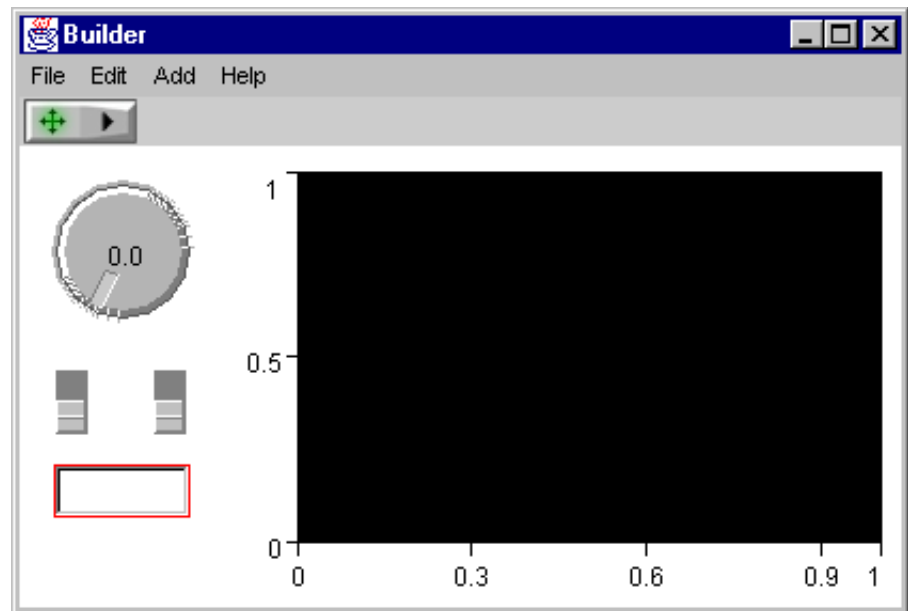


AppletBuilder User Guide



Overview of AppletBuilder

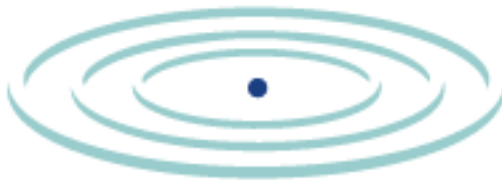
AppletBuilder has been designed to be easy to use. It features a "drag and drop then configure" interface. This means that you can insert a component, drag it into place, resize it, and then open its properties page to configure it.

AppletVIEW has familiar actions and familiar menus. You can save or open applet configuration files and you can add components from the Add menu. You can also switch between edit and run modes with the Run switch. The rest of this section will give you more details about the menu items available in AppletBuilder and methods you will use to create and edit your instrumentation applets.

The File Menu

New Starts a new, blank applet canvas. **Note:** You can only have one applet open at a time, so selecting New will remove any open applet in AppletBuilder that you have been working on.

Open Opens an applet configuration file and loads it into AppletBuilder. **Note:** You can only have one applet open at a time, so selecting Open will remove any open applet in AppletBuilder that you have been working on.



Save Saves your applet into a configuration file.

Save As ... Allows you to save your applet with a new name.

Read Server Allows you to "pull in" the configuration of a VI running in your LabVIEW system. See the next section Working In Applet Builder: Reading an Instrument from the Server, for further information.

HTML Opens a window that contains the <APPLET> tag you need to add to your HTML in order to serve your instrument.

Exit Quits AppletBuilder

The Edit Menu

Properties Opens a window where you can specify properties for a selected component. If no component is selected, choosing this item lets you specify properties for the overall applet. If a specific component is selected, choosing this item lets you specify the properties of the component. These vary among the various components. See the section below, Working In AppletBuilder: Component Editing, for specifics.

Cut Deletes a selected component. Note that there is no "Paste" option in this release of AppletBuilder. Also note that "Cut" cannot be undone.

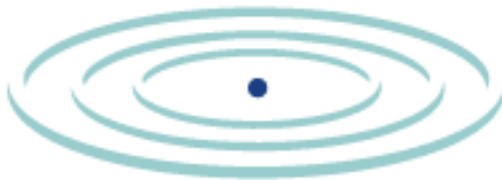
Align Allows you to align components if you have more than one selected. You can align components horizontally to the left-most edge, the average horizontal center, or the right-most edge. You can align components vertically to the top-most edge, the average vertical center, or the bottom-most edge.

