Borland®
(A MICRO FOCUS COMPANY)

MICRO FOCUS®

SilkTest 13.0

Silk4NET User Guide

Micro Focus
575 Anton Blvd., Suite 510
Costa Mesa, CA 92626

2012-05-24

Contents

Contacting Micro Focus 4
Information Needed by Micro Focus SupportLine 4
Product Notification Service 5
Silk4NET 6
SilkTest Product Suite 7
Creating a Silk4NET Project in Visual Studio 8
Creating a Silk4NET Test Overview 9
Adding a Silk4NET Test Template to a Visual Studio Project 10
Manually Creating a Silk4NET Test in Visual Studio 11
Recording a Test 12
Dragging and Dropping Recorded Test Steps 13
Exporting Recorded Test Steps to Visual Studio 14
Exporting a Silk4NET Project to Visual Studio 15
Running Silk4NET Tests 16
Analyzing Test Results 17
Silk4NET Sample Tests 18
Dynamic Object Recognition 19
XPath Basic Concepts 19
Supported XPath Subset 20
XPath Samples 21
Troubleshooting Performance Issues for XPath 22
xBrowser Frequently Asked Questions 24
How do I Verify the Font Type Used for the Text of an Element? 24
What is the Difference Between textContents, innerText, and innerHtml? 24
I Configured innerText as a Custom Class Attribute, but it Is Not Used in Locators 25
What Should I Take Care Of When Creating Cross-Browser Scripts? 25
How Can I Distinguish Firefox from Internet Explorer in My Scripts 25
Which Locators are Best Suited for Stable Cross-Browser Testing? 25
Logging Output of My Application Contains Wrong Timestamps 26
My Test Script Hangs After Navigating to a New Page 26
Recorded an Incorrect Locator 26
Rectangles Around Elements in Windows Internet Explorer are Misplaced 26
Link.Select Does Not Set the Focus for a Newly Opened Window in Internet Explorer 26
DomClick(x, y) Is Not Working Like Click(x, y) 26
FileInputField.DomClick() Will Not Open the Dialog 26
The Move Mouse Setting Is Turned On but All Moves Are Not Recorded. Why Not? 27
I Need Some Functionality that Is Not Exposed by the xBrowser API. What Can I Do? 27
Why Are the Class and the Style Attributes Not Used in the Locator? 27
Dialog is Not Recognized During Replay 27

Contacting Micro Focus

Micro Focus is committed to providing world-class technical support and consulting services. Micro Focus provides worldwide support, delivering timely, reliable service to ensure every customer's business success.

All customers who are under a maintenance and support contract, as well as prospective customers who are evaluating products are eligible for customer support. Our highly trained staff respond to your requests as quickly and professionally as possible.

Visit *http://supportline.microfocus.com/assistedservices.asp* to communicate directly with Micro Focus SupportLine to resolve your issues or email supportline@microfocus.com.

Visit Micro Focus SupportLine at *http://supportline.microfocus.com* for up-to-date support news and access to other support information. First time users may be required to register to the site.

Information Needed by Micro Focus SupportLine

When contacting Micro Focus SupportLine, please include the following information if possible. The more information you can give, the better Micro Focus SupportLine can help you.

- The name and version number of all products that you think might be causing an issue.
- Your computer make and model.
- System information such as operating system name and version, processors, and memory details.
- Any detailed description of the issue, including steps to reproduce the issue.
- Exact wording of any error messages involved.
- Your serial number.

To find out these numbers, look in the subject line and body of your Electronic Product Delivery Notice email that you received from Micro Focus.

Product Notification Service

The product notification service is an application that runs in your system tray and allows you to find out if updates are available for SilkTest. It also provides a link for you to click to navigate to the updates.

Running the Service

In the system tray, click the update notification icon and the Product Notification Service application opens.

Installed Version	Provides the version number of the currently installed SilkTest application.
Update Version	Provides a link and the version number of the next minor update, if one is available.
New Version	Provides a link and the version number of the next full release, if one is available.
Settings	Click the **Settings** button to open the **Settings** window. Select if and how often you want the notification service to check for updates.

