.2 gibt das VI “Registry: Wert lesen (einfach)” diesen Datentyp im gleichen Format aus wie das VI “Registry: Wert schreiben (einfach)”. Sie müssen keinen eigenen Parser mehr hinzufügen. Wenn Sie Ihren eigenen Parser mit diesen VIs in LabVIEW 8.2 verwenden, gibt das VI “Registry: Wert lesen (einfach)” ungültige Daten aus.

## VI “Signalverläufe erneut abtasten (einmalig)”

Der Standardwert des Eingangs **Offenes Intervall?** des VIs “Signalverläufe erneut abtasten (einmalig)” wurde von TRUE auf FALSE geändert. Es wird nun ein geschlossenes Intervall ausgewählt. Wenn Sie bestehenden Programmcode nicht entsprechend anpassen, gibt das VI ggf. nicht das erwartete Ergebnis aus.

## Audio-VIs

In den VIs “Audioaufnahme lesen” und “Audioaufnahme lesen (einfach)” gibt die Komponente **t0** des Ausgangs **Daten** den Zeitstempel des ersten gelesenen Abtastwerts aus. LabVIEW schätzt die Anfangszeit, zu welcher der erste Abtastwert gelesen wurde.

Das “VI Audioausgabe beenden” muss nicht mehr aufgerufen werden, um die Audioausgabe in einem fortlaufenden Audiotask zu beenden.

Das VI “Audioausgabe abwarten” arbeitet nun in den Modi **Kontinuierlich** und **Endliche Anzahl**.

## Signalverlaufs-VIs

In LabVIEW 8.2 wurden die folgenden Signalverlaufs-VIs geändert:

- “Einfache Triggererkennung”—In beiden Instanzen dieses VIs wurde der Eingang **Steigung** in **Triggerflanke** geändert.
- “Signalverlaufsabschnitt lesen”—Enthält folgende Instanzen: “WDT: Signalverlaufsabschnitt lesen (DBL)”, “WDT: Signalverlaufsabschnitt lesen (CDB)”, “WDT: Signalverlaufsabschnitt lesen (EXT)”, “WDT: Signalverlaufsabschnitt lesen (I16)”, “WDT: Signalverlaufsabschnitt lesen (I32)”, “WDT: Signalverlaufsabschnitt lesen (I8)” und “WDT: Signalverlaufsabschnitt lesen (SGL)”. Am Eingang **Format von Start/Dauer** gibt es nicht mehr die Option **Absolute Zeit**. Der Eingang **Start** wurde in **Start-Samples/Zeit** und der Ausgang **Tatsächlicher Start** in **Tatsächlich(e) Start-Samples/Zeit** geändert.
- “Zeit-Array für Signalverlauf lesen”—Der Datentyp des Ausgangs **X-Array** wurde von numerischem Fließkommawert doppelter Genauigkeit in Zeitstempel geändert.
- Y-Wert abfragen—Dieses polymorphe VI und dessen Instanzen wurden in “XY-Wert abfragen” umbenannt. Das VI “XY-Wert abfragen” enthält jetzt den Ausgang **X-Wert**. Der Ausgang **Datenwert** wurde in **Y-Wert** geändert.
- “Anzahl der Signalverlaufswerte”—Dieses polymorphe VI enthält folgende Instanzen: “WDT: Anzahl der Signalverlaufswerte (DBL)”, “Anzahl der Signalverlaufswerte (CDB)”, “WDT: Anzahl der Signalverlaufswerte (EXT)”, “WDT: Anzahl der Signalverlaufswerte (I16)”, “WDT: Anzahl der Signalverlaufswerte (I32)”, “WDT: Anzahl der Signalverlaufswerte (I8)” und “WDT: Anzahl der Signalverlaufswerte (SGL)”.
- “Signalverlauf aus Datei lesen”—Gibt bei einem EOF-Fehler im Ausgang **Fehler (Ausgang)** den Fehlerstatus TRUE aus.
- “Abschnitt ersetzen”—Der Eingang **Start** wurde in **Start-Samples/Zeit** und der Ausgang **Tatsächlicher Startwert** in **Tatsächlich(e) Start-Samples/Zeit** geändert.
- “Nach Digitalmustern suchen”—Der Eingang **Start** wurde in **Start-Index/Zeit** geändert.
- “Signalverlauf suchen”—Der Datentyp der Ausgänge **Zeitpunkt der besten Näherung** und **Zeit der Näherung** wurde von numerischem Fließkommawert doppelter Genauigkeit in Zeitstempel geändert.

- “Signalverlauf - Minimum und Maximum”—Der Datentyp der Ausgänge **Minimum-Zeit** und **Maximum-Zeit** wurde von “numerischer Fließkommawert doppelter Genauigkeit” in “Zeitstempel” geändert.
- “Signalverlauf zu XY-Paaren”—Der Datentyp des **x**-Elements des Ausgangs **XY-Paare** wurde von numerischem Fließkommawert doppelter Genauigkeit in Zeitstempel geändert.

## Änderungen der Funktionsweise von Eigenschaften, Methoden und Ereignissen

Die Funktionsweise folgender Eigenschaften, Methoden und Ereignisse hat sich in LabVIEW 8.2 geändert:

- Die Voreinstellung für den Eingang **Optionen** der ActiveX-Methode “GetVIReference” ist jetzt, den Benutzer zum Suchen nach fehlenden SubVIs aufzufordern. Es gibt jedoch eine neue Einstellung, `0x20`, bei der das Dialogfeld **Suchen** nicht angezeigt wird.
- Die Methode “Element hinzufügen” der Klasse “ProjektElement” gibt einen Fehler aus, wenn Sie versuchen, eine Umgebungsvariable zu einer Bibliothek hinzuzufügen, die nicht in einem Projekt geöffnet ist.
- Wenn der Eingang **Ref automatisch schließen** der Methode “VI ausführen” TRUE ist und die Methode einen Fehler ausgibt, wird die Referenz nicht geschlossen.
- Die LabVIEW-Umgebung kann auf vereinfachtes Chinesisch eingestellt werden. Setzen Sie dazu die Eigenschaft “Applikation:Sprache” auf `zh-cn`.
- .NET-Methoden, die Array-Datentypen anhand von Refnums verarbeiten, leiten in LabVIEW 8.0 alle Daten als Refnum-Datentyp weiter. Bei LabVIEW 8.2 geben die .NET-Methoden die Daten im tatsächlichen Datentyp weiter.
- Die Eigenschaft “Position bearbeiten” der Klassen “Binärtabelle”, “mehrspaltigesListefeld”, “Tabelle” und “Baumstruktur” gibt die Werte `(-2, -2)` aus. Das bedeutet, dass der Benutzer keine Änderungen am Text des Elements vornehmen kann. Die Eigenschaft “Zeile bearbeiten” der Klasse “Listefeld” gibt den Wert `-2` aus. Das bedeutet ebenfalls, dass der Benutzer keine Änderungen am Text des Elements vornehmen kann.
- In LabVIEW 8.0 gilt die Eigenschaft “Panel-Aktualisierungen verzögern” nicht für Frontpanels in einem Unterpanel. In LabVIEW 8.2 bezieht sich die Eigenschaft “Panel-Aktualisierungen verzögern” auch auf Unterpanel.
- Die Ereignisse “Applikationsinstanz beenden” und “Applikationsinstanz beenden?” wurden durch “Applikationsinstanz schließen” und “Applikationsinstanz schließen?” ersetzt. Wenn Sie das Ereignis

“Applikationsinstanz schließen” in einem VI verwenden, das außerhalb eines LabVIEW-Projekts ausgeführt wird, erzeugt LabVIEW ein Ereignis, wenn Sie das Programm über die Benutzeroberfläche oder programmatisch beenden. LabVIEW erzeugt das Ereignis “Applikationsinstanz schließen?”, wenn Sie das Programm über die Benutzeroberfläche schließen. Wenn Sie die Ereignisse “Applikationsinstanz schließen” und “Applikationsinstanz schließen?” für ein VI in einem LabVIEW-Projekt registrieren, erzeugt LabVIEW die Ereignisse, wenn die Applikationsinstanz geschlossen oder LabVIEW beendet wird.

## Veraltete Eigenschaften, Methoden und Ereignisse

Die folgenden Eigenschaften, Methoden und Ereignisse werden von LabVIEW 8.2 nicht mehr unterstützt:

- Die Anschlussfeld-Eigenschaft.
- Die Datentyp-Eigenschaft in der Klasse “Variable”. Verwenden Sie stattdessen die Eigenschaft “Datentyp (Variant)” in der Klasse “Variable”.

## Unbenannte Eigenschaften, Methoden und Ereignisse

Folgende Eigenschaften, Methoden und Ereignisse wurden in LabVIEW 8.2 umbenannt:

Klasse	Name in LabVIEW 8.0	Name in LabVIEW 8.2	Typ
Applikation	Verbindung mit Slave beenden	LVRT:Verbindung mit Slave beenden	Methode
Applikation	Applikation beenden	Applikationsinstanz schließen	Ereignis
Applikation	Applikation beenden?	Applikationsinstanz schließen?	Ereignis
Intensitätsgraph, Mischsignalgraph und Signalverlaufsgraph	Cursor-Palette sichtbar	Cursor-Legende sichtbar	Eigenschaft
Bibliothek	Bibliotheks-Tag löschen	Bibliotheks-Tag:Löschen	Methode
Bibliothek	Symbol ermitteln	Symbol:Ermitteln	Methode
Bibliothek	Bibliotheks-Tag ermitteln	Bibliotheks-Tag:Ermitteln	Methode

<b>Klasse</b>	<b>Name in LabVIEW 8.0</b>	<b>Name in LabVIEW 8.2</b>	<b>Typ</b>
Bibliothek	Bibliotheks-Tagnamen ermitteln	Bibliotheks-Tag:Namen ermitteln	Methode
Bibliothek	Sperrstatus ermitteln	Sperrstatus:Lesen	Methode
Bibliothek	Quellenbereich ermitteln	Quellenbereich:Ermitteln	Methode
Bibliothek	Speichern	Speichern:Bibliothek	Methode
Bibliothek	Kopie speichern	Speichern:Kopie	Methode
Bibliothek	Symbol festlegen	Symbol:Setzen	Methode
Bibliothek	Bibliotheks-Tag festlegen	Bibliotheks-Tag:Setzen	Methode
Bibliothek	Sperrstatus setzen	Sperrstatus:Setzen	Methode
Bibliothek	Quellenbereich festlegen	Quellenbereich:Setzen	Methode
Listenfeld, mehrspaltiges Listenfeld und Baumstruktur	Ziehen/Ablegen:Verschieben von Objekten möglich	Ziehen/Ablegen:Ziehen zulassen	Eigenschaft
Pfad und String	Ablegen möglich	Ablegen möglich	Eigenschaft
Projektelement	Tag löschen	Tag:Löschen	Eigenschaft
Projektelement	Tag lesen	Tag:Tag ermitteln	Eigenschaft
Projektelement	Tagnamen ermitteln	Tag:Namen ermitteln	Eigenschaft
Projektelement	XML-Tag lesen	Tag:XML-Tag ermitteln	Eigenschaft
Projektelement	Tag festlegen	Tag:Tag setzen	Eigenschaft
Projektelement	XML-Tag festlegen	Tag:XML-Tag festlegen	Eigenschaft
Projektelement	Bibliothekselement-Typ String	Bibliothekselement-Typ:String	Eigenschaft
Projektelement	Bibliothekselement-Typ	Bibliothekselement:Typ	Eigenschaft

# Upgrade von LabVIEW 7.x

Bei einem Upgrade von LabVIEW 7.x auf LabVIEW 8.2 können die nachfolgend aufgeführten Kompatibilitätsprobleme auftreten. Sonstige potentielle Probleme beim Upgrade von LabVIEW sind im Abschnitt [Upgrade von LabVIEW 8.0](#) beschrieben.

Alle Unterschiede, die es zwischen den einzelnen Versionen ab 7.x gibt, sind in den *Hinweisen zum Upgrade von LabVIEW* jeder Version beschrieben, die Sie unter [ni.com/manuals](http://ni.com/manuals) finden.



**Hinweis** Die *LabVIEW-Schnellübersicht* und die *LabVIEW-Grundlagen* haben sich bei LabVIEW 8.2 nicht geändert. Die PDF-Versionen dieser Dokumente finden Sie im Verzeichnis `labview\manuals`. Weitere Informationen zu diesen Dokumenten finden Sie im Abschnitt [Upgrade von LabVIEW 8.0](#) dieses Dokuments.

## Unterstützte Plattformen

Im Hinblick auf die Plattformkompatibilität gibt es bei LabVIEW 8.x folgende Änderungen:

- LabVIEW-Versionen ab 7.1 laufen nicht unter Windows Me/98/95. LabVIEW 8.x läuft nicht unter Windows NT.
- LabVIEW 8.x läuft weder unter Mac OS X 10.2 noch unter Vorgängerversionen davon.
- LabVIEW 8.x läuft nicht unter Sun Solaris.

## Systemvoraussetzungen

Für LabVIEW 7.x wird ein Arbeitsspeicher von mindestens 128 MB benötigt. Empfohlen werden jedoch 256 MB. Für LabVIEW 8.x wird ein Arbeitsspeicher von mindestens 256 MB benötigt. Empfohlen werden jedoch 512 MB.

Für LabVIEW 7.x ist eine Bildschirmauflösung von 800 × 600 Pixeln erforderlich. Empfohlen werden jedoch 1.024 × 768 Pixel. Für LabVIEW 8.x ist eine Bildschirmauflösung von 1.024 × 768 Pixeln erforderlich.

## Windows

LabVIEW 7.x erfordert mindestens einen Pentium III, einen Celeron-Prozessor mit 600 MHz oder einen gleichwertigen Prozessor. Es wird jedoch ein Pentium 4 oder ein gleichwertiger Prozessor empfohlen. Für LabVIEW 8.x ist ein Pentium III bzw. ein Celeron-Prozessor mit 866 MHz oder ein gleichwertiges Produkt Mindestanforderung. Es wird jedoch ein Pentium 4/M oder ein gleichwertiger Prozessor empfohlen.

LabVIEW 7.x benötigt für die LabVIEW-Minimalversion einen Festplattenspeicherplatz von mindestens 130 MB und für die vollständige Installation 550 MB. LabVIEW 8.x benötigt 1,2 GB Festplattenspeicherplatz für eine vollständige Installation.

## Mac OS

LabVIEW 7.x benötigt für die LabVIEW-Minimalversion einen Festplattenspeicherplatz von mindestens 280 MB und für die vollständige Installation 350 MB. LabVIEW 8.2 benötigt für die Minimalversion einen Festplattenspeicherplatz von mindestens 500 MB und für die vollständige Installation 700 MB.

## Linux

LabVIEW 7.x erfordert mindestens einen Pentium III, einen Celeron-Prozessor mit 600 MHz oder einen gleichwertigen Prozessor. Es wird jedoch ein Pentium 4 oder ein gleichwertiger Prozessor empfohlen. Für LabVIEW 8.x ist ein Pentium III bzw. ein Celeron-Prozessor mit 866 MHz oder ein gleichwertiges Produkt Mindestanforderung. Es wird jedoch ein Pentium 4/M oder ein gleichwertiger Prozessor empfohlen.

LabVIEW 7.x benötigt für die Minimalversion einen Festplattenspeicherplatz von mindestens 200 MB und für die vollständige Installation 300 MB. LabVIEW 8.2 benötigt für die Minimalversion einen Festplattenspeicherplatz von mindestens 430 MB und für die vollständige Installation 620 MB.

Für LabVIEW 7.x wird die GNU C Library (`glibc`) ab Version 2.1.3 benötigt. Empfohlen wird eine Version ab 2.2.4. Für LabVIEW 8.x ist die GNU C Library ab Version 2.2.4 erforderlich.

LabVIEW 7.x läuft nur unter Red Hat Linux ab 7.0, Mandrake Linux ab 8.0, SuSE Linux ab 7.1 oder Debian Linux ab 3.0. LabVIEW 8.x arbeitet nur mit Red Hat Enterprise Linux WS ab 3, MandrakeLinux/Mandriva ab 10.0 oder SuSE Linux ab 9.1.

## Benutzerdefinierte Palettenansichten

In LabVIEW 8.x gibt es keine benutzerdefinierten Palettenansichten mehr. Paletten können jetzt ohne Anlegen einer benutzerdefinierten Palettenansicht bearbeitet werden. Weitere Informationen zu Änderungen der Paletten in LabVIEW 8.0 finden Sie auf der Website von National Instruments [ni.com/info](http://ni.com/info) nach Eingabe des Infocodes `lv8palette`.

# Änderungen im Verhalten von VIs und Funktionen

Das Verhalten der nachfolgenden VIs und Funktionen hat sich in LabVIEW 7.1 oder 8.0 geändert.

## .NET-VIs und -Applikationen

Für die .NET-Funktionen und -Anwendungen in LabVIEW 8.x wird mindestens das .NET Framework 1.1 Service Pack 1 benötigt. Vor der Installation des .NET Framework 1.1 Service Pack 1 muss Microsoft .NET Framework 1.1 Hotfix KB886904 entfernt werden.

Wenn Sie ein .NET-VI laden, das zuletzt in LabVIEW 7.x gespeichert wurde, werden Sie möglicherweise in LabVIEW 8.x dazu aufgefordert, die Assemblys, auf welche das VI verweist, zu suchen, selbst dann, wenn sich die Assembly-Dateien im selben Verzeichnis wie das VI befinden oder wenn es im Dialogfeld **Werkzeuge»Fortgeschritten» .NET-Assembly-Referenzen** in LabVIEW 7.x registriert wurde.

## Algorithmen der Analyse-VIs

In LabVIEW-Versionen ab 7.1 arbeiten Analyse-VIs mit dem BLAS/LAPACK-Algorithmus. Dadurch liefern diese VIs nun genauere Ergebnisse. In LabVIEW 8.x befinden sich diese VIs auf den Paletten **Mathematik** und **Signalverarbeitung**.

## Express-VI “Signale anfügen”

Wenn in LabVIEW 7.x beim Express-VI “Signale anfügen” der Eingang **Eingangssignal A** ein leeres Array ist oder offen gelassen wird und Sie den Eingang **Eingangssignal B** mit einem einzelnen oder kombinierten Signal verbinden, ist auch der Ausgang **Angefügte Signale** leer. Wenn es in LabVIEW 8.x kein **Eingangssignal A** gibt oder dieser Eingang offen gelassen wird und Sie an **Eingangssignal B** ein einzelnes Signal anlegen, gibt das Express-VI **Eingangssignal B** aus. Wenn **Eingangssignal B** ein kombiniertes Signal enthält, werden alle Signale, aus denen es sich zusammensetzt, aneinander angehängt.

## Vergleichsfunktionen

In LabVIEW 7.x und Vorgängerversionen werden Variantdaten von den Vergleichsfunktionen zuerst in Bezug auf die Länge und anschließend bitweise verglichen. In LabVIEW 8.x beginnt der Vergleich der Variantdaten mit den Typencodes, in denen die tatsächliche Typinformation der Variantdaten kodiert ist, und wird dann bei den anderen typspezifischen Attributen fortgeführt.

## VI “Skalarprodukt”

In LabVIEW 7.0 wird das Skalarprodukt der Eingangsvektoren  $X$  und  $Y$  beim VI “Skalarprodukt” mit Hilfe folgender Gleichung berechnet:

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

In LabVIEW-Versionen ab 7.1 wird das Skalarprodukt komplexer Zahlen dagegen mit folgender Gleichung berechnet:

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

wobei  $y_i^*$  die komplex Konjugierte von  $y_i$  ist.

## VI “Einfacher Textreport” (Mac OS und Linux)

Das Anschlussfeld des VIs “Einfacher Textreport” hat sich geändert. Wenn Sie in LabVIEW 8.x ein VI öffnen, in dem das VI “Einfacher Textreport” verwendet wird und das zuvor in LabVIEW 7.x oder einer älteren Version gespeichert wurde, müssen Sie das SubVI mit der rechten Maustaste anklicken und aus dem Kontextmenü die Option **Mit SubVI neu verbinden** auswählen.

## Funktion “In String formatieren”

Wenn Sie in LabVIEW 7.x die %o- und %x-Syntaxelemente für Formatbezeichner mit der Funktion “In String formatieren” verwenden, werden die Werte des Fließkommazahl-Eingangs auf 32-Bit-Integer gerundet, bevor die Eingänge in einen String konvertiert werden.

In LabVIEW 8.x rundet diese Funktion aufgrund der Syntaxelemente für Formatbezeichner Fließkomma-Eingänge auf 64-Bit-Integer, bevor die Eingangswerte in Strings konvertiert werden.

## Funktion “Zahlen verbinden”

In LabVIEW 7.x und Vorgängerversionen konvertiert die Funktion “Zahlen verbinden” 32-Bit-Integer-Eingänge in 16-Bit-Integer, um einen 32-Bit-Integer zu erstellen. In LabVIEW 8.x wird mit der Funktion “Zahlen verbinden” aus 32-Bit-Integer-Eingängen ein 64-Bit-Integer erzeugt.



**Hinweis** Wenn Sie ein in LabVIEW 7.x erstelltes VI in LabVIEW 8.x öffnen, wandelt LabVIEW 32-Bit-Integer-Eingänge in 16-Bit-Integer um.

## Mathematische VIs und Matrizen

In LabVIEW 8.x arbeiten die **Mathematik**-VIs auch mit dem Datentyp "Matrix". Wenn Sie ein VI von LabVIEW 7.x in LabVIEW 8.x laden und das VI ein Mathematik-VI enthält, das mit einer matrixfähigen Funktion verbunden ist, wird ein rotes 7.x-Symbol angezeigt. Das bedeutet, dass die Funktion wie in LabVIEW 7.x arbeitet.

## Funktionen für die Konvertierung von Zahlen in Strings

In LabVIEW 7.x werden Fließkommazahlen bei den Funktionen "Zahl nach Hexadezimal-String", "Zahl nach Oktal-String" und "Zahl in String (Dezimaldarstellung)" vor der Umwandlung in einen String auf einen 32-Bit-Integer gerundet.

