openspan

OpenSpan Training

Learning the OpenSpan Windows and Scripting Containers for IBM Lotus:

CONTENTS

CONVENTIONS

You can save time using this training guide by understanding how screen elements, input data, and definitions are shown.

Convention	Meaning
Black bold characters	**Names of program elements that require emphasis, such as command buttons, menus, and dialog boxes, are shown in black bold text.**
Blue Bold Characters	**Text that you are supposed to type or data selections, such as from drop-lists, appear in blue boldface characters.**
Remember	**Definitions of terms and important concepts that bear remembering.**

This page intentionally left blank.

CHAPTER 1: Introduction to the OpenSpan Windows Container

This step-by-step guide provides an in-depth look at how to use the OpenSpan Windows Container for IBM Lotus to integrate Windows applications into your Lotus Expeditor composite applications. The OpenSpan Windows Container (OWC) exposes the functionality of Windows applications, isolates and uniquely identifies application controls with which you want to interact, and integrates and automates Windows applications in your composite applications.

What you do with the OpenSpan Windows Container

- Add it to your Composite application from the Component Palette **Containers** folder.
- Associate it with a Windows application by setting Container properties, such as the path to the application executable file.
- Interrogate the Windows application. This user-driven process is where you identify the Windows application components you want to use in your composite applications.

What the OpenSpan Windows Container does for you

- Once interrogated, your applications' user interface controls are automatically detected and uniquely identified (or matched) each time the application runs.
- Exposes controls to the Lotus Expeditor designer for use in wired composite applications.
- Smoothly integrates data between disparate applications based on your own configurable rules.

How the OpenSpan Windows Container does it

When a desktop application is running, it is actually the operating system that manages the user interface (GUI) — Text boxes, buttons, menus, toolboxes, icons, images, links, mouse clicks, events, etc. The operating system and application communicate with each other at all times, constantly passing data and events back and forth. OpenSpan technology intercepts and deciphers the communications between the operating system and desktop application, allowing you to control and manipulate Windows applications within your composite applications.

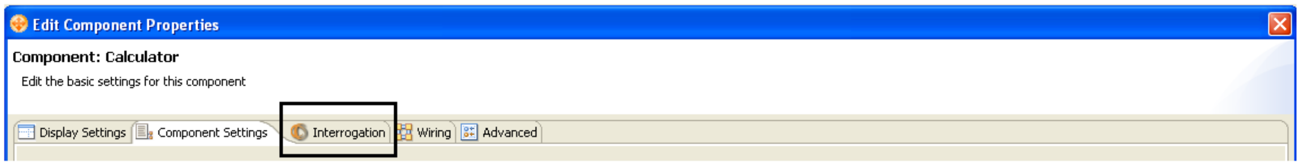
Getting to know the OpenSpan Windows Container Interface Features

The OWC provides an interface for interrogating and integrating Windows applications. Once the container is installed and a container component is added to your composite application, you access the OpenSpan Windows Container functionality from the Lotus Expeditor **Edit Component Properties** window.

The best way to learn to use the OpenSpan Windows Container is from the step-by-step exercises presented in this tutorial. However, before getting started with the exercises you should be familiar with the names and locations of the OpenSpan Windows Container tools and components that are referenced repeatedly in the exercises.

Note: This first glance at the OpenSpan Windows Container is intended only to help you become familiar with the component and its parts. How and why they are used is described in the exercises that follow.

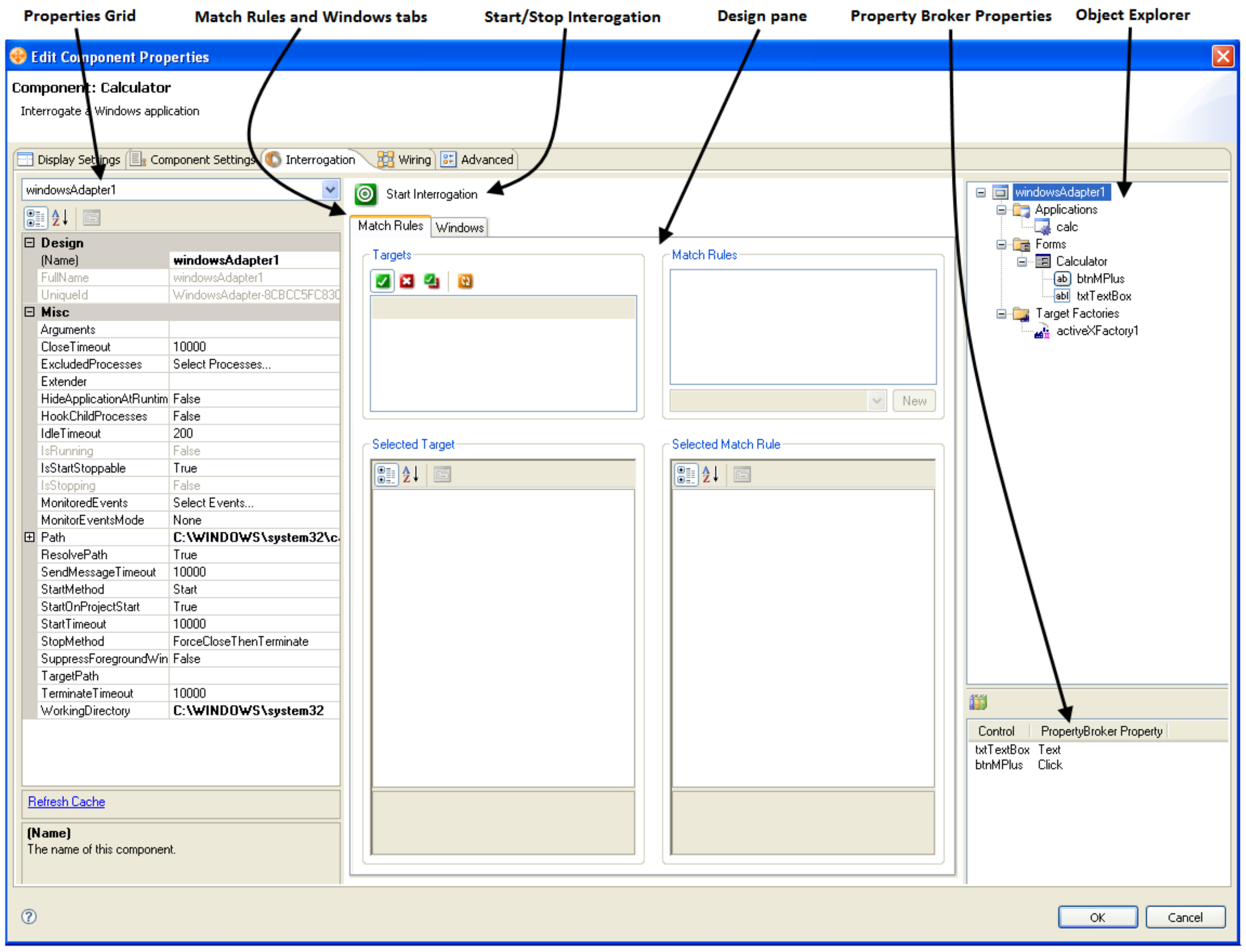
The OpenSpan Windows Container adds a tab labeled **Interrogation** to the Lotus Expeditor **Edit Component Properties** window. Note that the Interrogation tab replaces the Landmarks tab that is usually present on the **Edit Component Properties** window.


Edit Component Properties

Component: Calculator

Edit the basic settings for this component

Below is the **Interrogation** tab with the major screen elements identified:



Component: Calculator

Interrogate a Windows application

windowsAdapter1

Design	
(Name)	**windowsAdapter1**
FullName	windowsAdapter1
UniqueId	WindowsAdapter-8CBCC5FC83C
Misc	
Arguments	
CloseTimeout	10000
ExcludedProcesses	Select Processes...
Extender	
HideApplicationAtRuntim	False
HookChildProcesses	False
IdleTimeout	200
IsRunning	False
IsStartStoppable	True
IsStopping	False
MonitoredEvents	Select Events...
MonitorEventsMode	None
Path	**C:\WINDOWS\system32\c**
ResolvePath	True
SendMessageTimeout	10000
StartMethod	Start
StartOnProjectStart	True
StartTimeout	10000
StopMethod	ForceCloseThenTerminate
SuppressForegroundWin	False
TargetPath	
TerminateTimeout	10000
WorkingDirectory	**C:\WINDOWS\system32**

(Name)
The name of this component.

- windowsAdapter1
  - Applications
    - calc
  - Forms
    - Calculator
      - btnMPlus
      - txtTextBox
  - Target Factories
    - activeXFactory1

Control	PropertyBroker Property
txtTextBox	Text
btnMPlus	Click

This page intentionally left blank.