Silk4NET

Silk4NET is SilkTest's plug-in for Microsoft Visual Studio that allows you to efficiently create and manage functional, regression, and localization tests directly in the Visual Studio. With Silk4NET you can develop tests using either Visual Basic .NET or C#, run the tests as a part of a test plan in Microsoft's test environment or as a part of your build process, and view the test results all from within Visual Studio.

Silk4NET supports the testing of a broad set of application technologies including AJAX and Web 2.0, RCP, WPF, Windows Forms and Win32. Designed for realizing automation benefits even when applied to complex tests, Silk4NET brings true test automation capability directly to the developer's preferred environment and lets you easily cope with changes made in the test application.

Additionally, Silk4NET's powerful testing framework enables the high reusability of tests across multiple test projects, which further increases the achievable ROI. With less time spent on building and maintaining testing suites, your QA staff can expand test coverage and optimize application quality.

SilkTest Product Suite

The SilkTest product suite includes the following components:

- SilkTest Workbench – SilkTest Workbench is the new, native quality testing environment that offers .NET scripting for power users and innovative storyboard-based visual tests to make testing more accessible.
- Silk4NET – The Silk4NET Visual Studio plug-in enables you to create Visual Basic or C# test scripts directly in Visual Studio.
- Silk4J – The Silk4J Eclipse plug-in enables you to create Java-based test scripts directly in your Eclipse environment.
- SilkTest Recorder – SilkTest Recorder enables you to record and replay tests using a GUI and then export those tests to SilkTest Classic, Silk4J, or Silk4NET.
- SilkTest Classic – SilkTest Classic is the traditional, 4Test SilkTest product.
- SilkTest Agents – The SilkTest Agent is the software process that translates the commands in your tests into GUI-specific commands. In other words, the Agent drives and monitors the application you are testing. One Agent can run locally on the host machine. In a networked environment, any number of Agents can run on remote machines.

The product suite that you install determines which components are available. To install all components, choose the complete install option. To install all components with the exception of SilkTest Classic, choose the standard install option.

Creating a Silk4NET Project in Visual Studio

1. Click **File** > **New** > **Project** . The **New Project** dialog box appears.
2. Under **Installed Templates**, expand **Visual C#** or **Visual Basic**, and then select **Silk4NET**.
3. Type the project name, location, solution, and solution name, and then click **OK**. A new solution containing the Silk4NET project is created. Additionally, a Silk4NET test is created in the project with the following language-specific file name:
   - `UnitTest1.vb`
   - `UnitTest1.cs`

Creating a Silk4NET Test Overview

You can use the Silk4NET test template as a starting point for manually creating tests.

This template contains import namespaces for the required character encoding classes, SilkTest's Open Agent API, and Visual Studio's unit testing support. Additionally, this template contains a generic class and method in which you can create specific test steps designed to test functionality in your applications.

Another method is to use the SilkTest Recorder to automate the creation of tests by recording user actions made against a test application.

The SilkTest Recorder records the test in a neutral format that you can then export to your Visual Studio project in either Visual Basic .NET or C#.

The SilkTest Recorder also supports a drag-and-drop feature and a clipboard feature. Both features allow you to quickly move recorded test step into an existing Silk4NET test in Visual Studio. In this way, you can use a mixed-method approach in which you record specific parts of a test and manually code others.

Adding a Silk4NET Test Template to a Visual Studio Project

1. Choose **Project** > **Add New Item** . The **Add New Item** dialog box appears.
2. Under **Installed Templates**, choose **Common Items** > **Silk4NET** . The Silk4NET test template appears in the middle pane.
3. Select **Silk4NET Test**, and then click **Add**. A Silk4NET test template is added to your project with a default file name of `UnitTest1.cs` or `UnitTest1.vb` depending on your project's default programming language.

