CONTENTS

v Preface
v Requirements

1 Using Text Adapters

2 Overview

3 Project 1: Using the Windows adapter with the emulator application
   3 Exercise 1: Creating a project and adding a Windows adapter
   5 Exercise 2: Setting Windows adapter properties

7 Project 2: Using the generic WinHllapi Adapter
   7 Exercise 1: Adding a generic WinHllapi adapter
   8 Exercise 2: Setting text adapter properties
   9 Using the text adapter interrogation function
   10 Exercise 3: Interrogating a host session
   16 Exercise 4: Adding an automation to start the text adapter
   17 Exercise 5: Building and debugging the project
   18 Exercise 6: Adding an automation to navigate the host application screens
   20 Exercise 7: Building and debugging the project
   20 Exercise 8: Adding an automation to search the MCAT library
   22 Exercise 9: Executing an automation and enabling the Keyword Terms list box
   23 Exercise 10: Executing a search
   23 Exercise 11: Resetting the Windows form for a new search
   24 Exercise 12: Building and debugging the project

26 Project 3: Using partial screens
   27 Exercise 1: Interrogating the Results screen using partial screen matching
Exercise 2: Partial Screen Match Rule using the Complex Match Rule option
Exercise 3: Interrogating the title text
Exercise 4: Adding an automation to get titles for matching records
Exercise 5: Reading screen rows and getting titles
Exercise 6: Stopping processing when less than 14 titles are listed
Exercise 7: Ending the retrieval process
Exercise 8: Processing additional Results screens
Exercise 9: Adding an automation to stop the adapters
Exercise 10: Building and debugging the solution

Project 4: Using the Traceplayer application

Summary
Preface

Pega Robotic Automation Studio provides adapters to integrate with terminal emulator applications. Studio's text adapters let you create projects which integrate mainframe applications. Studio provides text adapter templates for the most commonly used emulator software, including:

- Attachmate
- Ericom
- IBM wIntegrate

Note: Studio supports emulators which use WinHllapi, EHIllapi, and some COM technologies.

In this tutorial, you will learn how to use the Windows adapter to integrate a standard WinHllapi emulator (Attachmate myEXTRA! Enterprise) and the generic WinHllapi adapter to interact with a mainframe session running in myEXTRA! Enterprise. By the end of this tutorial module, you will be able to perform the following tasks:

- Configure the Windows adapter to integrate a terminal emulator application
- Configure and use the Generic WinHllapi adapter to integrate screens and fields from a mainframe application
- Interrogate screens, fields, and regions within a mainframe application
- Create automations to navigate a mainframe application and return data from the application

Requirements

This tutorial assumes you have successfully completed the Pega Robotic Automation Architect Essentials training and have a working knowledge of Studio and the ability to create basic Windows-based projects.

This course requires the following software:

- Studio or the Studio plug-in
- Attachmate myEXTRA! Enterprise version 8.0
- Trace Player application
- UMich_large.BAT trace file for use with the TracePlayer (this file simulates an IBM-3278 terminal connected to the University of Michigan Libraries Databases)
Pega Robotic Automation Studio provides adapters to integrate with terminal emulator applications. Studio’s text adapters let you create projects which integrate mainframe applications. Studio provides text adapter templates for the most commonly used emulator software, including:

- Attachmate
- Ericom
- IBM wintegrate

This document includes these exercises and topics:

- **Overview**
- **Project 1: Using the Windows adapter with the emulator application**
- **Project 2: Using the generic WinHllapi Adapter**
- **Project 3: Using partial screens**
- **Project 4: Using the Traceplayer application**
- **Summary**

Note: Studio supports emulators which use WinHllapi, EHllapi, and some COM technologies.
Overview

In this training module, you will learn how to use the Windows adapter to integrate a standard WinHllapi emulator (Attachmate myEXTRA! Enterprise) and the generic WinHllapi adapter to interact with a mainframe session running in myEXTRA! Enterprise.