CHAPTER 2: Main Features and Common Tasks of the OWC

This step-by-step examples that follow cover the main features of the OpenSpan Windows Container (OWC) and OpenSpan Scripting Container (OSC), and walks you through some of the most common tasks required to create composite applications using the OpenSpan Windows and Scripting Containers. By the end of these exercises, you'll be more familiar with the OpenSpan Windows Container and the OpenSpan Scripting Container components, and will be able to get started creating your first composite application.

Note: The exercises that follow require a working knowledge of Lotus Expeditor, including the ability to build wires and program using JavaScript.

Exercise 1 – Using the OpenSpan Windows Container in a Composite Application

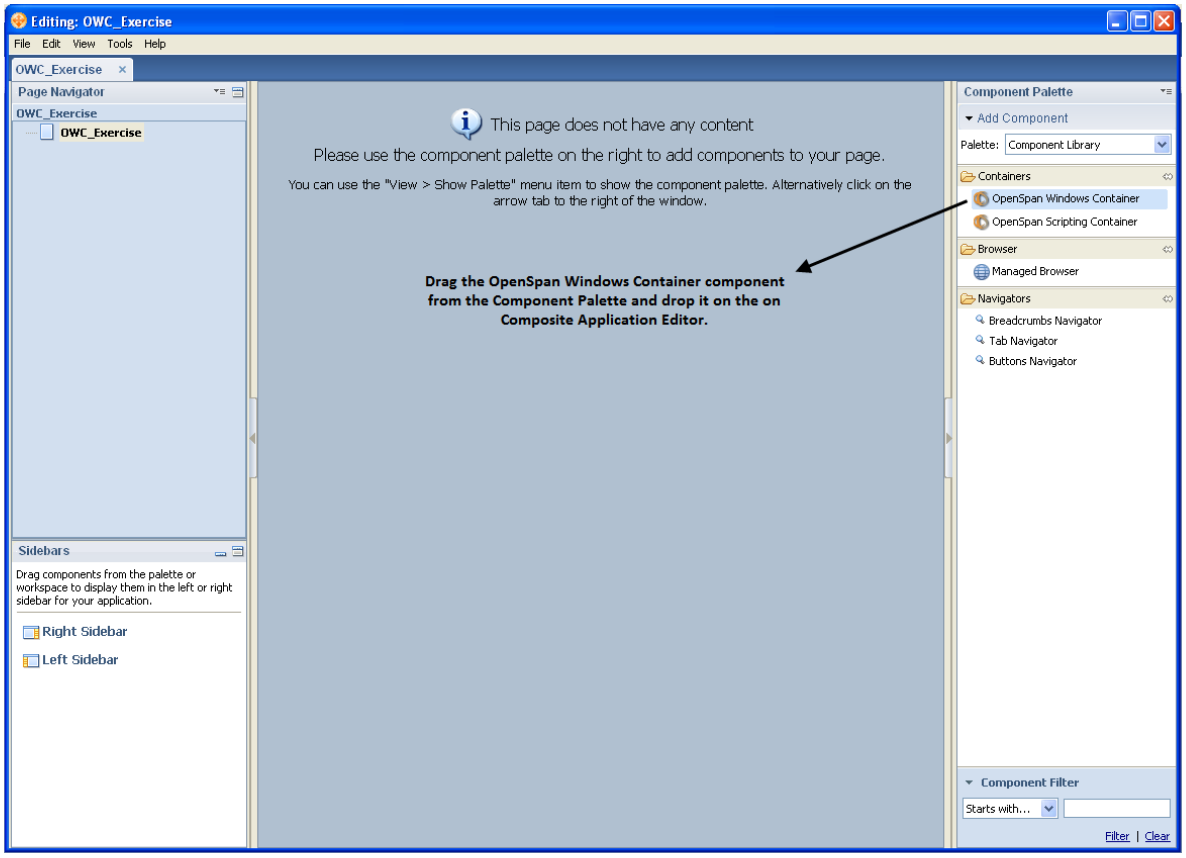
The following exercise builds a Composite Application (CA) using a sample Windows application installed with the OpenSpan Windows Container, named **MiniCRM**. The exercise also uses an OpenSpan training website named the **ACME Products Search System.** The exercise will:

- Change the label of an existing button in the MiniCRM application (from **OK** to **Find Store).**
- Change the behavior of that same button from closing the MiniCRM application to copying the MiniCRM Zip Code field entry to an equivalent field on the**, Locator** page of the **ACME Products Search System.**
- Initiate a search for the ACME Products store closest to the address (based on the zip code) that is displayed in the MiniCRM application.
- The CA will then copy and display the store address to the **Note**s field of the MiniCRM application.

Adding the OpenSpan Windows Container to a Composite Application

1. In Lotus Expeditor, create a new composite application by selecting **File** | **Application** | **New Composite Application**.
2. In the **New Composite Application** dialog, type OWC_Exercise in the **File name** field, enter a folder location or accept the default, and then click **OK**.

3. Add an OpenSpan Windows Container (OWC) to the composite application by dragging an OpenSpan Windows Container component from the **Component Palette** and dropping it on the Composite Application Editor (CAE).



This page does not have any content

Please use the component palette on the right to add components to your page.

You can use the "View > Show Palette" menu item to show the component palette. Alternatively click on the arrow tab to the right of the window.

Drag components from the palette or workspace to display them in the left or right sidebar for your application.

Drag the OpenSpan Windows Container component from the Component Palette and drop it on the on Composite Application Editor.

4. An OpenSpan Windows Container component appears in the editor and is added to the **Page Navigator** tree view.
5. Edit the **OpenSpan Windows Container** by right-clicking the component in the **Page Navigator** tree view and selecting **Edit Component Properties** from the context menu.
6. In the **Edit Component Properties** window, select the **Component Settings** tab.

OWC Component Settings

7. In the **Title** field, enter a unique name for the OpenSpan Windows Container component. You should use a name relevant to the application you are using in the composite application. Note that naming the component does not set any integration properties so it does not need to be exactly the same as the application name. However, it should easily identify the application to other users.

   <u>Remember:</u> Component names should be meaningful and relevant and should be unique. It is strongly encouraged that you give each OpenSpan Windows Container a meaningful and unique name that is different than any other component in this or any other composite application.

For this exercise, enter CRM in the **Title** field. CRM, or rather MiniCRM, is a sample application installed automatically with the OpenSpan Windows Container. The MiniCRM application is installed to the root of the installation drive and into a folder named **MiniCRM,** such as **C:\MiniCRM**.

8. Optionally enter a description for the application.
9. Component options:
   - **Reparent App While Running** causes all application top-level windows to be reparented into the container window. **True** is the default option. Setting this option to **False** causes the application windows to open independent of the composite application window. For this exercise accept the default setting **True**.
   - **Autostart App In Editor** causes the target application to open when the composite application is started. **False** is the default option – meaning the target application is not started automatically and must be started manually by right-clicking in Composite Application Editor and selecting **Start Application** from the context menu. Note that the application will also start when the **Start Interrogation** command button is selected. Select **True** for this exercise.
   - **Target Environment** property allows you to control whether or not the OpenSpan driver is used by the OpenSpan Windows Container. Use the OpenSpan driver when the target application (the application you need to monitor and/or control) is not directly launched by the component, such as when the target application is launched by the application started by the component or by a desktop shortcut selected by the user.

     There are two **Target Environment** settings:
     - **Driver:** This default option loads the OpenSpan driver when the project starts. Use the OpenSpan driver when the application started by an OpenSpan Windows Container component, as defined by the component's **Path** property, is not the application you need to control or is not the only application used. This setting makes all of the Start methods available: **Start**, **Start and Wait,** and **MonitorAll**. For this exercise accept the default option **Driver.**
     - **NoDriver**: The OpenSpan driver is not loaded when the project starts. Use this setting when the process specified by the **Path** property for all project adapters are the only processes used in the project. Note that the **NoDriver** option limits the Windows application Start Method only to **Start**.
10. Open the **Interrogation** tab.