   This template contains import namespaces for SilkTest's Open Agent API and Visual Studio's unit testing support. Additionally, this template contains a generic class and method in which you can create specific test steps, as shown in the following examples:

```
'Visual Basic .NET
Imports System.Text
Imports Microsoft.VisualStudio.TestTools.UnitTesting
Imports SilkTest.Ntf

<TestClass()>
Public Class UnitTest1

    Private ReadOnly _desktop As Desktop = Agent.Desktop

    <TestMethod()>
    Public Sub TestMethod1()
    End Sub

End Class
```

```
//C#
Using System;
Using Microsoft.VisualStudio.TestTools.UnitTesting;
Using SilkTest.Ntf;

namespace SilkTest.Ntf.SampleScripts.CSharp
{
    [TestClass]
    public class UnitTest1
    {
        private readonly Desktop _desktop = Agent.Desktop;

        [TestMethod]
        public void TestMethod1()
        {
            // Put your test here.
        }
    }
}
```

Manually Creating a Silk4NET Test in Visual Studio

1. Add a Silk4NET test template to your project. Choose **Project** > **Add New Item** > **Silk4NET** , and then click **Add**. The Silk4NET test template is added to your project.
2. Optionally, to add support for controls of a specific application technology, you must include an import statement at the beginning of the test that references the application technology namespace, as shown in the following examples:

```
'Visual Basic .NET
Imports SilkTest.Ntf.Wpf
Imports SilkTest.Ntf.XBrowser
Imports SilkTest.Ntf.Win32
```

```
//C#
Using SilkTest.Ntf.Wpf;
Using SilkTest.Ntf.XBrowser;
Using SilkTest.Ntf.Win32;
```

3. Configure the base state of the test application. For example:

```
'Visual Basic .NET
Dim baseState = New BaseState("C:\\Windows\\system32\\notepad.exe",
"/Window[@caption='Untitled - Notepad']")
baseState.WorkingDirectory = "%USERPROFILE%"
baseState.Execute()
```

```
//C#
BaseState baseState = new BaseState("C:\\Windows\\system32\\notepad.exe",
 "/Window[@caption='Untitled - Notepad']");
baseState.WorkingDirectory = "%USERPROFILE%"; baseState.Execute();
```

Note: The base state makes sure that the application that you want to test is running and in the foreground. This ensures that tests will always start with the same application state, which makes them more reliable. In order to use the base state, it is necessary to specify what the main window looks like and how to launch the application that you want to test if it is not running. Creating a base state is optional. However, it is recommended as a best practice.

4. Add test classes and methods that test the desired functionality of the test application.

Note: You can use the SilkTest Recorder clipboard feature to quickly copy and paste recorded test steps into a manually created Silk4NET test.

Recording a Test

1. On the Visual Studio menu bar, select **Silk4NET** > **Start Recorder** . The SilkTest Recorder appears.
2. Create a new script and start recording.

   For more information, refer to the *SilkTest Recorder Help*.

The Recorder saves the test actions in a neutral format from which you have the following options:

- Drag-and-drop the recorded test steps into an existing Silk4NET test. The test steps are automatically converted to either Visual Basic .NET or C# depending on your default programming language.
- Export the recorded test steps as a Silk4NET test that you can add to an existing Visual Studio project.
- Export the recorded test steps in a new test and create a new Silk4NET project in which to save the test.

Dragging and Dropping Recorded Test Steps

You must have created a Silk4NET test in which to drag-and-drop the recorded test steps.

1. In Visual Studio, open the Silk4NET test in which to drag-and-drop the recorded test steps.
2. On the Visual Studio menu bar, select **Silk4NET** > **Start Recorder** . The SilkTest Recorder appears.
3. Select **Settings** > **Global Preferences** . The **Preferences** dialog box appears.
4. In the **Default client** list, select your preferred programming language.
5. Click **OK**.
6. Create a new test and start recording. For more information, refer to the *SilkTest Recorder Help*. After stopping the recording, the recorded test steps appear in a neutral format on the **Actions** tab.
7. Select the steps to move into the Silk4NET test.
8. Drag-and-drop the test steps into the Silk4NET test.

The test steps are automatically converted into the default programming language and pasted in the test.

Exporting Recorded Test Steps to Visual Studio

Export a test created by the SilkTest Recorder that you can add to a Visual Studio project. During the export, you can choose to create the test in either Visual Basic .NET or C#. Additionally, you can use the clipboard feature to copy and paste the recorded test steps into an existing Silk4NET test.

