

# LabVIEW VI Analyzer Toolkit

The National Instruments LabVIEW VI Analyzer Toolkit is a tool that inspects LabVIEW VIs and recommends modifications to the front panel, block diagram, VI properties and documentation that improve code performance, usability and maintainability. The toolkit includes over 60 different tests that can be run on any VI. The VI Analyzer makes specific recommendations for improving VIs, but does not force changes to be implemented. However, regular use of the toolkit will improve any LabVIEW developer's programming technique.

This tutorial is designed to introduce you to some of the functionality included in the VI Analyzer Toolkit. Topics include using the VI Analyzer user interface to interactively inspect code, customizing the VI Analyzer tests, reviewing the analysis results for a VI Analyzer task and generating reports to document test failures over time. Feel free to explore other parts of the LabVIEW environment as you follow along – LabVIEW has numerous features to offer.

Analyzing LabVIEW VIs .....	2
Introduction .....	2
Creating and Configuring a VI Analyzer Task .....	2
Viewing VI Analyzer Results .....	7
Exporting VI Analyzer Results .....	10
Loading VI Analyzer Configuration Files .....	11
Customizing Tests within VI Analyzer Tasks .....	12
Using the VI Analyzer VIs .....	14
VI Analyzer Palette.....	14
Automating VI Analyzer Tasks .....	16
Summary .....	18

# Analyzing LabVIEW VIs

## Introduction

The VI Analyzer Toolkit enables you to interactively and programmatically test VIs for performance and style issues, and recommends improvements to the block diagram code, front panel, VI properties and documentation. The toolkit includes over 60 unique tests that can be run on any VI, and while the VI Analyzer Toolkit does not implement its recommendations for you, regular use of the toolkit will enhance the programming ability of any LabVIEW developer.

The VI Analyzer provides two options for creating, configuring and running VI Analyzer tasks. You can interact with the VI Analyzer by clicking on **Tools » VI Analyzer » Analyze VIs**. This method will walk you through configuring your code analysis settings, running your tests and generating reports using an intuitive user interface. You can also use the toolkit VIs installed on the **Functions » Addons » VI Analyzer** palette to create VI Analyzer tasks programmatically.

Using the VI Analyzer, you can specify tests you want to run on specific VIs. Not all tests are relevant to every VI you include in a VI Analyzer task. For example, it might be unnecessary to run front panel tests related to fonts on subVIs whose front panels are never shown to the user. Once you have configured your VI Analyzer test settings for a particular task, you can save them in a configuration file (.cfg), which you can then use in subsequent VI Analyzer tasks. This enables you to create a template for testing VIs on a regular basis to satisfy your own custom criteria.

After the VI Analyzer runs a task, the **VI Analyzer Results Window** shows test failures and recommends improvements. You can select which changes you want to make and then re-run the tests. Additionally, the report can also be saved as either a text file or HTML, so that it can be reopened and shared with other developers or stored in a source code control utility.

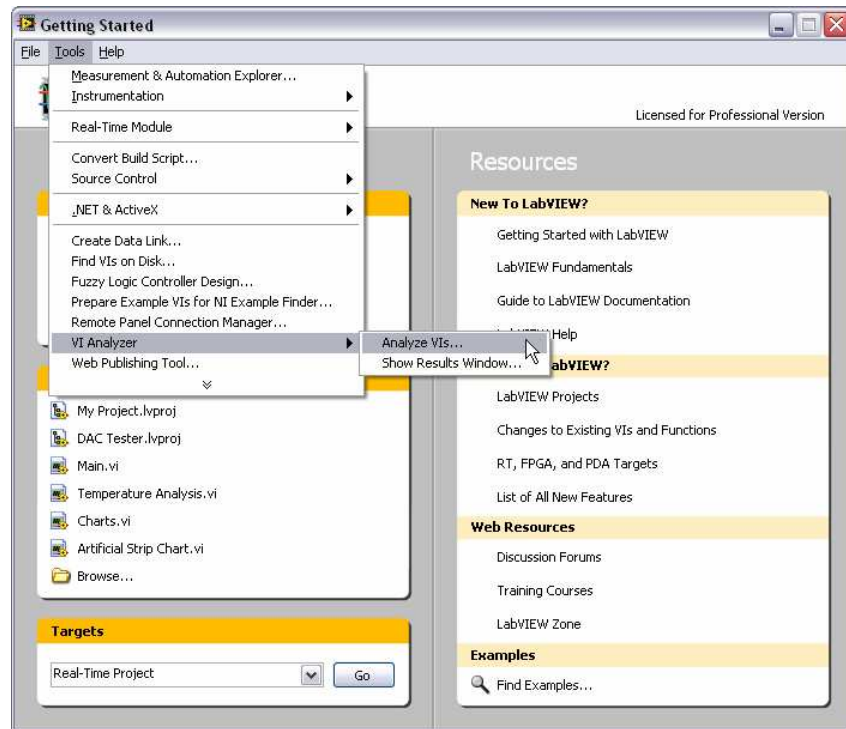
## Creating and Configuring a VI Analyzer Task

The VI Analyzer has a user interface that guides you through creating and configuring a VI Analyzer task. Using a series of dialog windows, you can select the VIs to be analyzed and choose the tests to be run.

