Version

1.1

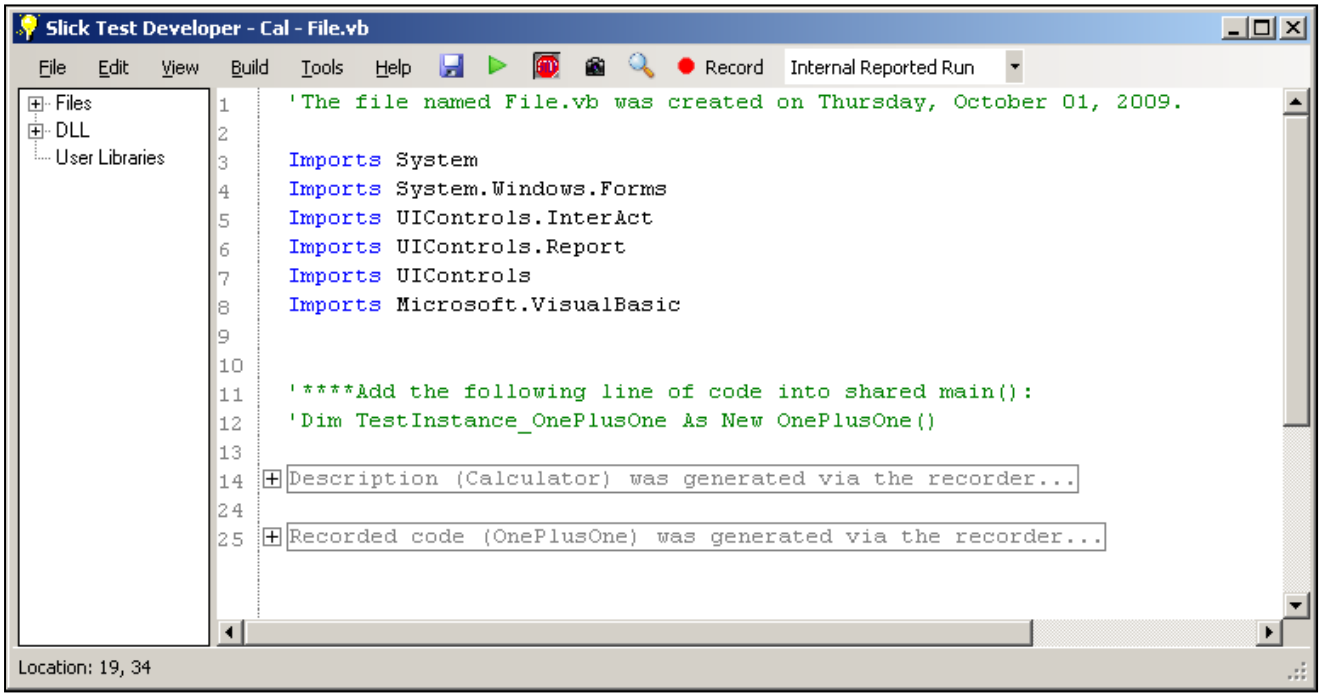
SLICK TEST DEVELOPER

Eternal Blue

# *Slick Test Developer User Guide*

ETERNAL BLUE©

# Slick Test Developer User Guide




Supported Technologies: Win32 and .net Windows Forms. Supports export to Visual Studios 2008 and Sharp Develop.

Requires: .net 3.5 to be installed on Windows XP or Windows Vista in 32 bit. Windows 7 as well as 64bit operating systems have not been fully tested.

TABLE OF CONTENTS

User Guide ..... 1
How to record a test ..... 3
Description Objects ..... 5
Using the IDE ..... 6
Using the Project Explorer ..... 9
Tests - Building, Running and Reporting ..... 10
Reporting ..... 11
Spying on objects and capturing images ..... 14
Additional features ..... 16
Export ..... 16
Canceling Tests ..... 16
Description Tester ..... 16
Find Window Via Hwnd ..... 16
Help Guides ..... 16
Templates ..... 16
Contact Information ..... 17

# User Guide

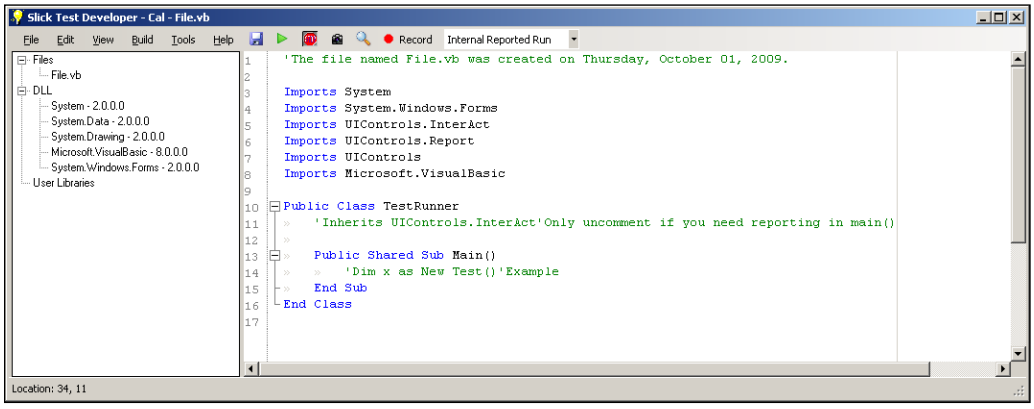
*An Introduction to the usage of Slick Test Developer*

Slick Test Developer is a fully featured automation solution in the windows environment. It includes capabilities to write, run and record result for testing.

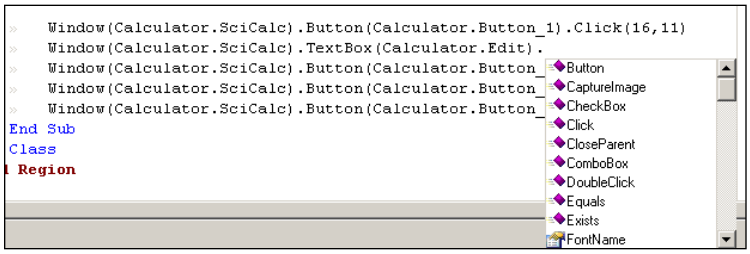
When you first start-up Slick Test Developer, you will be prompted to select a project. The project types that will be available to you include a Library Project and a Record and Go project. A Library Project will create a DLL and cannot directly run from Slick Test. A Record and Go project will allow the user to run tests in a console application. In addition to this, you can browse for an already created Slick Test Project (*.STP).