In LabVIEW 8.x runden diese Funktionen Fließkommazahlen auf 64-Bit-Integer, bevor die Eingangswerte in Strings konvertiert werden. Wenn Sie jedoch ein in LabVIEW 7.x erstelltes VI in LabVIEW 8.x öffnen, rundet LabVIEW aus Gründen der Kompatibilität und Funktion die Fließkommazahlen auf 32-Bit-Integer.

## Funktion "VI-Referenz öffnen"

Wenn in LabVIEW 7.x der Eingang **VI-Pfad** der Funktion "VI-Referenz öffnen" ein Pfad ist und sich im Speicher ein VI mit dem gleichen Namen befindet, gibt LabVIEW eine Referenz auf das VI im Speicher aus, selbst wenn der Pfad zu dem VI im Speicher nicht dem angegebenen Pfad entspricht.

Ist der Eingang **VI-Pfad** von "VI-Referenz öffnen" in LabVIEW 8.x ein String, wird ein VI nur geöffnet, wenn **VI-Pfad** mit dem vollständigen Namen des VIs im Speicher des Zielsystems übereinstimmt. Wenn **VI-Pfad** ein Pfad ist, sucht LabVIEW nach einem VI im Speicher mit demselben Zielpfad. Wenn kein VI mit passendem Pfad gefunden wird, versucht LabVIEW, das VI unter dem angegebenen Pfad zu laden. Bei Fehlen des VIs oder bei Konflikten mit anderen VIs im Speicher und Zielsystemen wird eine Fehlermeldung ausgegeben.

## VI "Schnellskalierung"

Wenn in LabVIEW 7.1 und Vorgängerversionen der Eingang **X** des VIs "Schnellskalierung 1D" bzw. "Schnellskalierung 2D" ein Array aus Nullen ist, gibt das VI **max|X|** als 0 und  $Y[i]=X[i]/\text{Max}|X|$  bzw.  $Y_{ij}=X_{ij}/\text{Max}|X|$  als Array aus NaN aus. Wenn in LabVIEW 8.x der Eingang **X** des VIs "Schnellskalierung" ein Array aus Nullen ist, gibt das VI unter **max|X|** den Wert 0 und  $Y[i]=X[i]/\text{Max}|X|$  bzw.  $Y_{ij}=X_{ij}/\text{Max}|X|$  als Array aus Nullen aus.

## VI “Schlüssel lesen”

In LabVIEW 7.x und Vorgängerversionen können mit Hilfe des VIs “Schlüssel lesen” japanische Multibyte-Zeichen gelesen werden, die in Shift-JIS kodiert sind. Dazu muss der Eingang **Multibyte-Kodierung** mit 1 oder <Shift-JIS> verbunden werden. In LabVIEW 8.x liest das VI “Schlüssel lesen” kodierte Multibyte-Zeichen automatisch. Dazu muss nur die Ländereinstellung des Betriebssystems auf die entsprechende Kodierung gesetzt werden.

## VI “Skalieren”

Wenn in LabVIEW 7.1 und Vorgängerversionen der Eingang **X** des VIs “1D skalieren” bzw. “2D skalieren” ein Array aus Nullen ist, gibt das VI unter **Skalierung** den Wert **0**, unter **Offset** den Wert **0** und unter  **$Y=(X-Offset)/Skalierung$**  ein Array aus NaN aus. Wenn in LabVIEW 8.x und Vorgängerversionen der Eingang **X** des VIs “Skalieren” ein Array aus Nullen ist, gibt das VI unter **Skalierung** den Wert **1**, unter **Offset** den Wert **0** und unter  **$Y=(X-offset)/scale$**  ein Array aus Nullen aus.

## Semaphor-VIs

In LabVIEW 7.x wird nicht versucht, die VIs “Semaphor freigeben” und “Semaphor belegen” auszuführen, wenn der Eingang **Fehler (Eingang)** einen Fehler empfängt. In LabVIEW 8.x wird auch beim Vorliegen eines Fehlers versucht, die VIs auszuführen. Wenn Sie allerdings ein in LabVIEW 7.x erstelltes VI in LabVIEW 8.x öffnen, wird der Funktionsumfang von LabVIEW 7.x beibehalten.

## SMTP-E-Mail-VIs

In LabVIEW 7.x und Vorgängerversionen können Sie einen Zeichensatz festlegen, indem Sie einen Wert mit dem Eingang **Zeichensatz** eines SMTP-E-Mail-VIs verbinden. In LabVIEW 8.x wird bei den SMTP-E-Mail-VIs davon ausgegangen, dass die Nachricht im Zeichensatz des Systems vorliegt. Mit diesen VIs wird die Nachricht vor dem Versenden im UTF-8-Format kodiert. Die Eingänge **Zeichensatz** und **Zeichenkonvertierung** gibt es bei den neuen SMTP-E-Mail-VIs nicht mehr.

## VI “Komplexe Zahlen sortieren”

Wenn Sie in LabVIEW 7.x und Vorgängerversionen den Eingang **Methode** des VIs “Komplexe Zahlen sortieren” auf **Betrag** stellen, wird die Reihenfolge der Elemente mit gleichem Betrag nicht geändert. Wenn Sie **Methode** in LabVIEW 8.x auf **Betrag** setzen, werden die Elemente mit gleichem Betrag zuerst nach ihrem Real- und dann nach ihrem Imaginärteil sortiert.

## VI “Einheitsvektor”

In LabVIEW 7.x und Vorgängerversionen wird die Norm des Eingangsvektors beim VI “Einheitsvektor” anhand der folgenden Gleichung berechnet:

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

In LabVIEW 8.x wird die Norm des Eingangsvektors beim VI “Einheitsvektor” anhand der folgenden Gleichung berechnet:

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

wobei  $X$  der Vektor (Eingang),  $\|X\|$  die Norm und  $y$  der Normtyp ist.

## Benutzer-VIs

VIs, die Sie in den Verzeichnissen `labview\help`, `labview\project` und `labview\wizard` ablegen, werden jeweils in den Menüs **Hilfe**, **Werkzeuge** und **Datei** angezeigt. VIs, die Sie in LabVIEW 7.x und Vorgängerversionen in diesen Verzeichnissen ablegen, funktionieren möglicherweise in LabVIEW 8.x nicht wie erwartet, da diese VIs in LabVIEW ab Version 8.0 in einer privaten Kopie der Anwendung geöffnet werden.

Eine Liste aller Benutzer-VIs im Arbeitsspeicher aller Applikationsinstanzen erhalten Sie mit Hilfe des VIs “VIMemory Get VIs in Memory” in der Bibliothek `labview\vi.lib\Utility\allVIsInMemory.llb`. Eine Referenz für die aktuelle Applikationsinstanz wird mit dem VI “Get User Application Reference” in der Bibliothek `labview\vi.lib\Utility\allVIsInMemory.llb` erzeugt. Weitere Informationen zu Applikationsinstanzen finden Sie in der *LabVIEW-Hilfe*.

## Veraltete VIs und Funktionen

Folgende VIs und Funktionen werden von LabVIEW 8.x nicht unterstützt:

- **(Mac OS)** Die PPC-VIs werden bei LabVIEW-Versionen ab 7.1 nicht mehr installiert. Stattdessen gibt es jetzt die TCP-VIs.
- Das VI “QR-Faktorisierung” wird von LabVIEW 8.x nicht mehr unterstützt. Verwenden Sie stattdessen das VI “QR-Zerlegung”.
- Die VIs “Levenberg Marquardt” und “Nichtlineare Levenberg-Marquardt-Anpassung” wurden in LabVIEW 8.x durch das VI “Nichtlineare Kurvenanpassung” ersetzt.

- In LabVIEW 8.x ist die Funktion “VISA: Statusinformation” nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie anstelle dieser Funktion das VI “Einfacher Fehlerbehandler” oder “Allgemeiner Fehlerbehandler”.
- Die VIs “Chi-Quadrat-Verteilung”, “F-Verteilung”, “Normalverteilung” und “T-Verteilung” wurden in LabVIEW 8.x durch die Instanzen “Chi-Quadrat”, “F”, “Normal” und “Student-t” des VIs “Stetige Verteilung” ersetzt.
- Die VIs “Inverse Chi-Quadrat-Verteilung”, “Inverse F-Verteilung”, “Inverse Normalverteilung” und “Inverse T-Verteilung” wurden in LabVIEW 8.x durch die Instanzen “Chi-Quadrat”, “F”, “Normal” und “Student-t” des VIs “Inverse stetige Verteilung” ersetzt.
- Die VIs “1D-Linearentwicklung” und “2D-Linearentwicklung” sind in LabVIEW 8.x nicht mehr auf der Palette **Funktionen** enthalten. Verwenden Sie stattdessen das VI “Linearentwicklung”.
- Die VIs “1D-Polynomentwicklung” und “2D-Polynomentwicklung” sind in LabVIEW 8.x nicht mehr auf der Palette **Funktionen** enthalten. Verwenden Sie stattdessen das VI “Polynomentwicklung”.
- Die VIs “1D: Kartesisch in polar” und “1D: Polar nach kartesisch” befinden sich in LabVIEW 8.x nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Funktionen “Re/Im nach Polar” bzw. “Polar nach Re/Im”.
- In LabVIEW 8.x befindet sich das VI “Frequenzanalyse” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen zur Messung von **Klirrfaktor** und **Oberwellenamplituden** das VI “Klirrfaktoranalyse” bzw. zum Messen der Ausgänge **SINAD** oder **Klirrfaktor plus Rauschen** das VI “SINAD-Analyse”.
- In LabVIEW 8.x befindet sich das VI “Netzwerkfunktionen (avg)” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die VIs “Übertragungsfunktion (Betrag-Phase)”, “Frequenzgangfunktion (Real-Im)”, “Kreuzspektrum (Betrag-Phase)” oder “Kreuzspektrum (Real-Im)”.
- In LabVIEW 8.x befindet sich das VI “Impulsparameter” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen zum Messen von **Anstiegszeit**, **Dauer**, **Überschwingen** oder **Vorschwingen** das VI “Signalvermessung”, zum Messen der Ausgänge **Periode**, **Impulsdauer** oder **Tastverhältnis** das VI “Impulsmessungen” und zum Messen von **Amplitude**, **High-Pegel** oder **Low-Pegel** das VI “Amplitude und Pegel”.
- In LabVIEW 8.x ist das VI “Übertragungsfunktion” nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie stattdessen die VIs “Übertragungsfunktion (Betrag-Phase)” bzw. “Übertragungsfunktion (Real-Im)”.

- In LabVIEW 8.x befindet sich das Express-VI “DIAdem-Report-Assistent” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen das Express-VI “DIAdem-Report”.
- In LabVIEW 8.x befinden sich die Konstanten “VISA-Ressourcenname” und “Logischer IVI-Name” nicht mehr auf der Palette **Funktionen**. Der VISA-Ressourcenname wird nun mit Hilfe des Eingangs **VISA-Ressourcenname** der VISA-VIs und der logische IVI-Name mit dem entsprechenden Eingang des Treiber-VIs festgelegt, mit dem das Gerät initialisiert wird.
- In LabVIEW 8.x ist die Fehler-Ringkonstante nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie stattdessen zur Eingabe des gewünschten Fehlercodes eine Konstante für einen vorzeichenlosen 32-Bit-Integer.
- **(Windows und Linux)** In LabVIEW 8.x befinden sich die Audio-VIs, die es in LabVIEW 7.x auf der Palette **Audio** gab, nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Audio-VIs von LabVIEW 8.x. Die mit LabVIEW 7.x gelieferten Beispiele gibt es in LabVIEW 8.x nicht mehr.
- **(Mac OS)** LabVIEW 8.x wird weiterhin mit den Audio-VIs von LabVIEW 7.1 geliefert. Die mit LabVIEW 7.x gelieferten Beispiele gibt es in LabVIEW 8.x nicht mehr.

## Datei-I/O-VIs und -Funktionen

In LabVIEW 8.x befindet sich das VI “Zeichen aus Datei lesen” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Funktion “Aus Textdatei lesen”.

In LabVIEW 8.x befindet sich das VI “Öffnen/Erstellen/Ersetzen einer Datei” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Funktion “Öffnen/Erstellen/Ersetzen einer Datei”. Die folgenden Funktionen enthalten einen Teil des Funktionsumfangs, den das VI “Öffnen/Erstellen/Ersetzen einer Datei” in LabVIEW 7.x und Vorgängerversionen hatte:

- Die Größe einer Datei lässt sich mit der Funktion “Dateigröße ermitteln” feststellen.
- Der Startpfad, das Dateimuster und der Standardname einer Datei oder eines Verzeichnisses für ein Dateidialogfeld werden im Express-VI “Dateidialog” festgelegt.
- Eine Refnum lässt sich mit Hilfe der Funktion “Refnum nach Pfad” in einen Pfad umwandeln.
- Mit der Funktion “In Binärdatei schreiben” können Sie plattformunabhängige Textdateien oder andere Arten von Binärdateien erstellen und

mit der Funktion “Aus Binärdatei lesen” können Sie die resultierende Binärdatei auslesen.

In LabVIEW 8.x befinden sich die Funktionen “Datei lesen” und “Datei schreiben” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Funktionen “Aus Binärdatei lesen” und “In Binärdatei schreiben”.

In LabVIEW 8.x befindet sich das VI “Zeichen in Datei schreiben” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Funktion “In Textdatei schreiben”.

In LabVIEW 8.x ist die Funktion “Zugriffsrechte” nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie stattdessen die Funktionen “Berechtigungen ermitteln” und “Berechtigungen festlegen”.

In LabVIEW 8.x befindet sich die Funktion “EOF” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Funktionen “Dateigröße ermitteln” und “Dateigröße festlegen”.

In LabVIEW 8.x ist die Funktion “Verzeichnis auflisten” nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie stattdessen die Funktion “Ordner anzeigen”.

In LabVIEW 8.x ist die Funktion “Bereich sperren” nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie stattdessen die Funktion “Zugriff verweigern”.

Wenn Sie ein in LabVIEW 7.x erstelltes VI öffnen und auf dem Blockdiagramm die Funktion “Neues Verzeichnis” enthalten ist, wird diese in LabVIEW 8.x durch die Funktion “Ordner erstellen” ersetzt. Wenn der von Ihnen im Eingang **Pfad** festgelegte Ordner nicht existiert, erstellt die Funktion “Ordner erstellen” das Verzeichnis. Die Funktion gibt keine Fehlermeldung aus, wie es bei der Funktion “Neues Verzeichnis” der Fall war.

In LabVIEW 8.x befindet sich die Funktion “Suchen” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die Funktionen “Dateiposition ermitteln” und “Dateiposition festlegen”.

In LabVIEW 8.x ist die Funktion “Datei- und Programmtyp” nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie stattdessen die Funktionen “Datei- und Programmtypen ermitteln” bzw. “Datei- und Programmtypen festlegen”.

In LabVIEW 8.x ist die Funktion “Datenträger-Info” nicht mehr auf der Palette **Funktionen** zu finden. Verwenden Sie stattdessen die Funktion “Datenträger-Info lesen”.

In LabVIEW 8.x befinden sich die Funktionen “Datei öffnen” und “Neue Datei” nicht mehr auf der Palette **Funktionen**. In LabVIEW 8.0 befindet sich das VI “Zeichen aus Datei lesen” ebenfalls nicht mehr auf der Palette **Funktionen**, wird jedoch aus Gründen der Kompatibilität mitgeliefert.

In LabVIEW 8.x befinden sich die VIs “Aus I16-Datei lesen”, “Aus Sgl-Datei lesen”, “In I16-Datei schreiben” und “In Sgl-Datei schreiben” nicht mehr auf der Palette **Funktionen**. Verwenden Sie stattdessen die VIs “Aus Binärdatei lesen” und “In Binärdatei schreiben”.

## **Änderungen der Funktionsweise von Eigenschaften, Methoden und Ereignissen**

Die Funktionsweise folgender Eigenschaften, Methoden und Ereignisse hat sich in LabVIEW 7.1 oder 8.0 geändert.

### **Eigenschaften und Methoden von Applikationen**

In LabVIEW 8.x hängt das Verhalten einiger Applikationseigenschaften und -methoden von der Kopie der Anwendung (Applikationsinstanz) ab, zu der sie gehören. So richtet sich zum Beispiel die Funktionsweise der Eigenschaft “Applikation:Alle VIs im Speicher” nach der Applikationsinstanz, in der sie sich befindet. Diese Eigenschaft gibt eine Liste mit allen VIs aus, die sich im Speicher derselben Applikationsinstanz befinden wie die Eigenschaft. Dagegen ist die Funktionsweise der Eigenschaft “Applikation:Verzeichnispfad” von der Applikationsinstanz unabhängig. Diese Eigenschaft gibt einen absoluten Pfad zu dem Verzeichnis aus, in dem sich die Applikation befindet. Diese Angabe ist für alle Applikationsinstanzen gleich.

Weitere Informationen zu Applikationsinstanzen finden Sie in der *LabVIEW-Hilfe*.

### **Methode “Frontpanel:Öffnen”**

In LabVIEW 7.1 wurde die Methode “FP öffnen” aus LabVIEW 7.0 “FP öffnen - Alt” genannt. LabVIEW 7.1 enthält eine andere Methode “FP öffnen”, die keine Fehlermeldung mehr ausgibt, wenn das Frontpanel bereits geöffnet ist. Die Methode “FP öffnen” aus LabVIEW 7.1 wurde in LabVIEW 8.x in “Frontpanel:Öffnen” umbenannt. Ersetzen Sie bei VIs, in denen die Methode “FP öffnen - Alt” enthalten ist, diese Methode durch “Frontpanel:Öffnen”.

## Methode “VI ausführen”

Wenn Sie in LabVIEW 7.1 den Eingang **Ref automatisch schließen** der Methode “VI ausführen” auf TRUE setzen, wird die Refnum nach Beendigung der VI-Ausführung automatisch geschlossen. In LabVIEW 8.x wird die Refnum auch dann automatisch geschlossen, wenn die Methode einen Fehler ausgibt. Dadurch kann in einem VI, das auf die Referenz angewiesen ist, während der Ausführung ein Fehler auftreten.

## Ereignisse “Taste gedrückt” und “Tastenwiederholung”

Das Datenfeld **VTaste** der Ereignisse “Taste gedrückt”, “Taste gedrückt?”, “Tastenwiederholung” und “Tastenwiederholung?” für VIs und Bedienelemente gibt nun zwei verschiedene Werte aus, je nachdem, ob die Taste <Return> im Buchstabenblock und die Taste <Enter> im Zahlenblock der Tastatur gedrückt wurde. Wenn in LabVIEW 7.x oder Vorgängerversionen durch Betätigen der <Enter>- oder <Return>-Taste eines dieser Ereignisse ausgelöst wird, gibt LabVIEW im Datenfeld **VTaste** immer **<Enter>** aus. Wenn in LabVIEW 8.x durch Betätigen der <Enter>- oder <Return>-Taste eines dieser Ereignisse ausgelöst wird, gibt LabVIEW unter **VTaste** entweder **<Enter>** oder **<Return>** aus.

**(Mac OS)** In LabVIEW 8.x können Kontextmenüs nur noch durch einen Klick bei gedrückter <Control>-Taste und nicht mehr bei gedrückter <Command>-Taste geöffnet werden. Wenn Sie dieses Verhalten mit einer Ereignisstruktur nachahmen, passen Sie Ihre VIs der neuen Funktionsweise entsprechend an.

## Eigenschaften für Listenfeld

Wenn Sie in LabVIEW 7.x oder Vorgängerversionen die Eigenschaft “Erste Zeile” in einem Listenfeld auf eine Zeile einstellen, die sich unter dem untersten Element im Listenfeld befindet, so fixiert LabVIEW die Reihe am letzten sichtbaren Element. In LabVIEW 8.x hat die Anzahl sichtbarer Einträge keinen Einfluss auf die Anzahl der Zeilen, die mit dieser Eigenschaft verbunden werden können.

LabVIEW 8.x unterstützt die Eigenschaft “Doppelklick” für einspaltige Listenfelder nicht. Verwenden Sie stattdessen die Methode “Doppelt angeklickte Zeile ermitteln”.

## Eigenschaft “Übergeordnetes VI”

In LabVIEW 7.x und Vorgängerversionen gibt die Eigenschaft “Übergeordnetes VI” eine Refnum für das VI aus, zu dem das Objekt gehört. Diese Refnum behält das VI im Speicher. In LabVIEW 8.x behält die Refnum, die von der Eigenschaft “Übergeordnetes VI” ausgegeben wird, das VI nicht

im Speicher. Wenn das übergeordnete VI nicht mehr im Arbeitsspeicher ist, wird die Refnum daher ungültig. Eine Refnum für ein VI, mit der das VI so lange im Speicher verbleibt, bis die Refnum per Hand geschlossen wird, kann mit der Funktion “VI-Referenz öffnen” erzeugt werden.

## **Texteigenschaft**

In LabVIEW 7.x und Vorgängerversionen gibt die Eigenschaft “Text” einen String in normaler Darstellung aus. In LabVIEW 8.x dagegen gibt die Eigenschaft einen String in derselben Darstellungsart aus wie das Frontpanel-Objekt. Wenn Sie das Frontpanel-Objekt beispielsweise auf Passwortanzeige gestellt haben, gibt die Eigenschaft den Text auch in der Passwortanzeige aus.

## **Eigenschaften für Baumstruktur-Element**

In LabVIEW 7.x und Vorgängerversionen geben “Eigenschaften aktiver Zellen:Zellengröße:Höhe” und “Eigenschaften aktiver Zellen:Zellengröße:Breite” 17 Pixel für jede Zeile im Baumstruktur-Element aus. In LabVIEW 8.x geben sie 16 Pixel für jede Zeile im Baumstruktur-Element aus.

## **VI-String-Methoden**

Strings, die Sie aus früheren LabVIEW-Versionen mit Hilfe der Methode “VI-Strings exportieren” exportieren, werden mit der Methode “VI-Strings:Importieren” möglicherweise nicht ordnungsgemäß in LabVIEW 8.x importiert.