In the following exercise, you will use the VI Analyzer user interface to create and configure a VI Analyzer task that tests a VI saved on disk.

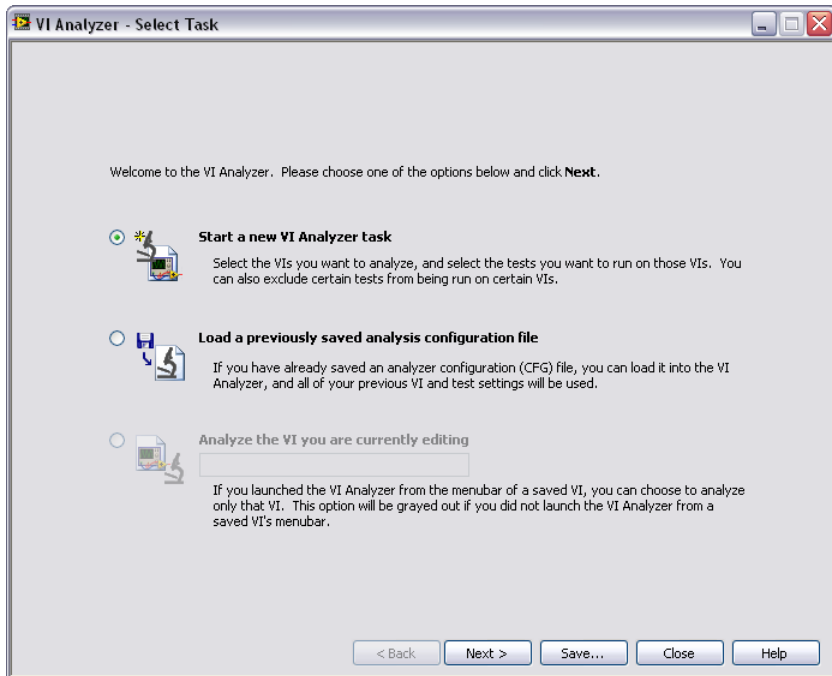
**You can complete the following exercise in approximately 10 minutes.**

1. From the **Tools** menu, select **VI Analyzer » Analyze VIs**. This will launch the VI Analyzer user interface, which will begin guiding you through selections for your code analysis settings.



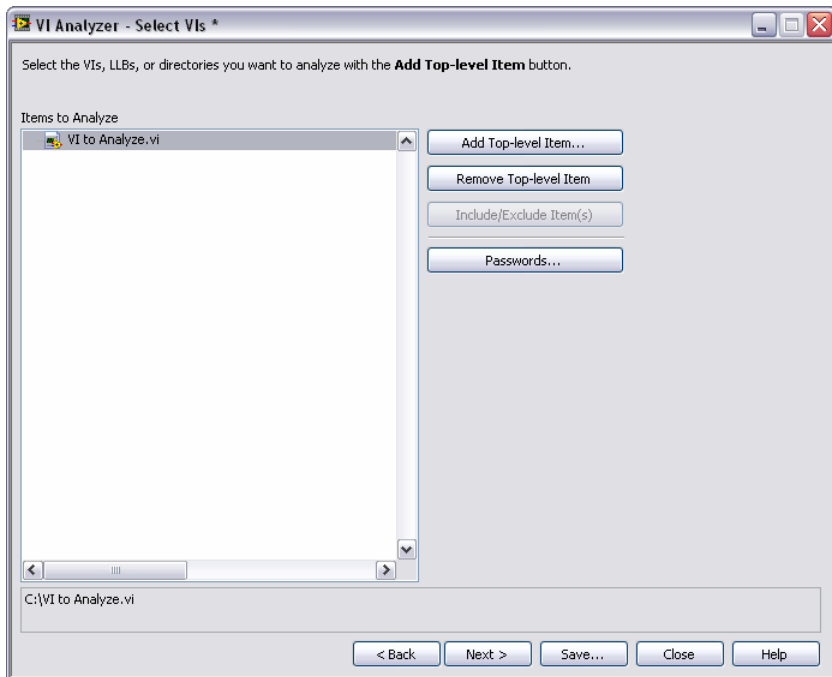
**Figure 1.1** – Launching the VI Analyzer

2. Once the **Select Task** dialog box appears, you will have three options: start a new VI Analyzer task, load a previous task, or analyze the VI from which you launched the VI Analyzer. Select **Start a new VI Analyzer task** and click the **Next** button.