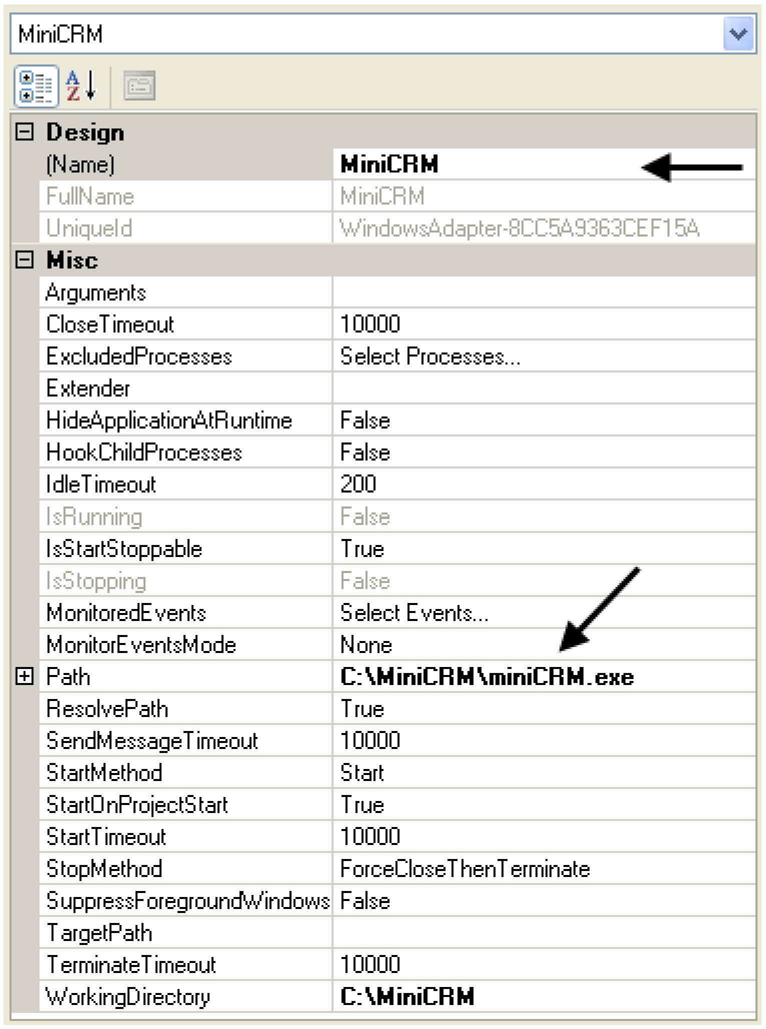
OWC Properties

11. In the **Properties** grid for the OpenSpan Windows component (e.g., windowsAdapter1):
    a. Change the **Design (Name)** field entry from **windowsAdapter1** to **MiniCRM**.

       Note: The adapter name should be changed to something recognizable and unique that is different than any other component in this or any other composite application.

    b. In the **Path** field, enter the path to the application you wish to interrogate and use in your composite application. For this example we will use the sample CRM application installed with the OpenSpan Windows Container, usually to **C:\MiniCRM\MiniCRM.exe**.

       Note: You can display the properties either alphabetically or categorized by selecting the associated icon at the top of the page.

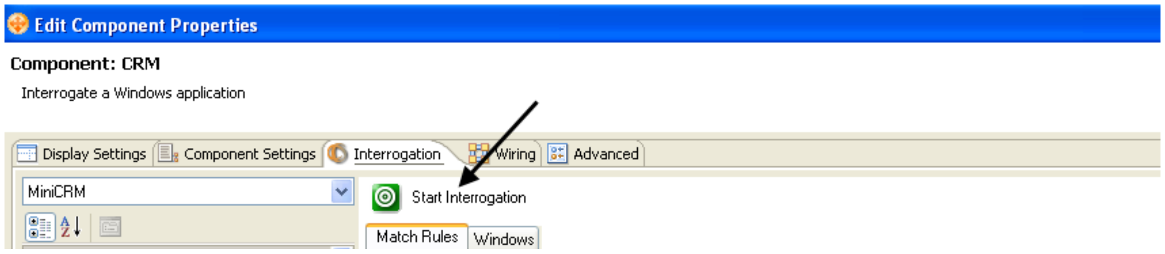


MiniCRM

Design	
(Name)	**MiniCRM**
FullName	MiniCRM
UniqueId	WindowsAdapter-8CC5A9363CEF15A
Misc	
Arguments	
CloseTimeout	10000
ExcludedProcesses	Select Processes...
Extender	
HideApplicationAtRuntime	False
HookChildProcesses	False
IdleTimeout	200
IsRunning	False
IsStartStoppable	True
IsStopping	False
MonitoredEvents	Select Events...
MonitorEventsMode	None
Path	**C:\MiniCRM\miniCRM.exe**
ResolvePath	True
SendMessageTimeout	10000
StartMethod	Start
StartOnProjectStart	True
StartTimeout	10000
StopMethod	ForceCloseThenTerminate
SuppressForegroundWindows	False
TargetPath	
TerminateTimeout	10000
WorkingDirectory	**C:\MiniCRM**

Interrogating a Windows Application

12. Interrogate the CRM application by clicking the **Start Interrogation** button at the top of the Design pane. The Interrogate function allows you to choose the controls/components in the Windows application that you want to use in the OpenSpan Windows Container.



The MiniCRM application launches and the **Interrogation Form** dialog opens.

Important: A **Start Interrogation** button also displays on the Composite Application Editor (CAE). That method does not allow access to the component's properties and match rules, and should be used only when exploring whether a target application's controls can be interrogated and not during composite application design. For composite application design you should always use the Interrogation method launched from the **Interrogation** tab, as shown in above.

13. Click the bulls-eye shaped icon in the **Interrogation Form** dialog, hold the mouse button down, and drag the icon over the control you want (e.g., **Zip Code** text box) to interrogate. When the control is highlighted (i.e., black outline appears around the control), release the mouse button.

    Interrogate the following MiniCRM application controls:

    a. **Notes** text box
    b. **OK** button
    c. **Account Number** text box
    d. **Zip Code** text box

    Important: As you interrogate an object the designer creates a new control corresponding to the selected target and adds it to the **Object Explorer** (i.e., the tree view on the right side of the **Edit Component Properties** window). If a target has parent elements that have not been interrogated, the designer also creates controls corresponding to these elements. The designer creates a set of match rules that uniquely identify the target for each control.

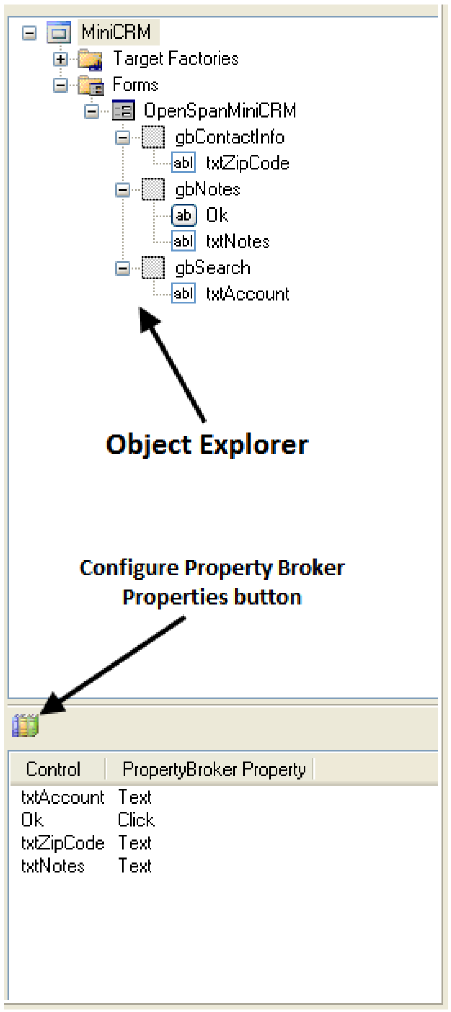
14. Stop interrogation by closing the MiniCRM application or clicking the **Stop Interrogation** button.

OWC Object Explorer

The Object Explorer, which is the tree view that displays adjacent-right of the Design pane, lists all of the interrogated targets and contains functions for exposing the properties of the objects. While interrogating an application, objects are displayed with a green icon indicator to show they are matched.

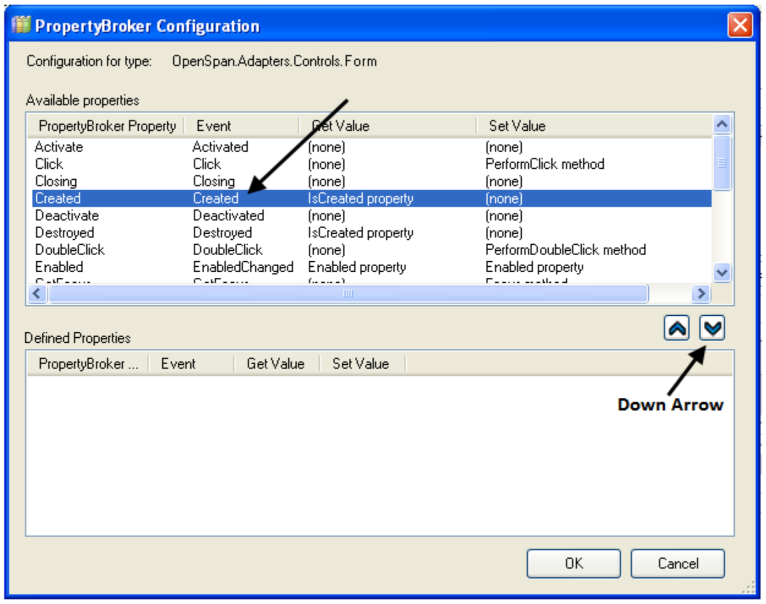
15. In the Object Explorer, select the **OpenSpanMiniCRM** object under **Forms** and then click the **Configure Property Broker Properties** button.

