LabVIEW™ Application Builder User Guide

Version 7.0

The LabVIEW Application Builder is an add-on package you can use to create stand alone applications and shared libraries (DLLs) with LabVIEW. You can distribute these applications and shared libraries without the LabVIEW development system. Refer to the *National Instruments Software License Agreement* located on the LabVIEW Professional Development System and Application Builder distribution CDs for the licensing requirements for distributing applications and shared libraries.

This user guide contains instructions for installing the Application Builder, describes the system requirements for applications created with this version, and describes the changes introduced between previous versions and the current version. This user guide also describes caveats and recommendations to consider when you build a VI into an application or shared library.

You must use Application Builder 7.0 with LabVIEW 7.0.

For more information…

Refer to the *LabVIEW Help* for step-by-step instructions for building an application or shared library and for caveats to consider before building an application or shared library.

Contents

April 2003
323563A-01

NATIONAL INSTRUMENTS™

Required System Configuration

Applications and shared libraries you create with the Application Builder have the same approximate requirements as the LabVIEW development system. Memory requirements depend on the size of the application or shared library. Typically, applications and shared libraries require about the same amount of memory required to run VIs in the LabVIEW development system.

LabVIEW applications and shared libraries use a directory for storing temporary files. Some temporary files are large, so National Instruments recommends that you have several megabytes of disk space available for this temporary directory. You can view or change the temporary directory by selecting **Tools»Options** and selecting **Paths** from the top pull-down menu.

Note If the application or shared library aborts unexpectedly, it might leave files in the temporary directory. Remove old files from the temporary directory to free disk space.

Mac OS X

To build a shared library, you must have the August 2002 or later Developer Tools including Project Builder installed. Refer to the Apple Developer Connection at `developer.apple.com` to download the correct version.

Mac OS 9.*x* or Earlier

To build a shared library, you must have a Macintosh Programmer's Workshop (MPW) including ToolServer installed. MPW must have an MrC compiler located in its tools directory. The MPW installed by CodeWarrior does not have this compiler. Refer to the Apple Developer Connection at `developer.apple.com` to download the correct version.

To call a LabVIEW shared library from a CodeWarrior application, you must have enough memory allocated to that application. Because the application uses the LabVIEW Run-Time Engine that is about 4 MB itself, you should allocate at least 5 MB for the shared libraries you build. The default amount of memory is 384 KB, and the error that results when you do not allocate enough memory does not indicate the source of the problem.

UNIX

LabVIEW applications and shared libraries that display front panels require an X Window System server, such as OpenWindows, CDE, or X11R6. These applications and shared libraries do not require a specific graphical user interface (GUI) such as Motif or OpenLook because LabVIEW uses `Xlib` to create its own GUI.

(Sun) Application Builder for Sun runs on Solaris 2.6 or later.

(Linux) Application Builder for Linux runs on Linux for Intel *x*86 processors with kernel version 2.0.*x*, 2.2.*x*, or 2.4.*x* and GNU C run-time library 2.1.92 or later (`glibc`).

Installing the Application Builder

The default installation for the LabVIEW Professional Development System includes the Application Builder. Complete the following steps to install the Application Builder if you purchased it separately.

Windows

Complete the following steps to install the Application Builder.

Note Some virus detection programs interfere with the installer. Disable any automatic virus detection programs before you install. After installation, check your hard disk for viruses and enable any virus detection programs you disabled.

1. Insert the installation CD.
2. Run `setup.exe`.
3. Change the path, if necessary, to your LabVIEW directory, and click the **Install** button.
4. Verify the installation as described in the *Verifying the Installation of the Application Builder* section of this document.

Mac OS

Complete the following steps to install the Application Builder.

Note Some virus detection programs interfere with the installer. Disable any automatic virus detection programs before you install. After installation, check your hard disk for viruses and enable any virus detection programs you disabled.

1. Insert the installation CD.
2. Double-click the **LabVIEW AppLibs Installer** icon.
3. After you click the **Install** button, the installer prompts you to select a destination folder. Select the LabVIEW folder.
4. Verify the installation as described in the *Verifying the Installation of the Application Builder* section of this document.