**NOTE**: Currently Slick Test Developer supports C# and VB.NET.

Once you select the project you wish to create and press Ok, you will be prompted to create a folder for the project to reside in. The folder name will also be the project name. At this point, Slick Test will automatically create all the folder structures and a file for the project (named File.vb).



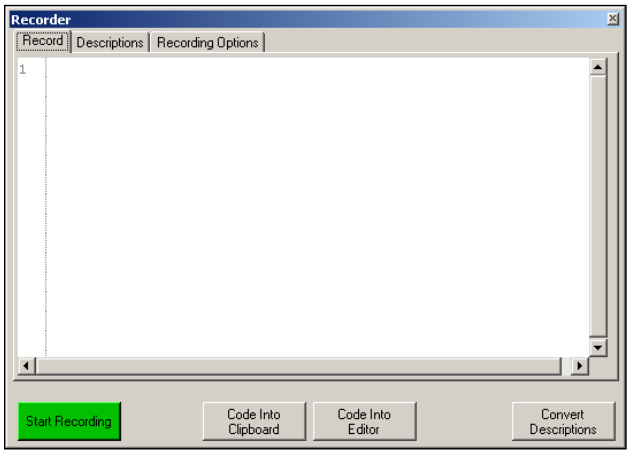
From here you can start to develop your automation solution using a variety of tools. The IDE (Integrated Development Environment) provides a text editor on the right which includes code highlighting, auto complete (including via ctrl + space) and other advanced features.




On the left side is a project explorer, which provides a list of your source code files, the included DLLs and any special user DLLs that maybe required. You can add, remove, delete and rename files from the tree. The default DLL list can be modified using the options menu, which will be explored further in other sections of this document.

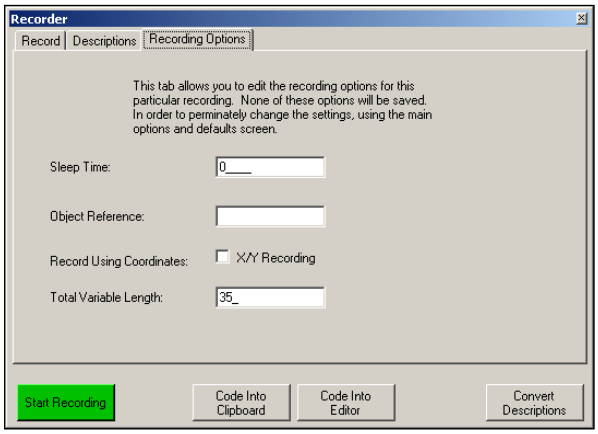
# How to record a test

One of the easiest ways to create a test is to record it. Recording provides the basics for development of your test cases. To get to the record dialog simply press the record button or select Tools and then choose Start Recording Button.


At this point, the recorder will appear. The user will be able to make any modifications to any recording settings before they start the recording by clicking the recording options tab.

The sleep time feature allows a user to put sleeps in between clicks or typing, allowing for a more realistic user experience. The Object Reference is any variable reference you need for the objects rather than using inheritance, which is the recommended method in Slick Test. Record Using Coordinates will use non-object based recording and Total Variable Length maintains a maximum length that any Description Object can be.

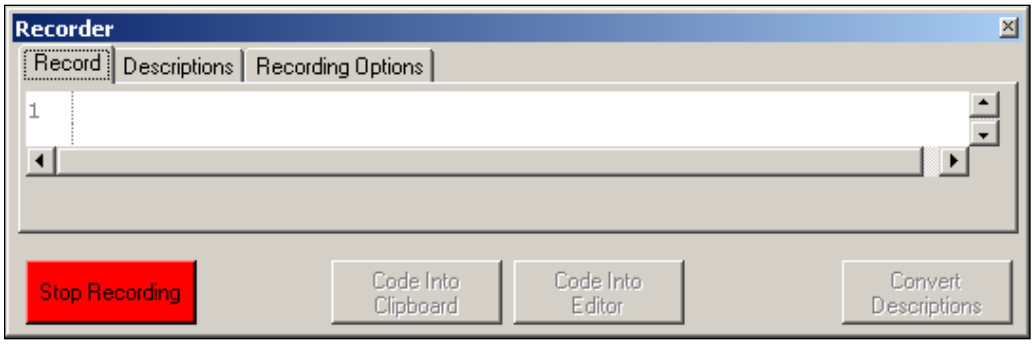



**NOTE**: All recording options can be saved by using the Defaults and Options section of the application.

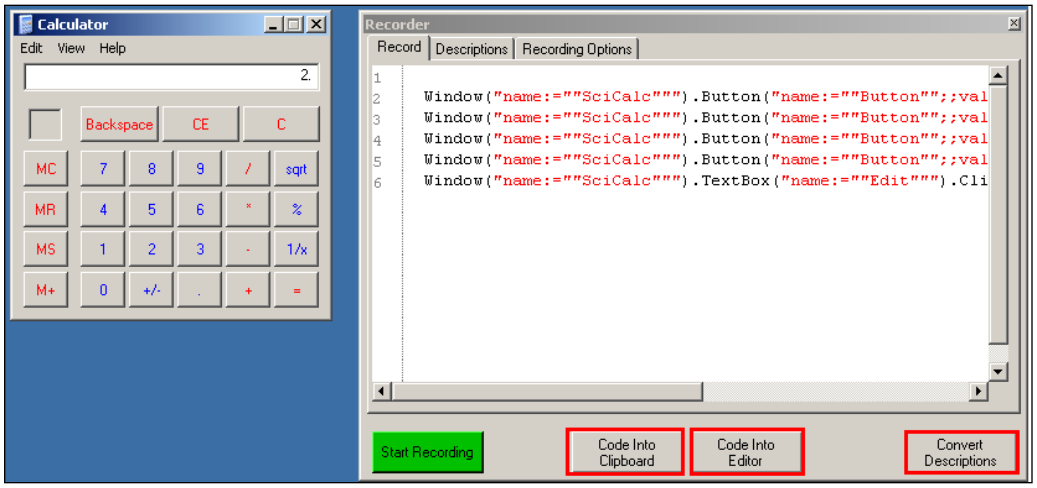
**NOTE**: It is recommended you finish reading this section before you press record.