    This opens the **PropertyBroker Configuration** window.



Control	PropertyBroker Property
txtAccount	Text
Ok	Click
txtZipCode	Text
txtNotes	Text

16. In the **PropertyBroker Configuration** dialog, select the **Created** property and click the down arrow to add it to the **Defined Properties** list box. Then click **OK** to save your changes and return focus to the **Edit Component Properties** window.


PropertyBroker Configuration

Configuration for type: OpenSpan.Adapters.Controls.Form

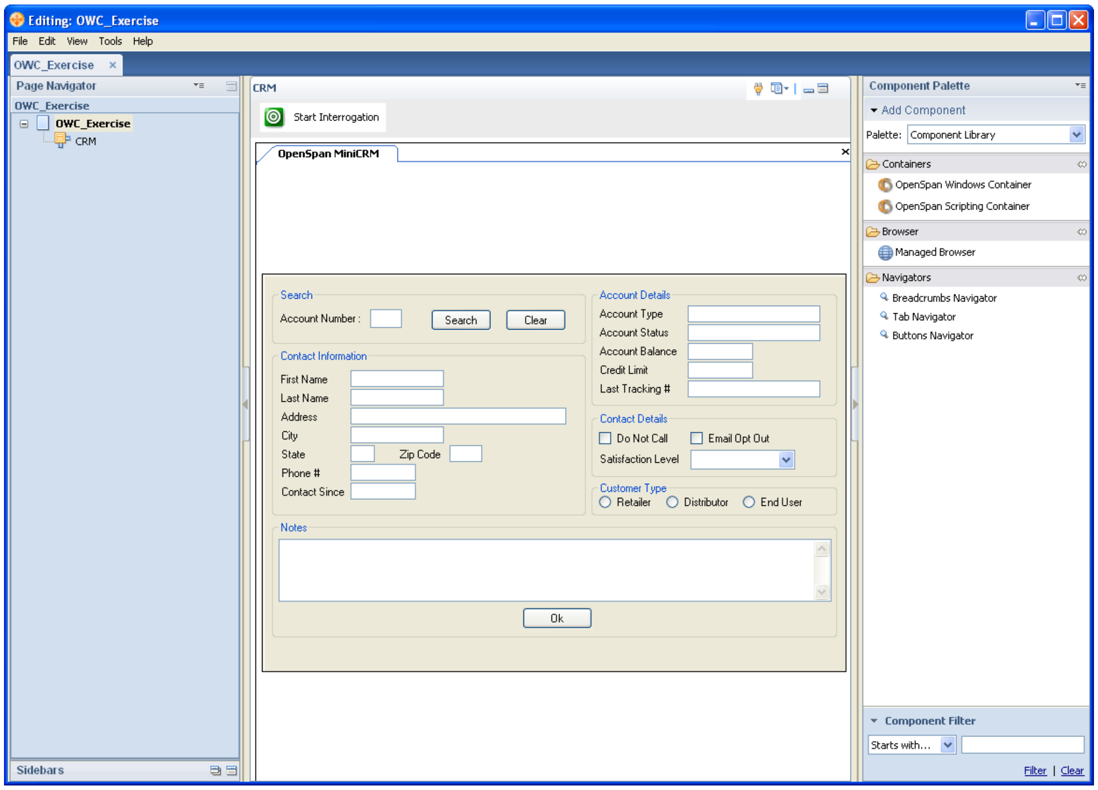
Available properties

PropertyBroker Property	Event	Get Value	Set Value
Activate	Activated	(none)	(none)
Click	Click	(none)	PerformClick method
Closing	Closing	(none)	(none)
Created	Created	IsCreated property	(none)
Deactivate	Deactivated	(none)	(none)
Destroyed	Destroyed	IsCreated property	(none)
DoubleClick	DoubleClick	(none)	PerformDoubleClick method
Enabled	EnabledChanged	Enabled property	Enabled property

Defined Properties

PropertyBroker ...	Event	Get Value	Set Value

17. In the **Edit Component Properties** window, click **OK** to save the selections and entries you made in the **Component Settings** tab, **Interrogation** tab, and **Property Broker Configuration** dialog, and to close the **Edit Component Properties** window.



This page intentionally left blank.

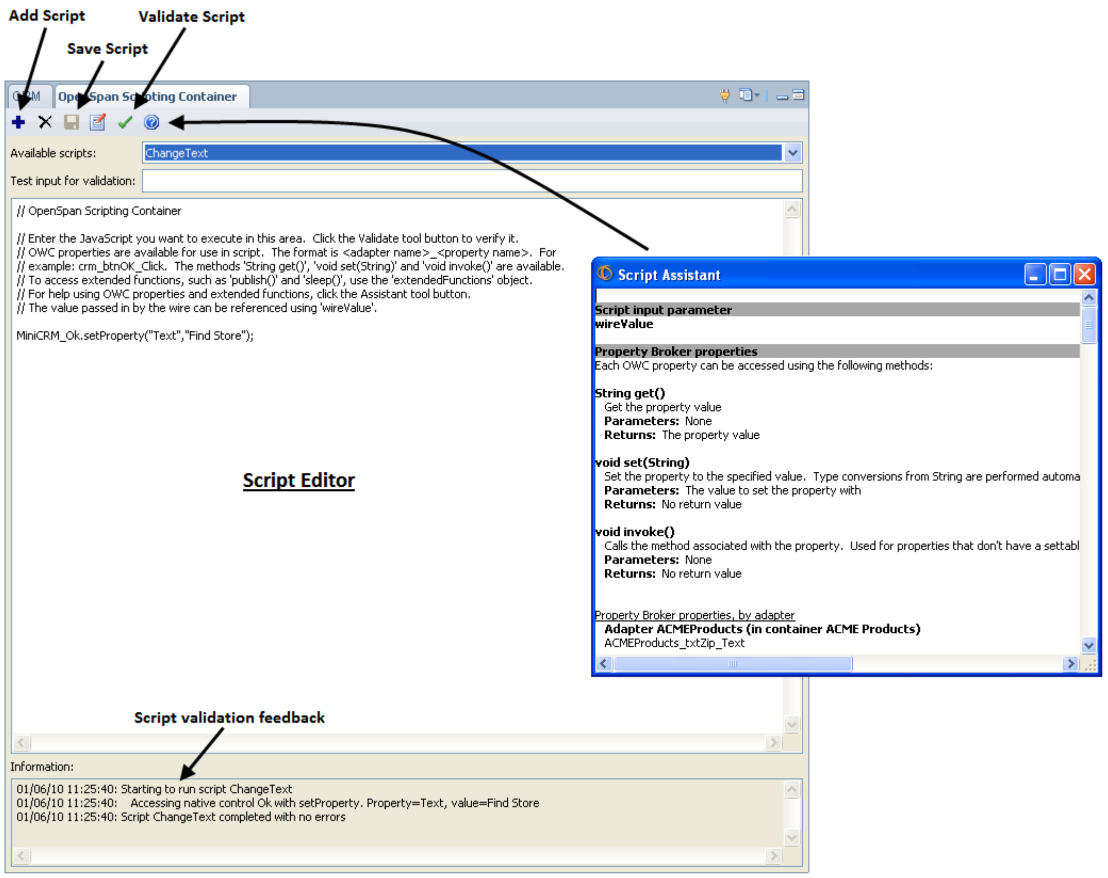
CHAPTER 3: Working with the OpenSpan Scripting Container

The OpenSpan Scripting Container (OSC) is used to create scripts (written in JavaScript) that can be executed via wires in a composite application. Additionally, each script has the ability to directly interact with any OpenSpan Windows Container, using Property Broker properties, or native control access. Scripts are used to automate applications. For example, navigating through a series of menus automatically or copying account information from one application to another when a change is made.

Getting to know the OpenSpan Scripting Container Interface Features

The OpenSpan Scripting Container toolbar contains the following command buttons:

- **Add script** – Adds a script to the OSC and launches the **Add Script Name** dialog.
- **Remove script** – Deletes the currently selected script.
- **Save script**
- **Rename script**
- **Validate script –** Finds errors in your JavaScript code.
- **Script Assistant** – Opens the **Script Assistant** window.