UNIX

Complete the following steps to install the Application Builder on Linux or Sun. Root privileges are not necessary to install these libraries, but you must be able to write to the LabVIEW directory where you want to install these libraries.

1. Mount the installation CD.
2. Enter the following UNIX command appropriate to the operating system:
   - **(Linux)**

     ```
     cd /mnt/cdrom/linux
     ```

     where *cdrom* is the directory where you mounted the CD.
   - **(Solaris 2)**

     ```
     volcheck
     cd /cdrom/cdrom0/solaris2
     ```

     where *cdrom* is the directory where you mounted the CD.
3. Run the installer by entering the following command:

   ```
   ./INSTALL
   ```
4. Follow the instructions on the screen.
5. Verify the installation as described in the *Verifying the Installation of the Application Builder* section of this document.

Verifying the Installation of the Application Builder

Launch LabVIEW after you install the Application Builder and select **Tools»Build Application or Shared Library (DLL)**. Verify that the LabVIEW directory contains an `applibs` directory. If this directory is not present, refer to the *Installing the Application Builder* section of this document and reinstall the Application Builder.

If the libraries are installed correctly, the `examples` directory contains the `appbuild.llb`.

Caveats and Recommendations when Building Applications and Shared Libraries

The following list describes some of the caveats and recommendations to consider when you build a VI into an application or shared library that you want to distribute to users:

- Some VI Server properties and methods are not supported in the LabVIEW Run-Time Engine. Avoid using these properties and methods in the VI you want to build into an application or shared library.
- Make sure that the settings in the **VI Properties** dialog box are correct. For example, you might want to configure the VI to run when opened, or you might want to hide the buttons on the toolbar.
- Consider distributing documentation with the application or shared library so users have the information they need to use the application or shared library. Refer to Chapter 5, *Creating Documentation*, of the *LabVIEW Development Guidelines* manual for more information about developing user documentation.

Note Do *not* distribute the LabVIEW product documentation with the applications or shared libraries you create. The LabVIEW product documentation is copyrighted material.

- If the VI loads other VIs dynamically using the VI Server or Call By Reference Nodes, you must add the dynamically loaded VIs by clicking the **Add Dynamic VI** button on the **Source Files** tab of the **Build Application or Shared Library (DLL)** dialog box.
- If the VI loads other VIs dynamically using the VI Server or Call By Reference Nodes, make sure the application or shared library creates the paths for the VIs correctly. When you include the dynamically loaded VIs in the application or shared library, the paths to the VIs

change. For example, if you build `foo.vi` into an application, its path is *`C:\..\Application.exe`*`\foo.vi`, where *`C:\..\Application.exe`* represents the path to the application and its filename.

- If the VI uses the Current VI's Path function, make sure the function works as expected in the application or shared library. In an application or shared library, the Current VI's Path function returns the path to the VI in the application file, and treats the application file as a VI library. For example, if you build `foo.vi` into an application, the function returns a path of *`C:\..\Application.exe`*`\foo.vi`, where *`C:\..\Application.exe`* represents the path to the application and its filename.
- When you close all front panels in an application, the application stops. If the VI you build into the application contains code that executes after the last front panel closes, this code does not execute in the application. Avoid writing block diagram code that executes after the last front panel closes.
- If the VI uses custom run-time menus, make sure all the application menu items that the VI uses are available in the LabVIEW Run-Time Engine.
- Consider distributing a preferences file that contains your LabVIEW work environment settings with the application.
- Consider creating an **About** dialog box to display information about the application.
- You must distribute the LabVIEW Run-Time Engine with the application or shared library.

Refer to the *LabVIEW Help* for more information about each of these caveats and recommendations when building an application or shared library, including a complete list of VI Server properties and methods that are not supported in the LabVIEW Run-Time Engine.

Changes to the Application Builder

The following sections describe the changes introduced between each of the most recent versions of the Application Builder.