1. On the SilkTest Recorder menu bar, choose **File** > **Export** . The **Export** wizard opens.
2. Double-click **Export as NTF Script**. The **Export to NTF** page opens.
3. From the **Export to** list box, select one of the following options:
   - **Clipboard** – Copies the recorded test steps to the clipboard. Choose this option to copy and paste the recorded test steps into an existing Silk4NET test of a Visual Studio project.

     Note: You do not have to specify the source location or base state when selecting this option.
   - **NTF Script** – Exports the recorded test steps as a test you can add to an existing Visual Studio project. Choose this option if you want to create a new test or overwrite an existing test.
4. From the **Programming language** list box, specify whether the test uses Visual Basic .NET or C#.
5. In the **Test method** box, specify a name for the test method. For example, type `TestAutoInput`.
6. In the **Namespace** box, specify the container name for the test.
7. In the **Test class** box, specify the class name to which the test belongs. For example, type `AutoTests`.
8. In the **Source folder** box, specify the location to which to export the test.

   Optionally, click ... and navigate to the folder that you want to use.
9. To include the base state in the exported test, check the **Use base state** check box.

   The base state makes sure that the application that you want to test is running and in the foreground. This ensures that tests will always start with the same application state, which makes them more reliable. In order to use the base state, it is necessary to specify what the main window looks like and how to launch the application that you want to test if it is not running. Creating a base state is optional. However, it is recommended as a best practice.
10. Click **Finish**. Recorder creates a Silk4NET test and exports it to the specified location or to the clipboard.
11. Add the exported Silk4NET project to your Visual Studio project. From the Visual Studio menu bar, choose **Project** > **Add Existing Item** , and then select the exported test.

Exporting a Silk4NET Project to Visual Studio

Export a Silk4NET test and create a new Visual Studio project that uses Visual Basic .NET or C# as the primary programming language.

1. On the SilkTest Recorder menu bar, choose **File** > **Export** . The **Export** wizard opens.
2. Double-click **Export as Silk4NET Project**. The **Export as Silk4NET Project** page opens.
3. From the **Programming language** list box, specify whether the project uses Visual Basic .NET or C#.
4. In the **Project location** text box, specify the location to which to export the project.

   Optional: Click  and navigate to the folder that you want to use.
5. In the **Project name** text box, specify the project name. For example, type `Visual Basic .NET Sample Project`.
6. In the **Namespace** text box, specify the container name for the project.
7. In the **Test class** text box, specify the class name to which the test belongs. For example, type `AutoTests`.
8. In the **Test method** text box, specify a name for the test method. For example, type `TestAutoInput`.
9. Click **Finish**. SilkTest Recorder creates a new project that includes the recorded test and exports the project to the specified location. From this location, you can open the project in Visual Studio.

Running Silk4NET Tests

1. Select **Test** > **Windows** > **Test View** . The **Test View** window opens and displays a list of tests. You can filter the list of projects by selecting filter criteria such as **Project**, **Namespace**, and **Test** class from the **Group by list** located at the top of the window.
2. Select the test(s) to run.
3. Click the **Run selection** button on the **Test View** window toolbar.

> In addition to running tests from the **Test View** window, run commands are also available on the **Test** menu, the context-sensitive menu of an opened test, and the **Test Results** window.

Analyzing Test Results

1. Run a Silk4NET test.
2. On the Visual Studio menu bar, choose **Test** > **Windows** > **Test Results** . The **Test Results** window appears.

The **Test Results** window displays the test run and the status of the run. To view the details of the test run, right-click the run, and then select **View Test Results Details**.

Silk4NET Sample Tests

The Silk4NET sample tests are packaged in a Visual Studio solution which you can open and view as well as run against the SilkTest sample applications.

Download and install the sample applications from *http://supportline.microfocus.com/websync/SilkTest.aspx*. After you have installed the sample applications, click **Start** > **Programs** > **Silk** > **SilkTest** > **Samples** > **Silk4NET Samples** to open the folder containing the Visual Studio solution file (`Silk4NET Samples.sln`) of the Silk4NET sample tests.