Once you have setup your recording options and have performed any setup required, press start recording.

After you press record, the recorder will shrink down and the Stop Recording button will appear. At any time you can press the Stop Recording button and the recorder will pause. You can later restart the recorder. While recording, you will have the ability to modify the code without affecting the recording and without pressing Stop Recording.



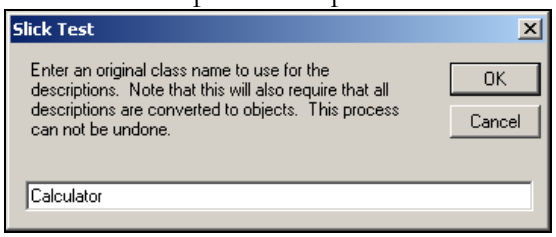

Once your recording is completed, press the stop recording button. At this point, you have five options. You can click the Code Into Clipboard, Code Into Editor, Convert Descriptions, manually copy the code or close the recorder. By copying the code into the clipboard or into the editor, you will have to convert the code into Description Objects. You can also just convert the code into Description Objects if you only need part of the code or wish to just update your descriptions.



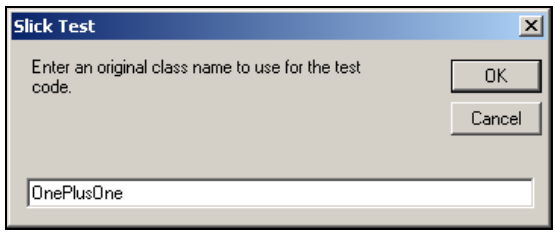
**Description Objects**

Description Objects are objects that describe a Windows Object (such as a button) in a textual way. Because text descriptions are so common, it is easier to maintain a Description Object and generally is faster for the Slick Test to process.

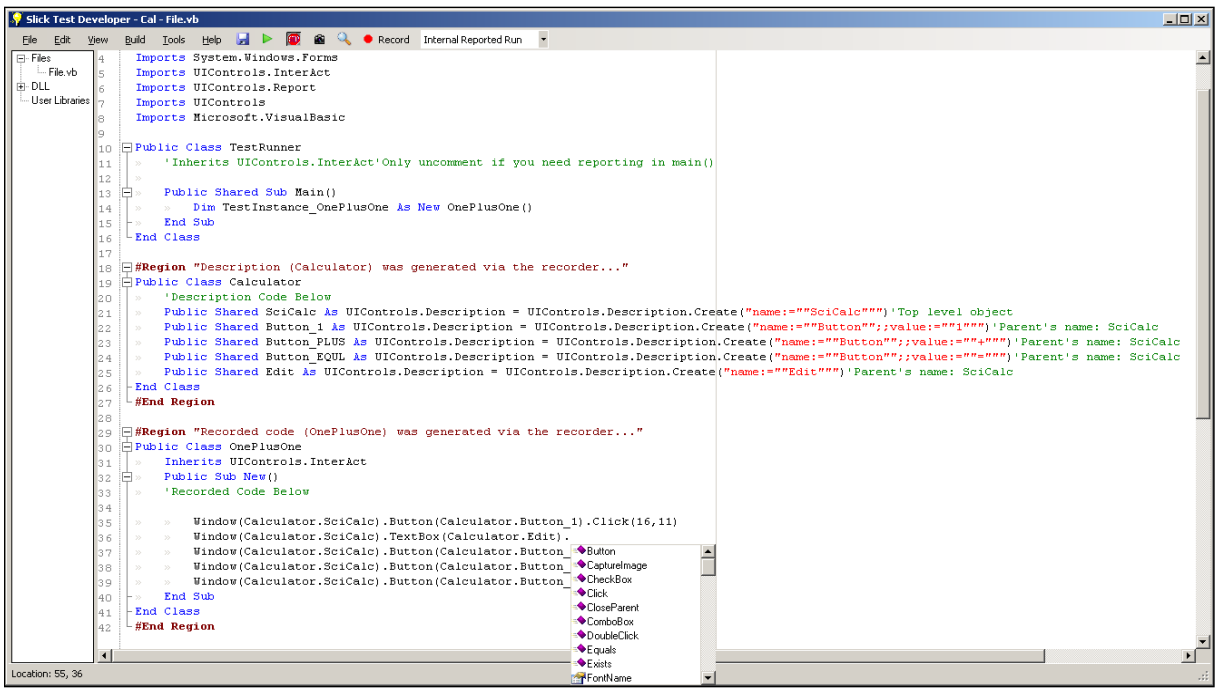
When you click to copy code into the editor (or the clipboard) it will prompt you to give it a unique name for the Descriptions so that it can create a class for the Descriptions. Keep in mind that this forces the system to convert the code to Description Objects.




Once you have given the Descriptions a class name, you will be prompted to give a unique class name for the test case you just recorded. This name will be assigned to a class, not a method. If you are using NUnit or some other third party testing tool, you may wish to move the code from the constructor of the class to a method or whatever format that tool requires. After the class name prompts, you will be prompted to either convert the code to descriptions (recommended) or leave the code as is. If you choose not to convert a description, only the test case name will be used.

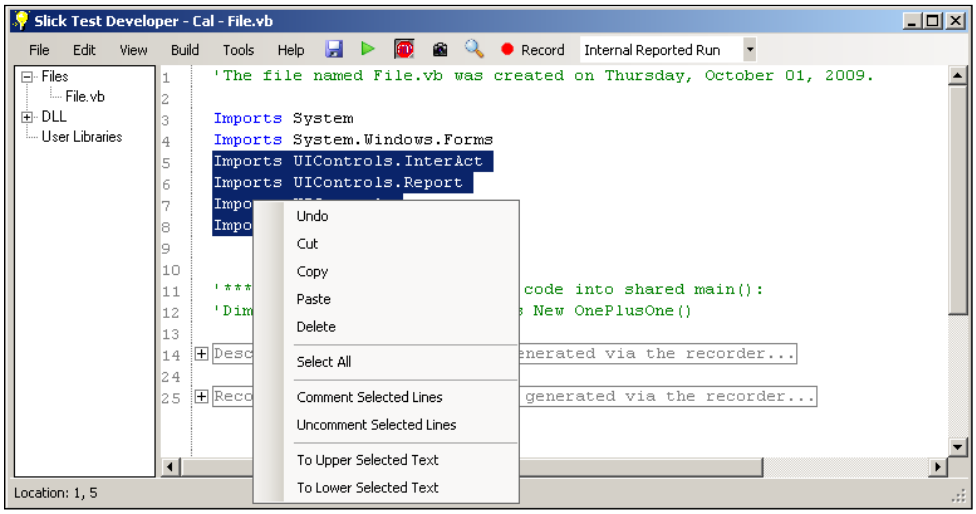



If you choose to use the Shared Sub Main() method for your testing, you will have to move a commented line of code provided by Slick Test into Main and make sure to uncomment the code. At this point you are ready to run the test or make any modifications as required.