Layering Allows you to change the stacking order of components. You can lower a component to the bottom of the stacking order or raise a component to the top of the stacking order.

The Add menu

Original Allows you to add a component. You can add the following types of components:

- Border
- Button
- Chart
- Checkbox
- ChoiceMenu



- Digital
- Graph
- Image
- Knob
- Label
- LED
- Slider
- Switch
- Textarea
- Textfield

The Help menu

About shows you a dialog box with the version of AppletVIEW you are using.

Help opens documentation in your default web browser.

The Run Switch

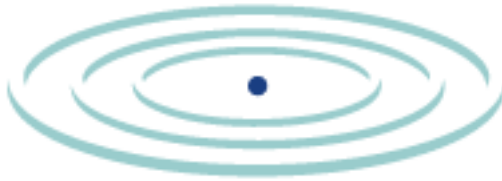
AppletBuilder has two modes: Edit and Run. They are controlled by the Run switch on the AppletBuilder toolbar. When the switch is green, you are in Edit mode and you can move, resize and change the properties of the components. If you click the Run switch, it will turn red. AppletBuilder will then be in Run mode, and you can see how the components behave to both mouse input and keyboard input at run-time.

Putting an instrument has a number of important implications because Run mode does more than simply allow you to see how the components behave. Run mode attempts to make a connection to the AppletVIEW Application Adapter running in LabVIEW. For a description of these implications see the next two sections, Working In AppletBuilder: Reading an Instrument from the Server and Working In AppletBuilder: AppletBuilder's Run Mode and the Application Adapter.

Working In AppletBuilder: Reading an Instrument from the Server

You do not need to recreate your instrument "by hand" in AppletBuilder. When you use AppletBuilder, you can connect to the AppletVIEW Application Adapter VI running in your local instrumentation system and "collect" or "pull in" the interface for any of your VIs.

To "pull in" the configuration of an instrument, first run the AppletVIEW Application Adapter in LabVIEW, and then run the instrument you want to "pull in". Then, in AppletBuilder, choose the Read Server option from the File menu. AppletBuilder will ask you to locate the Application Adapter on your network (normally this will be LOCALHOST since AppletVIEW is normally run on the same machine as your LabVIEW sys-



tem). AppletBuilder connects to the AppletVIEW application server and creates a list of running VIs. When you choose which VI you want to “pull in” to AppletBuilder, it will collect the configuration from the Application Adapter and build the front panel of your instrument.

Once you collect an interface, you can modify it in AppletBuilder. You can change the arrangement, color, size, position, etc. of the elements. When you are satisfied with your recreation of your instrument, save it as a configuration file, a Java Virtual Instrument (.jvi file).

At this point the process of recreation of your instrument is almost complete. If you prefer, you can rebuild your instrument “from scratch” in AppletBuilder by using the Add menu to add the components you want. Then you can resize, arrange, and configure them as you want.

Working In AppletBuilder: AppletBuilder's Run Mode and the Application Adapter

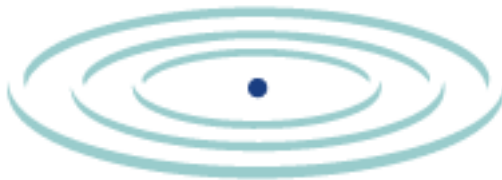
AppletBuilder's Run Mode does more than allow you to test how your instrument will react to user interaction. It attempts to connect to the AppletVIEW Application server running in LabVIEW. If AppletBuilder can connect to the Application Adapter, it tries to connect to its referenced VI. In other words, if you create an instrument by pulling in its configuration from the Application Adapter or open a previously created instrument and press AppletBuilder's Run button, AppletBuilder will try to connect to its companion instrument running in LabVIEW.

It is possible that AppletBuilder will fail to connect to the Application Adapter. If this occurs, It is possible that the Application Adapter is not running in LabVIEW, or the Application Adapter is already being used by another VI, or perhaps the referenced VI is not running in LabVIEW. The following list describes each situation and lets you know what to expect in each case.