Available scripts: ChangeText

Test input for validation:

```
// OpenSpan Scripting Container

// Enter the JavaScript you want to execute in this area.  Click the Validate tool button to verify it.
// OWC properties are available for use in script.  The format is <adapter name>_<property name>.  For
// example: crm_btnOK_Click.  The methods 'String get()', 'void set(String)' and 'void invoke()' are available.
// To access extended functions, such as 'publish()' and 'sleep()', use the 'extendedFunctions' object.
// For help using OWC properties and extended functions, click the Assistant tool button.
// The value passed in by the wire can be referenced using 'wireValue'.

MiniCRM_Ok.setProperty("Text","Find Store");
```

Script input parameter
wireValue

Property Broker properties
Each OWC property can be accessed using the following methods:

String get()
Get the property value
Parameters: None
Returns: The property value

void set(String)
Set the property to the specified value. Type conversions from String are performed automa
Parameters: The value to set the property with
Returns: No return value

void invoke()
Calls the method associated with the property. Used for properties that don't have a settabl
Parameters: None
Returns: No return value

Property Broker properties, by adapter
Adapter ACMEProducts (in container ACME Products)
ACMEProducts_txtZip_Text

Information:

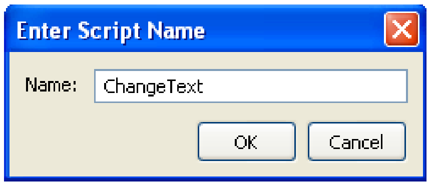
```
01/06/10 11:25:40: Starting to run script ChangeText
01/06/10 11:25:40:    Accessing native control Ok with setProperty. Property=Text, value=Find Store
01/06/10 11:25:40: Script ChangeText completed with no errors
```

Following the OpenSpan Scripting Container toolbar are the fields:

- **Available Scripts** - Drop-list containing the scripts created for the selected OSC.
- **Test Input for Validation** – If the wire that executes your script has a data value that the script will use, you can use the **Test Input for Validation** field to test how the script will work with a value.
- The Script Editor window appears in the center of the screen and th**e Information** field, directly below the Script Editor window, provides real-time feedback when you validate your JavaScript**.**

Adding the OpenSpan Scripting Container to a Composite Application

18. Add an OpenSpan Scripting Container (OSC) to the Lotus Expeditor editor by dragging an **OpenSpan Scripting Container** component from the **Component Palette** and dropping it on the **Composite Application Editor** (CAE).
19. Select the newly added OpenSpan Scripting Container in the **Page Navigator** tree view.
20. Click the Scripting Container toolbar **+** button to add a script, which opens the **Enter Script Name** dialog. Name the script ChangeText and then click **OK**.

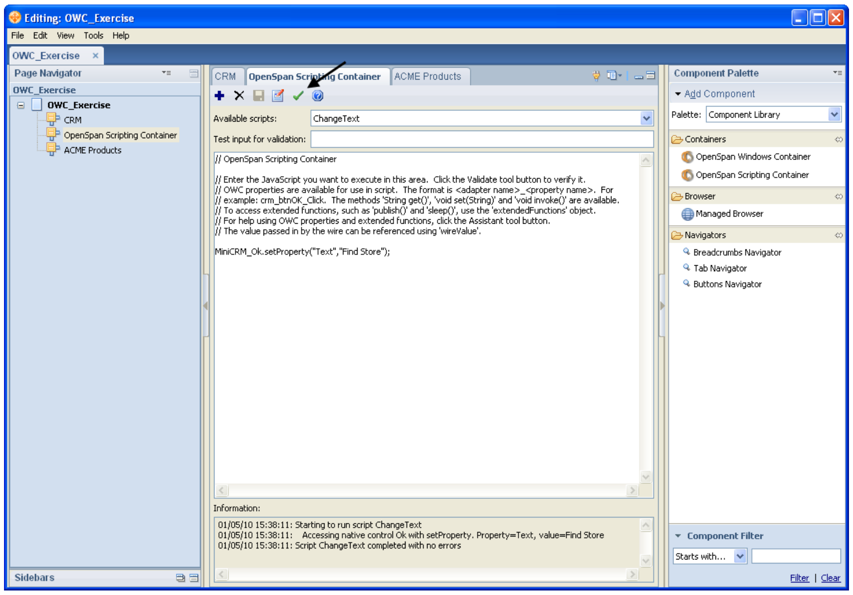

Enter Script Name

Name: ChangeText

Note: You can add an unlimited number of scripts to a single OpenSpan Scripting Container.

21. Add the following line to the script
MiniCRM_Ok.setProperty("Text", "Find Store");

22. Save and then validate the script by selecting the green checkmark icon on the OpenSpan Scripting Container toolbar. The information pane located below the script displays the results:



Available scripts: ChangeText

Test input for validation:

```
// OpenSpan Scripting Container

// Enter the JavaScript you want to execute in this area.  Click the Validate tool button to verify it.
// OWC properties are available for use in script.  The format is <adapter name>_<property name>.  For
// example: crm_btnOK_Click.  The methods 'String get()', 'void set(String)' and 'void invoke()' are available.
// To access extended functions, such as 'publish()' and 'sleep()', use the 'extendedFunctions' object.
// For help using OWC properties and extended functions, click the Assistant tool button.
// The value passed in by the wire can be referenced using 'wireValue'.

MiniCRM_Ok.setProperty("Text","Find Store");
```

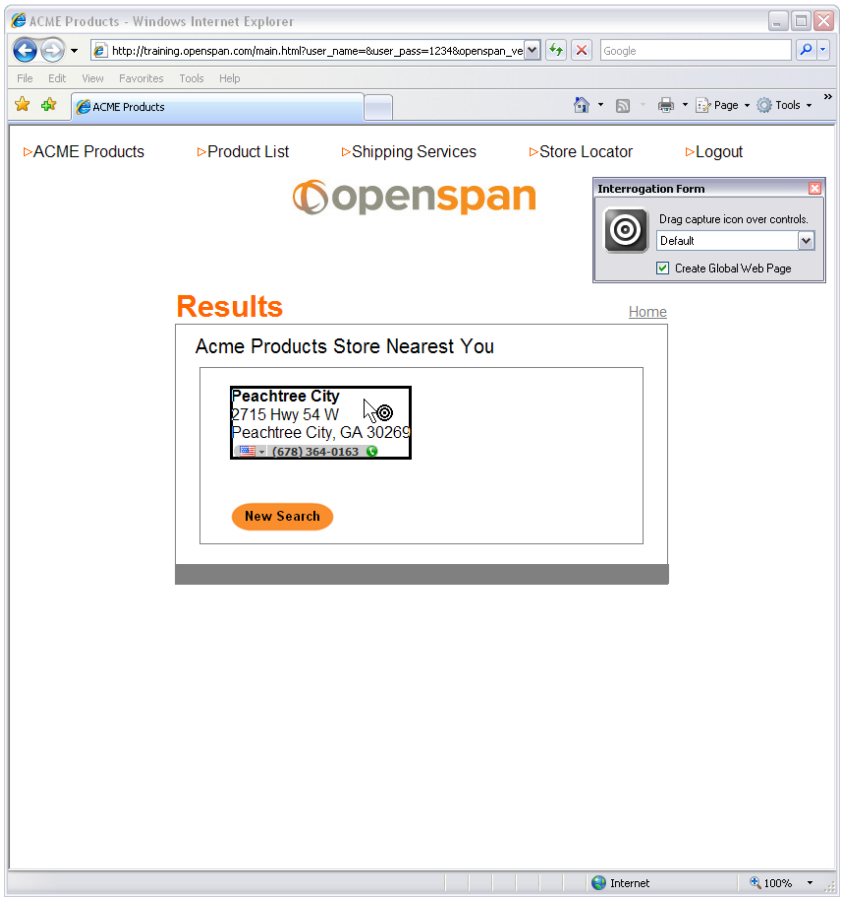
Information:

```
01/05/10 15:38:11: Starting to run script ChangeText
01/05/10 15:38:11:    Accessing native control Ok with setProperty. Property=Text, value=Find Store
01/05/10 15:38:11: Script ChangeText completed with no errors
```

23. In the **Page Navigator** tree view, select the **CRM** OWC.
24. Click the **Wires** button to open the **Create New Wire** dialog.
    a. **Source Component**: CRM→Select the **OpenSpanMiniCRM_Created** property.
    b. **Target Component:** OpenSpan Scripting Container → **Set ChangeText** property.
    c. Click the **Add Wire** button.
    d. Click **OK** to save the wire and close the **Create New Wire** dialog.
25. Add a second OpenSpan Windows Container (OWC) to the composite application.
26. Right-click the new OWC in the **Page Navigator** tree view and select **Edit Component Properties** from the context menu.
27. Open the **Component Settings** tab
    a. Enter the name **ACME Products**.
    b. Select **True** for the **Autostart App In Editor** option.
    c. Accept the default settings for the **Reparent App While Running** (True) and **Target Environment** (Driver) options.
28. Open the **Interrogation** tab.
29. Set the following properties:
    a. Change the **Design (Name)** to **ACMEProducts**.
    b. Enter in the **Path** property the full path to the location of the Microsoft Internet Explorer executable file **iexplore.exe**.
    c. Enter the **Argument** **-new http://training.openspan.com/main.html**. The leading ***–new*** in the argument creates a new thread for the adapter when it is started.