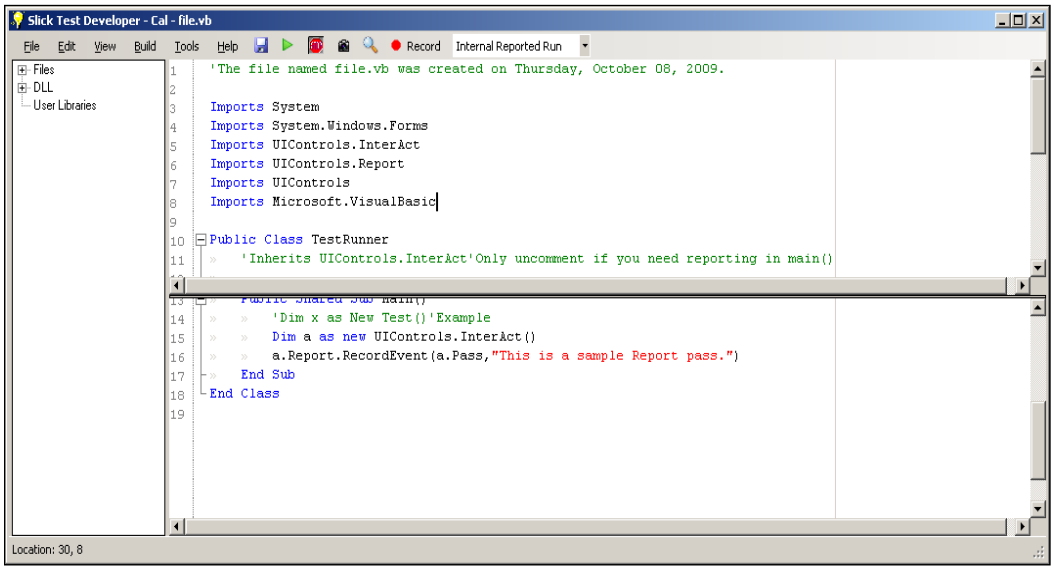


# Using the IDE

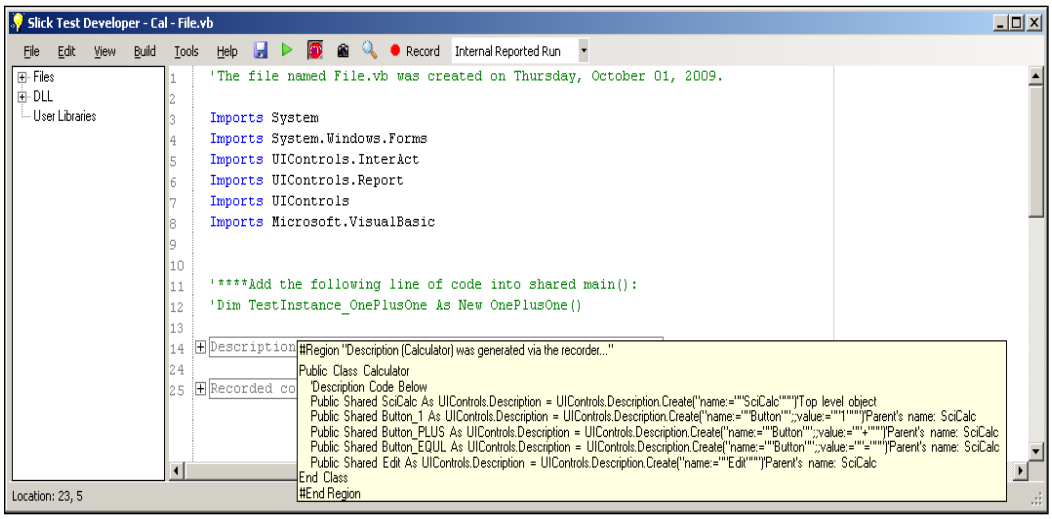
Slick Test's editor provides many handy features. A user can easily select a set of text and comment or uncomment lines of code. In addition to that, a user can set text to upper case or lower case. All of the traditional editing features such as copy, paste, delete, select all and undo are also provided.



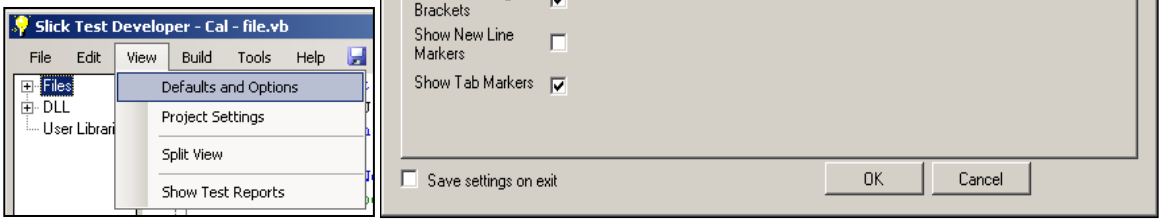

The IDE allows a user to split the editor into two sections, allowing the user to edit two separate sections of a file at once, allowing faster editing. This can be done by clicking View and then checking (selecting) Split View. To shut off the feature, simply uncheck Split View.