By the end of this training module, you will be able to:

• Configure the Windows adapter to integrate a terminal emulator application
• Configure and use the Generic WinHllapi adapter to integrate screens and fields from a mainframe application
• Interrogate screens, fields, and regions within a mainframe application
• Create automations to navigate a mainframe application and return data from the application
Using Text Adapters | Project 1: Using the Windows adapter with the emulator application

Project 1: Using the Windows adapter with the emulator application

You can use Studio’s Windows adapter to control starting and stopping an emulator application. Using the Windows adapter along with the Generic WinHllapi text adapter, lets you integrate any WinHllapi-based emulator application.

For this tutorial, the Attachmate myEXTRA! Enterprise emulator application is used. The role of the Windows adapter is to integrate with the emulator application. The host/mainframe application running in the emulator is integrated into the Studio project using the Generic WinHllapi adapter.

In this exercise, you will create a Studio project containing a Windows adapter with the adapter properties set to integrate the Attachmate myEXTRA! Enterprise emulator application.

Begin by launching the Studio application.

Exercise 1: Creating a project and adding a Windows adapter

1. Select File > Open Project/Solution. The Open Project dialog appears.

2. Select the OSTextAdptrTraining folder and select the OSTextAdptrTraining.ossln solution. A solution containing a solution Text file (SolutionNotes.txt), a Studio project (OSTextAdptrTraining), and the Windows Form1.os project item appear. Here is an example of Solution Explorer:

   ![Solution Explorer](image)

   Note: If you are using a VM provided by Studio for this course, a base solution that contains the Windows form that will be used in throughout this course is included in the Projects folder (see the C:\Documents and Settings\user\My Documents\OpenSpan Studio for VS 2010\Project\OSTextAdptrTraining folder).

3. Right-click on the Studio project and select Add > New Windows Application. The Add New Item dialog appears.

4. Select the Windows Application template from the Studio > General category. Name the adapter item WinEmulator. Here is an example of the completed Add New Item dialog:
5. Click **Add** to add the Windows Application adapter project item. The adapter project item is added to Solution Explorer:

6. Select **File > Save All** to save all solution files.
Exercise 2: Setting Windows adapter properties

In this sample solution, the Windows adapter integrates the emulator application into the project. Use the following steps to...

1. Set the Path property to the emulator application name and path
2. Interrogate the emulator application – session display window

1. Highlight the Windows adapter project item (WinEmulator.os) in the Solution Explorer and double-click the item to open it in the Designer (if it is not already open).
2. Open the Properties window.
3. Click in the Path property and then click Browse. The Open dialog appears.
4. Use the Open dialog to go to the location of the emulator application file (Extra.exe).

5. In the Arguments property, enter the name of the emulator session for the project.

6. Leave default entries for the remaining Windows Adapter properties.

7. Click the Start Interrogation button on the Design page for the Windows adapter. The Interrogation Form appears and the emulator application is launched.

8. Interrogate the session display area within the emulator application. See the following illustration:

Note: If you are using a VM provided by Pega for this course, the application is located in the C:\Program Files\Attachmate\E!E folder. In this case, set the Path property to:

    C:\Program Files\Attachmate\EXTRA\EXTRA.exe

Note: If you are using a VM provided by Pega for this course, the session is MCAT.edp and it is located in the C:\Documents and Settings\user\My Documents\Attachmate\EXTRA\sessions folder. In this case, set the Arguments property to:

    C:\Users\OpenSpan User\Documents\Attachmate\EXTRA\sessions\MCAT.edp

Note: If not auto-started in your VM or other environment, then before interrogation you must start this batch file:

    UMich_large.BAT

    If you have to manually start the batch file and you are using a VM provided by Pega for this course, you may already have a desktop shortcut that points to this file. Double-click either the shortcut or the batch file icon itself, found in this path:

    C:\Traceplayer-1\UMich_large.BAT

    After the UMICH.edp session loads, you are ready to proceed.