- If the adapter is being used, AppletBuilder displays a message saying so when put in Run Mode.
- If the referenced VI is not open, AppletBuilder displays a message saying so when put in Run Mode.
- If the adapter is closed while AppletBuilder is connected, AppletBuilder displays a message saying so.
- If the adapter is not running at all, AppletBuilder displays no message for that.

Working In AppletBuilder: Component Editing

Whether you have “pulled in” the components of an instrument interface or added them using the Add menu, AppletBuilder lets you edit components using familiar editing methods. You can drag and resize compo-



nents with the mouse, you can change its alignment and layering with options in the Edit menu, and you can select a component and open its properties window.

To move a component, click the mouse in the middle of the object and you will see the "north-south-east-west" arrow cursor. Drag the mouse to reposition the element.

To resize a component, click near an edge of the component and you will see either the "north-south" or "east-west" cursor. Drag the mouse to resize the component.

To align components, select two or more and choose the appropriate horizontal or vertical alignment option from the Edit menu.

To change a component's stacking order, select the component and choose the appropriate stacking order from the Edit menu's Alignment option.

To open a component's property window, select the component and choose Properties from the Edit menu or you can:

- Windows95/NT: Right-click on the component you wish to edit.
- MacOS: (command)-click on the component you wish to edit.

Some of the properties you can customize for all components are:

- Colors (background, foreground, etc.)
- Size
- Position
- Default value
- Font size, style, and type

Most components have many other properties you can set that are specific to that component. Let's take a slider as an example:

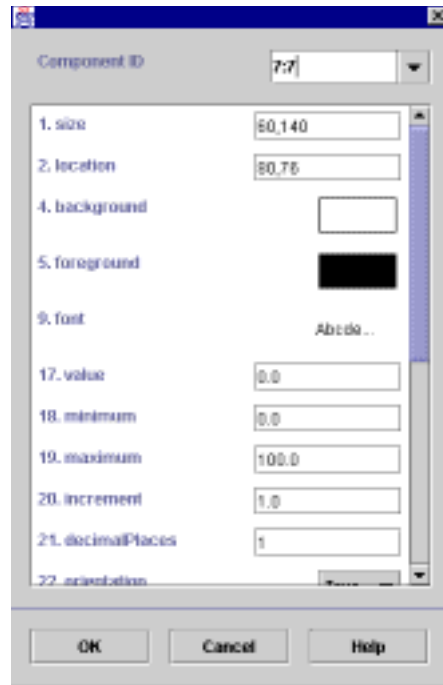
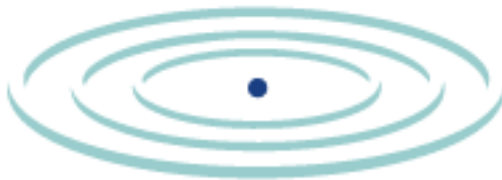
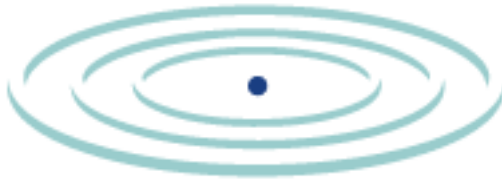


Figure 3. The Slider Property Window

Some properties define the component's appearance. For example, on a slider, you can set the orientation (True=vertical; False=horizontal), slider size, whether the value is visible, which side the value label should be on, etc.

Other properties define the component's behavior at run-time. For example, by default a slider sends its new value over the network only when the value changes after the mouse is released. On the other hand, you have the option of capturing the slider's movement while the mouse is pressed down and dragging it. You would do this by setting "sendDragEvents" to True. You can set how many values per second are sent over the network with "maxEventsPerSecond". The slider also has properties common to all components, such as size (in pixels) and location (in pixels, relative to the top left corner of the applet). Size, location, and color properties are useful for aligning and sizing components with precision for a neat layout. For a complete list of properties for each component (with the corresponding valid ranges, default values, and datatypes), see the section in Reference:Component Reference.