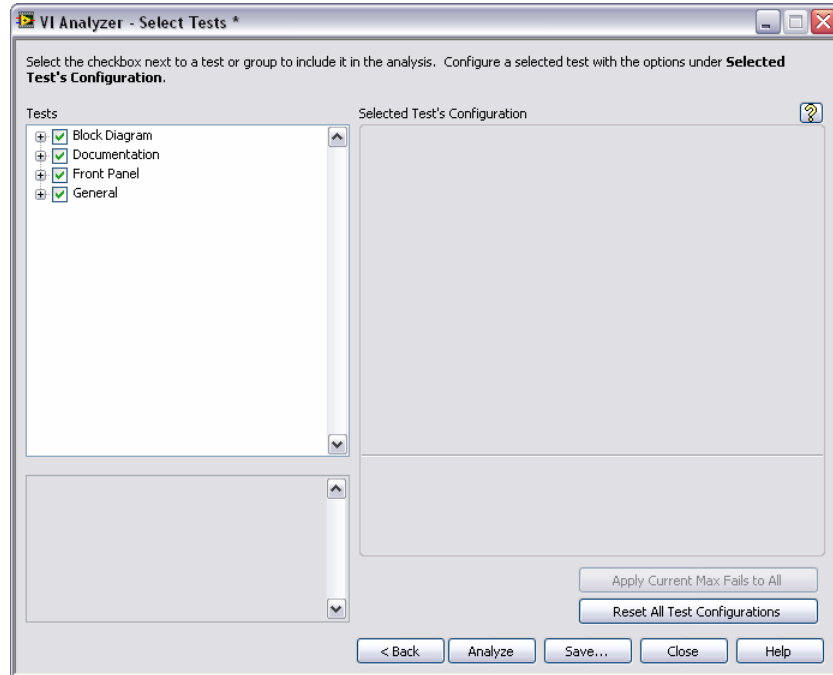
**Figure 1.2 – Select a VI Analyzer Task**

3. In the **Select VIs** dialog box shown in Figure 1.3, you will select VIs you wish to analyze, add any passwords for your VIs, and exclude VIs that you do not want to test. Click the **Add Top-level Item** button and open the **Exercises/VI Analyzer** folder on the Desktop. Select the LabVIEW VI named **VI to Analyze** and click **OK**. You will see the VI added in the **Items to Analyze** window.



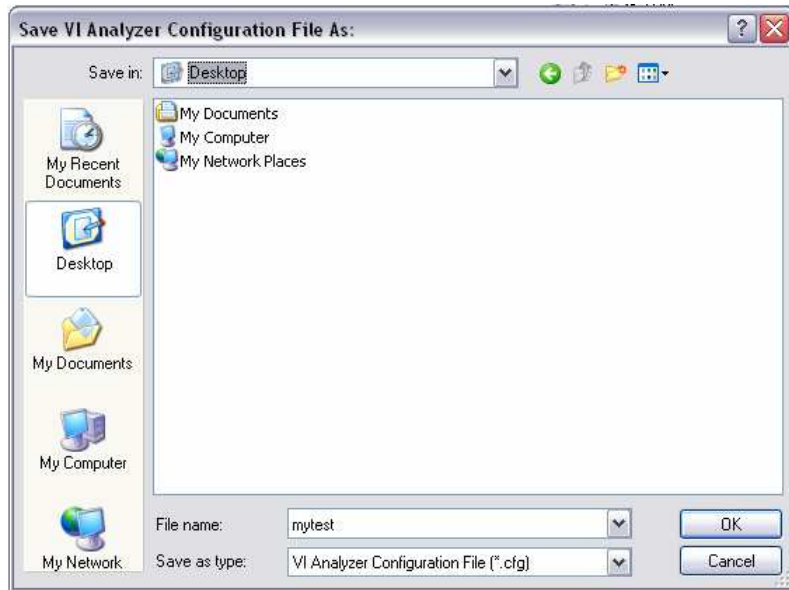
**Figure 1.3 – Select VIs to Analyze**

4. Click the **Next** button. The **Select Tests** dialog box appears. The **Select Tests** dialog box allows you to select the tests you want to run. The VI Analyzer user interface groups tests by category. Explore the various tests that are available, shown in Figure 1.4, but do not change the test selections in the **Tests** hierarchy.



**Figure 1.4** – Select Tests to Run

5. Click the **Save** button. In the file dialog box, seen in Figure 1.5, select the location where you want to save the configuration file. Name your file **mytest.cfg**. After you save a configuration file, you can load it into the VI Analyzer at a later time if you want to run an analysis task with the same settings. Refer to the **Loading VI Analyzer Configuration Files** section for more information about using configuration files.