## **Veraltete Eigenschaften, Methoden und Ereignisse**

Die folgenden Eigenschaften, Methoden und Ereignisse werden von LabVIEW 8.x nicht mehr unterstützt:

### **Eigenschaften für Cursor**

Die Eigenschaft “Cursor-Fixiermodus” wird in LabVIEW 8.x nicht mehr unterstützt. Verwenden Sie stattdessen die Eigenschaft “Cursor-Modus”.

### **Eigenschaften für Listenfelder, Tabellen, Binärtabellen, Baumstrukturen und Ereignisse**

In LabVIEW 8.x wird die Eigenschaft “Zellenvordergrundfarbe” für mehrspaltige Listenfelder nicht mehr unterstützt. Verwenden Sie stattdessen die Eigenschaft “Aktive Zelle:Schriftart der Zelle:Farbe”.

In LabVIEW 8.x wird die Eigenschaft “Zellenvordergrundfarbe” für Text- und Binärtabellen nicht mehr unterstützt. Verwenden Sie stattdessen die

Eigenschaft “Aktive Zelle:Schriftart für Zelle:Farbe” für Tabellen und Digitaltabellen.

In LabVIEW 8.x wird die Eigenschaft “Eigenschaften aktiver Zellen:Vordergrundfarbe” für Baumstrukturelemente nicht mehr unterstützt. Verwenden Sie stattdessen die Eigenschaft “Aktive Zelle:Schriftart der Zelle:Farbe”.

In LabVIEW 8.x werden die Ereignisse “Ziehen”, “Ziehen?”, “Ablegen” und “Ablegen?” in der Klasse “Baumstruktur” nicht mehr unterstützt. Verwenden Sie stattdessen die Ereignisse “Ziehen beendet”, “Ziehen - Eintritt”, “Ziehen - Austritt”, “Ziehen - Über”, “Ziehen - Quelle aktualisiert”, “Ziehen begonnen”, “Ziehen begonnen?” und “Ablegen” in der Klasse “Bedienelement”.

## **Eigenschaften für numerischesElementMitText**

In LabVIEW 8.x werden die Eigenschaften “Farben angegebenes NumElement”, “Farben angegebenes NumElement:Hintergrundfarbe” und “Farbe angegebenes numerisches Element:Textfarbe” für numerische Elemente nicht mehr unterstützt. Verwenden Sie stattdessen die Eigenschaften “Textfarben”, “Textfarben:Hintergrundfarbe” und “Textfarben:Textfarbe”.

## **Eigenschaften für Panel**

In LabVIEW 8.x wird die Eigenschaft “Farbe” in der Klasse “Frontpanel” nicht mehr unterstützt. In LabVIEW 8.x gilt die Eigenschaft nur für den Fensterbereich links oben. Verwenden Sie stattdessen die Eigenschaft “Farbe des Fensterbereichs” in der Klasse “Fensterbereich”.

## **Unterspanel-Eigenschaften**

Verwenden Sie in LabVIEW 8.x das Feld eines SubVIs in einem Unterpanel, um die Anzeige der Bildlaufleisten für Unterpanel-Elemente zu konfigurieren und das Frontpanel in den Unterpanel-Elementen zu skalieren.

In LabVIEW 8.x wird die Unterpanel-Eigenschaft “X-Bildlaufleiste sichtbar” nicht mehr unterstützt. Verwenden Sie stattdessen die Eigenschaft “Sichtbarkeit der horizontalen Bildlaufleiste” für den Fensterbereich.

In LabVIEW 8.x wird die Unterpanel-Eigenschaft “Y-Bildlaufleiste sichtbar” nicht mehr unterstützt. Verwenden Sie stattdessen die Eigenschaft “Sichtbarkeit der vertikalen Bildlaufleiste” für den Fensterbereich.

In LabVIEW 8.x wird die Unterpanel-Eigenschaft “Panel skalieren” nicht mehr unterstützt. Verwenden Sie stattdessen die Methode “Skalierungsmodus einstellen” für den Fensterbereich.

## Eigenschaften, Methoden und Ereignisse für VIs

In LabVIEW 8.x wird die Eigenschaft “Frontpanel-Fenster:Autom. zentrieren” nicht mehr unterstützt. Verwenden Sie stattdessen die Methode “Frontpanel:Zentriert”.

In LabVIEW 8.x wird die Eigenschaft “Frontpanel-Fenster:Bildschirmfüllend anzeigen” nicht mehr unterstützt. Verwenden Sie stattdessen die Eigenschaft “Frontpanel-Fenster:Status”.

In LabVIEW 8.x wird die Eigenschaft “Frontpanel-Fenster:Ursprung” in der VI-Klasse nicht mehr unterstützt. In LabVIEW 8.x gilt die Eigenschaft nur für den Fensterbereich links oben. Verwenden Sie stattdessen die Eigenschaft “Ursprung” in der Klasse “Fensterbereich”.

In LabVIEW 8.x wird die Eigenschaft “Frontpanel-Fenster:Bildlaufleisten anzeigen” in der VI-Klasse nicht unterstützt. In LabVIEW 8.x gilt die Eigenschaft nur für den Fensterbereich links oben. Verwenden Sie stattdessen die Eigenschaften “Sichtbarkeit der horizontalen Bildlaufleiste” und “Sichtbarkeit der vertikalen Bildlaufleiste” in der Klasse “Fensterbereich”.

In LabVIEW 8.x werden die Methoden “Frontpanel-Skalierungsmodus ermitteln” und “Frontpanel-Skalierungsmodus einstellen” in der VI-Klasse nicht mehr unterstützt. In LabVIEW 8.x gelten diese Methoden nur für den Fensterbereich links oben. Verwenden Sie stattdessen die Methoden “Skalierungsmodus ermitteln” und “Skalierungsmodus setzen” in der Klasse “Fensterbereich”.

In LabVIEW 8.x werden die Ereignisse “Maustaste gedrückt”, “Maustaste gedrückt?”, “Maus tritt in Bereich ein”, “Maus verlässt Bereich”, “Mausbewegung” oder “Maustaste losgelassen” in der VI-Klasse nicht mehr unterstützt. Verwenden Sie stattdessen die Ereignisse “Maustaste gedrückt”, “Maustaste gedrückt?”, “Maus tritt in Bereich ein”, “Maus verlässt Bereich”, “Mausbewegung” oder “Maustaste losgelassen” in der Klasse “Fensterbereich”.

## Tags der LabVIEW-Standardmenüpunkte

Folgende Applikationselement-Tags wurden aus LabVIEW 8.x entfernt:

- 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

Bei Laufzeitmenü-Dateien (\*.rtm-Dateien), die in einer Vorgängerversion von LabVIEW gespeichert wurden, wird das Tag in LabVIEW 8.x automatisch aus der \*.rtm-Datei entfernt, wenn die Datei im **Menü-Editor** gespeichert wird. Die gelöschten Applikationselement-Tags sind für LabVIEW reserviert und können nicht als Benutzer-Tags verwendet werden.

## Unterstützung von HiQ

HiQ wird in LabVIEW 8.x nicht mehr unterstützt. Wenn in einer Applikation HiQ-VIs verwendet werden, sollten diese eventuell durch Signalverarbeitungs- und Mathematik-VIs ersetzt werden.

## Fenster "Fehlerliste"

In LabVIEW 7.x und Vorgängerversionen werden im Abschnitt **VI-Liste** des Fensters **Fehlerliste** alle Fehler in den VIs im Speicher angezeigt. In LabVIEW 8.x werden in der **Fehlerliste** unter **Fehlerhafte Elemente** die Fehler an allen Objekten im Speicher, wie VIs oder Bibliotheken, angezeigt. Wenn mehrere Elemente den gleichen Namen haben, wird hier die Applikationsinstanz für jedes Element angezeigt. Weitere Informationen zu Applikationsinstanzen finden Sie in der *LabVIEW-Hilfe*.

## Syntax für VI-String-Dateien

Wenn mit der Option **Werkzeuge»Fortgeschritten»Strings importieren** oder der Methode "VI-Strings:Importieren" VI-String-Dateien importiert werden, sucht LabVIEW 8.x nach den Tags <GROUPER></GROUPER>. Diese Tags stehen für Frontpanel-Objekte, die zu einer Gruppe zusammengefasst sind. Daher können in LabVIEW 8.x keine VI-String-Dateien importiert werden, die in älteren Versionen von LabVIEW erstellt wurden.

In LabVIEW 7.1 und Vorgängerversionen werden Listenfeld-Strings im Abschnitt <ITEMS> der Privatdaten aufgelistet. In LabVIEW 8.x werden Strings in einem Listenfeld im Abschnitt <STRINGS> der Privatdaten aufgelistet. In LabVIEW 7.1 und Vorgängerversionen kann für ein Listenfeld nur eine Schriftart verwendet werden. Diese wird im Abschnitt <LBLABEL> der Privatdaten aufgeführt. In LabVIEW 8.x dagegen können in einem Listenfeld mehrere Schriftarten vorkommen. Diese werden im Abschnitt <CELL\_FONTS> der Privatdaten aufgelistet.

In LabVIEW 7.1 und Vorgängerversionen werden Strings in einem mehrspaltigen Listenfeld in den Standarddaten aufgelistet. Der Standardwert für ein mehrspaltiges Listenfeld ist jedoch ein Integer oder ein Array aus Integer. In LabVIEW 8.x werden Strings in einem mehrspaltigen Listenfeld in den Privatdaten aufgelistet.

In LabVIEW 7.1 und Vorgängerversionen werden weder Text noch Schriftarten aus Baumstrukturen exportiert. In LabVIEW 8.x können sowohl Baumstruktur-Strings als auch -Schriftarten exportiert werden. Diese werden in das gleiche Format exportiert wie bei einem Listenfeld bzw. mehrspaltigen Listenfeld.

In LabVIEW 8.x enthält jede Zeile einer Exportdatei nicht mehr als zwei Tags für Privat- oder Standarddaten. In LabVIEW 8.x werden die Elemente auch auf jeder Unterebene eingerückt.

Führen Sie folgende Schritte aus, um VI-String-Dateien in das Format von LabVIEW 8.x zu konvertieren:

1. Importieren Sie die VI-String-Datei in die vorherige LabVIEW-Version.
2. Speichern Sie das VI.
3. Laden Sie das VI in LabVIEW 8.x.
4. Wählen Sie **Werkzeuge»Fortgeschritten»Strings exportieren**, um die VI-String-Datei im Format von LabVIEW 8.x zu speichern.

## Konvertieren von Typdeskriptordaten in das oder aus dem Format von LabVIEW 7.x

Das Speicherformat von Typdeskriptoren hat sich in LabVIEW 8.x geändert. In LabVIEW 7.x werden Typdeskriptoren in Form von 16-Bit-Strings gespeichert. In LabVIEW 8.x werden Typdeskriptoren in Form von 32-Bit-Strings gespeichert. Dadurch wird die Beschränkung der Größe auf 64 KB für Typdeskriptoren aufgehoben.

In LabVIEW 8.x ist es allerdings möglich, die in LabVIEW 7.x gespeicherten Typdeskriptoren zu lesen und so zu speichern, dass sie mit LabVIEW 7.x kompatibel sind. Die Funktion "In String konvertieren" hat dazu nun den Kontextmenüpunkt **7.x-Daten konvertieren**. Wenn Sie die Funktion mit der rechten Maustaste anklicken und diesen Menüpunkt auswählen, wird mit den Eingangsdaten so verfahren, als ob in LabVIEW 7.x gearbeitet würde. Wenn Sie die Option **7.x-Daten konvertieren** aus dem Kontextmenü auswählen und am Ausgang **Daten-String** ein String anliegt, wird die Funktion in LabVIEW 8.x mit einem roten 7.x gekennzeichnet, um anzuzeigen, dass die Daten nicht im üblichen LabVIEW-8.x-Format vorliegen. Sollen die Daten nicht (mehr) umgewandelt werden, wird die

Menüoption **7.x-Daten konvertieren** beim erneuten Anklicken deaktiviert.

Wenn in LabVIEW 8.x ein VI geladen wird, das zuletzt in LabVIEW 7.x oder einer älteren Version gespeichert wurde, wird in LabVIEW 8.x bei der Funktion "In String konvertieren" die Einstellung **7.x-Daten konvertieren** automatisch aktiviert. Die Funktion arbeitet weiterhin wie in LabVIEW 7.x und Vorgängerversionen. Wenn für ein VI das Typdeskriptorformat von LabVIEW 8.x verwendet werden soll, klicken Sie mit der rechten Maustaste auf die Funktion "In String konvertieren" und deaktivieren Sie die Option **7.x-Daten konvertieren**. Wenn VIs weder mit Dateien arbeiten, die Daten im LabVIEW-7.x- oder einem älteren Format enthalten, noch Daten mit VIs austauschen, die in LabVIEW 7.x oder einer Vorgängerversion ausgeführt werden, empfiehlt sich in der Regel das LabVIEW-8.x-Typdeskriptorformat. Das alte Format wird unter Umständen in zukünftigen Versionen nicht mehr unterstützt.

## Umstellung vom integrierten LabVIEW-Versionsverwaltungs-Provider

Den Versionsverwaltungs-Provider von LabVIEW 7.x und Vorgängerversionen gibt es in LabVIEW 8.x nicht mehr. Wenn Sie in LabVIEW die Versionsverwaltung nutzen möchten, müssen Sie einen Versionsverwaltungs-Provider von einem Drittanbieter auswählen. Wenn Sie bisher den integrierten Provider verwendet haben, müssen Sie die Dateien auf einen anderen Versionsverwaltungs-Provider in LabVIEW umstellen. Die aktuelle Liste der Versionsverwaltungs-Provider von Drittanbietern, mit denen LabVIEW arbeitet, finden Sie auf der Website von National Instruments unter [ni.com/info](http://ni.com/info) nach Eingabe des Infocodes `exgucn`.

Beim Umstellen von Dateien auf einen neuen Versionsverwaltungs-Provider geht die Versionshistorie verloren, die im LabVIEW-Provider gespeichert war. Ältere Versionen der Dateien lassen sich nicht auf den neuen Provider übertragen.

Führen Sie folgende Schritte aus, um Dateien vom integrierten Versionsverwaltungs-Provider auf einen Versionsverwaltungs-Provider von einem Drittanbieter zu übertragen:

1. Vergewissern Sie sich, dass die Dateien im integrierten Versionsverwaltungs-Provider früherer LabVIEW-Versionen von allen Nutzern eingesehen sind.
2. Rufen Sie auf dem Computer, auf dem Sie den Versionsverwaltungs-Provider umstellen möchten, im integrierten Provider die neueste Version aller Dateien ab.
3. Checken Sie im integrierten Provider die betreffenden Dateien aus der Versionsverwaltung aus.

4. Nehmen Sie im Versionsverwaltungs-Provider des Drittanbieters die im neuen Versionsverwaltungsprojekt gewünschten Einstellungen vor.
5. Konfigurieren Sie LabVIEW für die Zusammenarbeit mit dem neuen Versionsverwaltungs-Provider. Informationen zur Konfiguration von LabVIEW für die Verwendung eines Versionsverwaltungs-Providers von Drittanbietern finden Sie im Buch **Grundlagen»Organisieren und Verwalten von Projekten»Anleitung»Verwendung der Versionsverwaltung in LabVIEW** im Register **Inhalt** der *LabVIEW-Hilfe*.
6. Erstellen Sie ein LabVIEW-Projekt. Fügen Sie dem Projekt die Dateien hinzu, die sich im integrierten Versionsverwaltungs-Provider befinden. Fügen Sie die Dateien der Versionsverwaltung hinzu, wenn Sie dazu aufgefordert werden. Sie können die Dateien auch direkt in den Provider des Drittanbieters einfügen. Informationen zum Erstellen eines LabVIEW-Projekts finden Sie im Buch **Grundlagen»Organisieren und Verwalten von Projekten»Anleitung»Erstellen eines LabVIEW-Projekts** im Register **Inhalt** der *LabVIEW-Hilfe*

## Umwandeln von NaN-Strings in Integer (Windows)

Wenn NaN in LabVIEW 7.x in einen Integer umgewandelt wird (entweder programmatisch oder durch den Benutzer), stellt dieser Wert den kleinsten Wert für diesen Integer-Datentyp dar. So ergibt die Umwandlung von NaN in einen 16-Bit-Integer mit Vorzeichen beispielsweise  $-32,768$ . Das ist der kleinstmögliche Wert für einen 16-Bit-Integer mit Vorzeichen.

Wenn NaN in LabVIEW 8.x in einen Integer umgewandelt wird (entweder programmatisch oder durch den Benutzer), stellt dieser Wert den größten Wert für diesen Integer-Datentyp dar. So ergibt die Umwandlung von NaN in einen 16-Bit-Integer mit Vorzeichen beispielsweise  $32,767$ . Das ist der größtmögliche Wert für einen 16-Bit-Integer mit Vorzeichen.

## Mit Case-Strukturen verbundene Konstanten

In LabVIEW 7.x und älteren Versionen können SubVIs im Speicher gehalten werden, indem eine Konstante mit einer Case-Struktur verbunden wird und die SubVIs in einem niemals aufgerufenen Case untergebracht werden. Wenn Sie zum Beispiel eine TRUE-Konstante mit einer Case-Struktur verbinden und ein SubVI im FALSE-Case unterbringen, lädt LabVIEW das SubVI zusammen mit dem aufrufenden VI. In LabVIEW 8.x werden alle nicht benötigten Blockdiagramm-Abschnitte entfernt. Wenn Sie also ein in einer älteren Version gespeichertes VI mit einer Konstante, die mit einer Case-Struktur verbunden ist, in LabVIEW 8.x laden, wird die Konstante in ein verborgenes Bedienelement umgewandelt, um die Funktionsweise der älteren Version beizubehalten.

## Verzögern von Meldungen des Betriebssystems

In LabVIEW 7.x werden Meldungen des Betriebssystems bearbeitet, während Callback-VIs zum Bearbeiten von .NET- und ActiveX-Ereignissen ausgeführt werden. In LabVIEW 8.x wird die Bearbeitung von Meldungen des Betriebssystems so lange verzögert, bis die Ausführung des Callback-VIs beendet ist bzw. bis Sie ein modales Dialogfeld geladen haben. Durch diese Verzögerung werden Callback-VIs ohne Unterbrechung ausgeführt. Es wird auch kein Ereignis in einem anderen Ereignis ausgelöst, was zu einem Sperrstatus führen kann.

Sie können von einem Callback-VI keine synchronen Aufrufe zu nicht modalen Dialogfenstern starten. Die Aufrufe eines nicht modalen Dialogfensters von einem Callback-VI müssen asynchron sein. Das wird erreicht, indem Sie die Methode “VI ausführen” im Dialogfenster aufrufen und eine boolesche Konstante mit dem Wert FALSE mit dem Eingang **Auf Ende der Operation warten** verbinden.

In LabVIEW 7.x werden Meldungen des Betriebssystems bearbeitet, während DLL- bzw. Shared-Library-Funktionen ausgeführt werden. In LabVIEW 8.x wird die Bearbeitung von Meldungen des Betriebssystems verzögert, bis die Aufrufe der DLL-Funktionen beendet sind bzw. bis Sie ein modales Dialogfeld von der DLL aus geladen haben. Durch diese Verzögerung werden DLL-Funktionen ohne Unterbrechung ausgeführt. LabVIEW ruft auch nicht die gleiche DLL auf, während eine DLL-Funktion ausgeführt wird, was zu einem Sperrstatus führen kann.

Ist dieses Standardverhalten eingestellt, können keine synchronen Aufrufe nicht modaler Dialogfelder erfolgen, während eine DLL ausgeführt wird. Die Aufrufe eines nicht modalen Dialogfensters von einer DLL aus müssen asynchron sein. Das wird erreicht, indem Sie die Methode “VI ausführen” im Dialogfenster aufrufen und eine boolesche Konstante mit dem Wert FALSE mit dem Eingang **Auf Ende der Operation warten** verbinden.

In DLLs, die Sie selbst erstellen, können Sie bestimmen, ob Meldungen des Betriebssystems verzögert werden sollen oder nicht. Klicken Sie mit der rechten Maustaste auf die DLL im **Projekt-Explorer**, wählen Sie **Eigenschaften** aus dem Kontextmenü und dann **Fortgeschritten** aus der Liste **Kategorie**. Deaktivieren Sie die Option **Betriebssystemmeldungen in DLL verzögern**, um während der Ausführung von DLL-Funktionen Meldungen des Betriebssystems bearbeiten zu können.

## Ressourcenmanager (Mac OS)

LabVIEW 7.x und frühere Versionen bieten undokumentierte Eigenschaften, mit denen Macintosh-Ressourcendateien gelesen und geschrieben werden können. Diese Methoden gibt es in LabVIEW 8.x jedoch nicht mehr. Hilfsprogramme, in denen diese undokumentierten Eigenschaften genutzt werden, funktionieren nicht mehr. Daher kann nicht mehr mit Macintosh-Ressourcendateien gearbeitet werden.

## Dialogfelder mit einer oder zwei Schaltflächen

In LabVIEW 7.x und Vorgängerversionen können VIs zur Anzeige von Dialogfeldern mit einer oder zwei Schaltflächen nicht programmatisch abgebrochen werden. In LabVIEW 8.x dagegen können Sie ein solches VI programmatisch abbrechen, indem Sie die Methode "VI abbrechen" aufrufen.

## Eigenschafts- und Methodenknoten

Wenn Sie in LabVIEW 7.x einen nicht sichtbar mit einem Objekt verbundenen Eigenschafts- oder Methodenknoten von einer Cursor-Legende aus erstellen, wird der Knoten gelöscht, wenn das VI in LabVIEW 8.x geöffnet wird.

## Update von DLLs

Wenn Sie in LabVIEW 7.x oder einer Vorgängerversion eine DLL erstellen, die mit `labview.lib` verknüpft ist, verbinden Sie die DLL in LabVIEW 8.x stattdessen mit `labviewv.lib`. Weitere Informationen zum Verknüpfen von DLLs mit `labviewv.lib` finden Sie in der *LabVIEW-Hilfe*.