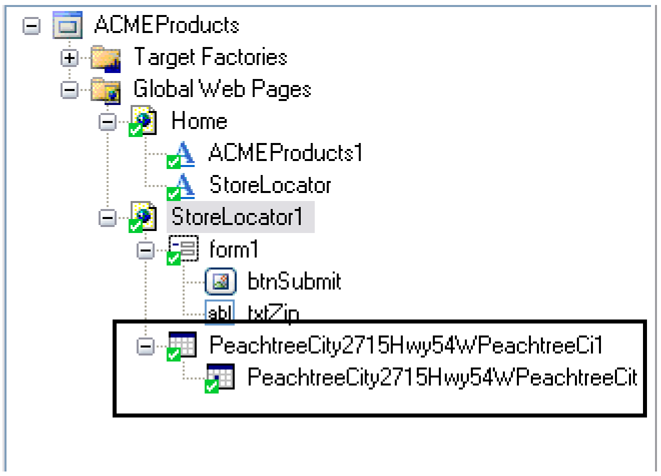
Note: If you are using Internet Explorer 8, you must set the **HookChildProcesses** property to **True**.

30. Click the **Start Interrogation** button. In the **Interrogation Form** dialog, click the **Create Global Web Page** checkbox. Interrogate the following controls:
    a. The **ACME Products** menu on the **ACME Product Search System** page.
    b. **Store Locator** menu. After interrogating the **Store Locator** menu, click it to advance to the **Locator** page.
    c. Interrogate the **Zip Cod**e text box on the **Locator** page. Enter zip code **30022**.
    d. Interrogate the **Find Store** button on the **Locator** page. Click the **Find Store** button to advance to the **Results** page.
    e. Interrogate the search results. It is important that you interrogate only the search results and not the surrounding controls, as shown in the following screen capture:


ACME Products - Windows Internet Explorer
http://training.openspan.com/main.html?user_name=&user_pass=1234&openspan_ve
Google
File Edit View Favorites Tools Help
ACME Products
Page Tools
▹ACME Products
▹Product List
▹Shipping Services
▹Store Locator
▹Logout
openspan
Interrogation Form
Drag capture icon over controls.
Default
Create Global Web Page
Results
Home
Acme Products Store Nearest You
New Search
Internet
100%

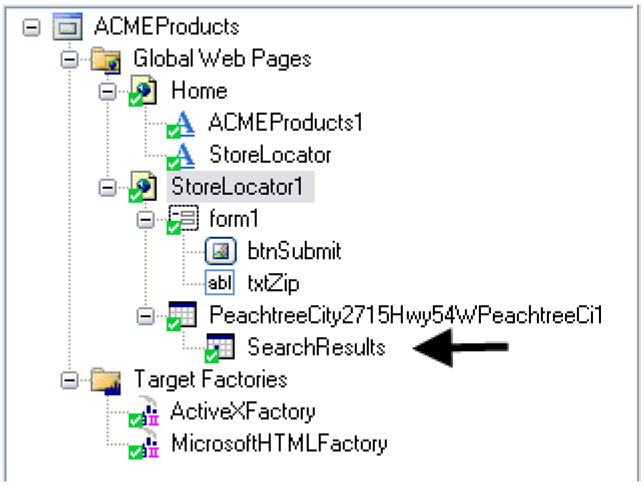
Peachtree City
2715 Hwy 54 W
Peachtree City, GA 30269
(678) 364-0163

31. Interrogating the store location results adds two items to the Object Explorer.
    - HTML table
    - HTML table cell (child of the table)



32. Shortly in this exercise, you will create a script that copies the results data from the ACME Web page to the MiniCRM application. Confirm that the table cell shown in the Object Explorer represents and matches the store location results by right-clicking it in the Object Explorer and selecting **Highlight** from the context menu. A rectangle will flash around the item on the **Results** Web page.
33. Rename the table cell in the Properties grid to **SearchResults**.

    Your Object Explorer should appear as follows:



34. Stop Interrogation by closing the **Interrogation Form** dialog box.
35. Click **OK** in the **Edit Component Properties** window to save your changes and close the window.

36. Add a script to the OSC you added in step 18. Name the new script **FindStore**.
37. Add the following lines to the script:

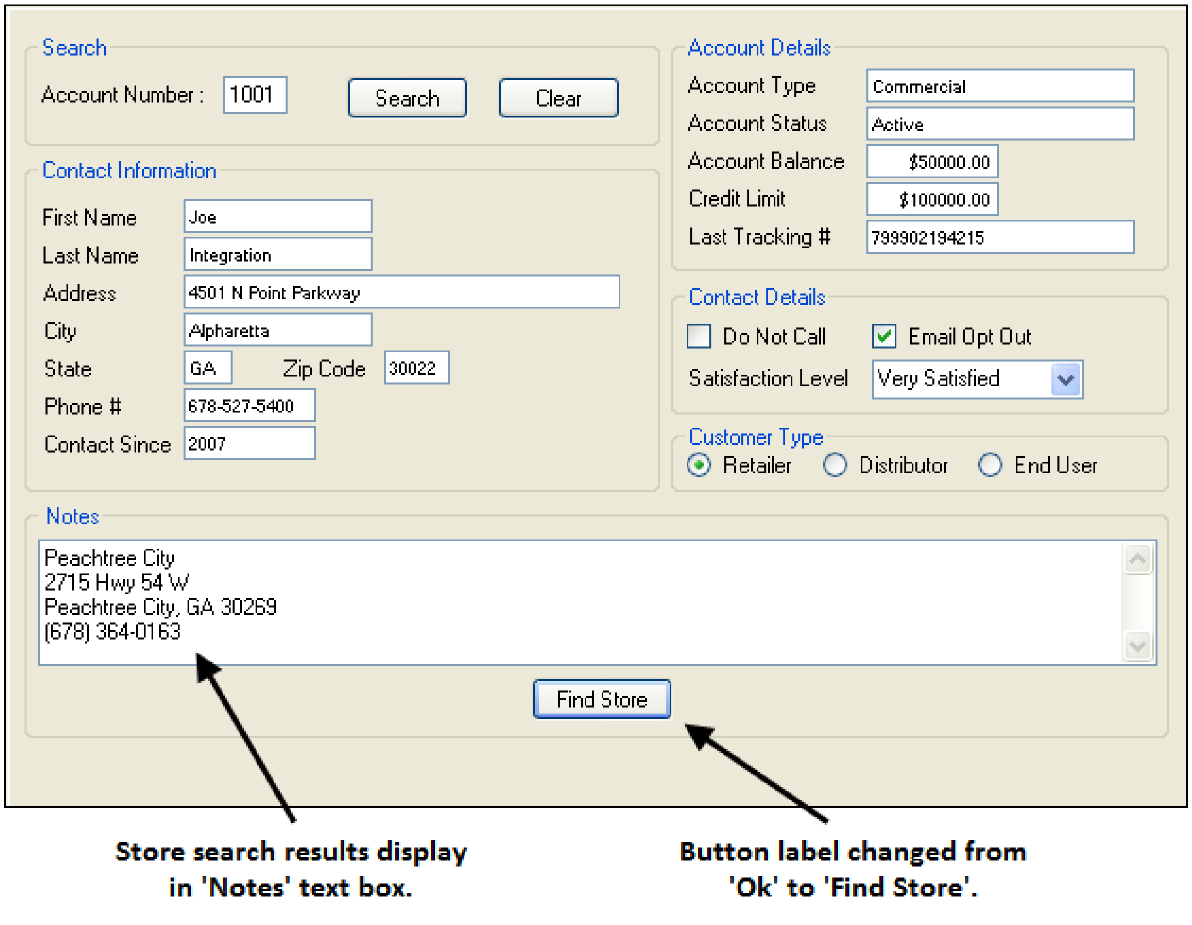
```
var zipCode, results;

// Get the Zip Code from Mini CRM
zipCode = MiniCRM_txtZipCode.getProperty("Text");

// Navigate and search for results in ACME Store Locator
ACMEProducts_StoreLocator.invokeMethod("PerformClick");
ACMEProducts_txtZip.setProperty("Text",  zipCode);
ACMEProducts_btnSubmit.invokeMethod("PerformClick");

//Capture results and add text to Mini CRM
results = ACMEProducts_SearchResults.getProperty("Text");
MiniCRM_txtNotes.setProperty("Text", results);

// Navigate back to ACME Products Search System home page
ACMEProducts_ACMEProducts1.invokeMethod("PerformClick");
```

38. The next step is to validate the script and before doing that the MiniCRM application must be populated with account data. In the **Page Navigator** tree view, select the **CRM** OWC. Enter **1001** in the **Account Number** field and click the **Search** button.
39. Return to the OpenSpan Scripting Container. Save and then validate the script.