Working In AppletBuilder: Saving and Viewing Your Applet

When you are finished creating your applet in AppletBuilder, it is important to save it. When you choose Save from the File menu, you will be prompted with a dialog to save your applet. You can choose either to save your applet as JVI or VIML. JVI is a compact binary format, while VIML is an easier read XML format. You normally should save your applets into your web path so you can view them with your web browser. Your web path is the root directory of the AppletVIEW HTTP server. This directory is normally the /www folder in your AppletVIEW installation directory. It's important to note that the AppletVIEW server uses this directory by default.

Working In AppletBuilder: The applet .jvi File

The .jvi file is a binary file that contains all the configuration information for your applet. It can only be read and modified by AppletBuilder, or loaded with AppletVIEW's Java classes through a web browser. Note that this file is not a Java .class file; that is, no Java compilation has occurred. Instead, the Java class files ("AppletVIEW.jar") dynamically build your applet at run-time based on the .jvi file. This is why sometimes the .jvi file is called a configuration file. This gives the client browser a big advantage when browsing different applets, because the Java class files are only loaded once. The applet configuration files load very quickly because they are usually very small.

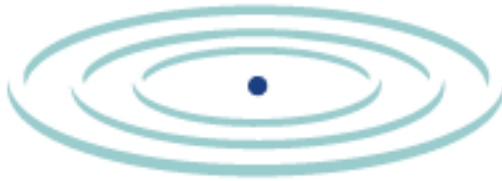
Working In AppletBuilder: Embedding Your Applet in an HTML Page

For a web browser to load an applet, it must load an HTML file that references your applet. This reference is created with an <APPLET> tag inside an HTML page. When you browse to this HTML page on a web server, the server sends both the Java class files and the .jvi configuration file and to create your instrument in the web page.

AppletBuilder writes the <APPLET> tag for you. If you save your applet and then choose the HTML option from the File menu, a window opens containing the applet tag for your applet. Copy and paste this tag into your own HTML file, save your HTML with your .jvi file, and your applet instrument is ready.

You can also add some optional parameter tags inside the applet tag to modify the applet's default behavior. For example, suppose you created an applet and saved it as "MyApplet.jvi". The <APPLET> tag created by AppletBuilder will look like this:

```
<APPLET
  CODE="com.nacimiento.appletview.applet"
  CODEBASE="/AppletVIEW/"
  ARCHIVE="AppletVIEW.jar,AppletVIEWui.jar"
  WIDTH="400" HEIGHT="400">
<PARAM NAME="ConfigFile" VALUE="MyApplet.jvi">
</APPLET>
```



The following table shows you the tag names and values inside the <APPLET> tag, what they mean, and if/how you should change them.

Table 1. <APPLET> tag name and value pairs

Tag Name	Optional?	User variable?	Default Value	Meaning
CODE	No	No	com.nacimiento.appletview.applet	The main class for the Java source code
ARCHIVE	No	No	AppletVIEW.jar, AppletVIEWui.jar	The Java archive (JAR) file containing the Java class files
CODE-BASE	Yes	Yes	/AppletVIEW/	Path to the JAR file, relative to Web root. If you leave this tag out, it looks for the JAR file in the same directory as the HTML file.
WIDTH	No	Yes	set by Applet Builder	Width in pixels of the applet
HEIGHT	No	Yes	set by Applet Builder	Height in pixels of the applet

In addition to the tag names inside the <APPLET> tag, you can also specify some parameter names and value pairs using the <PARAM> tag. The <PARAM> tags should always be inside the <APPLET></APPLET> tag pairs. The "ConfigFile" parameter is required; otherwise no applet components will load. The following table shows some of the available parameters for AppletVIEW applets:

Table 2. <PARAM> names and value pairs for AppletVIEW applets.

Param Name	Optional?	Value	Meaning
ConfigFile	No	set by Applet Builder	Specifies the config file (.jvi file) to load as an applet.
DataPort	Yes	Default value is 4747 Recommended value range is 4000 to 8000	If present, sets the communication between the applet and LabVIEW to occur over this port. If this parameter is left out, the data port defaults to 4747.
Debug	Yes	False (default): no debug statements True: debug statements to Java console	You can turn this on to debug your code. When set to true, everything your applet and its components do will be printed to the Java console on the browser. Note that this will reduce performance significantly.

Working In AppletBuilder: Examples

The AppletVIEW software comes with several example VIML files you can look at to get ideas for building your own interfaces. Look for these examples in:

docs/examples