## Seitenränder beim Drucken

In LabVIEW 7.x und Vorgängerversionen werden die Seitenränder in der Option **Seitenränder** der Seite **Drucken** im Dialogfeld **Optionen** in Zentimetern angegeben. In LabVIEW 8.x werden die **Seitenränder** in Millimetern angegeben.

## Upgrade von LabVIEW 6.x

Bei einem Upgrade von LabVIEW 6.x auf LabVIEW 8.2 können die nachfolgend aufgeführten Kompatibilitätsprobleme auftreten. Alle weiteren potentiellen Probleme beim Upgrade von LabVIEW sind in den Abschnitten [Upgrade von LabVIEW 7.x](#) und [Upgrade von LabVIEW 8.0](#) beschrieben.

Alle Unterschiede zwischen den einzelnen Versionen ab 6.x sind in den *Hinweisen zum Upgrade von LabVIEW* jeder Version beschrieben, die Sie unter [ni.com/manuals](http://ni.com/manuals) finden.

## Änderungen am Datentyp “Signalverlauf”

Der Datentyp “Signalverlauf” verwendet in LabVIEW 7.0 anstelle einer Fließkommazahl mit doppelter Genauigkeit für die Komponente **t0** den Datentyp “Zeitstempel”. Wenn Sie Daten in diesem Datenformat in LabVIEW 6.x speichern, ohne die Änderung in LabVIEW-Versionen ab 7.x zu berücksichtigen, tritt beim Abrufen der Daten unter Umständen ein Fehler auf.

In LabVIEW 7.x und späteren Versionen wandelt das VI “Signalverlauf aus Datei lesen” das alte Signalverlaufsdatentyp-Format in einer Datei in ein neues um. Ein Dialogfeld öffnet sich, in dem Sie aufgefordert werden, die Umwandlung zu akzeptieren. In der LabVIEW Runtime-Engine kann das VI “Signalverlauf aus Datei lesen” diese Umwandlung jedoch nicht vornehmen. Daher wird eine Fehlermeldung ausgegeben. Wenn Sie auf der Website [ni.com/info](http://ni.com/info) den Infocode `exd9zq` eingeben, erfahren Sie, was es im Hinblick auf Signalverläufe bei der Umstellung von LabVIEW 6.x auf LabVIEW 7.x und spätere Versionen zu beachten gibt.

## VIs für serielle Kompatibilität

In LabVIEW 7.x und späteren Versionen wird das VI “Serielle Kompatibilität” nicht mehr auf der Palette **Funktionen** angezeigt. VIs zur Kommunikation mit VXI-Modulen sollten mit VISA-VIs und -Funktionen erstellt werden.

In LabVIEW-Versionen ab 7.x wird zur Kommunikation mit dem seriellen Treiber des Betriebssystems nicht mehr mit dem `serpdrv`-Treiber gearbeitet. LabVIEW enthält kompatible VIs, die auf VISA basieren. Verwenden Sie in neuen Anwendungen zur Steuerung von Geräten mit serieller Schnittstelle die VIs und Funktionen der VISA-Palette. Alle VIs, die in Vorgängerversionen von LabVIEW enthalten waren und mit seriellen VIs gearbeitet haben, sind auch in LabVIEW 7.1 und späteren Versionen funktionstüchtig.

Wenn in der ursprünglichen LabVIEW-Anwendung Änderungen an der Zuweisung der Portnummern vorgenommen wurden, ist die Änderung anzugeben. Verwenden Sie dazu das VI “Set Serial Alias Ports” im Verzeichnis `labview\vi.lib\Instr\_sersup.llb`. Verbinden Sie ein String-Array mit dem Eingang **VISA Aliases** des VIs und geben Sie die Portbezeichnungen ein, die im Eingangsarray verwendet werden. Jedes Element im Array sollte einen Port darstellen. Wenn Sie beispielsweise Port 0 das VISA-Alias “MySerialPort” zugewiesen haben, geben Sie in das

Array **VISA Aliases** als erstes Element `MySerialPort` ein. Vor dem VI “Set Serial Alias Ports” muss immer das VI “VISA: Seriellen Port konfigurieren” aufgerufen werden.

Anwendungsbeispiele für die VISA-VIs und -Funktionen zur Steuerung serieller Geräte finden Sie unter `labview\examples\instr\smp1ser1.llb`.

## Standardwerte in Schleifen

In LabVIEW 6.0 und Vorgängerversionen werden bei Nicht-Ausführung von For-Schleifen undefinierte Werte ausgegeben. In LabVIEW-Versionen ab 6.1 geben For-Schleifen voreingestellte Werte aus, wenn für die Anzahl der Durchläufe 0 angegeben wird oder ein Eingang mit aktivierter Auto-Indizierung mit einem leeren Array verbunden wird. In diesem Fall wird die Schleife nicht ausgeführt, und für alle nicht automatisch indizierten Ausgabetunnel werden die Standardwerte für den Tunnel-Datentyp ausgegeben.

## Lizenzen für netzwerkgesteuerte Frontpanel

Das LabVIEW Full Development System und der Application Builder enthalten eine Lizenz, mit der es einem Client möglich ist, ein Frontpanel über ein Netzwerk zu bedienen. Das LabVIEW Professional Development System wird mit einer solchen Lizenz für fünf Client-Rechner ausgeliefert.

Diese Lizenzen sind auf mehrere Benutzer erweiterbar.

## Vergabe mehrerer Ausführungsthreads

In LabVIEW-Versionen ab 7.1 werden zur Ausführung von VIs mehr Threads vergeben als in den Vorgängerversionen. Daher kann es bei der Verarbeitung mehrerer Threads zu Fehlern kommen, wenn ein Funktionsknoten zum Aufruf externer Bibliotheken fälschlicherweise als ablaufinvariant gekennzeichnet wurde, obwohl die DLL eigentlich nicht ablaufinvariant ist. In der *LabVIEW-Hilfe* finden Sie weitere Informationen zum Knoten für den Aufruf externer Bibliotheken und zur Ablaufinvarianz.

Mit dem VI “Threadconfig” in der Bibliothek `labview\vi.lib\Utility\sysinfo.llb` können Sie die Zuweisung von Threads ändern. Sie können die Ablaufinvarianz von VIs deaktivieren, indem Sie **Datei» VI-Einstellungen** wählen, dann im Pulldown-Menü **Kategorie auf Ausführung** klicken und die Option **Ablaufinvariante Ausführung** deaktivieren.

Weitere Informationen zum Zuweisen von Threads finden Sie in der *LabVIEW-Hilfe*.

## Treiber

Bei LabVIEW-Versionen ab LabVIEW 7.x ist die LabVIEW-Instrumententreiber-CD nicht im Lieferumfang enthalten. Im Instrument Driver Network von National Instruments unter [ni.com/idnet](http://ni.com/idnet) stehen die Treiber jedoch weiterhin zum Herunterladen zur Verfügung. Auf der Treiber-CD von National Instruments sind unter anderem NI-DAQ und NI-VISA enthalten.

## Einheiten und Umrechnungsfaktoren

In LabVIEW-Versionen ab 7.x muss die zusätzliche Einheit nach Verwendung der Funktion "Mehrfacharithmetik" nicht mit der Funktion "Einheit konvertieren" entfernt werden.

Die Umrechnungsfaktoren für Einheiten wurden in LabVIEW 7.1 und späteren Versionen den Richtlinien des vom National Institute for Standards and Technology (NIST) veröffentlichten *Guide for the Use of the International System of Units (SI)* weiter angepasst. Außerdem wurde **Kalorie** in **Kalorie (thermisch)** und **Pferdestärke** in **Pferdestärke (elektrisch)** umbenannt. Die Abkürzungen für diese Einheiten haben sich nicht geändert. Nachfolgend finden Sie alle Änderungen an den Umrechnungsfaktoren für Einheiten zwischen LabVIEW 6.1 und 7.x:

Einheit	Definition (6.1)	Definition (ab 7.x)
Astronomische Einheit (AU)	149.498.845.000 m	149.597.900.000 m
British Thermal Unit (Mittelwert)	1055,79 J	1055,87 J
Elektronenvolt (eV)	1,602e-19 J	1,60217642e-19 J
Foot-Candle	10,764 lx	10,7639 lx
Pferdestärke verglichen mit Pferdestärke (elektrisch)	745,7 W	746 W. Die neue Umrechnung ist exakt.
Gallone (imperial)	4,54596 l	4,54609 l
Lichtjahr	9,4605 Pm	9,46073 Pm
Pound Force	4,448 N	4,448222 N
Rod	502,92 cm	5,029210 m
Slug	32,174 lb	14,59390 kg
Unitäre atomare Masseneinheit (u)	1,66057e-27 kg	1,66053873e-27 kg

## Eigenschaft “Panel-Aktualisierungen verzögern”

Bei LabVIEW 6.1 und Vorgängerversionen werden Elemente, die sich inzwischen geändert haben, erst wieder aktualisiert, wenn der Wert FALSE an die Eigenschaft übermittelt wird. Wenn diese Eigenschaft in LabVIEW ab Version 7.0 den Wert TRUE hat, werden alle Frontpanel-Objekte mit noch ausstehenden Änderungen neu angezeigt und die nächste Aktualisierung der Objekte wird um eine bestimmte Zeit verschoben. In einigen Fällen nimmt die Neudarstellung der Elemente zusätzliche Zeit in Anspruch.

## Wertebereiche numerischer Bedienelemente

In LabVIEW 6.1 und Vorgängerversionen haben einige numerische Elemente per Voreinstellung einen Mindestwert von 0,00, einen Höchstwert von 0,00, einen Inkrement-Wert von 0,00 und die Einstellung **Ignorieren** bei einer Bereichsüberschreitung. Für diese Elemente werden in LabVIEW ab Version 7.x die Werte des Standardwertebereichs für den jeweiligen Datentyp verwendet.

## Typumwandlungspunkte und -definitionen

In den LabVIEW-Versionen ab 6.1 sind in den Verbindungen Angaben über Typdefinitionen enthalten, so dass im Blockdiagramm mehr Typumwandlungspunkte vorliegen können. Wenn Sie ein Typ-Definitionselement mit einem VI oder einem Knoten verbinden, der kein Typ-Definitionsknoten ist, zeigt LabVIEW einen Typumwandlungspunkt an.

Typumwandlungspunkte werden auch angezeigt, wenn ein Ausgangsanschluss, der eine Typdefinition hat, mit einer Anzeige verbunden wird, die keine Typdefinition hat. Diese Typumwandlungspunkte zeigen, an welchen Stellen im VI Typdefinitionen nicht einheitlich verwendet werden. Die Ausführungsgeschwindigkeit des VIs wird davon nicht beeinträchtigt.

Um Typdefinitionen in Strings zu konvertieren, verwenden Sie das VI “Daten in String konvertieren”, zu dem Sie in der *LabVIEW-Hilfe* nähere Einzelheiten finden.

## Schaltflächenbeschriftungen im Dialogfeld “Datei”

In LabVIEW 6.1 und Vorgängerversionen wird im Fenster der Dateidialog-Funktion die Schaltfläche **Speichern** angezeigt, wenn der Benutzer einen neuen Dateinamen eingeben kann. Andernfalls lautet die Beschriftung der Schaltfläche **Öffnen**. In LabVIEW 8.x lautet die Schaltflächenbeschriftung im Dateidialogfeld, das vom Express-VI “Dateidialog” angezeigt wird, in allen Fällen **OK**. Die Beschriftung lässt sich nachträglich mit Hilfe des Eingangs **Schaltflächenbeschriftung** des

Express-VIs “Dateialog” ändern. Wenn dieses Express-VI in einem anderen VI verwendet wird, überprüfen Sie das VI, um sicherzustellen, dass die neue Standardbeschriftung **OK** auch dem Zweck Ihres VIs gerecht wird.

## Die Funktion “Online-Hilfe steuern”

Der Eingang **Pfad zur Hilfedatei** der Funktion “Online-Hilfe steuern” kann jetzt nicht mehr offen gelassen werden. An den Anschluss kann entweder ein Dateiname mit der Erweiterung `.chm` oder `.hlp` übergeben werden oder der vollständige Pfad zur Datei. Wenn Sie nur den Namen einer kompilierten Hilfedatei angegeben haben, sucht LabVIEW im Verzeichnis `labview\help` nach der Datei.

## Anzeigen des Frontpanels nach dem Laden

Wenn Sie in LabVIEW ab Version 7.x ein VI so einstellen, dass das Frontpanel beim Laden des VIs sofort sichtbar ist, wird das Frontpanel beim Laden durch den VI-Server nicht angezeigt. In diesem Fall muss das Frontpanel mit Hilfe der Methode “Frontpanel:Öffnen” programmatisch geöffnet werden.

## Funktion “VI-Referenz öffnen”

Wird in LabVIEW 6.1 und Vorgängerversionen der Parameter **Optionen** der Funktion “VI-Referenz öffnen” nicht verbunden, so erstellt LabVIEW ein VI anhand einer Vorlage, sofern sich die Vorlage nicht bereits im Speicher befindet. Befindet sich die Vorlage im Speicher, erstellt LabVIEW für die Vorlage eine Refnum. Wenn Sie in LabVIEW 7.0 und 7.1 mit Hilfe der Funktion “VI-Referenz öffnen” eine Refnum für eine Vorlage erstellen, die sich bereits im Speicher befindet, gibt die Funktion eine Fehlermeldung aus, es sei denn, Sie legen `0x02` im Parameter **Optionen** fest. Ab LabVIEW 8.0 wird bei Erstellen einer Referenz auf eine Vorlage mithilfe der Funktion “VI-Referenz öffnen” ein VI von der Vorlage erstellt, auch wenn sich diese Vorlage bereits im Speicher befindet.

## Exponentialschreibweise

In LabVIEW-Versionen vor 6.1 wird zur Exponentialschreibweise in einem Formelknoten das Symbol `^` verwendet. In den LabVIEW-Versionen ab 6.1 ist eine Potenzierung mit dem Symbol `**` gekennzeichnet (zum Beispiel `x**y`). Das Symbol `^` steht nun für das bitweise Exklusiv-ODER (XOR).

## IVI-Konfigurationsdateiformat

Bei Dateien zum Speichern von IVI-Konfigurationen wird nun zwischen Groß- und Kleinschreibung unterschieden. Wenn in Ihrer Applikation mit logischen, virtuellen oder Treiber-Session-Namen gearbeitet wird, sollten Sie sich vergewissern, dass der in der IVI-Konfigurationsdatei verwendete Name mit dem angegebenen Namen übereinstimmt.

## Formular für technische Unterstützung

In LabVIEW 7.x und späteren Versionen wird `techsup.11b` nicht mehr installiert. Bei Problemen oder Fragen bezüglich der Installation, Konfiguration oder Ausführung von Anwendungen besuchen Sie bitte die Website [ni.com/support](http://ni.com/support).

## Upgrade von LabVIEW 5.x oder älteren Versionen

Informationen zum Upgrade von LabVIEW 5.x und Vorgängerversionen auf LabVIEW 8.2 finden Sie auf der Website von National Instruments [ni.com/info](http://ni.com/info) nach Eingabe des Infocodes `ext8h9`.

## Änderungen und Neuerungen in LabVIEW 8.2

---

In der *LabVIEW-Hilfe* finden Sie Informationen zu Funktionen in LabVIEW 8.2, Grundlagen der Programmierung mit LabVIEW, schrittweise Anleitungen und Referenzen. Zum Öffnen der *LabVIEW-Hilfe* klicken Sie auf **Hilfe»LabVIEW-Hilfe durchsuchen**.

Weitere Informationen zu bekannten Problemen, zusätzlichen Kompatibilitätsproblemen und Informationen zu zuletzt hinzugefügten Funktionen in LabVIEW 8.2 finden Sie in der Datei `readme.html` im `labview`-Verzeichnis.

## LabVIEW-Dokumentation

In LabVIEW 8.2 wird in komplexen Hilfethemen rechts oben ein Seitenüberblick mit Verknüpfungen zu den Unterthemen angezeigt. Klicken Sie in der Übersicht auf einen Link, um zum entsprechenden Thema zu wechseln.

## Neue Beispiel-VIs

Im Ordner **Neue Beispiele für LabVIEW 8.x** auf der Registerkarte **Suchen** der NI-Beispielsuchmaschine finden Sie neue Beispiel-VIs und Beschreibungen dazu. Die Beispiele können direkt gestartet werden.

## Verbesserter Startvorgang

LabVIEW 8.2 kann aufgrund von Leistungsoptimierung schneller gestartet werden als LabVIEW 8.0.

## Verbesserungen des Blockdiagramms

In LabVIEW 8.2 gibt es folgende Verbesserungen am Blockdiagramm und dazugehörigen Funktionen:

### Änderungen an den Standardfarben

Die Standardfarben der folgenden Blockdiagrammkomponenten wurden zur Verbesserung der Benutzerfreundlichkeit geändert:

- Die Verbindungen und Anschlüsse von Fehlerclustern werden auf dem Blockdiagramm dunkelgelb statt rosa dargestellt.
- Typumwandlungspunkte werden per Voreinstellung rot statt grau angezeigt. Wählen Sie zum Ändern dieser Farbe im Menü **Werkzeuge»Optionen** und wählen Sie aus der Liste **Kategorie** den Punkt **Farben** aus. Deaktivieren Sie zum Auswählen einer anderen Farbe das Kontrollkästchen **Standardfarben** und klicken Sie auf das Farbkästchen **Typumwandlungspunkt**.

### Entfernen von Haltepunkten aus einer VI-Hierarchie

Wählen Sie in einem VI die Option **Bearbeiten»Haltepunkte aus Hierarchie entfernen**, um alle Haltepunkte aus der VI-Hierarchie zu entfernen. Haltepunkte müssen in dynamisch aufgerufenen VIs und in VIs, die von der Funktion "Statische VI-Refnum" verwendet werden, manuell entfernt werden.

### Durch Konstanten optimierte Leistung

Mit Hilfe der Konstantenfaltung wird in LabVIEW die Leistung von VIs optimiert. LabVIEW speichert konstante Werte, wenn VIs kompiliert werden, d. h., sie werden nicht während der Laufzeit berechnet. Bei mit Strukturen verbundenen Konstanten berechnet LabVIEW während der Kompilierung von VIs die Ausgangswerte der Strukturen und speichert die Werte, so dass sie während der Ausführung zur Verfügung stehen.

Die Rauten zur Kennzeichnung einer Konstantenfaltung können im Blockdiagramm angezeigt werden, indem Sie **Werkzeuge»Optionen** und **Blockdiagramm** aus der Liste **Kategorie** wählen und die Optionen **Konstantenfaltung von Verbindungen anzeigen** und **Konstantenfaltung von Strukturen anzeigen** aktivieren. Wenn Sie die Option **Konstantenfaltung von Verbindungen anzeigen** aktivieren, werden an den Leitungen, die mit entsprechenden Konstanten verbunden sind, Rauten

angezeigt. Wenn Sie die Option **Konstantenfaltung von Strukturen anzeigen** aktivieren, wird in den Strukturen, die mit Konstanten verbunden sind, ein graues Rautenzeichen angezeigt. Die Raute wird jedoch u. U. erst nach dem Starten oder Speichern des VIs angezeigt.

LabVIEW 8.2 führt die Faltung auch bei berechneten Konstanten durch, die mit Selektoranschlüssen einer Case-Struktur und Bedingungsanschlüssen einer While-Schleife verbunden sind.

## Verschiedene Verbesserungen am Blockdiagramm

In LabVIEW 8.2 gibt es folgende weitere Verbesserungen am Blockdiagramm:

- Wenn Sie mit der rechten Maustaste auf den Ausgang **Referenz (Ausgang)** eines Eigenschafts- oder Methodenknotens klicken und **Erstellen** aus dem Kontextmenü wählen, zeigt das Menü **Erstellen** die Eigenschaften oder Methoden in derselben Klasse wie den Eigenschaftsknoten oder die -methode an.
- Die Eigenschaft “Sichtbare Objekte:Hierarchielinien sichtbar” akzeptiert folgende Einstellungen: Bei **sichtbar** werden immer die Linien angezeigt, um die Hierarchie der Objekte als vertikale und horizontale Linien links neben den Objekten in der Baumstruktur darzustellen. Bei Auswahl von **Nicht sichtbar** werden die Linien immer ausgeblendet. Wenn **Voreinstellung** ausgewählt ist, werden die Hierarchielinien nur dann angezeigt, wenn Baumstrukturen unter diesem Betriebssystem mit Hierarchielinien angezeigt werden.

## Verbesserungen am Frontpanel

In LabVIEW 8.2 gibt es folgende Verbesserungen am Frontpanel und dazugehörigen Funktionen:

### Festlegen von Hintergrundbildern für Fensterbereiche

Sie können Hintergrundbilder für Fensterbereiche festlegen und importieren. Klicken Sie mit der rechten Maustaste auf die Bildlaufleiste des Fensterbereichs und wählen Sie **Eigenschaften** aus dem Kontextmenü aus. Wählen Sie ein Bild im Dialogfeld **Eigenschaften für Fensterbereich** aus der Liste **Hintergrund** aus.

Zur Auswahl eines Bilds, das nicht in der Liste **Hintergrund** angezeigt wird, klicken Sie auf die Schaltfläche **Durchsuchen**. LabVIEW unterstützt Dateien im \*.bmp-, \*.jpeg- und \*.png-Format als Hintergrundbilder. Zum programmatischen Einstellen eines Hintergrundbilds für einen Fen-

sterbereich können Sie auch die Eigenschaft “Hintergrundbild” verwenden.



**Hinweis** Wenn Sie ein Bild auswählen, das nicht in der Liste **Hintergrund** angezeigt wird, fügt LabVIEW dieses nicht dauerhaft der Liste hinzu. Speichern Sie das Bild im Verzeichnis `labview\resource\backgrounds`, um es dauerhaft der Liste hinzuzufügen.