40. Select the CRM OWC in **Page Navigator** tree view. Click the **Wires** command button. Add a wire:
    a. **Source Component:** CRM → OK_Click
    b. **Target Component:** OpenSpan Scripting Container → Set FindStore
    c. Click the **Add Wire** button.
    d. Click **OK** to save the wire and close the **Create New Wire** dialog.

Test the Composite Application - Search for a Store Location

41. Close the Component Application Editor window and save your changes.
42. In the Lotus Expeditor, enter Account number 1001 in the OpenSpan MiniCRM application:
43. Click the **Search** command button to populate the MiniCRM window.
44. Click the **Find Store** command button to initiate the store location search.
45. Results of the search display in the **Notes** text box.



Account Number : 1001

First Name	Joe
Last Name	Integration
Address	4501 N Point Parkway
City	Alpharetta
State	GA
Zip Code	30022
Phone #	678-527-5400
Contact Since	2007

Account Type	Commercial
Account Status	Active
Account Balance	$50000.00
Credit Limit	$100000.00
Last Tracking #	799902194215

☐ Do Not Call ☑ Email Opt Out
Satisfaction Level: Very Satisfied

[✓ Retailer] Distributor End User

Notes:
Peachtree City
2715 Hwy 54 W
Peachtree City, GA 30269
(678) 364-0163

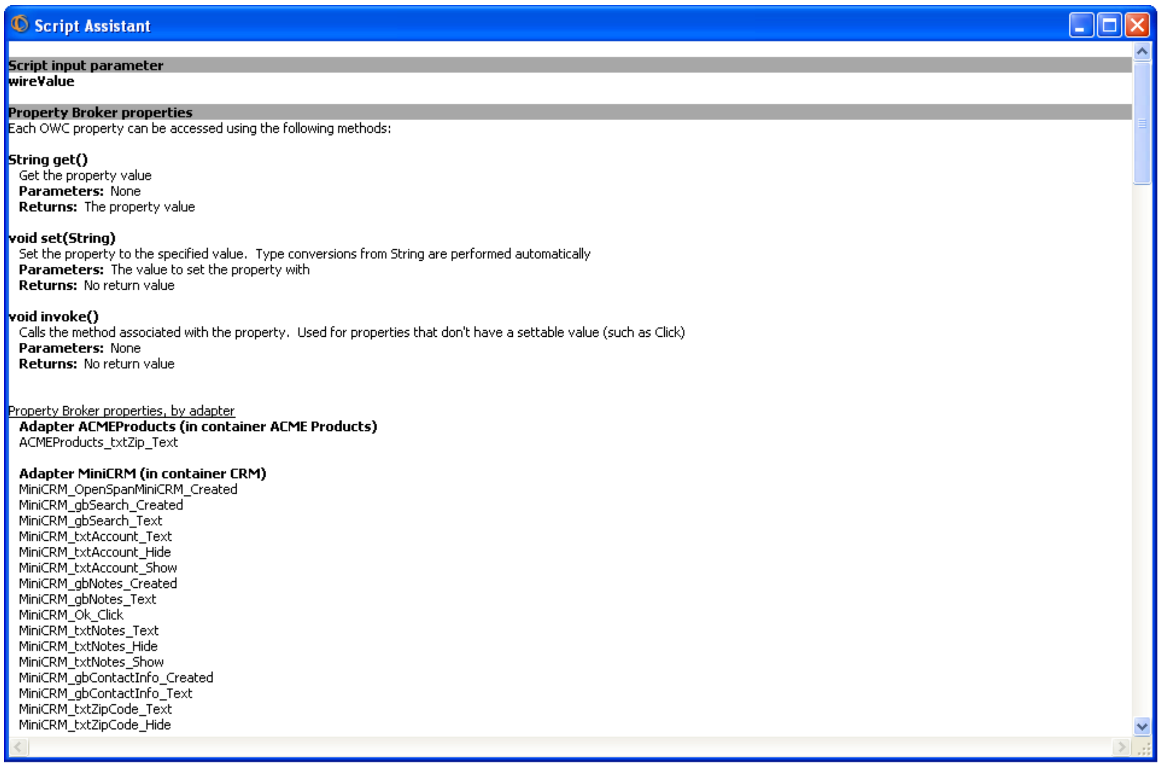
Store search results display in 'Notes' text box.

Button label changed from 'Ok' to 'Find Store'.

Script Assistant

The **Script Assistant** window, opened by clicking the question mark button on the OpenSpan Scripting Container toolbar, will help you to create syntactically correct code by copying and pasting properties and methods from the Script Assistant directly into the scripting editor. The Script Assistant is populated automatically with properties and methods from the adapters and controls in your composite application, which are presented in five sections:

- **Script input parameter** – This is the wire value that executes the script.
- **Property Broker properties** – These are the defined properties for interrogated controls listed in the Property Broker Properties list box. The **Property Broker properties** is the list box that displays directly below the Object Explorer (on the **Interrogation** tab of the **Edit Component Properties** window) and lists the previously selected and default properties for interrogated components.
- **Interrogated controls** - Any interrogated control's properties and methods can be accessed natively using the methods listed in this section of the Script Assistant.
- **Adapter Methods** – Adapters can be controlled (e.g., started, stopped, shown, hidden) using the methods listed.
- **Extended Functions –** These functions don't interact with the controls or the adapter. Rather, they are additional functions you can use to show message boxes, pause the execution of a script, or write to log files.



Script input parameter
wireValue

Property Broker properties
Each OWC property can be accessed using the following methods:

String get()
Get the property value
Parameters: None
Returns: The property value

void set(String)
Set the property to the specified value. Type conversions from String are performed automatically
Parameters: The value to set the property with
Returns: No return value

void invoke()
Calls the method associated with the property. Used for properties that don't have a settable value (such as Click)
Parameters: None
Returns: No return value

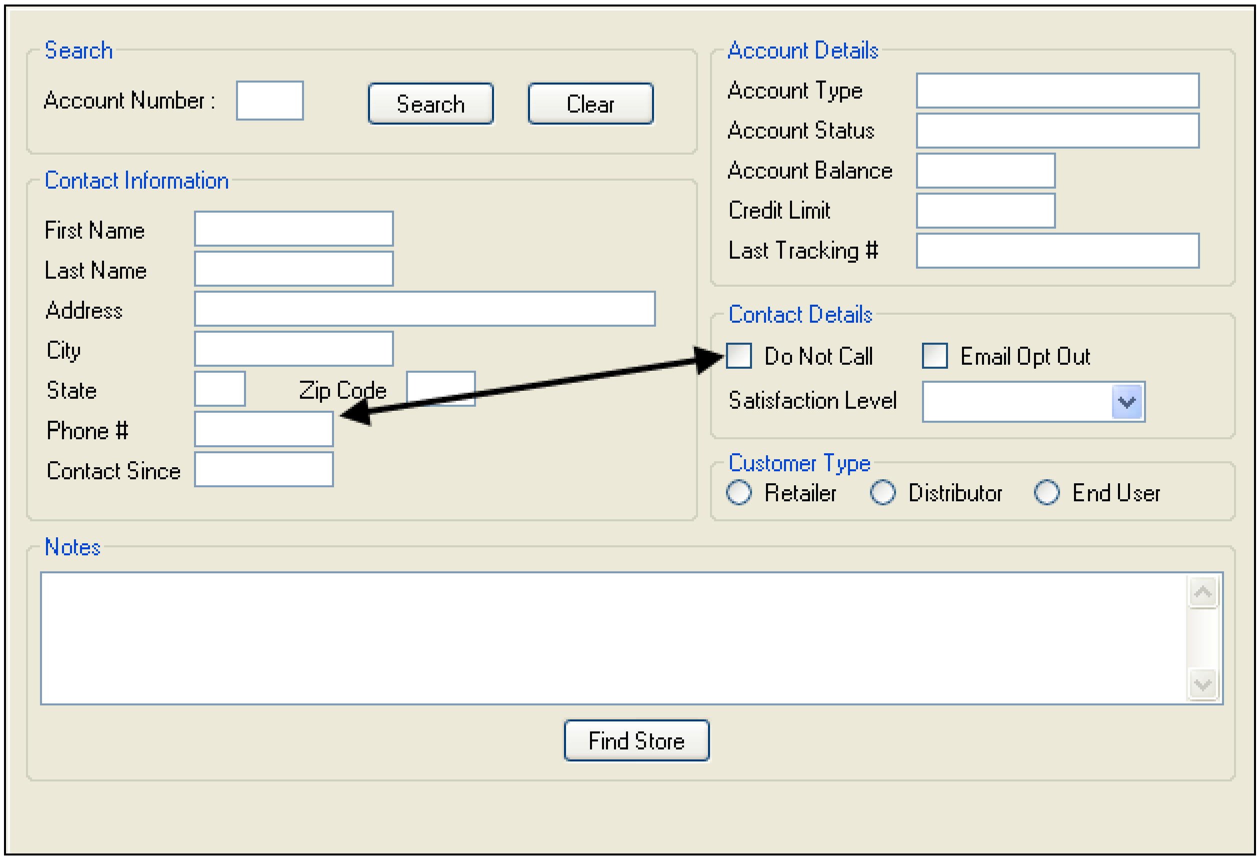
Property Broker properties, by adapter
Adapter ACMEProducts (in container ACME Products)
ACMEProducts_txtZip_Text