Note: If you are using a VM provided by Pega for this course, the MCAT session opens in the myEXTRA! Enterprise application.
These controls are added to Object Explorer:

9. Highlight Control (the child of Control1) and delete the ControlID match rule. This ID changes between application sessions and can cause matching errors. Repeat this process for Control1.

10. Stop the interrogation by closing the Interrogation Form.

11. Save all solution files.
Project 2: Using the generic WinHllapi Adapter

While Pega provides adapters for specific emulators, such as the Attachmate EXTRA! WinHllapi, this course illustrates how to use the generic WinHllapi adapter to integrate host applications running in any emulator which uses WinHllapi communications.

When using the generic adapter, you must provide logic in the project for starting the emulator and opening the session. In the previous exercise, you added the Windows adapter and configured it to start the myEXTRA! application and load the MCAT.edp session. In this exercise, you will learn how to interrogate screens and screen text.

Begin by making sure the Traceplayer application is running the UMICH_Large.bat file.

Exercise 1: Adding a generic WinHllapi adapter

1. Right-click the OSTextAdptrTraining project in Solution Explorer and select Add > New Item. The Add New Item dialog appears.
2. Select the General Generic WinHllapi template from the Studio Text Adapter category.
3. Enter this name for the item:
   
   GenWinHllapi.os

   Here is an example of the completed New Item dialog:

4. Click Add. The adapter item is added to Solution Explorer:
Exercise 2: Setting text adapter properties

Here are the key text adapter properties used in basic text adapter projects:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DllName</td>
<td>Sets the name and full path of the HLLAPI (High Level Language Application Program Interface) assembly used by the emulator. For Attachmate myEXTRA!, the emulator uses the WHLAPI32.dll assembly to communicate with mainframe applications.</td>
</tr>
<tr>
<td>SessionID</td>
<td>Identifies the HLLAPI Shortname that has been set up in the emulator application for the session. In Attachmate myEXTRA!, the HLLAPI Shortname is set on the Global Preferences dialog (Options &gt; Global Preferences &gt; Advanced tab).</td>
</tr>
<tr>
<td>StartOnProjectStart</td>
<td>Defines whether the adapter and associated session are started when the project starts. When this property is False, you must add project logic to start the adapter, such as use the Adapter.Start method.</td>
</tr>
</tbody>
</table>

1. Begin by highlighting the GenWinHllapi.os text adapter project item in Solution Explorer.
2. Double-click on the text adapter to open the item in the Designer if not already open.
3. Open the Properties window and click in the DllName property to display a Browse button.
4. Click Browse. The Open dialog appears. Select the HLLAPI assembly used by the adapter.
5. Set the SessionID property. If you are using a VM provided by Pega, the session shortname is A, so set the SessionID to A.
6. Set the StartOnProjectStart property to False. For this training solution, the adapter will be started based on creation of the session window control interrogated using the Windows adapter.
7. Save all solution files.
Using the text adapter interrogation function

When interrogating a mainframe (text-based) application, you define screens, regions, and fields. Studio uses the position, length, and text of selected screen areas to identify the object and match it. The steps to matching targets in a text application are listed here:

- Identify the screen containing the text you want to include in the project
- Select target text as either a field (screen area in which text can be typed) or a region (display-only text area).
- Use partial screens to identify targets that are repeated within the Screen presentation. (See “Project 3: Using partial screens” on page 26 for more information on partial screens.)
- Modify match rules from simple to complex when you need to generalize the text to accommodate matching (see exercise 3 for details on modifying match rules).

To interrogate the session screens within Studio’s Designer, Studio must start the emulator application. For this course, you will start the interrogation function for the Windows adapter (WinEmulator.os) and then start the interrogator for the text adapter (GenWinHllapi.os).