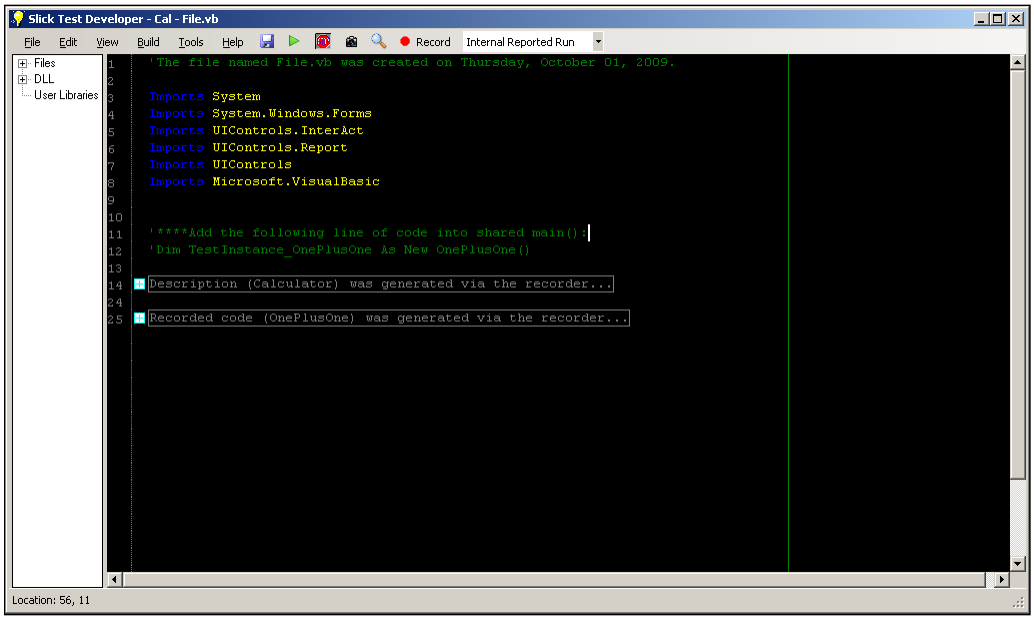
Slick Test Developer also provides advanced features such as auto complete and region folding. A user can mouse over any folded region and will be provided with a tool tip of the code in the region. A tool tips will also appear over any piece of code, such as a method that you place your mouse over. Auto complete works for on any attached DLLs and the currently opened files. Other files in the project are currently not processed for auto complete. This will be changed in a future release.




A large number of options for the editor are provided in the UI section of the Defaults and Options dialog. The Defaults and Options dialog can be found by clicking View and then clicking Defaults and Options. Once there, select the UI tab. Inside of the UI options, you can set the font, and enable or disable various formatting displays. In addition to that, you can enable edit line highlighting to see the current line.



Some advanced UI options can only be accessed through VBNET.xshd found in the same folder as the Slick Test Executable. While Slick Test Developer is close, open VBNET.xshd using notepad or other text editor.

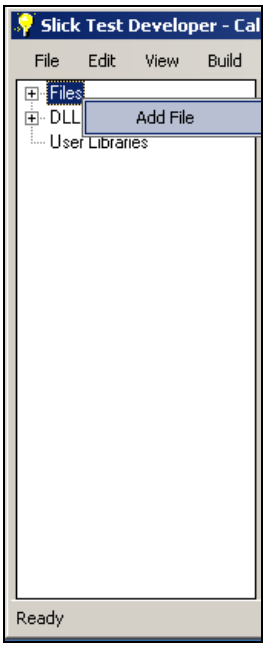


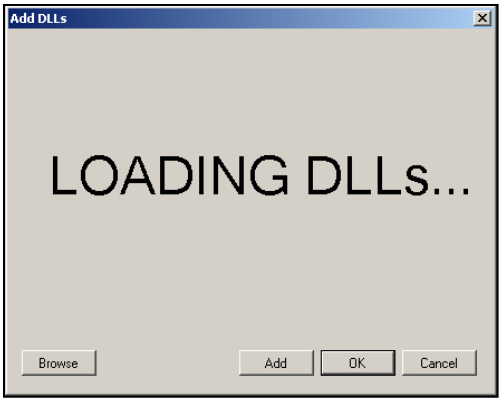

An example environment setup is provided but commented out at the top of the file. Simply remove the XML comments from the environment section at the top of the document, save and open Slick Test. This guide will not walk you through the specifics of how to adjust the editor.

# Using the Project Explorer

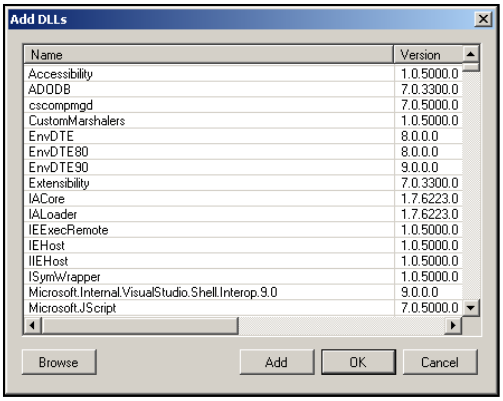
The Project Explorer can be seen to the left of the text editor in Slick Test Developer. Three primary areas are contained within the project explorer window.

There is the Files section, which lists all *.vb (or *.cs when C# is supported) in the source director based upon loading of the project. If you right click on the Files you can add additional files by right clicking Files and choosing Add File. If you right click a file, you can add, delete, open or rename a file.



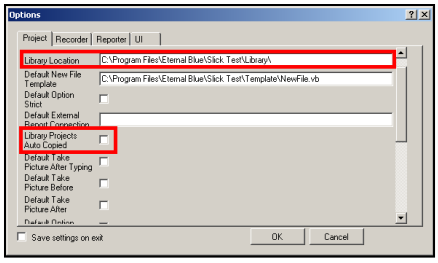



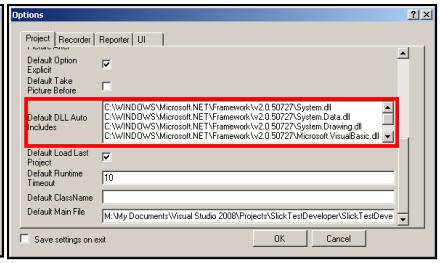

Users can add or delete DLLs by right clicking DLLs and clicking the Add DLL menu item. Once you've clicked Add DLL, a dialog will pop up with a "Loading DLLs" displayed for a while and then a list of DLLs will appear. A user can then add DLLs from the list displayed or browser for additional DLLs using the browse button. Users can remove a DLL reference by right clicking on the specific DLL and clicking Remove DLL. All DLLs listed also show the version information in case multiple versions of the same DLL exist.