Adapter MiniCRM (in container CRM)
MiniCRM_OpenSpanMiniCRM_Created
MiniCRM_gbSearch_Created
MiniCRM_gbSearch_Text
MiniCRM_txtAccount_Text
MiniCRM_txtAccount_Hide
MiniCRM_txtAccount_Show
MiniCRM_gbNotes_Created
MiniCRM_gbNotes_Text
MiniCRM_Ok_Click
MiniCRM_txtNotes_Text
MiniCRM_txtNotes_Hide
MiniCRM_txtNotes_Show
MiniCRM_gbContactInfo_Created
MiniCRM_gbContactInfo_Text
MiniCRM_txtZipCode_Text
MiniCRM_txtZipCode_Hide

Exercise 2 - Using Property Broker Methods in the Scripting Container

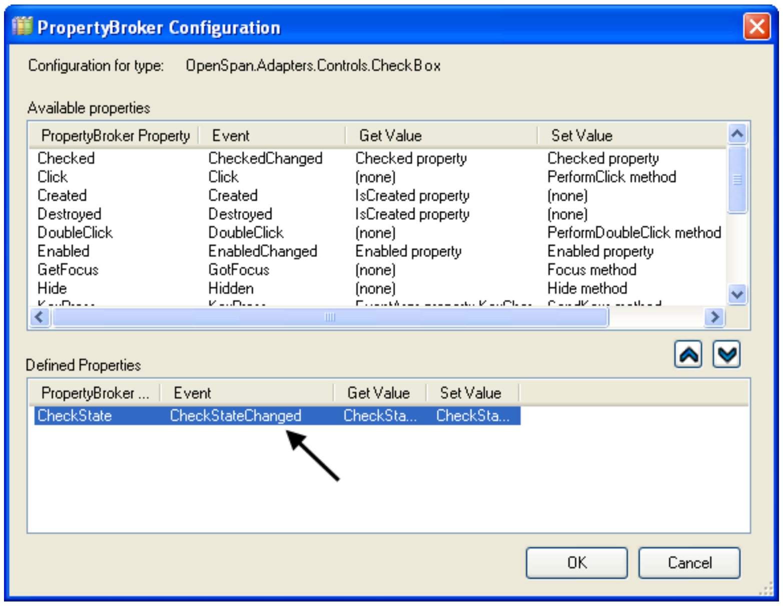
The first exercise in this tutorial used the Interrogated Controls methods in the scripting container and this second example will use Property Broker Properties methods. Both method types are listed in the Script Assistant.

Use of Property Broker Properties methods will be demonstrated by extending the composite application you built in Exercise 1 to hide the MiniCRM P**hone #** text box when the MiniCRM **Do Not Call** check box is selected.



1. Open the **Edit Component Properties** window for the MiniCRM application.
2. Open the **Interrogation** Tab.
3. Start Interrogation of the MiniCRM application and interrogate the following controls:
    a. **Do Not Call** checkbox.
    b. **Phone Number** text box.
4. Stop Interrogation by closing the **Interrogation Form** dialog box.
5. Select the **DoNotCall** check box in the Object Explorer.
6. Click the **Configure Property Broker Properties** button.

7. Add the **CheckState** property to the list of Defined Properties. Click **OK** to save your selection and return to the **Edit Component Properties** window.



Configuration for type: OpenSpan.Adapters.Controls.CheckBox

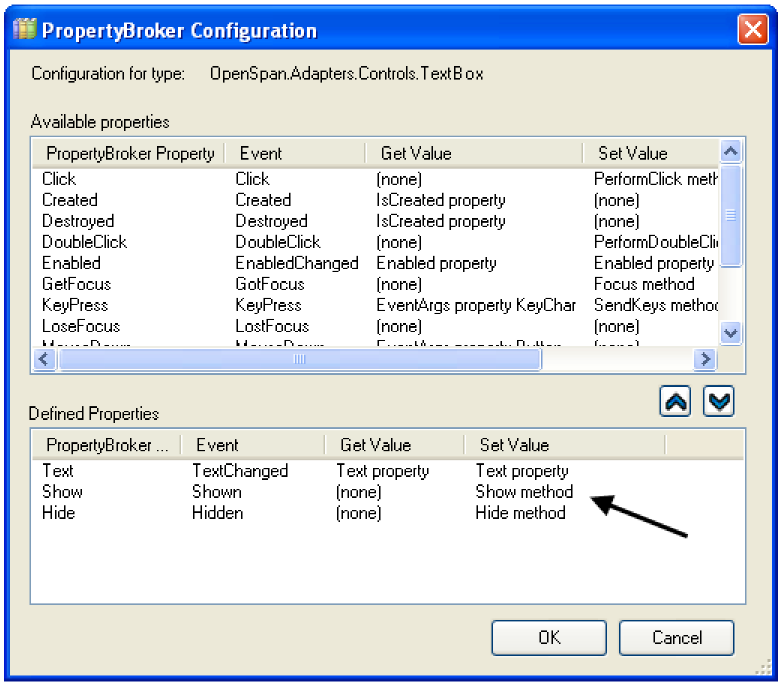
Available properties

PropertyBroker Property	Event	Get Value	Set Value
Checked	CheckedChanged	Checked property	Checked property
Click	Click	(none)	PerformClick method
Created	Created	IsCreated property	(none)
Destroyed	Destroyed	IsCreated property	(none)
DoubleClick	DoubleClick	(none)	PerformDoubleClick method
Enabled	EnabledChanged	Enabled property	Enabled property
GetFocus	GotFocus	(none)	Focus method
Hide	Hidden	(none)	Hide method

Defined Properties

PropertyBroker ...	Event	Get Value	Set Value
CheckState	CheckStateChanged	CheckSta...	CheckSta...

8. Select the **txtPhone** component in the Object Explorer.
9. Click the **Configure Property Broker Properties** button.
10. Add the **Text**, **Show**, and **Hide** properties to the list of Defined Properties. Click **OK** to save your selection and return to the **Edit Component Properties** window.



Configuration for type: OpenSpan.Adapters.Controls.TextBox

Available properties

PropertyBroker Property	Event	Get Value	Set Value
Click	Click	(none)	PerformClick meth
Created	Created	IsCreated property	(none)
Destroyed	Destroyed	IsCreated property	(none)
DoubleClick	DoubleClick	(none)	PerformDoubleCli
Enabled	EnabledChanged	Enabled property	Enabled property
GetFocus	GotFocus	(none)	Focus method
KeyPress	KeyPress	EventArgs property KeyChar	SendKeys methoc
LoseFocus	LostFocus	(none)	(none)

Defined Properties

PropertyBroker ...	Event	Get Value	Set Value
Text	TextChanged	Text property	Text property
Show	Shown	(none)	Show method
Hide	Hidden	(none)	Hide method

11. Open the OpenSpan Scripting Container and add a script named DoNotCall.
12. Add the following lines to the script and save:

```
 var ckstate;
ckstate  = MiniCRM_DoNotCall_CheckState.get();
  if(ckstate=="Unchecked"){
     MiniCRM_txtPhone_Show.invoke();
  }
  else if(ckstate=="Checked") {
    MiniCRM_txtPhone_Hide.invoke();
  }
```

13. Select the CRM OpenSpan Windows Container in the **Page Navigator**.
14. Click the Wires button and add a wire:
    a. **Source Component:** CRM → DoNotCall_CheckState
    b. **Target Component:** OpenSpan Scripting Container → Set DoNotCall
    c. Click the **Add Wire** button.
    d. Click **OK**.
15. Close the Component Application Editor window and save your changes.

Test the Composite Application – Hide the Phone # Text Box

16. On the MiniCRM application select (check) the **Do Not Call** check box. The **Phone #** should be hidden.
17. Unselect the **Do Not Call** check box and the **Phone #** text box should reappear.

Getting more Information about OpenSpan Containers

For more information on using the OpenSpan Windows and Scripting Containers, see the many articles in the IBM Lotus Expeditor Wiki at http://www-10.lotus.com/ldd/lewiki.nsf.