Wenn Sie ein VI mit einem Hintergrundbild laden, speichert LabVIEW das VI zusammen mit dem Bild.

## Fixieren von Drehschaltern und -reglern an Minimal- und Maximalwerten

Drehschalter und Drehregler können per Voreinstellung nicht weiter als bis zum Minimal- bzw. Maximalwert gedreht werden.

Zum Deaktivieren dieser Fixierung klicken Sie mit der rechten Maustaste auf den Drehregler oder -schalter und wählen Sie aus dem Kontextmenü die Option **Eigenschaften** aus. Deaktivieren Sie das Kontrollkästchen **An Minimum und Maximum fixieren**. Bei einem fixierten Drehelement, kann der Wert nicht vom Minimum auf das Maximum oder umgekehrt springen. Wenn Sie diese Funktion deaktivieren, kann es möglicherweise zu unbeabsichtigten Sprüngen zwischen den Werten kommen.

Wenn Sie mit LabVIEW 8.2 ein VI öffnen, das zuletzt mit einer früheren LabVIEW-Version gespeichert wurde, ist die Fixierung deaktiviert. Zum Aktivieren der Option aktivieren Sie das Kontrollkästchen **An Minimum und Maximum fixieren**.

## Ziehen mehrerer Objekte innerhalb von Baumstrukturen und Listenfeldern

Sie können mehrere Elemente in Baumstrukturen und Listenfeldern per Drag-and-Drop verschieben. Klicken Sie mit der rechten Maustaste auf die Baumstruktur oder das Listenfeld und wählen Sie **Auswahlmodus»0 oder mehr Objekte** oder **Auswahlmodus»1 oder mehr Objekte** aus dem Kontextmenü, um das Verschieben mehrerer Elemente per Drag-and-Drop zu aktivieren. Wird der Ziehvorgang mehrerer Objekte begonnen, so werden alle ausgewählten Objekte verschoben.

## Verbesserter digitaler Signalverlaufsgraph

Weitere Informationen zu digitalen Signalverlaufsgraphen finden Sie im Buch **Grundlagen»Graphen und Diagramme** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

In LabVIEW 8.2 gibt es folgende Verbesserungen an digitalen Signalverlaufsgraphen.

## Festlegen der Liniendicke

**Linienstil** ersetzt **Position der fetten Linie** im Kontextmenü der Plot-Legende des digitalen Signalverlaufsgraphen. Klicken Sie mit der rechten Maustaste auf die Kurve in der Plot-Legende und wählen Sie zum Festlegen der Liniendicke **Punktstil** aus dem Kontextmenü. LabVIEW 8.2 enthält eine neue **Linienstil**-Option, mithilfe welcher die gesamte Linie fett dargestellt werden kann.

## Dunklere Vergleichsdaten

Wenn ein digitaler Signalverlaufsgraph in beiden logischen Zuständen digitale Daten enthält, werden die Vergleichsdaten standardmäßig dunkler als die Festplattendaten angezeigt. Wenn die Vergleichsdaten nicht dunkler angezeigt werden sollen, klicken Sie mit der rechten Maustaste auf den Plot und deaktivieren Sie die Option **Fortgeschritten»Vergleichsdaten verdunkeln**. Mithilfe der Eigenschaft "Vergleichsdaten verdunkeln" können Sie die Vergleichsdaten auch programmatisch verdunkeln.



**Hinweis** Diese Eigenschaft eignet sich hauptsächlich für die Erzeugung digitaler I/O-Signale. Weitere Informationen zu Vergleichsdaten finden Sie im Buch **Grundlagen»Graphen und Diagramme»Allgemeines»Individuelle Gestaltung von Graphen und Diagrammen** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

## Verschiedene Verbesserungen am Frontpanel

In LabVIEW 8.2 gibt es folgende weitere Verbesserungen am Frontpanel:

- In LabVIEW 8.0 wurde beim ersten Anzeigen eines Untertitels eines Frontpanel-Objekts die Beschriftung zur Seite geschoben. In LabVIEW 8.2 wird die Beschriftung ausgeblendet und nur der Untertitel angezeigt.
- Im Dialogfeld **Graph-Eigenschaften: XY-Graph** wurden das Pull-down-Menü **Optionale Ebene anzeigen** und die Optionen zum Konfigurieren der ausgewählten Ebene von der Seite **Skalierungen** auf die Seite **Erscheinungsbild** verschoben.
- Mithilfe des Farbwerkzeugs kann die Hintergrundfarbe einer Systemtabelle verändert werden.
- In mehrspaltigen Systemlistenfeldern, einfachen Listenfeldern, Tabellen und Baumstrukturen können Sie die Farbe der Kopfzeilen und der Zellen ändern.

- Die Optionen **Standardwert wiederherstellen**, **Daten ausschneiden** und **Daten einfügen** sind nicht im Kontextmenü eines Anzeigeelements verfügbar, wenn sich das VI im Ausführungsmodus befindet. Diese Kontextmenüoptionen sind nur für Bedienelemente im Ausführungsmodus verfügbar.
- Klicken Sie zur Größenanpassung des Registerelements mit der rechten Maustaste auf das Element und wählen Sie **Fortgeschritten» Größe anpassen** aus dem Kontextmenü.
- Wenn Sie mit der rechten Maustaste auf ein Enum-, Ring- oder Kombinationsfeld -Element klicken und **Objekte bearbeiten** aus dem Kontextmenü wählen, müssen Sie doppelt in eine Zelle klicken, um ein Element in der Spalte **Objekte** oder **Werte** zu bearbeiten. Diese Änderung trifft auch zu, wenn Sie mit der rechten Maustaste auf Beschriftungen von Schieberegler oder Drehschaltern klicken und **Objekte bearbeiten** aus dem Kontextmenü wählen.
- Wenn Sie in LabVIEW 8.0 **Datei»Änderungen übernehmen** im Element-Editor wählen, behält LabVIEW Beschriftung, Untertitel und Wert des ursprünglichen Elements bei. In LabVIEW 8.2 werden alle im Element-Editor durchgeführten Änderungen übernommen und Daten des ursprünglichen Elements verworfen. Dieses Verhalten trifft nur auf benutzerdefinierte Elemente zu, es gilt nicht für Typdefinitionen.
- Bei der automatischen Skalierung der Achsen eines Graphen oder Diagramms werden ausgeblendete Plots nicht mit einbezogen. Wenn ausgeblendete Plots bei der automatischen Skalierung berücksichtigt werden sollen, sollten diese nicht ausgeblendet sondern transparent dargestellt werden. Klicken Sie zur Änderung der Plot-Farbe mit der rechten Maustaste auf die Plot-Legende und wählen Sie **Farbe** aus dem Kontextmenü.
- **(Windows)** Optionsfelder und Kontrollkästchen verhalten sich in LabVIEW genauso wie in Windows. Optionsfelder und Kontrollkästchen können über die Leertaste aktiviert und deaktiviert werden. Dialogfeldschaltflächen können über die <Enter>-Taste im alphanumerischen Block, die <Enter>-Taste im Ziffernblock oder die Leertaste bedient werden. Das Drücken der Taste <T> oder <F> hat keinerlei Auswirkung auf Dialogfeldschaltflächen.

- LabVIEW 8.2 weist neue Token für eine fortgeschrittene Bearbeitung der Zeitstempel-Bedienelemente auf. Klicken Sie zur Anzeige des Dialogfeldes **Zeitstempel-Eigenschaften**: mit der rechten Maustaste auf das Element und wählen Sie aus dem Kontextmenü die Option **Eigenschaften**. Wählen Sie auf der Seite **Format und Genauigkeit** den Punkt **Fortgeschrittener Bearbeitungsmodus**, um die Liste **Formatierungs-codes für absolute Zeit** anzuzeigen. In LabVIEW 8.2 gibt es folgende neue Formatcodes:

Formatcode	Wert
<%^<>T>	Container für Weltzeit
<%z>	Unterschied zwischen lokaler Zeit und Weltzeit

## Verbesserungen an der Umgebung

In LabVIEW 8.2 gibt es folgende Verbesserungen an der LabVIEW-Umgebung:

### Automatisches Zwischenspeichern

Bei einem unerwarteten Absturz oder Neustart des Systems werden von allen ungespeicherten VIs, VI-Vorlagen, Bedienelementen und Bedienelement-Vorlagen (\*.vi-, \*.vit-, \*.ctl- und \*.ctt-Dateien) Sicherungskopien in einem temporären Ordner erstellt. LabVIEW legt keine Sicherungskopien von Projekten, Projektbibliotheken, XControls und LabVIEW-Klassen an (\*.lvproj-, \*.lvlib-, \*.xctl- und \*.lvclass-Dateien).

Bei einem unerwarteten Neustart oder Systemabsturz wird bei eingestellter Zwischenspeicherung beim nächsten Start von LabVIEW automatisch das Fenster **Wiederherzustellende Dateien auswählen** geöffnet. Wählen Sie die Dateien aus, die wiederhergestellt werden sollen und klicken Sie auf **Wiederherstellen**. Wenn keine Dateien wiederhergestellt werden sollen, heben Sie die Markierung aller Dateien auf und klicken Sie auf **Verwerfen**. Zum Verschieben aller ausgewählten Dateien in das Unterverzeichnis `LVAutoSave\archives` des Standarddatenverzeichnisses klicken Sie auf **Abbrechen**.

Wählen Sie **Werkzeuge»Optionen** und anschließend **Umgebung** aus der Liste **Kategorie**. Hier lässt sich die Option "Automatisch Sicherungskopien anlegen" aktivieren bzw. deaktivieren. Sie können auch festlegen, wie oft Sicherungskopien angefertigt werden sollen.

## Verbesserungen an Dialogfeldern

In LabVIEW 8.2 wurden folgende Verbesserungen an Dialogfeldern vorgenommen:

### Verbesserungen und Änderungen am Optionen-Dialogfeld

In LabVIEW 8.2 wurden die nachfolgenden Verbesserungen am Dialogfeld **Optionen** vorgenommen. Informationen zum Umgehen von Problemen bzgl. der alten Optionen finden Sie auf der Website von National Instruments [ni.com/info](http://ni.com/info) nach Eingabe des Infocodes `ex6rkc`.

- Die Seite **Leistung und Speicher** einschließlich der Optionen **Beim Start verfügbare Datenträgerkapazität überprüfen** und **Mehrere Threads ausführen** wurde entfernt.
- Auf der Seite **Frontpanel** wurden die Optionen **Die Funktionstasteinstellungen des Systems übergehen** und **Grafik beim Zeichnen glätten** entfernt.
- Auf der Seite **Blockdiagramm** wurde die Option **Verbindungshilfen anzeigen** entfernt.
- Auf der Seite **Pfade** wurde die Option **Bibliotheksverzeichnis** aus dem Pulldown-Menü entfernt.
- Auf der Seite **Umgebung** wurde die Option **Kurzmenüs verwenden** entfernt.
- Auf der Seite **Umgebung** ist die Option **Just-In-Time-Hilfe aktivieren** nicht mehr per Voreinstellung aktiviert.
- Auf der Seite **Blockdiagramm** gilt die Option **Beim Einfügen SubVI-Namen anzeigen** auch für globale Variablen. Aktivieren Sie diese Option, wenn die Beschriftung einer globalen Variablen bei Einfügen dieser in das Blockdiagramm angezeigt werden soll.
- Auf der Seite **Webserver: Einsehbare VIs** wurde die Option **Momentan ausgewähltes VI in Sichtbares VI** geändert.

### Verbesserungen am LLB-Manager

In LabVIEW 8.2 gibt es folgende Verbesserungen am **LLB-Manager**:

- Die Schaltfläche **Durchsuchen** wird rechts neben dem Feld **Verzeichnis** angezeigt. Klicken Sie auf diese Schaltfläche, um die zu bearbeitende LLB auszuwählen.
- In der Symbolleiste wird eine **Löschen**-Schaltfläche angezeigt. Wählen Sie eine Datei aus der Liste **Dateien** und klicken Sie auf diese Schaltfläche, um die Datei aus der LLB zu entfernen.
- Durch Rechtsklick auf die Spalten kann in der **Dateien**-Liste nach Spalten sortiert werden.

## Seite “Datenbindung”

In LabVIEW 8.2 gibt es folgende Verbesserungen an der Registerkarte **Datenbindung** des Dialogfelds **Eigenschaften** für alle Frontpanel-Elemente:

- Die Textfelder **Ausgewähltes Netzwerkelement** und **Ausgewähltes Projektelement** wurden zum Textfeld **Pfad** zusammengefasst.
- Der Bereich **Modus** wurde in das Pulldown-Menü **Zugriffsart** geändert.
- Die Option **Blinken bei aktivem Alarm** wurde entfernt.
- Die Schaltfläche **Durchsuchen**, die nach Auswahl der Option **Engine für Umgebungsvariablen (NI-PSP)** aus dem Pulldown-Menü **Datenbindungsauswahl** aktiviert wird, startet anstelle des Dialogfelds **Netzwerkelement auswählen** den Dialog **Quellobjekt auswählen**. Das Dialogfeld **Netzwerkelement auswählen** wurde entfernt.
- Das Pulldown-Menü **Netzwerkveröffentlichte Quelle** wird im Dialogfeld **Quellobjekt auswählen** und nicht auf der Seite **Datenbindung** angezeigt.

## Verschiedene Verbesserungen an Dialogfeldern

In LabVIEW 8.2 gibt es folgende weitere Änderungen an Dialogfeldern:

- Das Fenster **VI-Metrik** enthält eine Zeile in der Tabelle **Metrik-Statistiken** zur Anzeige des Durchschnitts aller Werte in einer Spalte. Dateien eines bestimmten Ordners können durch Auswahl der Option **Dateien in diesem Ordner von Statistiken ausschließen** ausgelassen werden.
- Im Dialogfeld **VI-Einstellungen** wurde die Seite **Sicherheit** in **Schutz** geändert.
- Das Fenster **Polymorphes VI** enthält eine Schaltfläche **Lizenzwarnung anzeigen**, die angezeigt wird, wenn das polymorphe VI zu einer Projektbibliothek mit einer Evaluierungslizenz oder einer ungültigen Lizenz gehört. Beim Anklicken dieser Schaltfläche wird eine Warnung angezeigt. Klicken Sie für weitere Informationen zum Lizenzstatus auf die **Hilfe**-Schaltfläche in der Warnung.
- In den Dialogfeldern **Format-String bearbeiten** und **Such-String bearbeiten** können Sie im Pulldown-Menü **Ausgewählte Operation** SI-Schreibweise als Umwandlungstyp wählen.
- Im Dialogfeld **Vereinfachtes Bild exportieren** wurde die Option **In Zwischenablage speichern** in **In Zwischenablage exportieren** geändert.
- Im Dialogfeld **Gerätetreiber-VI erstellen** wurde die Option **Kein** aus dem Pulldown-Menü **Ausgabe-Datentyp** der Seite **Steuereinstellungen** entfernt.

- Im **Fehlercoden-Editor** wurde die Option **Aktueller Fehlercode mit Beschreibung** in die Optionen **Fehlercode** und **Fehlertext** geändert.

## Anzeige ausgeblendeter Bedien- und Anzeigeelemente

Durch Auswahl von **Bearbeiten»Ausgeblendete Bedien- und Anzeigeelemente anzeigen** können Sie alle ausgeblendeten Bedien- und Anzeigeelemente auf dem Frontpanel von benutzerdefinierten Elementen und globalen Variablen anzeigen. Diese Option steht nur im Menü **Bearbeiten** von benutzerdefinierten Elementen und globalen Variablen zur Verfügung.

Sie können auch Anzeige- und Bedienelemente für VIs anzeigen, die weder benutzerdefinierte Elemente noch globale Variablen sind, indem Sie das VI “ShowHidden Core” ausführen, das sich im Verzeichnis `labview\project\_ShowHidden` befindet.

## Verbesserungen am Fenster “Fehlerliste”

In der Liste **Fehlerhafte Elemente** im Fenster **Fehlerliste** werden fehlerhafte Elemente mit einem roten x neben dem Namen angezeigt. LabVIEW zeigt diese Elemente am Anfang der Liste **Fehlerhafte Elemente** an. Elemente, die Fehler in anderen Elementen verursachen, da Sie diese bearbeiten, werden mit einem Stift neben dem Namen angezeigt. Fehlerhafte Elemente, neben denen kein Symbol angezeigt wird, sind fehlerhaft, weil sie von einem fehlerhaften Element abhängig sind.

## Verbesserungen an “Suchen und Ersetzen”

In LabVIEW 8.2 gibt es folgende Verbesserungen an den Such- und Ersetzenvorgängen sowie dazugehörigen Funktionen.

- Bei Aktivierung der Option **Kopien ignorieren** in den **Optionen für Textsuche** werden Frontpanel-Kopien oder ablaufinvariante Frontpanel bei der Suche nach Text in einem VI ignoriert.
- Die Schaltfläche **Objekt wählen** im Dialogfeld **Suchen** wird mit dem Namen des ausgewählten Objekts angezeigt.
- Das Fenster **Suchergebnisse** wird nur angezeigt, wenn LabVIEW mindestens zwei Objekte findet. Wenn nur ein Objekt gefunden wird, markiert LabVIEW dieses auf dem Frontpanel oder im Blockdiagramm.
- Im Dialogfeld **Suchen** lassen sich nun auch Express-VIs, Knoten (wie die Funktion “Regulären Ausdruck suchen”) und Variablen durchsuchen.

## Handhabung geöffneter Fenster

Im Menü **Fenster** werden maximal zehn geöffnete Fenster angezeigt. Die geöffneten Fenster werden durch Auswahl des Menüpunkts **Fenster»Alle Fenster** oder mit der Tastenkombination <Strg + Shift + W> verwaltet. Im rechten Teil des Dialogfelds **Alle Fenster** finden Sie Schaltflächen zum Anzeigen oder Schließen eines Fensters oder Speichern eines zu einem Fenster gehörenden Objekts.

Fensterobjekte, die seit dem letzten Speichern geändert wurden, sind am Ende des Namens in der Spalte **Titel** mit einem Sternchen gekennzeichnet.

## Verbesserungen an den Paletten

In LabVIEW 8.2 gibt es folgende Verbesserungen an den Paletten:

- Im Dialogfeld **Favoriten ordnen** kann die Anordnung der Elemente in der Palettenkategorie **Favoriten** verändert werden.
- Das Dialogfeld **Menü-Dokumentation** wurde in **Palettendokumentation** geändert. Das Feld **Menübeschreibung** heißt nun **Palettenbeschreibung**.
- Bei Paletten für eine Projektbibliothek kann der Pfad zur Bibliothek angezeigt werden. Wählen Sie dazu **Werkzeuge»Fortgeschritten»Palette bearbeiten**, klicken Sie mit der rechten Maustaste auf eine Palette und wählen Sie **Palettendateipfad anzeigen** aus dem Kontextmenü. LabVIEW zeigt den eigentlichen Pfad der Palette und den Pfad der übergeordneten Bibliothek an, wenn die Palette zu einer Bibliothek gehört.
- Das Dialogfeld **Elemente- und Funktionen-Palette bearbeiten** enthält die Option **Vorschau der Änderungen vor dem Speichern anzeigen**. Aktivieren Sie die Option **Vorschau der Änderungen vor dem Speichern anzeigen** und klicken Sie auf die Schaltfläche **Änderungen speichern**, um das Dialogfeld **Vorschau der Palettenänderungen** anzuzeigen.
- Der Kontextmenüpunkt **VI ablegen** wurde in **VI anordnen** geändert. Dieser Kontextmenüpunkt wird angezeigt, wenn Sie mit der rechten Maustaste auf ein VI auf der **Funktionen**-Palette klicken.
- In LabVIEW 8.2 gibt es Paletten, die bis zur Installation von Add-Ons leer bleiben. Die Unterpalette **Reglerdesign und Simulation** der Palette **Elemente** ist leer, bis ein LabVIEW-Produkt für Reglerdesign und Simulation installiert wird.

## Speichern einer Hierarchie-Kopie

Sie können ein VI und seine SubVIs als Hierarchiekopie ohne Erstellung eines Quellcodepakets speichern. Klicken Sie dazu auf **Datei»Speichern unter** und wählen Sie die Option **Hierarchie an neue Position kopieren**, um das VI und seine Hierarchie an einer anderen Stelle zu speichern.

## Verschiedene Verbesserungen an der Umgebung

In LabVIEW 8.2 gibt es folgende weitere Verbesserungen am Frontpanel:

- In der Studenten- und der Evaluierungsversion von LabVIEW erscheint auf SubVIs in einem Unterpanel kein Wasserzeichen mehr. Es wird auch kein Wasserzeichen mehr in der LabVIEW Debug Deployment Version angezeigt.
- Wenn Sie keine gültige Lizenz für Ihre LabVIEW-Version besitzen, können Sie möglicherweise polymorphe VIs nicht bearbeiten.
- Sie können Elementreferenzen mit strikter Typdefinition erstellen. Klicken Sie dazu mit der rechten Maustaste auf das Element in der Typdefinition und wählen Sie **Erstellen»Referenz** aus dem Kontextmenü.
- Im Menü **Anzeigen** gibt es die Optionen **Bibliothek dieses VIs**, **XControl dieses VIs** und **Klasse dieses VIs**, mit deren Hilfe die Projektbibliothek, das XControl oder die LabVIEW-Klasse, der das gegenwärtig im **Projekt-Explorer** angezeigte VI angehört, markiert wird. Der Menüpunkt ändert sich in Abhängigkeit des Typs der Bibliothek, dem das VI angehört. Befindet sich die Bibliothek, das XControl oder die Klasse nicht in einem LabVIEW-Projekt, wird ein neues Fenster geöffnet, in dem nur die entsprechende Bibliothek, das XControl oder die Klasse enthalten ist, der das aktuelle VI angehört.
- In LabVIEW 8.0 wurde die Option **Hilfe»Fehler erklären** im Fenster **Erste Schritte** entfernt. In LabVIEW 8.2 wird dieser Menüpunkt wieder im Fenster **Erste Schritte** angezeigt.
- **(Mac OS)** Zum Ein- oder Ausblenden des Fensters **Kontexthilfe** wählen Sie **Hilfe»Kontexthilfe anzeigen** oder drücken Sie die Tasten <Command + Shift + H>. Mit <Command + H> wird die geöffnete LabVIEW-Anwendung ausgeblendet. Weitere Informationen zu benutzerdefinierten Tastenkombinationen finden Sie in der *LabVIEW-Hilfe*.