**Figure 1.5** – Saving the VI Analyzer Configuration File

6. You can add a password to a configuration file if you want to distribute the file and prohibit users from loading the file, as shown in Figure 1.6. In the **Password Protect Configuration File?** dialog box, select the default **No** option to disable password protection for the configuration file. Click the **OK** button.



**Figure 1.6** – Password Protecting the Configuration File

7. Click the **Analyze** button. The **Analyzing VIs** dialog box displays the progress of the VI Analyzer task as VIs are loaded into memory and analyzed.



**Note** You will use the results of this VI Analyzer task in the next exercise, so **do not close** the VI Analyzer at this time.

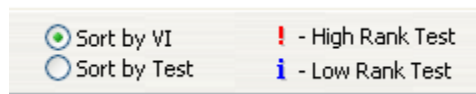
## Viewing VI Analyzer Results

The **VI Analyzer Results Window**, which appears after the analysis is complete, displays the results of the analysis task. The window allows you to view and organize results from all the tests run on your VIs.

Complete the following exercise to examine the VI Analyzer results from the previous exercise.

**You can complete the following exercise in approximately 10 minutes.**

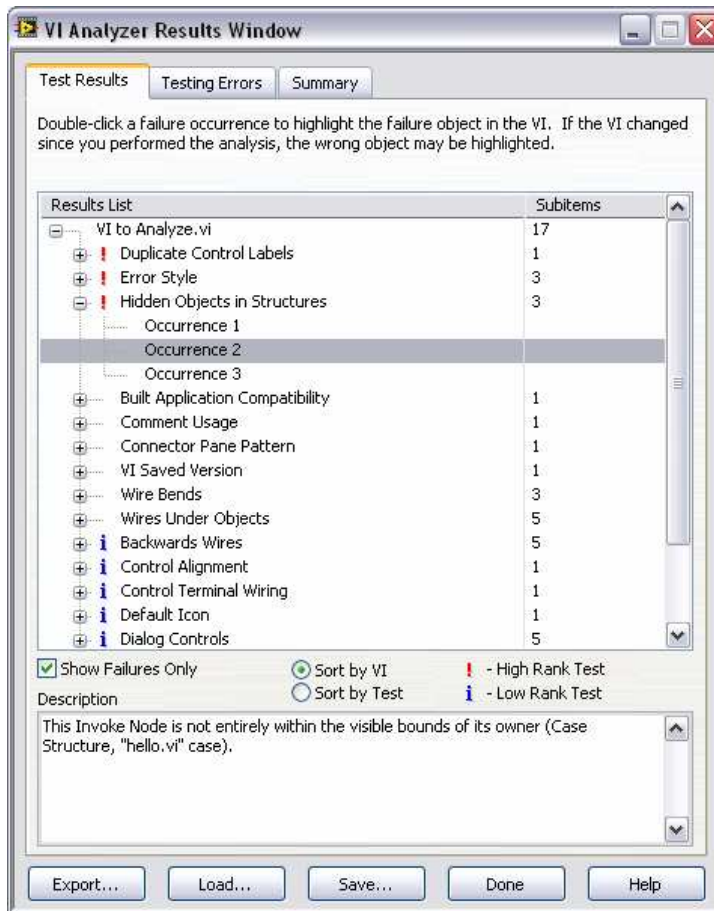
1. On the **Test Results** page, click the **Sort by Test** option to view the results by test. The **Results List** hierarchy displays the high-ranking tests first, marked with a red exclamation point, and displays the low-ranking tests last, marked with a blue i. The **VI Analyzer Results Window** includes a legend for the test ranking icons, shown below in Figure 1.7.



**Figure 1.7** – Sorting by VI or Test

High-ranking tests involve issues that have a significant impact on VI performance. Low-ranking tests involve minor style or cosmetic issues that do not significantly affect VI performance. Tests that do not have a ranking icon next to them fall into a middle-ranking area. The default ranking for each test matches National Instruments recommendations for style and performance issues.