In addition to the installed Silk4NET sample applications, the set of Silk4NET sample tests includes several tests for the following SilkTest Web-based sample applications:

Insurance Co. Web site	*http://demo.borland.com/InsuranceWebExtJS/*
Green Mountain Outpost Web	*http://demo.borland.com/gmopost/*

Dynamic Object Recognition

Dynamic object recognition enables you to write test methods that use XPath queries to find and identify objects. Dynamic object recognition uses a `Find` or `FindAll` method to identify an object in a test method. For example, the following query finds the first button with the caption "ok" that is a child of a given window:

```
Dim okButton = window.find("//PushButton[@caption=ok]")
```

Examples of the types of test environments where dynamic object recognition works well include:

- In any application environment where the graphical user interface is undergoing changes.

  For example, to test the **Check Me** check box in a dialog that belongs to a menu where the menu and the dialog name are changing, using dynamic object recognition enables you to test the check box without concern for what the menu and dialog name are called. You can then verify the check box name, dialog name, and menu name to ensure that you have tested the correct component.
- In a Web application that includes dynamic tables or text.

  For example, to test a table that displays only when the user points to a certain item on the Web page, use dynamic object recognition to have the test method locate the table without regard for which part of the page needs to be clicked in order for the table to display.
- In an Eclipse environment that uses views.

  For example, to test an Eclipse environment that includes a view component, use dynamic object recognition to identify the view without regard to the hierarchy of objects that need to open prior to the view.

Benefits of Using Dynamic Object Recognition

The benefits of using dynamic object recognition include:

- Dynamic object recognition uses a subset of the XPath query language, which is a common XML-based language defined by the World Wide Web Consortium, W3C.
- Dynamic object recognition requires a single object rather than a repository of objects for the application that you are testing. Using XPath queries, a test case can locate an object using a `Find` command followed by a supported XPath construct.

XPath Basic Concepts

Silk4NET supports a subset of the XPath query language. For additional information about XPath, see *http://www.w3.org/TR/xpath20/*.

Basic Concepts

XPath expressions rely on the *current context*, the position of the object in the hierarchy on which the `Find` method was invoked. All XPath expressions depend on this position, much like a file system. For example:

- `"//Shell"` finds all shells in any hierarchy relative to the current object.
- `"Shell"` finds all shells that are direct children of the current object.

Additionally, some XPath expressions are context sensitive. For example, `myWindow.find(xPath)` makes `myWindow` the current context.

Supported XPath Subset

Silk4NET supports a subset of the XPath query language. Use a `FindAll` or a `Find` command followed by a supported construct to create a test case.

The following table lists the constructs that Silk4NET supports.

Supported XPath Construct	Sample	Description
Attribute	`MenuItem[@caption='abc']`	Finds all menu items with the given caption attribute in their object definition that are children of the current context. The following attributes are supported: caption (without caption index), priorlabel (without index), windowid.
Index	`MenuItem[1]`	Finds the first menu item that is a child of the current context. Indices are 1-based in XPath.
Logical Operators: and, or, not, =, !=	`MenuItem[not(@caption='a' or @windowid!='b') and @priorlabel='p']`	
.	`TestApplication.Find("//Dialog[@caption='Check Box']/./.")`	Finds the context on which the `Find` command was executed. For instance, the sample could have been typed as `TestApplication.Find("//Dialog[@caption='Check Box']")`.
..	`Desktop.Find("//PushButton[@caption='Previous']/../PushButton[@caption='Ok']")`	Finds the parent of an object. For instance, the sample finds a PushButton with the caption "Ok" that has a sibling PushButton with the caption "Previous."
/	`/Shell`	Finds all shells that are direct children of the current object.  **Note:** "`/Shell`" is equivalent to "`Shell`".
/	`/Shell/MenuItem`	Finds all menu items that are a child of the current object.
//	`//Shell`	Finds all shells in any hierarchy relative to the current object.
//	`//Shell//MenuItem`	Finds all menu items that are direct or indirect children of a Shell that is a direct child of the current object.
//	`//MenuItem`	Finds all menu items that are direct or indirect children of the current context.