- Unter **Werkzeuge»Instrumentierung»Fortgeschrittene Entwicklung** finden Sie Optionen zur Entwicklung spezialisierter Applikationen unter Verwendung von Gerätetreibern. Die Optionen lauten:
  - **Treiberrichtlinien anzeigen** zeigt die Richtlinien des National Instrument Instrument Driver Networks ([ni.com/idnet](http://ni.com/idnet)) zur Entwicklung von Treibern an.
  - **Icon Art Glossary anzeigen** zeigt das Icon Art Glossary des National Instruments Instrument Driver Networks ([ni.com/idnet](http://ni.com/idnet)) an.
  - **Andere Ressourcen** zeigt die Entwicklungswerkzeuge und Ressourcen des National Instruments Instrument Driver Networks ([ni.com/idnet](http://ni.com/idnet)) an.

## Verbesserungen an neuen und geänderten VIs, Funktionen und Knoten

In LabVIEW 8.2 gibt es die folgenden neuen und veränderten VIs und Funktionen. Weitere Informationen zu VIs, Funktionen und Knoten finden Sie im Buch **VI- und Funktionsreferenz** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

### Neue VIs und Funktionen

LabVIEW 8.2 enthält folgende neue VIs und Funktionen:

#### Fortgeschrittene Datei-VIs

Auf der Palette **Dateifunktionen (Fortgeschritten)** gibt es folgende neue VIs:

- “Prüfen, ob Datei oder Ordner existiert”
- “Zwei Pfade vergleichen”
- “Pfad für temporäre Datei erzeugen”
- “Dateierweiterung ermitteln”
- “MD5-Prüfsummendatei”
- “Rekursive Dateiliste”

#### Funktionen für digitale Signalverläufe

Auf der Palette **Digitaler Signalverlauf** gibt es folgende neue Funktionen:

- “Signalverlauf erstellen”
- “Digitale Daten erstellen”
- “Signalverlaufskomponente lesen”
- “Digitale Datenkomponenten aufrufen”

## Mathematik-VIs

Die Palette **Mathematik** enthält im LabVIEW Full und Professional Development System folgende neue VIs:

- “Kronecker-Produkt”
- “Lyapunov-Gleichungen”

## .NET-VIs

Auf der Palette **.NET** gibt es folgende neue VIs:

- “In .NET Objekt”
- “.NET-Objekt in Variant”

## Skalierungs-VIs

Auf der Palette **Skalierung** gibt es folgende neue VIs:

- “RTD-Messwerte konvertieren”
- “DMS-Messwerte konvertieren”
- “Thermistor-Messwerte konvertieren”
- “Thermoelement-Messwerte konvertieren”

## VIs zur Signalverarbeitung

Die Palette **Signalverarbeitung** enthält in den LabVIEW Full und Professional Development Systemen folgende neue VIs:

- “Bohman-Fenster”
- “Dezimieren (kontinuierlich)”
- “Dezimieren (einmalig)”
- “FIR-Filter”
- “FIR-Filter mit Ausgangsbedingungen” VI
- “Gaußmoduliertes Sinusmuster”
- “Gaußscher Monopuls”
- “Inverse Chirp-Z-Transformation”
- “Modifiziertes Bartlett-Hann-Fenster”
- “Parzen-Fenster”
- “Periodisches Sinc-Muster”
- “Impulsfolge”
- “Rationales Neuabtasten”
- “Savitzky-Golay-Filter”
- “Savitzky-Golay-Filterkoeffizienten”

- “Dreiecksmuster”
- “Upsampling”
- “Welch-Fenster”

## TDM-Streaming-VIs und Funktionen

Auf der Palette **TDM-Streaming** gibt es folgende neue VIs und Funktionen:

- “TDMS-Dateimonitor”
- “TDMS: Schließen”
- “TDMS: Defragmentieren”
- “TDMS: Entleeren”
- “TDMS: Eigenschaft lesen”
- “TDMS: Inhalt auflisten”
- “TDMS: Öffnen”
- “TDMS: Lesen”
- “TDMS: Eigenschaft setzen”
- “TDMS: Schreiben”

Auf der Palette **Datenspeicherung** gibt es folgende neue VIs:

- “TDM in TDMS konvertieren”
- “TDMS in TDM konvertieren”

Weitere Informationen zum TDM-Streaming-Dateiformat finden Sie im Abschnitt [TDM-Streaming-Dateiformat](#) dieses Dokuments.

## Veränderte VIs, Funktionen und Knoten

Folgende VIs, Funktionen und Knoten haben sich in LabVIEW 8.2 geändert.

### DataSocket-Funktionen

Auf der Palette **DataSocket** gibt es folgende geänderte VIs:

- Mit Hilfe der Funktion “DataSocket: Schreiben” lässt sich die synchrone Benachrichtigung für NI-PSP-Datenobjekte aktivieren. Durch Anhängen von `?sync="true"` an das Ende der `psp`-URL bei aktivierter synchroner Benachrichtigung kann ein von Null verschiedener **ms timeout**-Wert angegeben werden. Die Funktion wartet, bis die Operation abgeschlossen oder das Timeout erreicht ist. Durch die Aktivierung synchroner Benachrichtigungen kann sich die Ausführungsgeschwindigkeit verlangsamen, insbesondere bei RT-Systemen.

- Die Funktion “DataSocket: Lesen” weist den Ausgang **Status** auf, über welchen Warnungen oder Fehler von einem PSP-Server oder FieldPoint-Controller gemeldet werden.

## Mathematik-VIs

Im LabVIEW Base Package enthält das VI “A x B” die neuen Instanzen “Vektor x A” und “Vektor x A (komplex)”.

Die Palette **Mathematik** enthält in den LabVIEW Full und Professional Development Systemen folgende geänderte VIs:

- Das VI “Exponentialanpassung” enthält den neuen Eingang **Gewicht**, der das Array mit Gewichtungen für die Beobachtungswerte (**X**, **Y**) angibt. Negative Werte, die oft durch Signalrauschen verursacht werden, führen nicht mehr zu fehlerhafter Exponentialanpassung.
- Das VI “Größter gemeinsamer Teiler von p(x) und q(x)” enthält den neuen Eingang **Algorithmus**, über den der Algorithmus bestimmt wird, mit dem der größte gemeinsame Teiler des Polynoms berechnet werden soll.
- Das VI Allgemeine Polynomanpassung enthält den neuen Eingang **Gewicht**, der das Array mit Gewichtungen für die Beobachtungswerte (**X**, **Y**) angibt.
- Die VIs “Histogramm”, “Histogramm (Punkt für Punkt)”, “Allgemeines Histogramm” und “Allgemeines Histogramm (Punkt für Punkt)” enthalten den Ausgang **Histogrammgraph**, in dem der Graph des Histogramms der Eingangsfolge **X** angezeigt wird. Die y-Achse stellt die Histogramm-Zählung dar und die x-Achse die Zentralwerte der Intervalle (Unterteilungen) des Histogramms.
- Das VI “Kleinste gemeinsames Vielfaches von p(x) und q(x)” enthält den neuen Eingang **Algorithmus**, über den der Algorithmus bestimmt wird, mit dem das kleinste gemeinsame Vielfache des Polynoms berechnet werden soll.
- Das VI “Partialbruchzerlegung” enthält den neuen Eingang **Option**, über den die Handhabung der Kofaktoren von **Zähler** und **Nenner** festgelegt wird.
- Das VI “Sylvester-Gleichungen” enthält den neuen Eingang **Matrixtyp**, über den der Typ der Eingänge **A** und **B** festgelegt wird. Dadurch kann **X** schneller berechnet werden.
- Der Eingang **Richtungscosinuswerte** der Array-Instanz des VIs “Rotation kartesischer 3D-Koordinaten (Richtung)” wurde in **Rotationsmatrix** geändert.

## Numerische Funktionen

Auf der Palette **Numerisch** gibt es folgende geänderte Funktionen:

- Die Funktion “Quotient und Rest” berechnet nun auch bei großen negativen 64-Bit-Teilern genaue Ergebnisse.
- Der Eingang **Beliebiges Objekt** der Funktionen “Bytes tauschen” und “Words tauschen” wurde in **Daten** geändert. Der Eingang arbeitet mit Strings, Tags, Pfaden, booleschen Werten, Nicht-Integern, Enums, Fehler-Clustern, Bildern und Matrizen sowie mit Arrays und Clustern dieser Datentypen. Des Weiteren können Daten unverändert weitergeleitet werden. Werden 16-, 32- oder 64-Bit-Integer mit dem Eingang verbunden, so tauscht die Funktion jedes Byte-Paar in jedem Word oder jedes Word-Paar in jedem Longword aus. Occurrences, Refnums und Variant-Daten können nicht mit dem Eingang verbunden werden.
- Bei x86-Plattformen funktioniert die Funktion “Mit Potenz von 2 multiplizieren” jetzt auch ordnungsgemäß, wenn die Eingänge **x** und **n** Fließkommazahlen sind.

## Protokoll-VIs und -Funktionen

Auf der Palette **Protokolle** gibt es folgende geänderte VIs und Funktionen:

- Die Funktion “UDP: Öffnen” und das VI “UDP: Multicast öffnen” enthalten den neuen Ausgang **Port**, der die Portnummer ausgibt, die von der Funktion verwendet wurde.
- Die Funktion “UDP: Öffnen” und das VI “UDP: Multicast öffnen” haben den neuen Eingang **Netzwerkadresse**, über den angegeben wird, an welcher Netzwerkadresse das Gerät auf Signale warten soll.
- Der Eingang **Adresse** der Funktion “TCP: Verbindung herstellen” wurde von optional auf empfohlen geändert.
- Das VI TCP: Listen enthält einen Eingang **Auflösen der Netzwerkadresse**, der angibt, ob die Funktion “IP nach String” an der Netzwerkadresse aufgerufen werden soll. Die Voreinstellung lautet TRUE.

## VI “Quadratische Programmierung”

Am VI “Quadratische Programmierung” gab es folgende Änderungen:

- Das VI “Quadratische Programmierung” ist polymorph mit folgenden Instanzen: “Quadratische Programmierung (IP)” und “Quadratische Programmierung (AS)”. Die Instanz “Quadratische Programmierung (IP)” hat die gleiche Funktion wie das VI “Quadratische Programmierung” in LabVIEW 8.0. Die Instanz “Quadratische Programmierung (AS)” bestimmt das Minimum mithilfe eines Active-Set-Algorithmus.

- Beide Instanzen des VIs “Quadratische Programmierung” enthalten den Eingang **Start**, der den Punkt in der  $n$ -Dimension festlegt, an welchem der Optimierungsprozess beginnen soll.
- Beide Instanzen des VIs “Quadratische Programmierung” enthalten den Eingang **Max. Zeit (s)**, der die maximale Zeit zwischen Anfang und Ende des Optimierungsprozesses festlegt.
- Die Instanz “Quadratische Programmierung (AS)” des VIs “Quadratische Programmierung” enthält den Eingang **Warmstart?**, über welchen festgelegt wird, ob die Nebenbedingungen der Optimierung ermittelt werden müssen oder nicht.

## VIs zur Signalverarbeitung

Die Palette **Signalverarbeitung** enthält in den LabVIEW Full und Professional Development Systemen folgende veränderte VIs:

- Im VI “Median-Filter” wurde der Eingang **Rang** aus LabVIEW 8.0 durch die Eingänge **linker Rang** und **rechter Rang** ersetzt. Dieses VI wendet einen Medianfilter auf die Eingangsfolge **X** an. Der Rang des Filters ist **rechter Rang**, wenn **rechter Rang** größer als 0 ist oder **linker Rang**, wenn **rechter Rang** kleiner als 0 ist.



**Hinweis** Wenn Sie ein in LabVIEW 8.0 oder Vorgängerversionen erstelltes VI mit einem Median-Filter-VI öffnen, bei welchem ein Wert mit dem Eingang **Rang** verbunden ist, wird dieser Wert in LabVIEW 8.2 für den Eingang **Linker Rang** verwendet.

- Der Eingang **f** in den VIs “Sinusschwingung”, “Sinusschwingung (Punkt für Punkt)”, “Dreieckschwingung”, “Dreieckschwingung (Punkt für Punkt)”, “Sägezahnschwingung”, “Sägezahnschwingung (Punkt für Punkt)”, “Rechteckschwingung”, “Rechteckschwingung (Punkt für Punkt)” und “Arbiträre Kurve” wurde in **Frequenz** umbenannt.
- Der Eingang **Anz. Seitenpunkte** des VIs “Savitzky-Golay-Filter (Punkt für Punkt)” wurde in **Seitenpunkte** geändert.
- Die Symbole der VIs für Punkt-für-Punkt-Analysen wurden geändert.
- Der Eingang **Fenster** der VIs “Symmetrisches Fenster” und “Skalierter Zeitbereichsfenster” kann nun auch auf “Modifiziertes Bartlett-Hann-Fenster”, “Bohman”, “Parzen” und “Welch” eingestellt werden.

## Audio-VIs (Linux)

Für die **Audio-VIs** wird der Open-Sound-System-Treiber (OSS-Treiber) benötigt. Informationen zum OSS-Urheberrecht finden Sie im Buch **Wichtige Informationen»Urheberrechte** auf der Registerkarte **Inhalt** der *LabVIEW-Hilfe*.



**Hinweis** Geräte werden anhand des Namens `/dev/dsp` oder `/dev/dspX` erkannt, wobei `X` eine Nummer zwischen 0 und 16 ist. LabVIEW versucht dann, jedes Gerät zur Aufnahme und Wiedergabe zu öffnen. Wenn Ihre Soundkarte nicht erkannt wird, vergewissern Sie sich, dass es auf dem lokalen System eine Gerätedatei mit dem Namen `/dev/dsp` oder `/dev/dspX` gibt und dass Sie Lese- und Schreibrechte für das Gerät haben. Sollten Sie das Gerät an eine andere Stelle als die voreingestellte verschoben haben, reicht auch ein symbolischer Link aus.

Die Audio-VIs unter Linux enthalten folgende Änderungen:

- Die VIs sind für Mono und Stereo geeignet.
- Audiodaten werden durch einen Signalverlauf dargestellt. Die Daten des **Y**-Arrays können in Form von vorzeichenlosen 8-Bit-, vorzeichenbehafteten 16-Bit- oder vorzeichenbehafteten 32-Bit-Integern sowie mit Datentypen einfacher und doppelter Genauigkeit dargestellt werden. Jeder Signalverlauf steht für einen Kanal.
- Die Audiodaten werden durch Puls-Code-Modulation digitalisiert.
- Die VIs sind zur fortlaufenden Audioausgabe in der Lage.
- Mit den VIs ist Streaming der Wave-Dateien möglich.
- Die VIs sind in Bezug auf die Fehlerprüfung verbessert worden.

## String-Funktionen

Auf der Palette **String** gibt es folgende geänderte Funktionen:

- Der Eingang **Format-String** der Funktion “In String suchen” arbeitet nun auch mit Werten in SI-Schreibweise.
- Die Funktion “Tabellenstring in Array” funktioniert fehlerfrei, wenn der Eingang **Array-Typ** einen komplexen numerischen Wert enthält.

## VISA-Funktionen

Wenn Sie mit der rechten Maustaste auf bestimmte VISA-Funktionen klicken und aus dem Kontextmenü **Erstellen»Konstante, Erstellen»Bedienelement** oder **Erstellen»Anzeigeelement** wählen, wird ein Ringobjekt angezeigt. Das Ringobjekt wird als Pulldown-Menü angezeigt, in denen der Benutzer eine Auswahl treffen kann. Alle von LabVIEW unterstützten Werte werden im Pulldown-Menü angezeigt.

Diese Änderung wirkt sich auf folgende Funktionen und Anschlüsse aus:

<b>Funktion</b>	<b>Anschlüsse</b>
VISA: Interrupt anfordern	Modus
VISA: Trigger freigeben	Protokoll
VISA: Hilfssignal anfordern	Bussignal
VISA: Ereignis deaktivieren	Ereignis-Typ
VISA: Ereignis verwerfen	Ereignis-Typ
VISA: Ereignis aktivieren	Ereignis-Typ
VISA: Ressource finden	Suchmodus
VISA: GPIB steuert ATN	Modus
VISA: GPIB steuert REN	Modus
VISA: Trigger zuweisen	Triggerquelle, Triggerziel
VISA: Öffnen	Zugriffsmodus
VISA: Trigger freigeben	Triggerquelle, Triggerziel
VISA: VXI-Befehl oder -Abfrage	Modus
VISA: Auf Ereignis warten	Ereignistyp (Eingang), Ereignistyp (Ausgang)

## **Knoten zum Aufruf externer Bibliotheken**

Der “Knoten zum Aufruf externer Bibliotheken” enthält Fehleranschlüsse. Das Dialogfeld der **Funktion zum Aufruf externer Bibliotheken** enthält folgende Änderungen:

- Das Dialogfeld enthält mehrere Seiten zur einfacheren Konfiguration von Parametern.
- Auf der Seite **Funktion** können Sie den Knoten auf programmatische Angabe des Bibliothekspfads einstellen, indem Sie die Option **Pfad im Blockdiagramm angeben** aktivieren.
- Wenn Sie String- oder Array-Parameter konfigurieren, können Sie auf der Seite **Parameter** mit Hilfe der Option **Mindestgröße** Speicher genau zuweisen.
- Auf der Seite **Callbacks** können Sie angeben, welche Aufrufe der Knoten vornehmen soll.

## Verschiedene Änderungen an VIs, Funktionen und Knoten

In LabVIEW 8.2 gibt es folgende Änderungen an VIs, Funktionen und Knoten:

- Die Optionen für **Mathematische Operation** im Konfigurationsdialogfeld des Express-VIs “Mathematik im Zeitbereich” wurden von **Differentiell** in **Ableitung (dX/dt)**, von **Differenz** in **Unterschied (dX)**, von **Integral** in **Integral (Sum[Xdt])** und von **Summierung** in **Summierung (Sum[X])** geändert.
- **(Windows)** Der Eingang **Refnum (Eingang)** der Instanz “Audiodatei-Info (Refnum)” des VIs “Angaben zur Audiodatei” wurde in **Audiodatei (Refnum)** geändert.
- **(Mac OS, Linux)** Die vom VI “Fußzeile für Report festlegen” erzeugten Fußzeilen haben eine geringfügig andere Größe als in früheren LabVIEW-Versionen. Das VI “Fußzeile für Report festlegen” enthält auch den neuen Eingang **HTML-Footer-Größe**.
- Das VI “Start zeitgesteuerter Strukturen synchronisieren” enthält einen Eingang **löschen**, über den alle zeitgesteuerten Strukturen entfernt werden und die gesamte Gruppe gelöscht wird, bevor die zeitgesteuerten Strukturen der angegebenen Gruppe hinzugefügt werden. Mithilfe dieses Eingangs werden alle zeitgesteuerten Strukturen entfernt, die keiner zeitgesteuerten Schleife entsprechen. Dieses VI enthält auch den Ausgang **Namen von zeitgesteuerten Strukturen (Ausgang)**, in welchem die Namen aller zeitgesteuerten Strukturen in der Gruppe ausgegeben werden, nachdem sie zur angegebenen synchronisierten Gruppe hinzugefügt wurden.
- **(Windows)** In LabVIEW 8.0 werden von der Funktion “Ordner anzeigen” dem Eingang **Muster** die Zeichen `. *` angehängt, wenn das **Muster** keine Erweiterung hat. In LabVIEW 8.2 ändert die Funktion den Eingang **Muster** nicht. Dem Dateinamen wird aber ein `.` angehängt, wenn der Dateiname keine Erweiterung hat, aber das **Muster**.
- Klicken Sie nach dem Einfügen der Funktion “Statische VI-Refnum” im Blockdiagramm doppelt auf die Funktion, um ein Dialogfeld zur Auswahl eines VIs anzuzeigen. In LabVIEW 8.0 führte der Doppelklick auf eine Funktion zur Anzeige des Fehlerdialogfelds wegen einer fehlenden Datei.
- Der Eingang **Format-String** der Funktionen “In Datei formatieren”, “In String formatieren”, “Aus Datei einlesen” und “In String suchen” kann jetzt auch offen gelassen werden.
- Wenn Sie ein VI mit einem Umgebungsvariablenknoten in einem Projekt öffnen, die dazugehörige Umgebungsvariable jedoch nicht im **Projekt-Explorer** zu finden ist, kommt es zu einem Fehler am Knoten. Das Gleiche gilt für alle zur fehlenden Umgebungsvariable gehörigen Frontpanel-Elemente. Das Problem ist Windows-spezifisch

und tritt nur beim Öffnen des VI-Knotens in einem Projekt auf. Wenn Sie das VI in der Haupt-Applikationsinstanz öffnen, erhalten Sie keine Meldung über fehlende Umgebungsvariablen. In Vorgängerversionen von LabVIEW wurde nicht angezeigt, dass der Umgebungsvariablenknoten die dazugehörigen Variablen im Projekt-Explorer nicht finden konnte.