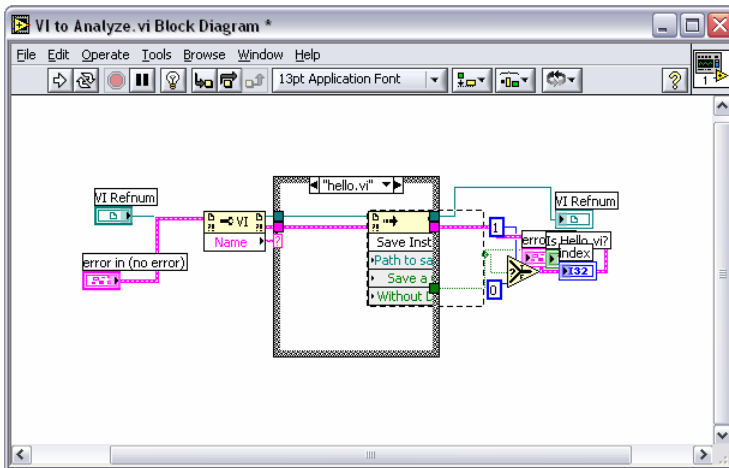
2. Expand the **Hidden Objects in Structures** item to view a list of failure occurrences for that particular test.
3. Click the **Occurrence 2** item. Notice the text in the text box describes why the VI failed the test, as shown in Figure 1.8.



**Figure 1.8** – Viewing the Test Results

According to the **Description** text box, the VI failed this test due to an invoke node on the block diagram that was not entirely visible within the bounds of a case structure.

4. Double-click **Occurrence 2** item in the **Hidden Objects in Structures** failure list. The block diagram for the VI opens and the VI Analyzer automatically highlights the test failure. As the **Description** text box displayed earlier indicated, an invoke node is partially obscured by the border of the **hello.vi** case, as shown in Figure 1.9. If the entire node had been hidden, this case would have appeared to be empty.



**Figure 1.9** – Viewing Problems in the Block Diagram

You can reposition the invoke node so that entire node is visible and the VI passes this test the next time you run the VI Analyzer task.

5. Click the **Testing Errors** tab of the **VI Analyzer Results Window**, as shown in Figure 1.8. If errors occur during the analysis, the **Testing Errors** page displays a list of VIs and tests that did not load, tests that did not run, and tests in which errors occurred. For example, if you attempt to run tests on a password-protected VI and do not provide a password, the **Errors List** hierarchy displays a **Test Error Out** error and the **Error Description** text box displays the reason.



**Figure 1.10** – Testing Errors Window

For this exercise, the **Testing Errors** tab does not list any tests because no errors occurred.

8. Click the **Summary** tab of the **VI Analyzer Results Window**. This page displays several types of summary results from the analysis, such as the number of tests that ran, the number that passed/failed, and the time it took to complete the VI Analyzer task.



**Note** You will use the results shown in the **Results Window** for this VI Analyzer task in the next exercise, so **do not close** the VI Analyzer window at this time.

**End of Exercise 1-2**

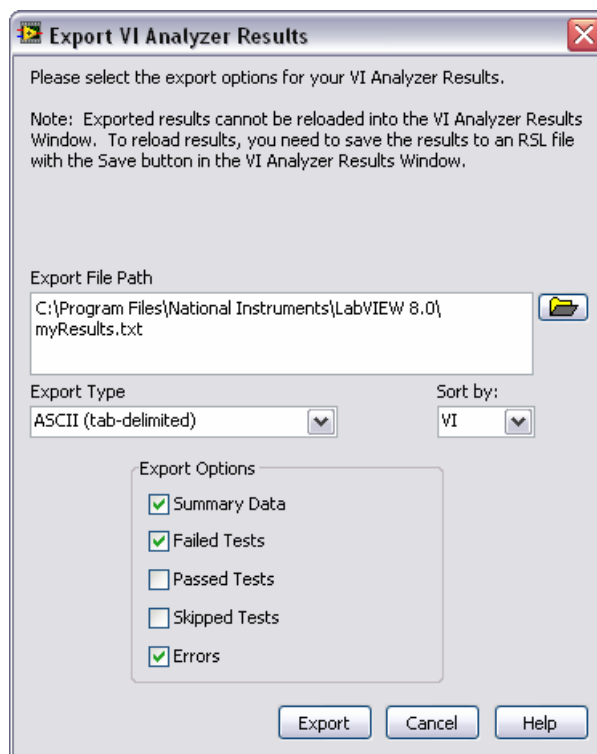
# Exporting VI Analyzer Results

The code analysis results of the VI Analyzer task can be saved as either an ASCII text file or as HTML so that you can reopen the report at any time. You can share the results with other developers or store the file in a source code control utility along with the VIs.

In the following exercise, you will export the contents of the **VI Analyzer Results Window** to either a text file or HTML file.

**You can complete the following exercise in approximately 5 minutes.**

1. Click the **Export** button. The **Export VI Analyzer Results** dialog box appears.



**Figure 1.11** – Exporting the VI Analyzer Results

2. Enter the path and filename for the results file in the **Export File Path control**. The default is a filename consisting of the current date and time, located in the LabVIEW default data directory.
3. From the **Export Type** list, select the file format you want. You can export the results either as a tab-delimited ASCII file, which is useful with spreadsheet applications, or as an HTML file, which is useful for viewing in Web browsers.
4. From the **Sort by** list, select whether you want the results in the file sorted by VI name or by test name.