Supported XPath Construct	Sample	Description
*	`*[@caption='c']`	Finds all objects with the given caption that are a direct child of the current context.
*	`//MenuItem/*/Shell`	Finds all shells that are a grandchild of a menu item.

The following table lists the XPath constructs that Silk4NET does not support.

Unsupported XPath Construct	Example
Comparing two attributes with each other	`PushButton[@caption = @windowid]`
An attribute name on the right side is not supported. An attribute name must be on the left side.	`PushButton['abc' = @caption]`
Combining multiple XPath expressions with 'and' or 'or'.	`PushButton [@caption = 'abc'] or .// Checkbox`
More than one set of attribute brackets	`PushButton[@caption = 'abc] [@windowid = '123']` (use `PushButton [@caption = 'abc and @windowid = '123']` instead)
More than one set of index brackets	`PushButton[1][2]`

Any construct that does not explicitly specify a class or the class wildcard, such as including a wildcard as part of a class name	`//[@caption = 'abc']`
	(use `//*[@caption = 'abc']` instead)
	`"//*Button[@caption='abc']"`

XPath Samples

The following table lists sample XPath queries and explains the semantics for each query.

XPath String	Description
`desktop.find("/Shell[@caption='SWT Test Application']")`	Finds the first top-level Shell with the given caption.
`desktop.find("//MenuItem[@caption='Control']")`	Finds the MenuItem in any hierarchy with the given caption.
`myShell.find("//MenuItem[@caption!='Control']")`	Finds a MenuItem in any child hierarchy of myShell that does not have the given caption.
`myShell.find("Menu[@caption='Control']/MenuItem[@caption!='Control']")`	Looks for a specified MenuItem with the specified Menu as parent that has myShell as parent.
`myShell.find("//MenuItem[@caption='Control' and @windowid='20']")`	Finds a MenuItem in any child hierarchy of myWindow with the given caption and windowid.
`myShell.find("//MenuItem[@caption='Control' or @windowid='20']")`	Finds a MenuItem in any child hierarchy of myWindow with the given caption or windowid.

XPath String	Description
`desktop.findAll("/Shell[2]/*/PushButton")`	Finds all PushButtons that have an arbitrary parent that has the second top-level shell as parent.
`desktop.findAll("/Shell[2]//PushButton")`	Finds all PushButtons that use the second shell as direct or indirect parent.
`myBrowser.find("//FlexApplication[1]//FlexButton[@caption='ok']")`	Looks up the first FlexButton within the first FlexApplication within the given browser.
`myBrowser.findAll("//td[@class='abc*']//a[@class='xyz']")`	Finds all link elements with attribute class `xyz` that are direct or indirect children of `td` elements with attribute class `abc*`.

Troubleshooting Performance Issues for XPath

When testing applications with a complex object structure, for example complex web applications, you may encounter performance issues, or issues related to the reliability of your scripts. This topic describes how you can improve the performance of your scripts by using different locators than the ones that Silk4NET has automatically generated during recording.

Note: In general, we do not recommend using complex locators. Using complex locators might lead to a loss of reliability for your tests. Small changes in the structure of the tested application can break such a complex locator. Nevertheless, when the performance of your scripts is not satisfying, using more specific locators might result in tests with better performance.

The following is a sample element tree for the application MyApplication:

```
Root
  Node id=1
    Leaf id=2
    Leaf id=3
    Leaf id=4
    Leaf id=5
  Node id=6
    Node id=7
      Leaf id=8
      Leaf id=9
    Node id=9
      Leaf id=10
```

You can use one or more of the following optimizations to improve the performance of your scripts:

- If you want to locate an element in a complex object structure , search for the element in a specific part of the object structure, not in the entire object structure. For example, to find the element with the identifier 4 in the sample tree, if you have a query like `Root.Find("//Leaf[@id='4']")`, replace it with a query like `Root.Find("/Node[@id='1']/Leaf[@id='4']")`. The first query searches the entire element tree of the application for leafs with the identifier 4. The first leaf found is then returned. The second query searches only the first level nodes, which are the node with the identifier 1 and the node with the identifier 6, for the node with the identifier 1, and then searches in the subtree of the node with the identifier 1 for all leafs with the identifier 4.
- When you want to locate multiple items in the same hierarchy, first locate the hierarchy, and then locate the items in a loop. If you have a query like `Root.FindAll("/Node[@id='1']/Leaf")`, replace it with a loop like the following:

```
Public Sub Main()
  Dim node As TestObject

  node = _desktop.Find("//Node[@id='1']")
  For i As Integer = 1 To 4 Step 1
```

```
    node.Find("/Leaf[@id='"+i+"']")
  Next

End Sub
```

xBrowser Frequently Asked Questions

This section includes a collection of questions that you might encounter when testing your Web application.

How do I Verify the Font Type Used for the Text of an Element?

You can access all attributes of the `currentStyle` attribute of a DOM element by separating the attribute name with a ":".

Windows Internet Explorer 8 or earlier	`wDomElement.GetProperty("currentStyle:fontName")`
All other browsers, for example Windows Internet Explorer 9 or later and Mozilla Firefox	`wDomElement.GetProperty("currentStyle:font-name")`

What is the Difference Between textContents, innerText, and innerHtml?

- `textContents` is all text contained by an element and all its children that are for formatting purposes only.
- `innerText` returns all text contained by an element and all its child elements.
- `innerHtml` returns all text, including html tags, that is contained by an element.

Consider the following html code.

```
<div id="mylinks">
  This is my <b>link collection</b>:
  <ul>
    <li><a href="www.borland.com">Bye bye <b>Borland</b> </a></li>
    <li><a href="www.microfocus.com">Welcome to <b>Micro Focus</b></a></li>
  </ul>
</div>
```

The following table details the different properties that return.

Code	Returned Value
`browser.DomElement("//div[@id='mylinks']").GetProperty("textContents")`	`This is my link collection:`
`browser.DomElement("//div[@id='mylinks']").GetProperty("innerText")`	`This is my link collection:Bye bye Borland Welcome to Micro Focus`
`browser.DomElement("//div[@id='mylinks']").GetProperty("innerHtml")`	`This is my <b>link collection</b>: <ul> <li><a href="www.borland.com">Bye bye <b>Borland</b></a></li>`

Code	Returned Value
	Welcome to Micro Focus

I Configured innerText as a Custom Class Attribute, but it Is Not Used in Locators

A maximum length for attributes used in locator strings exists. `InnerText` tends to be lengthy, so it might not be used in the locator. If possible, use `textContents` instead.

What Should I Take Care Of When Creating Cross-Browser Scripts?

When you are creating cross-browser scripts, you might encounter one or more of the following issues:

- Different attribute values. For example, colors in Windows Internet Explorer are returned as `"#FF0000"` and in Mozilla Firefox as `"rgb(255,0,0)"`.
- Different attribute names. For example, the font size attribute is called `"fontSize"` in Windows Internet Explorer 8 or earlier and is called `"font-size"` in all other browsers, for example Windows Internet Explorer 9 or later and Mozilla Firefox.
- Some frameworks may render different DOM trees.

How Can I Distinguish Firefox from Internet Explorer in My Scripts

- The `BrowserApplication` class provides a property `"browsertype"` that returns either "Firefox" or "Internet Explorer". You can add this to a locator in order to define which browser it matches.
- The `BrowserWindow` provides a method `GetUserAgent` that returns the user agent string of the current window.

Which Locators are Best Suited for Stable Cross-Browser Testing?

The built in locator generator attempts to create stable locators. However, it is difficult to generate quality locators if no information is available. In this case, the locator generator uses hierarchical information and indices, which results in fragile locators that are suitable for direct record and replay but ill-suited for stable, daily execution. Furthermore, with cross-browser testing, several AJAX frameworks might render different DOM hierarchies for different browsers.

To avoid this issue, use custom IDs for the UI elements of your application.

Logging Output of My Application Contains Wrong Timestamps

This might be a side effect of the synchronization. To avoid this problem, specify the HTML synchronization mode.