- In LabVIEW 8.0.1 ist der Ausgang **Aktivierungsgrund** der “zeitgesteuerten Schleife” ein Ring. In LabVIEW 8.2 hat der Ausgang wie in LabVIEW 8.0 den Typ “Enum”.
- Die Namen kopierter Ausgänge folgender VIs und Funktionen wurden von **Kopie [Ausgang]** in **[Ausgang]** geändert. Diese Veränderung hat jedoch auf die Funktionsweise der VIs und Funktionen keinen Einfluss.
  - “Aufruf über Referenz”
  - “Abfrageausdruck ausführen”
  - “Datei-/Verzeichnisinformation”
  - “Objektinfo lesen”
  - “Eigenschaft lesen”
  - “Methodenknoten”
  - “Ordner anzeigen”
  - “Eigenschaftsknoten”
  - “String nach Tokens durchsuchen”
  - “Refnum nach ID”
  - “Eigenschaft setzen”

## Neue Eigenschaften, Methoden und Ereignisse

In LabVIEW 8.2 gibt es neue VI-Server-Klassen, -Eigenschaften, -Methoden und -Ereignisse. Eine Liste neuer Klassen, Eigenschaften, Methoden und Ereignisse finden Sie im Buch **Änderungen und Neuerungen in LabVIEW 8.2»Neue VI-Server-Klassen, -Eigenschaften, -Methoden und -Ereignisse** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

LabVIEW 8.2 enthält folgende neue VISA-Eigenschaften: “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” und “PXI/PCI Settings:D-Star Set”.

# Verbesserungen an LabVIEW-MathScript

Informationen zu LabVIEW MathScript finden Sie im Buch **Grundlagen» Formeln und Gleichungen** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

In LabVIEW 8.2 gibt es folgende Verbesserungen und Veränderungen an MathScript:

## Neue MathScript-Funktionen

LabVIEW 8.2 enthält folgende neue MathScript-Funktionen. Diese Funktionen können im **LabVIEW MathScript-Fenster** oder im MathScript-Knoten verwendet werden.

- Klasse “plots”: 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, ploty, polar, quiver, scatter, scatter3, shg, stem3, strips, surf, treepplot und waterfall.
- Klasse “dsp”: 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 und zp2tf.
- Klasse “support”: csvread, csvwrite, dlmread, dlmwrite, labviewroot, type, uiload und what.
- Klasse “string”: eval.
- Klasse “libraries”: loadlibrary, calllib, unloadlibrary, libisloaded und libfunctionsview. Sie können mithilfe dieser Funktionen DLLs vom **LabVIEW MathScript-Fenster** oder

MathScript-Knoten aufrufen. Beispiele für das Aufrufen von DLLs von MathScript finden Sie in `MathScript Shared Libraries.lvproj` im Verzeichnis `labview\examples\MathScript\MathScript Shared Libraries`.

## Verschiedene Neuerungen und Änderungen an MathScript

In LabVIEW 8.2 gibt es folgende weitere Veränderungen an MathScript:

- Die **LabVIEW-MathScript-Fenster** arbeitet schneller. Ebenso wird im MathScript-Knoten schneller kompiliert.
- Die LabVIEW-Runtime-Engine arbeitet mit MathScript. MathScript-Knoten können in allen ausführbaren Anwendungen und DLLs enthalten sein, die mit dem Application Builder erzeugt werden. Die LabVIEW-Runtime-Engine unterstützt derzeit jedoch noch nicht alle MathScript-Funktionen. Wenn ein Skript diese nicht unterstützten Funktionen enthält, müssen Sie das Skript vor dem Erzeugen einer ausführbaren Anwendung oder DLL möglicherweise ändern. Die vollständige Liste aller MathScript-Funktionen, die in LabVIEW nicht möglich sind, finden Sie in der *LabVIEW-Hilfe* unter *MathScript-Funktionen, die nicht von der LabVIEW-Runtime-Engine unterstützt werden*.
- Wenn Sie den MathScript-Befehl `help` aufrufen, wird Hilfe in einem HTML-Hilfefenster angezeigt. Wählen Sie zur Anzeige der Hilfe im **Ausgabefenster** von **LabVIEW MathScript** den Menüpunkt **Datei»MathScript-Einstellungen** und deaktivieren Sie die Option **HTML-Hilfe anzeigen?** Wenn Sie eine Funktion definieren, wird die Hilfe zu dieser Funktion immer im **Ausgabefenster** angezeigt.
- Der MathScript-Knoten verhält sich nicht wie andere Skriptknoten, wenn Sie mit dem Eingangsanschluss einen nicht unterstützten Datentyp verbinden. Bei einem MathScript-Knoten wird der Datentyp entweder in einen unterstützten Typ konvertiert oder es wird eine fehlerhafte Verbindung angezeigt. Wenn LabVIEW den Datentyp umwandelt, wird am Anschluss, an dem der Datentyp umgewandelt wurde, ein Typumwandlungspunkt angezeigt. Wenn der Datentyp des Eingangs angezeigt werden soll, klicken Sie (nachdem Sie den Eingang eines MathScript-Knotens verbunden haben) mit der rechten Maustaste auf den Eingangsanschluss und wählen Sie **Datentyp anzeigen** aus dem Kontextmenü aus.

- MathScript unterstützt alle Nicht-Unicode-Zeichen in Textstrings, nicht aber in Variablenamen. In Variablenamen können ausschließlich ASCII-Zeichen verwendet werden. So können Sie zum Beispiel  $\acute{a}$  in einem Textstring verwenden, aber nicht das folgende Skript im **LabVIEW-MathScript-Fenster** oder dem MathScript-Knoten aufrufen:

```
 $\acute{a}$ =rand(50, 1)
plot( $\acute{a}$ )
```

Sie können Daten in Pfade speichern, die andere Zeichen als Unicode-Zeichen enthalten. Wenn Sie LabVIEW in einem Verzeichnis installieren, dessen Pfad andere Nicht-Unicode-Zeichen enthält, funktionieren die MathScript-Funktionen fehlerfrei.

- Zusammengesetzte logische Ausdrücke in MathScript werden in LabVIEW mit Hilfe der Short-Circuit-Evaluation ausgewertet. Wenn Sie zum Beispiel den Befehl `if 0 == 0 || xyz(a) == 2` ausführen, führt LabVIEW `xyz(a)` nicht aus, da der erste Teil des Ausdrucks bereits TRUE ist. Wenn Sie den Befehl `if 0 ~= 0 && xyz(a) == 2` ausführen, führt LabVIEW `xyz(a)` nicht aus, da der erste Teil des Ausdrucks bereits FALSE ist.

Da LabVIEW 8.0 alle Komponenten von zusammengesetzten logischen Ausdrücken in MathScript evaluiert, unabhängig davon, ob diese Ausdrücke TRUE sind oder nicht, werden Skripte aus LabVIEW 8.0, die zusammengesetzte logische Ausdrücke enthalten, in LabVIEW 8.2 möglicherweise nicht wie erwartet ausgeführt.

Da LabVIEW 8.0 zum Beispiel `xyz(a)` im Befehl

```
if 0 == 0 || xyz(a) == 2
```

ausführt, kann das Ergebnis von `xyz(a)` zum Definieren einer Variable in einem Skript verwendet werden. In LabVIEW 8.2 wird das gleiche Skript auf andere Weise ausgeführt. Da LabVIEW 8.2 `xyz(a)` nicht im Befehl `if 0 == 0 || xyz(a) == 2` ausführt, kann das Ergebnis von `xyz(a)` in keinem Skript verwendet werden. Wenn LabVIEW die Short-Circuit-Evaluation nicht verwenden soll, müssen die zusammengesetzten logischen Ausdrücke aus dem Code entfernt werden.

- MathScript unterstützt die `nargin`- und `nargout`-Funktionen in benutzerdefinierten Funktionen.
- MathScript unterstützt das Schlüsselwort `return`. MathScript unterstützt ebenfalls das Schlüsselwort `end` für die Matrixindizierung.
- Die Funktionen `prod` und `sum` haben nun einen Eingang **b**, in dem die Dimension festgelegt werden kann, entlang der das Produkt oder die Summe berechnet wird.

## 3D-Bildelement

Weitere Informationen zum 3D-Bildelement finden Sie im Buch **Grundlagen»VIs für Audio und Grafik** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

Die VIs, Eigenschaften und Methoden der Palette **3D-Bildsteuerung** wandeln 3D-Objekte in eine 3D-Szene um, die Sie anzeigen und bearbeiten können. Sie können mehrere 3D-Objekte erzeugen und deren Größe, Form, Bewegung, Darstellung und Beziehung zu anderen Objekten in der Szene angeben.

Anhand folgender VIs, Eigenschaften und Methoden wird die Darstellung eines 3D-Objekts festgelegt:

- Mithilfe der **Objekt**-VIs werden 3D-Objekte erzeugt oder gesucht. Sie können 3D-Objekte auch mit den Szenenobjekt-Eigenschaften und Szenenobjekt-Methoden in eine Szene einfügen und Merkmale programmatisch festlegen.
- Mit den VIs zum **Laden von Dateien** können bestehende Modell- oder Scaledateien in eine 3D-Szene eingebunden werden.
- Die geometrische Form eines 3D-Objekts wird mithilfe der VIs **Geometrische Formen** sowie der Szenenobjekt-Eigenschaften und -Methoden bestimmt.
- Mit den **Transformations**-VIs werden Objekte in einer 3D-Szene angeordnet.

Mit den **Hilfs**-VIs werden in 3D-Szenen gängige Arbeitsschritte ausgeführt. Sie können ein separates Fenster für eine Szene konfigurieren, neue Clip-Ebenen innerhalb einer Szene erstellen, Lichtquellen hinzufügen, Texturen auf 3D-Objekte anwenden und LabVIEW-Farbwerte zur Anzeige in einem 3D-Bildelement umwandeln.

Mithilfe folgender Eigenschaften und Methoden kann eine 3D-Szene auch programmatisch konfiguriert werden.

- Mit den Szenenfenster-Eigenschaften können Sie die Szene in einem separaten Fenster rendern, Einstellungen zum Fenster vornehmen und die Bewegung der Kamera in der Szene festlegen.
- Mit den Szenen-Clipsebenen-Eigenschaften werden Ebenen in der Szene angegeben, in denen ein Objekt angezeigt oder abgeschnitten wird.
- Mit den Szenenlicht-Eigenschaften können Sie eine Lichtquelle für die Szene konfigurieren.
- Mit den Szenentextur-Eigenschaften und -Methoden werden 3D-Objekte texturiert.

Ein Beispiel zum Erstellen einer 3D-Szene mit einem 3D-Bildelement ist das VI "Solarsystem" im Verzeichnis `labview\examples\picture\3D Picture Control`.

## Objektorientierte Programmierung in LabVIEW

Weitere Informationen zu objektorientierter Programmierung in LabVIEW finden Sie im Buch **Grundlagen»Objektorientierte Programmierung in LabVIEW** auf der Registerkarte **Inhalt** der *LabVIEW-Hilfe*.

Die objektorientierte Programmierung in LabVIEW richtet sich nach objektorientierten Programmiersprachen wie C++ und Java, die mit Klassenstrukturen, Datenkapselung und Vererbung arbeiten. Mit objektorientierter Programmierung lassen sich Abschnitte von Applikationen erstellen, die einfacher zu bearbeiten und zu erweitern sind, ohne dass sich die Änderungen auf den Rest der Applikation auswirken. Die objektorientierte Programmierung kann auch für die Erstellung benutzerdefinierter Datentypen verwendet werden.

### Erstellen von LabVIEW-Klassen

In LabVIEW werden benutzerdefinierte Datentypen durch Erstellen von LabVIEW-Klassen erzeugt. LabVIEW-Klassen definieren mit einem Objekt zusammenhängende Daten und Methoden, die mit den Daten durchführbare Operationen definieren.

In LabVIEW sind die Daten einer Klasse privat, d. h., nur VIs der Klasse können auf diese Daten zugreifen. Die Daten einer Klasse werden im Privatdatenelement definiert. Beim Erstellen und Speichern einer LabVIEW-Klasse erstellt LabVIEW eine Klassenbibliotheksdatei (`*.lvclass`), in der ein neuer Datentyp definiert wird. Die Klassenbibliotheksdatei umfasst das Privatdatenelement und Informationen zu allen von Ihnen erstellten VIs der Klasse, z. B. eine Liste der VIs und verschiedene Eigenschaften der VIs. Die Klassenbibliothek ähnelt der Projektbibliothek (`.lvlib`). Allerdings definiert die Klassenbibliothek einen neuen Datentyp.

Klassen können wie folgt erstellt werden:

- Klicken Sie mit der rechten Maustaste im **Projekt-Explorer** auf **Mein Computer** und wählen Sie **Neu»Klasse** aus dem Kontextmenü.
- Wählen Sie **Datei»Neu** zur Anzeige des Dialogfelds **Neu** und anschließend die Option **Klasse** aus der Liste **Neu erstellen**.

## Definieren von Privatdatenelementen

LabVIEW erzeugt automatisch ein Privatdatenelement der Klasse, wenn Sie eine LabVIEW-Klasse erstellen.

Mithilfe des Element-Editors können Sie das Privatdatenelement der Klasse benutzerspezifisch anpassen. LabVIEW öffnet den Element-Editor, wenn Sie im **Projekt-Explorer** doppelt auf das Privatdatenelement klicken. Sie können Bedien- und Anzeigeelemente im **Cluster der Klassenprivatdaten** ablegen, um den Privatdatentyp einer LabVIEW-Klasse zu definieren. Die für die Elemente im **Cluster der Klassenprivatdaten** festgelegten Standardwerte gelten auch als Standardwerte für die Klasse.

## Erstellen von VIs einer Klasse

Die für eine LabVIEW-Klasse erstellten Methoden werden in den darin befindlichen VIs implementiert. Sie erstellen die VIs einer Klasse, um Operationen an den Privatdaten der Klasse durchführen zu können. Die VIs gehören immer zu der Klasse, in der sie erstellt wurden und werden im **Projekt-Explorer** unter dem Privatdatenelement der Klasse angezeigt. Die meisten Methoden können mithilfe eines einzigen VIs einer Klasse definiert werden. Sie können aber auch Methoden mithilfe mehrerer VIs der gesamten Klassenhierarchie erstellen.

Die VIs einer Klasse können von einem leeren VI, einer VI-Vorlage (die Fehlerbehandlung und Klassenobjekte umfasst) oder einem Vorgänger-VI erstellt werden. Klicken Sie mit der rechten Maustaste auf die Klasse und wählen Sie einen der folgenden Kontextmenüpunkte:

- **Neu»VI**—Öffnet ein leeres VI einer Klasse.
- **Neu»Dynamisches VI**—LabVIEW fügt dem neuen Member-VI die Cluster **Fehler (Eingang)** und **Fehler (Ausgang)**, eine Case-Struktur für die Fehlerbehandlung, die Eingangsklasse und die Ausgangsklasse hinzu.
- **Neu»Überschreib-VI**—Erstellt ein VI auf Grundlage eines übergeordneten.

## Weitergabe einer LabVIEW-Klasse an andere Benutzer

Von Ihnen entwickelte LabVIEW-Klassen können an andere Entwickler und Anwender weitergegeben werden. Es gibt verschiedene Möglichkeiten für die Weitergabe von Klassen, aus denen Sie die für Sie zutreffende auswählen sollten. Sie können mithilfe des Application Builders ein Zip-Datei erstellen, um Klassen zu verteilen. Sie können auch vor der Verteilung die LabVIEW-Klasse sperren, um den Zugriff der Benutzer auf die Privatdaten und die VIs der Klasse zu beschränken. Das Sperren der Klasse kann auch dem Einfügen von Fehlern in die Applikation vorbeugen.

# Verbesserungen an LabVIEW-Projekten

In LabVIEW 8.2 gibt es folgende Verbesserungen an LabVIEW-Projekten und dazugehörigen Funktionen:

## Verbesserungen beim Application Builder

In LabVIEW 8.2 gibt es folgende Verbesserungen am Application Builder im LabVIEW Professional Development System:

- **Produktversion automatisch erhöhen**—Wenn Sie einen Installer mehrere Male erstellen, kann die Versionsnummer des Produkts für jede neue Version automatisch erhöht werden. Die Option **Produktversion automatisch erhöhen** wird auf der Seite **Produktinformationen** im Dialogfeld **Installer-Eigenschaften** angezeigt.
- **Festlegen der Datenträgergröße für das Speichern des Installers**—Beim Speichern von Installer-Komponenten können Sie die Datenträgergröße anpassen. Die Option **Speichermedium auswählen** wird auf der Seite **Fortgeschritten** des Dialogfelds **Installer-Eigenschaften** angezeigt. Das Pulldown-Menü mit Datenträgern enthält nun die Option **Benutzerdefiniert**. Mit Hilfe dieser Option kann eine beliebige Datenträgergröße im Textfeld **Datenträgergröße (MB)** eingegeben werden.
- **Mindestens Windows 2000**—Sie können angeben, dass für die Ausführung des Installers mindestens Microsoft Windows 2000 Service Pack 3 erforderlich ist. Die Option **Windows 2000 oder höher** wird im Abschnitt **Systemvoraussetzungen** der Seite **Fortgeschritten** im Dialogfeld **Installer-Eigenschaften** angezeigt.
- **Speichern von Installer-Komponenten**—Wenn Sie ein Installationsprogramm mehr als einmal erstellen und der Installer zusätzliche Installer oder Komponenten enthält, muss nicht bei jedem Installer-Build ein Speicherort für die zusätzlichen Komponenten festgelegt werden, wenn Sie “Speichern” aktivieren. Beim ersten Build des Installers werden Sie im Dialogfeld **Softwarepaket suchen** aufgefordert, das Softwarepaket, in dem die zusätzlichen Komponenten enthalten sind, zu suchen. Aktivieren Sie die Option **Komponenten dieses Softwarepakets speichern**, um Dateien des Softwarepakets dauerhaft auf dem lokalen System zu speichern. Beim nächsten Erstellen eines Installationsprogramms mit diesen Komponenten kopiert der Application Builder die Komponenten vom lokalen Speicherort und Sie werden nicht aufgefordert, eine CD einzulegen.
- **Dynamische VIs und Abhängigkeiten**—Wenn ein in den **Einstellungen der Quelldateien (Applikation)** als dynamisch festgelegtes VI in eine andere Datei als die erstellte Applikation einbezogen wird, verschiebt der Application Builder alle Abhängigkeiten des dynamischen VIs an den neuen Speicherort. Wenn mehrere dynamische oder

Haupt-VIs ein VI aufrufen und versuchen, es an zwei verschiedene Speicherorte zu verschieben, ordnet der Application Builder das betreffende VI samt SubVIs der erstellten Applikation zu. Wenn ein VI auf der Seite **Einstellungen der Quelldateien** mit dem Attribut **Nur einbeziehen, wenn verwendet** an einen neuen Speicherort verschoben werden soll, müssen Sie es als dynamisches VI kennzeichnen.

- **Verwendung von Projekt-Alias-Dateien mit DLLs**—Die Seite **Fortgeschritten** des Dialogfelds **DLL-Eigenschaften** enthält die Option **Standard-Alias-Datei des Projekts verwenden**, über welche die Projekt-Alias-Datei mit der DLL in Verbindung gebracht wird. Bei Deaktivierung dieser Option muss eine Alias-Datei im Textfeld **Alias-Datei im Projekt** angegeben werden.
- **Einbeziehen zusätzlicher LabVIEW-Header-Dateien in DLLs**—Die Seite **Fortgeschritten** des Dialogfelds **DLL-Eigenschaften** enthält die Option **Zusätzliche LabVIEW-Header-Dateien einschließen**, über welche zusätzliche LabVIEW-Header-Dateien kopiert werden, auf die von der während der Erstellung erzeugten Header-Datei verwiesen wird. Dadurch können Sie eine in LabVIEW erzeugte DLL in C oder anderen Programmiersprachen verwenden, bei denen solche Header-Dateien benötigt werden.
- Die Schaltfläche **Neues Zielverzeichnis** auf der Seite **Quellcodepaket-Einstellungen** des Dialogfelds **Quellcodepaket** wurde in **Hinzufügen** geändert. Des Weiteren wurden alle Optionen bzgl. des Ausschließens von Dateien von einem Quellcodepaket auf die Seite **Zusätzliche Ausschließungen** verschoben.
- Die Schaltfläche **Neues Zielverzeichnis** auf der Seite **Zielverzeichnisse** der Dialogfelder **Eigenschaften für Applikation** und **DLL-Eigenschaften** wurde in **Hinzufügen** geändert.
- Applikationen und Quellcodepakete können ohne vorheriges Speichern der VIs erstellt werden.
- Wenn Sie im Application Builder erstellten ZIP-Dateien nicht die Erweiterung `.zip` anhängen, wird diese Einstellung in LabVIEW übernommen.

## Neue Dialogfeldseiten

In LabVIEW 8.2 wurden dem Dialogfeld **Application Builder** folgende Seiten hinzugefügt:

- Dialogfeld **Installer-Eigenschaften**
  - **Dialogfeldinformationen**—Auf dieser Seite wird die Benutzeroberfläche für den Installer entworfen. Sie können die Sprache für den Text festlegen, benutzerdefinierte Readme- und Lizenzdateien anzeigen sowie einen Begrüßungstitel und eine Begrüßungsnachricht erstellen. Diese Seite ersetzt den Abschnitt **Dialogfeld-**

**informationen** auf der Seite **Produktinformation** in LabVIEW 8.0.



**Hinweis** Die Option **Benutzerdefinierten Lizenzvertrag einschließen** der Seite **Dialogfeldinformationen** ersetzt die Option **Lizenzdatei** der Seite **Produktinformation**.

- **Hardware-Konfiguration**—Mit Hilfe dieser Seite lässt sich die Quelle für die Informationen zur Hardware-Konfiguration festlegen, die im Installer enthalten sein sollen. Der neue Abschnitt **Importmodus** enthält mehr Optionen für den Import von Gerätekonfigurationsdateien aus dem Measurement & Automation Explorer. Diese Seite ersetzt den Abschnitt **Hardware-Konfiguration** auf der Seite **Fortgeschritten** in LabVIEW 8.0.
- Dialogfelder **Eigenschaften für Applikation, DLL-Einstellungen** und **Quellcodepaket-Einstellungen**
  - **Zusätzliche Ausschließungen**— Auf dieser Seite können Sie Typdefinitionen, unbenutzte Instanzen von polymorphen VIs und nicht benötigte Dateien von Projektbibliotheken entfernen. Mit Hilfe dieser Einstellungen lässt sich die Größe des Builds verringern.