Refer to the *LabVIEW Application Builder Release Notes* for more information about changes between versions 5.0 and 6.0.

Changes Introduced between Versions 6.1 and 7.0

The following list describes the changes between versions 6.1 and 7.0:

- If an application or shared library requires serial or parallel port support or hardware configuration information, you can include that support and information in the installer by placing checkmarks in the appropriate checkboxes on the **Advanced Installer Settings** dialog box.
- If you have the LabVIEW Real-Time Module installed, you can include support for the LabVIEW Real-Time Module in the installer by placing a checkmark in the **LabVIEW RT Support** checkbox on the **Advanced Installer Settings** dialog box.
- If you want a program to run after the installation is complete but before the installer exits, place a checkmark in the **Wait until done** checkbox on the **Advanced Installer Settings** dialog box.
- If you use the Application:Command Line Arguments property to read the user-defined command-line arguments passed when an application or shared library launches, you can place a checkmark in the **Pass all command line arguments to application** checkbox on the **Application Settings** tab of the **Build Application or Shared Library (DLL)** dialog box to pass all arguments as user-defined arguments so you do not need to enter two hyphens before user-defined arguments in the command line.
- When you build an application or shared library from a VI that contains custom controls or type definitions, the Application Builder disconnects the controls and type definitions so you do not have to include the files with the application or shared library. If the VI contains a polymorphic VI, the Application Builder removes all unused instances from the polymorphic VI, which decreases the size of the application or shared library.
- Shared libraries you build in LabVIEW use ANSI types in the generated header instead of LabVIEW data types.

Changes Introduced between Versions 6.0 and 6.1

The following list describes the changes between versions 6.0 and 6.1:

- The **Destination** dialog box is two separate dialog boxes. The first dialog box, **Build Destination Settings**, accessible from the **Source Files** tab of the **Build Application or Shared Library (DLL)** dialog box, describes the file-by-file build destination settings. The second dialog box, **Installer File Settings**, accessible from the **Installer Settings** tab of the **Build Application or Shared Library (DLL)** dialog box, describes the file-by-file installer settings.

- After you build an application or shared library and save the build script file, you can click the arrow next to the **Load** button to display a list of the most recently used build script files.
- **(Windows)** LabVIEW generates Microsoft Installer (MSI) installers. You can select from 10 installation directories.
- **(Windows)** The **Media Size** and **Extra Space on first disk (kB)** buttons were removed from the **Installer Settings** tab of the **Build Application or Shared Library (DLL)** dialog box because the Microsoft Installer (MSI) does not support disk spanning.
- **(Windows)** The **Create Uninstaller** option was removed from the **Advanced Installer Settings** dialog box because the Application Builder always creates an uninstaller. Additionally, you can configure the installer to install only the parts of the LabVIEW Run-Time Engine that you want.
- To control an application remotely, you must include the NI License Manager utility in the installer by clicking the **Advanced** button in the **Installer Settings** tab of the **Build Application or Shared Library (DLL)** dialog box and placing a checkmark in the **Remote Panel License Support** checkbox. For more information about the NI License Manager, select **Start»Programs»National Instruments»NI License Manager**.
- **(Windows)** The number of languages for which you can create installers decreased from 12 to 4 (English, French, German, and Japanese).
- If you are developing an application or shared library, you must distribute any relevant user-defined or third-party error code text files with the application or shared library. If you use the Application Builder to create an installer for the application or shared library, the Application Builder prompts you to select the error code text files you want to distribute. The Application Builder also configures the installation subdirectory for the files.

  If you use a third-party utility to create an installer, locate the error code text files you want to distribute and configure the proper installation subdirectory for the files. User-defined error code text files are located in `labview\user.lib\errors`, and third-party files are located in `labview\projects\errors`. If you are creating an installer for an application, install the error code text files in a `user.lib\errors` subdirectory of the application directory. If you are creating an installer for a shared library, install the error code text files in the `National Instruments\shared\errors` directory. Each of these directories also can contain a language subdirectory with translated error codes.

323563A-01 Apr03