DLLs can also be listed in the Defaults and Options dialog so that they are automatically included with every created project. To do this, simply click View and then select Defaults and Options. Then in the Project tab go to Default DLL Auto Includes. To add an additional DLL, simply type in the file path of the DLL in a new line and press OK to save the settings.

User Libraries are user created libraries that get automatically updated on every build. The Library Location (folder) can be found or changed by click View, selecting Defaults and Options and then going to the Project tab. When creating Library Projects, a user can have it automatically copied to the Library Location by checking Library Projects Auto Copy in the Project tab of Defaults and Options.



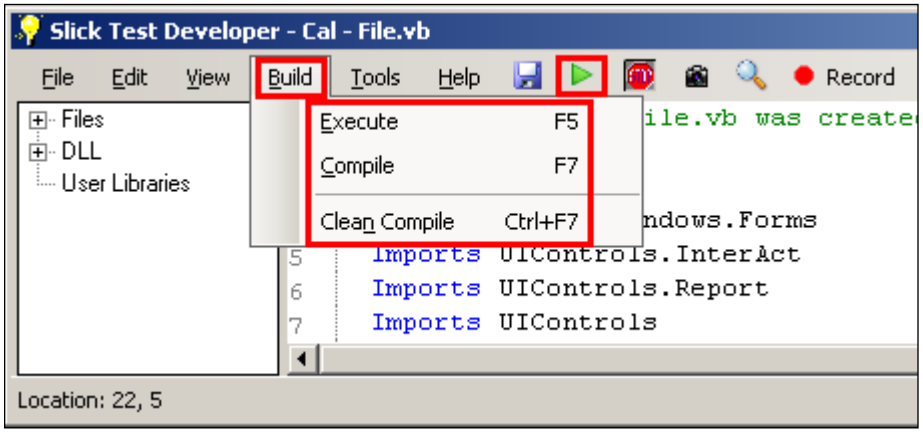



# Tests - Building, Running and Reporting

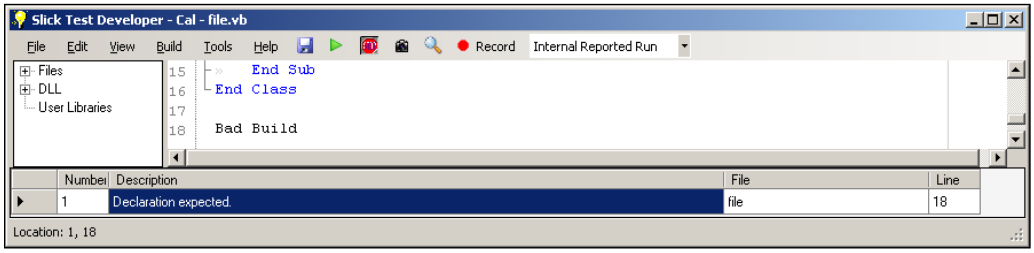
A user has multiple ways to build and run a test. As long as a user is in a Record and Go project, a user can use the green Run button, press F5 or click Build and then select Execute. The user should know that Execute will automatically compile, even if compiling isn't required. In the future compile sensing will be added.

If a user just wishes to compile a project they can select Compile from the build menu or press F7. If a user wishes to make sure all references are the latest (for example the user just upgraded Slick Test) they can choose Clean Compile from the build menu or press Ctrl + F7.

**Note**: User Libraries are always the latest as they are copied on every build.



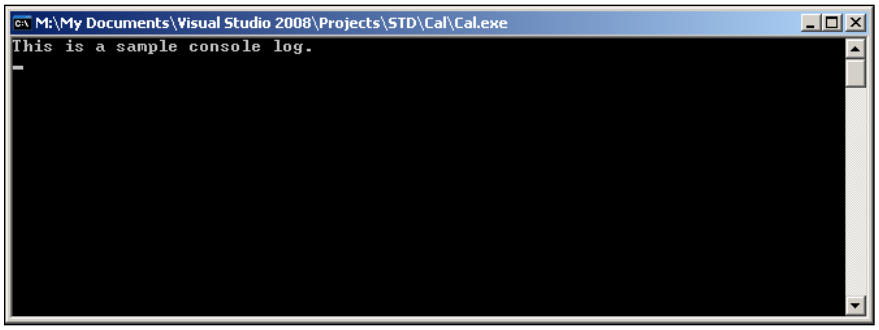

Sometimes while building, a user will find that the build failed due to some sort of error. The errors list will pop up from the bottom of Slick Test and provide the user with a full list of known compiler errors. Keep in mind some errors will uncover other errors, so it may take multiple builds before a test will compile. When an error does occur, you can double click the error and the editor will automatically take you to the file and line number.




**Note**: Be sure to select the type of reporting before pressing the run button. For more information, see the reporting section below.

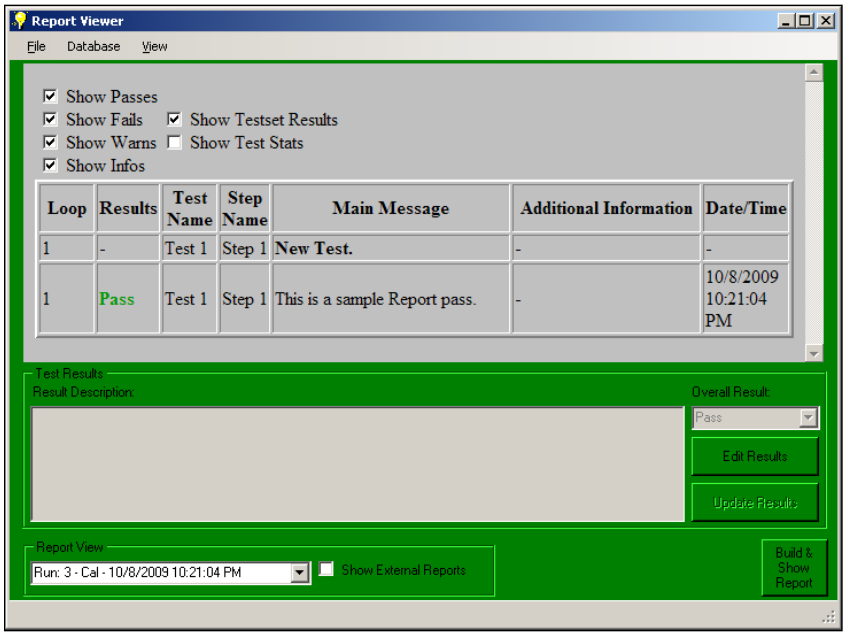
**Reporting**

When you press Execute (or Run) a console window will appear, and the test will begin. Depending on your settings and the code you have written, the console may provide the user with some details about the test run. These may also be recorded into the reporter.