5. Place checkmarks in the checkboxes next to the results you want to include in the exported file. By default, the **Summary Data**, **Failed Tests**, and **Errors** checkboxes contain checkmarks.
6. Click the **Export** button to generate the file.
7. In the VI Analyzer Results Window, click the **Done** button to close the window.
8. Navigate to the location of the exported results file on the computer and double-click on it. If you saved the results as text file, the file will automatically open in Notepad. If you saved the results in the HTML format, the file will automatically open in your default Web browser.
9. Click the **Yes** button to save the results in a VI Analyzer results file (.rsl) that you can view later in the **VI Analyzer Results Window**. A file dialog box appears. Select the path and filename you want for the results file.
10. Click the **No** button to close the VI Analyzer.

**End of Exercise 1-3**

## Loading VI Analyzer Configuration Files

You can create and edit VI Analyzer configuration files that contain the settings you selected for a VI Analyzer task. Using a configuration file, you can customize your test settings and apply the customized tests to the same VI multiple times as you make improvements, or to multiple VIs to enforce consistent testing procedures on all your code.

In the following exercise, you will load a previously created configuration file into the VI Analyzer.

**You can complete the following exercise in approximately 5 minutes.**

1. Open the VI Analyzer. In the **Select Task** dialog box, select the **Load a previously saved analysis configuration file** option and click the **Next** button.
2. From the file dialog, navigate to the **mytest.cfg** file you created earlier. Double-click the filename to select it.
3. In the **Select VIs** dialog box, notice that the settings for VIs and folders in the **Items to Analyze** hierarchy are the ones you selected previously.
4. Click the **Next** button. The **Select Tests** dialog box appears. In the next exercise, you will use the **Select Tests** dialog box to customize tests in the VI Analyzer task.



**Note** You will use the **Selects Tests** dialog window in the next exercise, so **do not close** the VI Analyzer window at this time.

End of Exercise 1-4

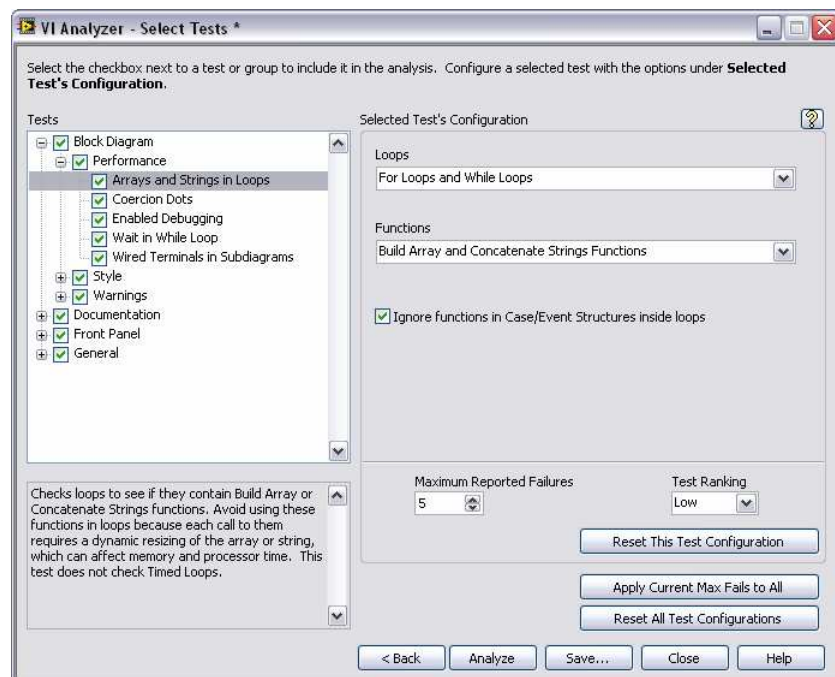
## Customizing Tests within VI Analyzer Tasks

Many of the VI Analyzer tests allow you to set custom criteria for passing or failing. You can set your own test criteria and save your settings to a VI Analyzer configuration file, which was discussed in an earlier section.

In the following exercise, you will customize the test criteria for one of the VI Analyzer tests.

You can complete the following exercise in approximately 5 minutes.

1. From the **Select Tests** dialog box, expand the **Block Diagram** category in the **Tests** window. Expand the **Performance** subcategory to display the specific tests for the category.
2. Click the **Arrays and Strings in Loops** test. The configuration options for the test appear in the **Selected Test's Configuration** frame on the right, as shown in Figure 1.12.



**Figure 1.12** – Customizing Individual VI Analyzer Tests

3. Select **While Loop** from the **Loops** control. The test now searches only for instances containing While Loops.
4. Click the **Save** button. Save the file as **mytest2.cfg**.