Unlike Windows and Web application interrogation, there is no interrogator icon to drop over targets. Instead, you left-click and drag the cursor through the screen text displaying in the Designer. This highlights the selected text. You then right-click on the highlighted text (or field area) to select from these options:

- Save As Match Region
- Select Field
- Save As Field
- Save As Region

Definitions of these options follow:

**Save As Match Region**

Use this function to select screen text that uniquely identifies a screen within the application. You can set a Match Region as either Simple or Complex. Simple match regions use the text exactly as it appears on the screen to match the screen. If you select Complex, you can use regular expression (regex) syntax to generalize the text for matching.

This is useful when the screen identifying text contains an account number or date which will change during runtime. You can use the Complex match rule options to match the screen text based on regex syntax. See “Project 3: Using partial screens” on page 26 for more information on Complex match rules.

**Select Field**

Use this function to determine the field length of screen fields. Place the cursor in the screen area for the field, right-click and select this option. Studio highlights the entire area defined by the host application as the field.

This will help you determine whether to use the application’s definition of a specific field or whether you will want to select a region to define a customized field.
Using Text Adapters

Project 2: Using the generic WinHllapi Adapter

**Save As Field**

Use this to select a field and create a Studio control that represents the field.

**Note** Use the Select Field option to highlight the entire field area and then use the Save As Field option to create a control for the field. This ensures that when data is entered in the field or retrieved from the field, the entire entry is recognized by Studio.

**Save As Region**

Use this function to select areas of text for use in a project. In some cases the host application has defined a very large field length and therefore when you use the Select Field command, a large area is highlighted. The Save as Region command lets you limit this area or select a custom area of text as to return as a region to Studio.

**Exercise 3: Interrogating a host session**

You interrogate host applications through the Designer. For this training module, the project will navigate the University of Michigan Libraries application to the MCAT database, query the database using a keyword search term, and retrieve book titles from the search results for display on a Windows form. The following screens and text areas are used in the training solution:

- Screen - University of Michigan Libraries Database Selection Menu
  - Database Selection field
- Screen – UMich Online Catalog Introduction (Welcome to MCAT)
  - Next Command field
- Screen – UMich Online Catalog Keyword Index (Search Results screen)
  - Search Results field
  - Result Records (Date, Title, Author)
- <F8> FORward page text

Before you begin, make sure the `UMICH_Large.bat` file is running. For more information on the Traceplayer, see “Project 4: Using the Traceplayer application” on page 44.

1. In Studio, start the emulator application by selecting the Windows adapter project item (`WinEmulator.os`) and clicking the **Start Interrogation** button in the Designer. The emulator application launches and the UMICH.edp session loads.

2. Open the Text Adapter project item, `GenWinHllapi.os`, and select **Start Interrogation** from the Designer. University of Michigan Libraries Database Selection Menu screen is loaded into the Designer. Interrogate the session using the screen displayed in the Designer for the text adapter. Here is an example of the Designer:
3. Identify the Database Selection Menu screen by highlighting the text in the Designer. Left-click just before the “D” of DATABASE and drag the cursor through the “Database Selection Menu” text string. Studio highlights the text. Right-click on the highlighted text to display the interrogation options.

4. Select **Save as Match Region > For Screen > Simple**. A screen control is created and displays in Object Explorer.

5. In the Properties window, use the **Name** property to assign this name to the screen:

    `scrDBSelectMenu`

Your Object Explorer should look similar to the following:

Interrogate the **Database Selection** field (the empty space just to the right of the Database Selection text). Right-click anywhere in the field and choose the **Select Field** option. Studio highlights the entire data entry field.
6. Right-click anywhere in the highlighted field and choose the **Save As Field > For Screen** option. A control for the field is added to Object Explorer.

7. Use the Properties window to rename the control to:

   `fldDbSelect`

   Your Object Explorer should look like the following:

8. On the Database Selection Menu screen displayed in the emulator, enter **MCAT** in the Database Selection field and press Enter to go to the UMich Online Catalog screen. Here is an example:

9. For the Welcome screen of the MCAT database, use the Introduction text to identify the screen by selecting (highlighting) the text Introduction. Then right-click on the selected text and choose **Save As Match Region > For Screen > Simple**.
10. Next, right-click the **Next Command** data entry field (not the field label) and select the **Select Field** option.