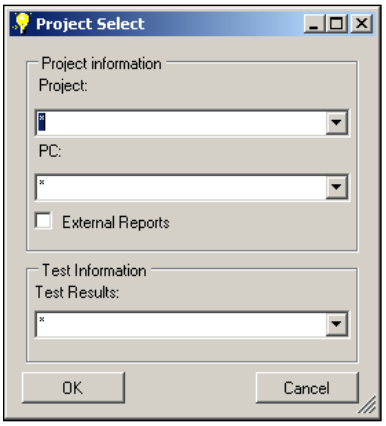

Once the test has completed, the Report Viewer may appear. If it does not, simply click View and then select Show Test Reports. In order to automatically show the reporter after a test has completed, go to View and select Defaults and Options. Select the Reporter tab and check Show Report.

**Note**: The Report Viewer will not appear if you never report anything in the test.

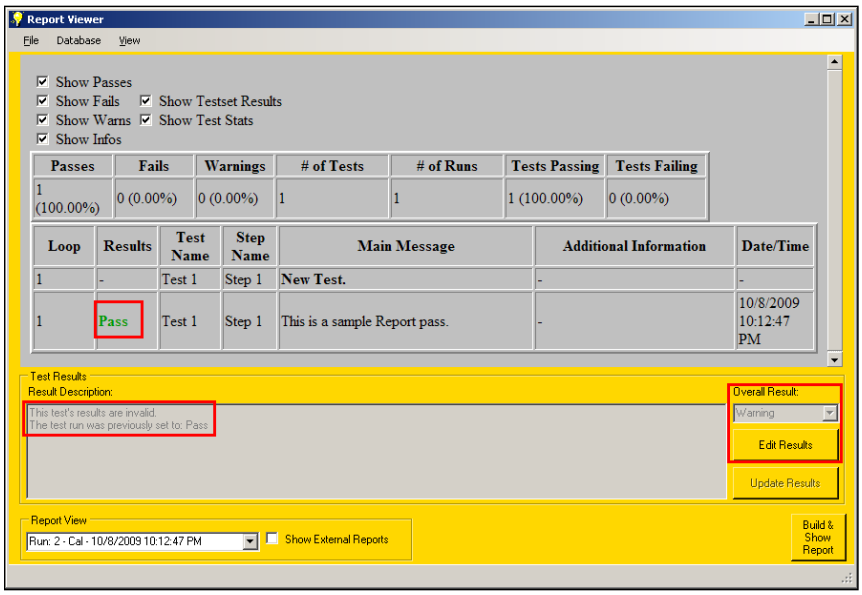



The report is rendered in HTML and provides the ability to filter data to allow for easier searching. At present, a single project run is considered one test result, whether you run one or more tests.

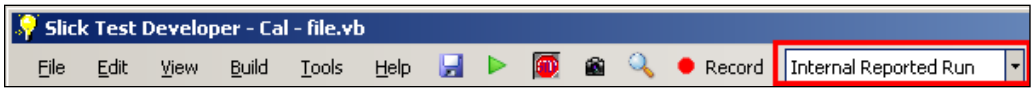
Each run is displayed at the bottom in the Report View drop down. You can select an old report and press Build & Show Report to see the selected report. By default, the currently opened project is filtered, but the user can edit the filtering by clicking View and selecting Project Filters.




A user can edit the overall results of the run and provide information about the run by pressing edit results and then clicking update results.



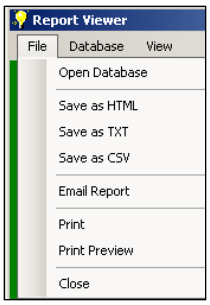

Users must be aware that there are two forms of reports, those that are considered internal to a user and those that are considered external.




In order to set the reporting type, in the main Slick Test Editor, click the report drop down and select the report type you want.

The reporting distinction is designed to allow the user to organize their development reports so that they are clear and separate from reports against known good automation. In the future direct SQL reporting will be implemented for External Reports, while internal reports will only go to a local database.

You can export a report to HTML, CSV or a text file, as well as Print or Email it (Email is only supported if you have Microsoft's Outlook™ installed).




Users have several database functions available to them for cleaning up the database. Clear Database clears all but the latest entry from the entire database for Internal Reports. External Reports can never be clean up without deleting the database file. You can also Shrink the database, which will do some memory cleanup while Compacting the database will do the most cleaning, but requires a temporary database be created, and has a larger risk of corrupting data.

**WARNING:** Do not share the report database out and have multiple users report to the same database. This will eventually cause errors in the database and a loss of data. It is recommended you do not have a remote user even read from the database while a local user is writing to the database.

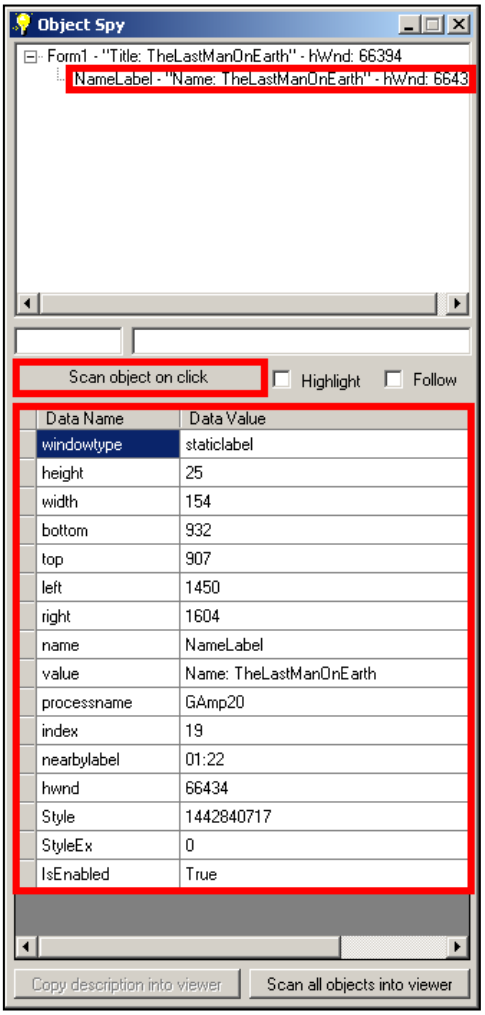
Users can open a database, even a database on a remote share to view results. Simply click File and select Open Database in the report viewer.