5. Click the **Analyze** button to run an analysis with the new settings.
6. In the **VI Analyzer Results Window**, review the analysis results.
7. Close the LabVIEW VI Analyzer.

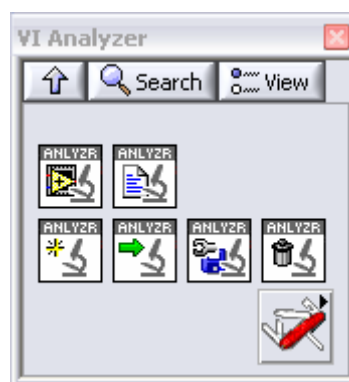
**End of Exercise 1-5**

# 2

## Using the VI Analyzer VIs

### VI Analyzer Palette

The **VI Analyzer** palette contains VIs that you can use to programmatically create or run VI Analyzer tasks. The programmatic option provides you with flexibility when you need to check programs regularly. To view the VI Analyzer palette, shown in Figure 2.1, go to **Functions » Addons » VI Analyzer**.



**Figure 2.1** – VI Analyzer Palette

The names of the VI Analyzer VIs begin with **VIAn** to identify them as part of the toolkit. The VIAn Easy Analyze VI and VIAn Easy Report VI enable you to create basic VI Analyzer tasks quickly. Other VIs require familiarity with the analysis process and with LabVIEW programming.

The VIAn Easy Analyze VI, shown below in Figure 2.2, uses an existing VI Analyzer configuration file to run an analysis. The file contains the path to the VIs to include in the analysis as well as the specific tests to run. The VI returns the test results in a single cluster.



**Figure 2.2** – VIAn Easy Analyze VI

The VIAn Easy Report VI, shown below in Figure 2.3, converts the analysis results from the VIAn Easy Analyze VI into a report. You can specify the

format of the report (ASCII or HTML), which test results to include, and where to save the report file.

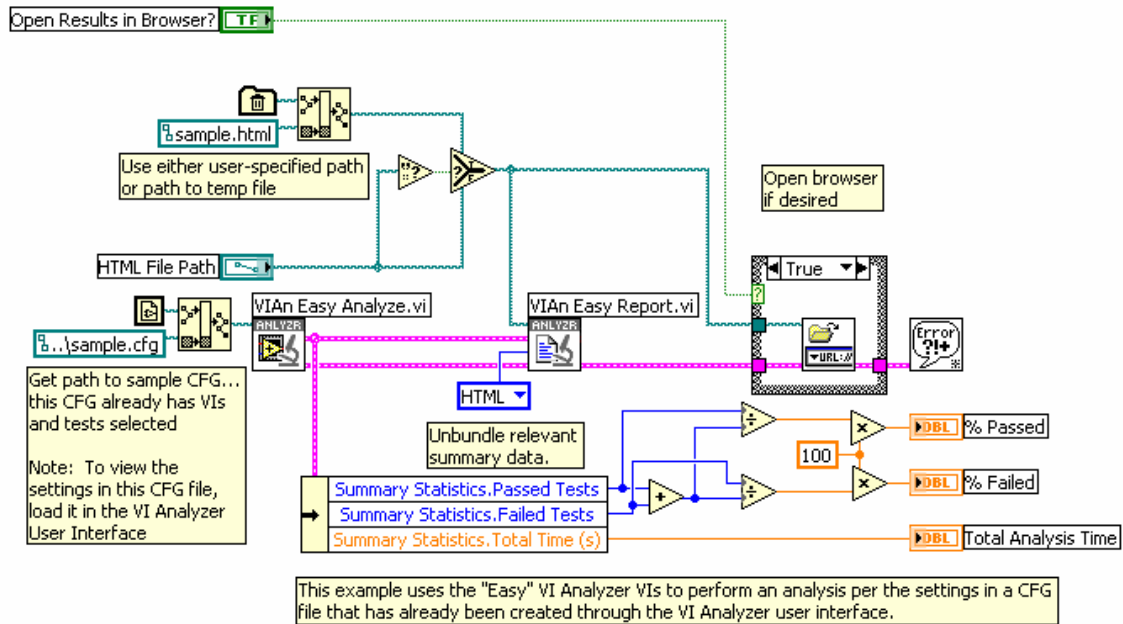


**Figure 2.3** – VIAn Easy Report VI

In the next exercise you will run a VI Analyzer task created using the VI Analyzer VIs. This programmatically created task will run tests on several VIs and generate an HTML report based on the results.