## Kopieren und Neuordnen von Build-Spezifikationen

Build-Spezifikationen können im **Projekt-Explorer** kopiert werden. Klicken Sie dazu mit der rechten Maustaste auf eine Build-Spezifikation und wählen Sie aus dem Kontextmenü die Option **Kopieren** aus.

Daneben kann der Inhalt von **Build-Spezifikationen** auch mit der Maus verschoben werden, um die Abfolge beim Build-Prozess zu verändern.

## Aktivieren des Application Builder (Windows)

Wenn Sie eine aktivierte Version des LabVIEW Base Package oder Full Development Systems haben, wählen Sie **Hilfe»Application Builder aktivieren**, um den Application Builder zu aktivieren. Die Lizenz wird beim Neustart von LabVIEW wirksam.

## Programmatisches Erstellen von Projektbibliotheken und Hinzufügen von Umgebungsvariablen

Mit dem VI “CreateOrAddLibrary” im Verzeichnis `labview\vi.lib\Utility\Variable` wird einem Projekt oder Elternobjekt (z. B. einem Zielsystem, Ordner oder einer anderen Bibliothek) programmatisch eine Bibliothek hinzugefügt.

Mit dem VI “AddSharedVariableToLibrary” im Verzeichnis `labview\vi.lib\Utility\Variable` können einer Bibliothek Umgebungsvariablen programmatisch hinzugefügt werden.

## Speichern von LabVIEW-Projekten und Projektvariablen für eine Vorgängerversion

Sie können LabVIEW-Projekte und Projektbibliotheken für LabVIEW 8.0 speichern. Bei LabVIEW-Projekten wählen Sie im **Projekt-Explorer** den Menüpunkt **Datei»Für vorige Version speichern** aus. Bei Projektbibliotheken klicken Sie die Datei im **Projekt-Explorer** mit der rechten Maustaste an und wählen aus dem Kontextmenü die Option **Für vorige Version speichern** aus oder öffnen die Projektbibliothek und wählen **Datei»Für vorige Version speichern**.

## Auswahl einer Applikationsinstanz

LabVIEW erzeugt für jedes Zielsystem in einem LabVIEW-Projekt eine Applikationsinstanz. Wenn Sie ein VI vom **Projekt-Explorer** aus öffnen, wird das VI in der Applikationsinstanz für das Zielsystem geöffnet. LabVIEW erstellt darüber hinaus eine Art Hauptapplikationsinstanz, in der geöffnete VIs enthalten sind, die nicht Teil eines Projekts sind, oder VIs, die nicht von einem Projekt aus geöffnet wurden. Die Applikationsinstanz, in der ein VI geöffnet werden soll, lässt sich in einem Kontextmenü auswählen. Klicken Sie mit der rechten Maustaste auf den Namen der aktuellen Instanz in der linken unteren Ecke des Frontpanels oder Blockdiagramms, um das Kontextmenü anzuzeigen und die gewünschte Instanz auszuwählen. Bei Auswahl einer neuen Applikationsinstanz wird das VI in der ausgewählten Applikationsinstanz erneut geöffnet. Das VI bleibt aber auch in der ursprünglichen Applikationsinstanz geöffnet.

## Verbesserungen an Umgebungsvariablen

In LabVIEW 8.2 gibt es folgende Verbesserungen an Umgebungsvariablen:

- Die Puffergröße für eine Netzwerk-Umgebungsvariable wird mit den Eigenschaften “Netzwerk:Puffergröße”, “Netzwerk:ElemGröße” und “Netzwerk:PunkteProSignalverlauf” berechnet.
  - Bei Netzwerk-Umgebungsvariablen, die mit Skalarzahlen arbeiten, wird die Anzahl der zu puffern den Werte mit der Eigenschaft “Netzwerk:Puffergröße” festgelegt.
  - Bei Netzwerk-Umgebungsvariablen, die mit Arrays und Strings arbeiten, werden die Eigenschaften “Netzwerk:Puffergröße” und “Netzwerk:ElemGröße” verwendet.

- Bei Netzwerk-Umgebungsvariablen, die mit Signalverläufen arbeiten, werden die Eigenschaften “Netzwerk:Puffergröße” und “Netzwerk:PunkteProSignalverlauf” verwendet.
- Bei Netzwerk-Umgebungsvariablen, die mit Arrays aus Signalverläufen arbeiten, werden die Eigenschaften “Netzwerk:Puffergröße”, “Netzwerk:ElemGröße” und “Netzwerk:PunkteProSignalverlauf” verwendet.
- Der Variableneingang des Knotens “Umgebungsvariable” wird benötigt, wenn der Knoten für das Schreiben von Daten konfiguriert ist.
- Die Seite **Variable** der **Eigenschaften für Umgebungsvariable** wurde wie folgt geändert:
  - Die Option **Benutzerdefiniert** des Pulldown-Menüs **Datentyp** wurde in **Von benutzerdef. Element** geändert.
  - Das Pulldown-Menü **Datentyp** enthält die neue Option **Variante**.
  - Die Elementgröße für den Netzwerkpuffer wird nicht mehr angezeigt.
  - Über die Optionen **nur lesen** oder **nur schreiben** für die **Zugriffsart** können Sie Umgebungsvariablen mit oder ohne Schreibschutz erstellen. Wenn die Umgebungsvariable zu einem Objekt gehört, das nur als Datenquelle fungieren kann, ist die Option **In Schreiben ändern** deaktiviert. Bei Objekten, die nur als Datensinke dienen können, ist **In Lesen ändern** deaktiviert.
  - In LabVIEW 8.0 müssen Sie mit der rechten Maustaste auf eine Umgebungsvariable klicken und aus dem Kontextmenü die Option **Zeitstempel einblenden** auswählen, um Zeitstempelangaben zur Umgebungsvariable zu erhalten. In LabVIEW 8.2 müssen Sie mit der rechten Maustaste auf die Umgebungsvariable klicken, **Eigenschaften** aus dem Kontextmenü wählen und die Option **Zeitstempel aktivieren** auf der Seite **Variable** aktivieren, um Zeitstempelangaben zu einer Einzelprozess-Umgebungsvariablen zu erhalten. Um die Zeit auszulesen und der Umgebungsvariablenknoten einen **Zeitstempel**-Ausgang hinzuzufügen, klicken Sie den Knoten mit der rechten Maustaste an und wählen Sie **Zeitstempel einblenden**. Wenn Sie eine in LabVIEW 8.0 erstellte Einzelprozess-Umgebungsvariable in LabVIEW 8.2 laden, wird der Zeitstempel per Voreinstellung aktiviert.
- In LabVIEW 8.0 werden Elemente, die nicht aus der Liste im Dialogfeld **Variable auswählen** ausgewählt werden können, ausgegraut. In LabVIEW 8.2 wird die Schaltfläche **OK** ausgegraut, wenn Sie ein ungültiges Element aus der Liste auswählen.

## Verschiedene Verbesserungen an Projekten

In LabVIEW 8.2 gibt es folgende Verbesserungen an Projekten:

- Wenn Sie in LabVIEW 8.0 eine Typdefinition in einer Projektbibliothek erstellen und den Zugriffsbereich dieser Typdefinition als privat festlegen, kann auch in einem VI, das sich nicht in der Projektbibliothek befindet, auf diese Typdefinition verwiesen werden. In LabVIEW 8.2 kann außerhalb der Bibliothek nicht mehr auf private Typdefinitionen verwiesen werden.
- Klicken Sie mit der rechten Maustaste auf ein XControl und wählen Sie **Neu»VI** aus dem Kontextmenü, um ein VI innerhalb dieses XControls zu erstellen.
- In LabVIEW 8.0 muss ein XControl-Leistungsmerkmal, eine Eigenschaft oder ein Methoden-VI gespeichert werden, nachdem während der Fehlersuche ein Haltepunkt gesetzt wurde. In LabVIEW 8.2 müssen Leistungsmerkmale, Eigenschaften und Methoden-VIs nicht mehr gespeichert werden, nachdem Haltepunkte gesetzt wurden.
- Wenn Sie ein Passwort für eine Bibliothek in einem LabVIEW-Projekt eingeben, das keine gültige Lizenz für die erworbene LabVIEW-Version aufweist, können keine Elemente in die Bibliothek oder aus dieser verschoben werden.
- Wenn eine Projektbibliothek passwortgeschützt ist und sich das Passwort nicht im entsprechenden Cache befindet, können Sie mit der rechten Maustaste auf die Projektbibliothek klicken und **Passwort eingeben** aus dem Kontextmenü wählen, um die Projektbibliothek freizugeben.

## Steuerung von VIs von mehreren Clients aus

Applikationen und VIs können gleichzeitig von mehreren Clients im Netzwerk gesteuert werden. Für die simultane Steuerung muss ein VI ablaufinvariant sein. Wählen Sie zur Festlegung eines VIs als ablaufinvariant **Datei»VI-Einstellungen** und anschließend **Ausführung** aus der Liste **Kategorie** und aktivieren Sie die Option **Ablaufinvariante Ausführung**. Für jede Client-Anfrage auf die Steuerung des Frontpanels wird eine Kopie des ablaufinvarianten VIs geöffnet. Mit Hilfe der Eigenschaft "Webserver: VI-Zugriffsliste" können Sie den Zugriff auf Kopien, die sich bereits im Speicher des Netzwerk-Frontpanels befinden, programmatisch beschränken.

## Importieren von Funktionen aus einer DLL

Sie können VIs für exportierte Funktionen aus Windows \*.dll-Dateien, Mac OS \*.framework-Dateien oder Linux \*.so-Dateien erzeugen und ändern. Wählen Sie **Werkzeuge»Importieren»Shared Library (.dll)** und folgen Sie den Eingabeaufforderungen zur Erstellung von Wrapper-VIs für DLLs. Sie müssen den Namen einer DLL und einer Header-Datei (\*.h) angeben.

Weitere Informationen zum Erstellen von Wrapper-VIs für DLLs finden Sie unter **Grundlagen»Aufruf textbasierten Programmcodes»Anleitung»Importieren von Funktionen aus einer DLL** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

## Vorlagen für Gerätetreiber

In LabVIEW 8.2 gibt es folgende neue Vorlagen für den Assistenten zur Erstellung neuer Gerätetreiber

- **Allgemein (registerbasiert)**—Wird für registerbasierte Geräte verwendet, für welche es keine klassenspezifische Vorlage gibt. Zu den gängigen registerbasierten Geräten gehören unter anderem VXI und PXI.
- **Spektrum-Analysator**—Hier werden einfache Parameter, wie z. B. Frequenzbereich und Kopplung des Durchlaufs eingestellt. Die Vorlage enthält auch Funktionen zur Konfiguration und Abfrage der Markierung.

## Verbesserungen an .NET und ActiveX (Windows)

Für .NET-Elemente gibt es keine UI-Thread-Abhängigkeit mehr (User Interface - Benutzeroberfläche). Es muss kein VI oder SubVI mehr im UI-Thread für .NET-Element-Operationen ausgeführt werden. Per Voreinstellung legt LabVIEW für jedes .NET-Element die Ausführung im entsprechenden Thread fest. In der Regel ist das der UI-Thread.

LabVIEW unterstützt nicht die Fehlersuche in ActiveX- und .NET-Callback-VIs. Wenn Sie Haltepunkte in ActiveX- und .NET-Callback-VIs verwenden, hält LabVIEW an diesen nicht an.

In LabVIEW 8.0 und Vorgängerversionen wurde von der .NET-Refnum-Sonde nur der Hexadezimalwert der Referenz angezeigt. In LabVIEW 8.2 zeigt die .NET-Refnum-Sonde das Ergebnis des Aufrufs der ToString()-Methode für ein Objekt an, das durch die .NET-Refnum, den Objekttypen, den Hash-Code des Objekts und den Hexadezimalwert der Referenz gekennzeichnet ist. Die Sonde lässt sich so konfigurieren, dass sie einen Haltepunkt setzt, wenn der Wert eine ungültige Refnum ist.

LabVIEW 8.2 enthält folgende Änderungen an.NET- und ActiveX-Menüs:

- Die Menüpunkte **Werkzeuge»NET & ActiveX**, **.NET-Elemente zur Palette hinzufügen** und **ActiveX-Elemente zur Palette hinzufügen** wurden in **Werkzeuge»Importieren»NET-Elemente zur Palette** und **Werkzeuge»Importieren»ActiveX-Elemente zur Palette** geändert.
- Der Menüpunkt **Werkzeuge»NET & ActiveX»ActiveX-Eigenschaften durchsuchen** wurde in **Anzeigen»ActiveX-Eigenschaftsbrowser** geändert.

## Neuerungen an der Beispielsuchmaschine

Der Beispiellordner **Zuletzt verwendete** wird per Voreinstellung auf der Registerkarte **Suchen** in der NI-Suchmaschine für Beispiele angezeigt. Sie können die Anzahl der maximal im Beispiellordner **Zuletzt verwendete** angezeigten Beispiele durch Klicken auf die Schaltfläche **Einstellungen** und Wahl der Registerkarte **Allgemein** in der NI-Suchmaschine festlegen.

In der *Hilfe zur NI-Suchmaschine für Beispiele* finden Sie weitere Informationen zu den Verbesserungen an der NI-Suchmaschine für Beispiele. Klicken Sie in der NI-Suchmaschine auf die Schaltfläche **Hilfe**, um die *Hilfe zur NI-Suchmaschine für Beispiele* anzuzeigen.

## Verbesserungen an der Versionsverwaltung

Weitere Informationen zur Versionsverwaltung in LabVIEW finden Sie auf der Registerkarte **Inhalt** der *LabVIEW-Hilfe* im Buch **Grundlagen»Organisieren und Verwalten von Projekten**.

Folgende Versionsverwaltungs-Provider von Drittanbietern wurden für die Integration in das LabVIEW 8.2 Professional Development System getestet:

- Seapine Surround SCM
- Borland StarTeam
- Telelogic Synergy
- PushOK (CVS- und SVN-Plugins)
- ionForge Evolution



**Hinweis** Derzeit arbeitet ionForge Evolution ab Version 2.8 mit LabVIEW.

Es gibt auch noch weitere Versionsverwaltungs-Provider von Drittanbietern, die kompatibel mit dem LabVIEW-Prozess für den graphischen Vergleich von VIs sind.

# Versionsverwaltungsoperationen an LabVIEW-Projektordnern

Sie können in einem LabVIEW-Projekt Operationen der Versionsverwaltung an einem Ordner mit Elementen durchführen. LabVIEW führt die Operation an allen geeigneten Elementen innerhalb der Hierarchie durch. Wenn Sie zum Beispiel Dateien zur Versionsverwaltung hinzufügen, sind nur Dateien im Ordner betroffen, die noch nicht zur Versionsverwaltung hinzugefügt wurden. Wenn Sie an einem Ordner Versionsverwaltungsoperationen durchführen, werden alle Konfigurationsoptionen zur Versionsverwaltung angezeigt, die für die Elemente des Ordners zur Verfügung stehen.



**Hinweis** Befindet sich eine Projektbibliothek in einem Ordner, gelten die Operationen der Versionsverwaltung nicht für den Inhalt der Bibliothek. Bei Dateibewegungen an Teilen der Projektbibliothek müssen die gewünschten Elemente manuell ausgewählt werden. Sie können aber Operationen der Versionsverwaltung an einem Ordner in einer Projektbibliothek durchführen.

## Ungespeicherte Dateien bei Versionsverwaltungsoperationen

Wenn Sie versuchen, Dateien mit ungespeicherten Änderungen einzuchecken, werden Sie über das Dialogfeld **Nicht gespeicherte Dateien** aufgefordert, die Änderungen zu speichern oder zu verwerfen. Dieses Dialogfeld wird angezeigt, wenn Sie im Dialogfeld **Versionsverwaltungsoperationen** auf die Schaltfläche **OK** klicken und von der Operation betroffene LabVIEW-Dateien ungespeicherte Änderungen haben. Außerdem erscheint das Dialogfeld, wenn Sie in einer Bibliothek oder Projektdatei **Werkzeuge»Versionsverwaltung»Unterschiede anzeigen** anklicken und die Datei ungespeicherte Änderungen hat.

## Verschiedene Verbesserungen an der Versionsverwaltung

In LabVIEW 8.2 gibt es folgende Verbesserungen an der Versionsverwaltung:

- In LabVIEW 8.0 steht ungespeicherten Dateien der Menüpunkt **Zu Versionsverwaltung hinzufügen** im Menü **Werkzeuge»Versionsverwaltung** zur Verfügung. In LabVIEW 8.2 steht der Menüpunkt **Zu Versionsverwaltung hinzufügen** ungespeicherten Dateien nicht zur Verfügung.
- Wählen Sie **Werkzeuge»Versionsverwaltung»Einstellungen zur Versionsverwaltung** und aktivieren Sie die Optionen **vi.lib ausschließen** und **instr.lib ausschließen**, um die Dateien der Verzeichnisse

`vi.lib` und `instr.lib` vom Hinzufügen von Dateien zur Versionsverwaltung auszuschließen. Wenn beim Hinzufügen von Dateien zur Versionsverwaltung Abhängigkeiten eingeschlossen werden, werden durch diese Optionen nicht notwendige Dateien von der Operation ausgeschlossen.

- Das Dialogfeld **Perforce-Versionshistorie** enthält die Optionen **Mit dieser Version synchronisieren** und **Änderungsliste beschreiben**. Klicken Sie für den Zugriff auf diese Optionen mit der rechten Maustaste auf eine Version in der **Historie**-Liste.
- Im Feld **Aktionen in der Änderungsliste** des Dialogfelds **Perforce-Versionshistorie** werden an der aktuellen Änderungsliste vorgenommene Aktionen angezeigt. Mögliche Aktionen sind Hinzufügen, Bearbeiten, Löschen und Integrieren. In diesem Feld wird auch der Dateityp angezeigt, an dem die Aktion vorgenommen wurde.
- **(Windows)** Wenn Sie den Core Perforce Windows Installer installieren, wird **Perforce SCM** im Pulldown-Menü **Provider für Versionsverwaltung** auf der Seite **Versionsverwaltung** des Dialogfelds **Optionen** angezeigt.

## TDM-Verbesserungen

In LabVIEW 8.2 gibt es folgende Verbesserungen an TDM-Dateien:

### TDM-Streaming-Dateiformat

LabVIEW 8.2 enthält das TDM-Streaming-Format (`*.tdms`) für das Speichern binärer Daten. Mit dem `*.tdms`-Dateiformat werden Dateien wesentlich schneller ausgelesen als mit dem `*.tdm`-Dateiformat älterer LabVIEW-Versionen. Das `*.tdms`-Format bietet auch eine benutzerfreundlichere Oberfläche für die Definition von Eigenschaften.

Weitere Informationen zu TDMS-VIs und -Funktionen finden Sie im Abschnitt [TDM-Streaming-VIs und Funktionen](#) dieses Dokuments.

### Benutzerdefinierte TDM- und TDMS-Eigenschaften

Sie können für Binärdateien mit Messwerten benutzerdefinierte Eigenschaften festlegen und DAQmx-Eigenschaften angeben. Konfigurieren Sie benutzerdefinierte Eigenschaften für `*.tdm`- und `*.tdms`-Dateien mithilfe der Express-VIs “Messwerte in Datei schreiben”, “Daten schreiben”, “Eigenschaften setzen” und “Eigenschaften lesen”.

Klicken Sie zum Festlegen der Eigenschaften in der Konfigurationsseite des Express-VIs “Messwerte in Datei schreiben” auf die Schaltfläche **Erweitert**. Öffnen Sie die Konfigurationsseite der Express-VIs “Daten

schreiben”, “Eigenschaften setzen” und “Eigenschaften lesen”, um diese Eigenschaften festzulegen.

## Importieren von Webdiensten (Windows)

In LabVIEW 8.2 können Webdienste einfach verwaltet werden. Sie können jeden beliebigen Webdienst in eine Projektbibliothek aus VIs umwandeln. Mithilfe dieser Bibliothek kann der Webdienst dann so programmiert werden, als befände er sich unabhängig auf dem lokalen Computer. Sie müssen aber eine gültige URL für eine Web Service Description Language (WSDL) angeben. WSDL ist eine Sprache im XML-Format, mit der der Webdienst und seine Funktion beschrieben werden. Über **Werkzeuge»Importieren»Webdienst** wird der Assistent gestartet, mit dem Sie durch das Importieren der Webdienst-Methoden und das Erstellen einer VI-Bibliothek geführt werden.



**Hinweis** Für den Assistenten zum Importieren von Webdiensten muss das .NET Framework ab 1.1 installiert sein.

Weitere Informationen zum Import von Webdiensten finden Sie im Buch **Grundlagen»ActiveX und .NET»Allgemeines»Importieren von Webdiensten** auf der Registerkarte **Inhalt** in der *LabVIEW-Hilfe*.

## Änderungen an externen Code-Funktionen

Der Datentyp `int32` wird an einigen Stellen im Speichermanager bei der Verwendung von externem Quellcode durch `size_t` ersetzt. Diese Änderung ist mit den vorhandenen DLLs kompatibel.

Weitere Informationen zu Funktionen der Speicherverwaltung finden Sie im Abschnitt *Funktionen zur Speicherverwaltung* in der *LabVIEW-Hilfe*.

National Instruments, NI, ni.com und LabVIEW sind Marken der Firma National Instruments Corporation. Nähere Informationen zu den Marken von National Instruments finden Sie im Abschnitt *Terms of Use* unter [ni.com/legal](http://ni.com/legal). MATLAB® ist eine eingetragene Marke der Firma The MathWorks, Inc. Sonstige hierin erwähnte Produkt- und Firmenbezeichnungen sind Marken oder Handelsnamen der jeweiligen Unternehmen. Nähere Informationen über Patente auf Produkte von National Instruments finden Sie unter **Hilfe»Patente** in Ihrer Software, in der Datei `patents.txt` auf Ihrer CD oder unter [ni.com/patents](http://ni.com/patents). Für Urheberrechtshinweise, Nutzungsbedingungen und rechtliche Hinweise betreffend Komponenten, die in USI genutzt werden (Xerces C++, ICU und HDF5), siehe `USICopyrights.chm`.