**WARNING:** Do not write to a database using on a remote share. There is a chance that data corruption could occur.

# Spying on objects and capturing images

The Object Spy tool provides the ability to see details of objects without recording a test. The Object Spy can be found by clicking the magnifying glass or by clicking Tools, then selecting Object Spy. Users can turn on the follow and highlight features by checking them and a red square will appear around whatever object your mouse is over along with limited information provided by the object.

A user can then click on "Scan object on click" and they will see their mouse cursor change into a hand. The very next click will spy on the object the user clicked on as well as give the user a tree showing ownership of the object.

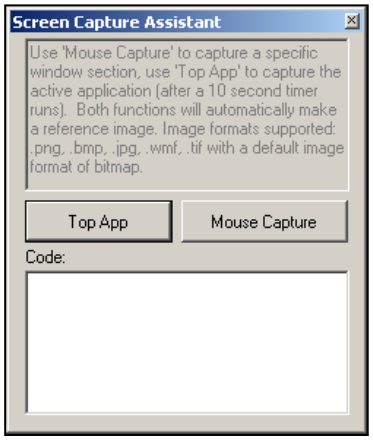



In the screenshot to the right, the NameLabel was spied on with a parent name Form1. Currently NameLabel is displayed in the details but by selecting any item in the tree view you can see the details of said object. You can also right click the items in the tree to highlight the specific object or to dump most information known about the window and its children.

**NOTE**: CamelCased Data Names cannot be used as part of a description. All lowercase Data Names can be used as descriptions.

If you were to select an item in the tree, the Copy description into viewer would become enabled. When you click on the Copy description into viewer button, a simple window will appear showing you the code to create a description for the given object. It should be noted that this will not provide the simplest description based upon the heuristics of the recorder since the object spy does not search for an optimal description.

Additionally, the Object Spy provides the ability to search for all objects within a window or to capture all objects that would be typically displayed in a tree. In order to capture all objects related to a window, click the "Scan all objects into viewer" button and while holding down alt, click on the window. If you simply want to create descriptions based upon the tree, do the same as scanning all objects except don't hold down the alt key. It should be noted that this will provide the simplest description based upon the heuristics of the recorder.


The Code Viewer for copying or editing descriptions into your code base.

The screen capture tool can be found by click on the camera or clicking Tools and then selecting Capture Screenshot. At this point a window will appear. A user can either capture the top application or capture a section of a screen by clicking mouse capture.

Once a user clicks the Top App button, the user will be prompted to provide a path to save the screenshot in. This will be the initial screenshot if you wish to later do comparison testing. The file extensions supported are explicitly listed in the application, but they do include png and bmp. If no extension is provided, the application will be set to bmp. After 10 seconds, an image will be captured and code will be written for you to be able to create another screenshot.

If you want to capture just a section of a window, or an object within the window, a user should use Mouse Capture. Simply click on the mouse capture button and then enter in a file name you wish to use. After pressing ok on the file path prompt, the button will change the text to "Top-Left Corner." At this point the user can right click anywhere they wish to record the top left corner of the future image. After the first right click, the button will change to "Bottom-Right Corner." Right click again and an image will be saved with the button changing back to "Mouse Capture."

# Additional features

There are several additional miscellaneous features that are useful enough to be covered, but do not fall into any broad category. These features will be explained here.



**Export**

Users can export the project to for external usage in other IDEs that support *.vbproj or *.csproj project files. To export the project simply click File then select "Export for external usage" in the menu. Exporting the project will not move any of the files to the export location. If you want to move the files to an external project folder you will have to manually edit the project file using notepad or other text editor. Keep in mind if you do export the Slick Test Project, any changes done in the external editor will not be retained. Furthermore, Slick Test does not currently open vbproj or csproj files.

**Canceling Tests**

Sometimes a user will start up a test by accident. In order to cancel the test, the user simply must click the Stop sign image in the tool bar. The test will be canceled and will be marked as a fail. All results gathered up at that point will not be retained.

**Description Tester**

The description tester allows a user to verify a description via .Exits(1) or get a property. When you press run, type enter or press f5 the line will be executed and the text will output the value in a message box.

**Find Window Via Hwnd**

This is a simple tool that allows the user to search the primary monitor for a specific Hwnd or Windows Handle. IF found, it will bring that window to the top and highlight the area that window exists in.

**Help Guides**

Three guides have been provided to assist in the creation of automation. This guide is designed to provide a high level over view of how the system works and the features it provides. The Automation Primer is a more technical guide with various how-to guides and does not focus on the Slick Test Developer UI. The API guide is designed to document the entire API provided by Slick Test.

**Templates**

Two templates are currently provided for Slick Test Developer. One template is for the default file that is included with every new project and can be found <installation directory> \Template\NewProjectFile.vb. The second template is provided for every new file added after the first. It can be found at < installation directory>\Template\NewFile.vb. When each of these files is added to a project, a system automatically modifies the document with the following macros:

- {{FILENAMEWITHNOEXT}} – The name of the newly created file without an extension.
- {{DATECREATED}} – The date the file was created on.
- {{USERNAME}} – The user name of the user creating the file (i.e. <Domain>\<Username>)

**Contact Information**

You can find contact information by going to the web at http://sourceforge.net/projects/slicktest/. Slick Test was created by an Automation Engineer who was tired of tools that did not provide everything needed for the job. This tool is a first attempt to fix that. I have attempted to provide a good deal of polish to this system to provide a true automation solution that compares to many of the vender's automation solutions. There is still more work to be done, but I hope you enjoy using this tool. If you find bugs, please feel free to report them on source forge.