My Test Script Hangs After Navigating to a New Page

This can happen if an AJAX application keeps the browser busy (open connections for Server Push / ActiveX components). Try to set the HTML synchronization mode. Check the *Page Synchronization for xBrowser* topic for other troubleshooting hints.

Recorded an Incorrect Locator

The attributes for the element might change if the mouse hovers over the element. Silk4NET tries to track this scenario, but it fails occasionally. Try to identify the affected attributes and configure Silk4NET to ignore them.

Rectangles Around Elements in Windows Internet Explorer are Misplaced

- Make sure the zoom factor is set to 100%. Otherwise, the rectangles are not placed correctly.
- Ensure that there is no notification bar displayed above the browser window. Silk4NET cannot handle notification bars.

Link.Select Does Not Set the Focus for a Newly Opened Window in Internet Explorer

This is a limitation that can be fixed by changing the Browser Configuration Settings. Set the option to always activate a newly opened window.

DomClick(x, y) Is Not Working Like Click(x, y)

If your application uses the `onclick` event and requires coordinates, the `DomClick` method does not work. Try to use `Click` instead.

FileInputField.DomClick() Will Not Open the Dialog

Try to use `Click` instead.

The Move Mouse Setting Is Turned On but All Moves Are Not Recorded. Why Not?

In order to not pollute the script with a lot of useless `MoveMouse` actions, Silk4NET does the following:

- Only records a `MoveMouse` action if the mouse stands still for a specific time.
- Only records `MoveMouse` actions if it observes activity going on after an element was hovered over. In some situations, you might need to add some manual actions to your script.

I Need Some Functionality that Is Not Exposed by the xBrowser API. What Can I Do?

You can use `ExecuteJavaScript()` to execute JavaScript code directly in your Web application. This way you can build a workaround for nearly everything.

Why Are the Class and the Style Attributes Not Used in the Locator?

These attributes are on the ignore list because they might change frequently in AJAX applications and therefore result in unstable locators. However, in many situations these attributes can be used to identify objects, so it might make sense to use them in your application.

Dialog is Not Recognized During Replay

When recording a script, Silk4NET recognizes some windows as `Dialog`. If you want to use such a script as a cross-browser script, you have to replace `Dialog` with `Window`, because some browsers do not recognize `Dialog`.

For example, the script might include the following line:

```
/BrowserApplication//Dialog//PushButton[@caption='OK']
```

Rewrite the line to enable cross-browser testing to:

```
/BrowserApplication//Window//PushButton[@caption='OK']
```

Index

A

AJAX applications
    script hangs 26

C

Chrome
    cross-browser scripts 25
contact information 4
Customer Care 4

D

Dialog
    not recognized 27
downloads 4
dynamic object recognition
    overview 19
    sample queries 21

F

FAQs
    xBrowser 24
Firefox
    cross-browser scripts 25
    distinguishing from IE 25
    locators 25

I

innerHTML 24
innerText 24, 25
Internet Explorer
    cross-browser scripts 25
    distinguishing from Firefox 25
    link.select focus issue 26
    locators 25

L

locators
    incorrect in xBrowser 26
    xBrowser 25

P

Product Support 4
product updates 5

R

replay
    Dialog not recognized 27

S

serial number 4
SupportLine 4

T

test cases
    sample queries 21
textContents 24
timestamps 26
troubleshooting XPath 22

U

updates 5

W

WebSync 4
Windows Internet Explorer
    misplaced rectangles 26
works order number 4

X

xBrowser
    browser type distinctions 25
    class and style not in locators 27
    cross-browser scripts 25
    Dialog not recognized 27
    DomClick not working like Click 26
    exposing functionality 27
    FAQs 24
    FieldInputField.DomClick not opening dialog 26
    font type verification 24
    innerText not being used in locators 25
    link.select focus issue 26
    mouse move recording 27
    navigating to new pages 26
    recording an incorrect locator 26
    recording locators 25
    textContents, innerText, innerHTML 24
    timestamps 26
    Windows Internet Explorer misplaces rectangles 26
XPath
    basic concepts 19
    overview 19
    samples 21
    supported subset 20
    troubleshooting 22