**You can complete the following exercise in approximately 10 minutes.**

1. Select **Help » Find Examples** and select the **Directory Structure** view. Open the **VIAnalyzer** folder to find the Easy VI Analysis VI. This example VI runs a VI Analyzer task and generates an HTML-based report with the results.
2. Open the block diagram. Notice that the example VI includes subVIs such as the VIAn Easy Analyze VI and the VIAn Easy Report VI, as shown in Figure 2.4.



**Figure 2.4** – Easy VI Analysis VI

3. Run the Easy VI Analysis VI. After the example VI runs, front panel indicators display the percentage of tests passed and failed, as well as the total analysis time. A Web browser opens to display the HTML-based report. This report will look similar to the report you created in the Exporting VI Analyzer Results section.

## End of Exercise 2-1

You can use the Easy VI Analysis VI as a model when you create your own programs using the VI Analyzer VIs. Refer to the *LabVIEW Help* for more information about specific VI Analyzer VIs.

# Automating VI Analyzer Tasks

You can also use the VI Analyzer VIs to create a program that dynamically creates and modifies a VI Analyzer task.

You can complete the following exercise in approximately 10 minutes.

1. Select **Help » Find Examples** and select the Directory Structure view. Open the VIAnalyzer folder to find the Advanced VI Analysis VI. This example VI allows you to exclude specific VIs and specific tests from a VI Analyzer task.
2. Open the block diagram. SubVIs in the example include a number of VI Analyzer VIs you can use to perform a VI Analyzer task, generate an HTML report, and provide a list of the VIs analyzed and tests run in the VI Analyzer task.
3. Figure 2.5 shows some of the VI Analyzer VIs on the Advanced VI Analysis VI block diagram. The block diagram for this VI is very wide and includes subVIs and other objects not visible in the figure. The VIAn Remove Item VI excludes the Container VI from the analysis because the Container VI is a placeholder VI that contains all the other VIs in the LLB. The VIAn Modify Test Settings VI excludes certain tests from the analysis. The VIAn Modify Test Config Value VI deselects the Control Descriptions option in the VI Documentation test. You can use the front panel controls to change the criteria that these VIs set.

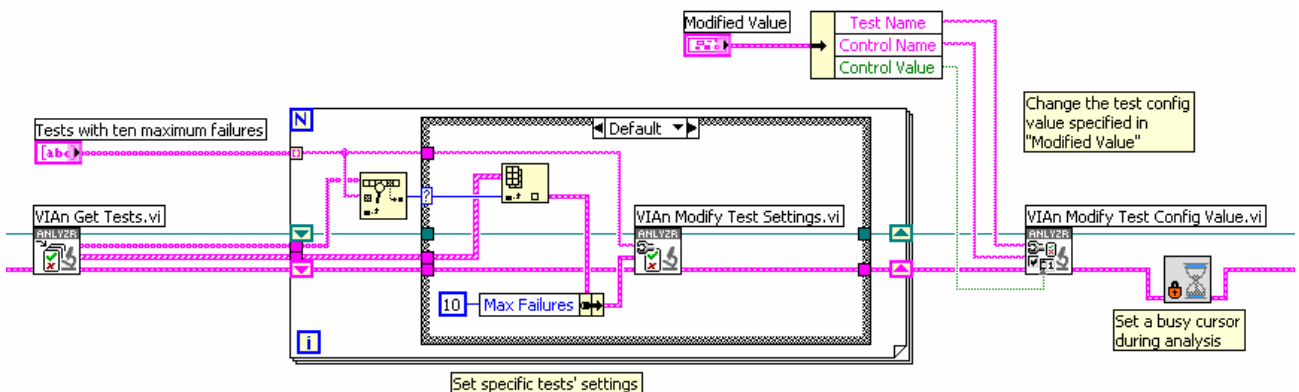


Figure 2.5 – Advanced VI Analysis VI

4. Run the Advanced VI Analysis VI. After the example VI runs, front panel indicators display lists of the VIs and tests that the VI Analyzer task included. A web browser opens to display an HTML-based report.

**End of Exercise 3-1**

You can use the Advanced VI Analysis VI as a model when you create VIs for more advanced VI Analyzer tasks. Refer to the *LabVIEW Help* for more information about specific VI Analyzer VIs.

# 3

## Summary

During this short tutorial you have successfully accomplished the following tasks using the functionality included in the LabVIEW VI Analyzer Toolkit:

- Create a VI Analyzer analysis task and analyze a LabVIEW VI
- Interactively view the analysis results and step through the failures in the code
- Saving test configurations to a VI Analyzer configuration file (.cfg)
- Export the VI Analyzer results to an ASCII or HTML report that can be viewed later in Notepad or a Web browser
- Load a previously created configuration file into the VI Analyzer
- Customize VI Analyzer tests to suit the requirements of the developer

Now that you are familiar with the functionality included in the VI Analyzer Toolkit, please feel free to continue exploring the other features LabVIEW has to offer.