11. Right-click the highlighted **Next Command** data entry field and select **Save As Field > For Screen**.

12. In the Properties window, rename Screen1 to **scrMCATIntro** and Field1 to **fldSrchCmd**. Your Object Explorer should look like the following:

13. In the **Next Command** field on the UMich Online Catalog-Introduction screen (displayed in the Designer), enter this command to search for books related to the keyword “*baseball*”: 
**k=baseball**

Press Enter to display the search results. The results display in the UMICH Online Catalog - Keyword Index screen as shown here:

14. Use the Keyword Index text to identify the UMich Online Catalog Keyword Index screen by selecting (highlighting) the Keyword Index label.

15. Then right-click the Keyword Index text and select **Save As Match Region > For Screen > Simple**.

16. Select (highlight) the Search Results region, leaving enough spaces to accommodate larger result sets. Right-click the highlighted region, and select **Save As Region > For Screen**.
17. Highlight this text near the bottom of the screen:

   <F8> FORward page

This text indicates that there are additional screens of search results. The project will use the text to determine whether additional records exist for the search. Right-click on the highlighted text and select Save as Region > For Screen.

18. On the Properties window, rename the interrogated controls:

<table>
<thead>
<tr>
<th>For this control</th>
<th>Change the name to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screen1</td>
<td>scrKeyword</td>
</tr>
<tr>
<td>Region1 (search results)</td>
<td>rgnResults</td>
</tr>
<tr>
<td>Region2 (F8 Forward)</td>
<td>rgnF8Forward</td>
</tr>
</tbody>
</table>

Your Object Explorer should look like the following:

19. Click Stop Interrogation for the Text adapter (GenWinHllapi), then close the Interrogation Form for the Windows adapter, WinEmulator. When prompted, you can disconnect from the Attachmate session.

20. Save all solution files.
Exercise 4: Adding an automation to start the text adapter

For this course, the Windows adapter will automatically launch the emulator application (Attachmate myEXTRA! Enterprise). Once the emulator is launched, the text adapter (GenWinHllapi.os) is started by calling the Start method on the adapter. The first automation of this project starts the text adapter.

Begin by highlighting the OSTextAdptrTraining project in Solution Explorer.

1. Add a new automation to the project. Assign this name to the automation:

   StartSession.os

   Your Solution Explorer should look similar to the following:

2. Add two Studio messageDialog components to the automation – Local tab. Select this overload option for both components:

   (String message): DialogResult

3. Set the message parameters for the dialogs as follows:

<table>
<thead>
<tr>
<th>On this dialog</th>
<th>Set the message parameter to</th>
</tr>
</thead>
<tbody>
<tr>
<td>messageDialog1</td>
<td>Unable to start the emulator. This message appears if Studio cannot start the Windows emulator application (myEXTRA! Enterprise).</td>
</tr>
<tr>
<td>messageDialog2</td>
<td>Unable to start the session. This message appears if Studio cannot load the UMICH session in the emulator through the text adapter.</td>
</tr>
</tbody>
</table>

4. Add these methods and events to the automation:
   - WinEmulator.Started event
   - WinEmulator.Control.WaitForCreate method – this is the child control under Control1 for the Windows adapter.
   - GenWinHllapi.Start method
   - ScrDBSelectMenu.WaitForCreate method
Using Text Adapters | Project 2: Using the generic WinHllapi Adapter

- `messageDialog1.Show(message=Unable to start emulator)` method
- `messageDialog1.Show(message=Unable to start session)` method

5. Arrange the connection blocks and connect the event path as shown here:

6. Save all solution files.

**Exercise 5: Building and debugging the project**

Make sure the `UMICH_Large.bat` file is running and then perform these steps:

1. Select the Debug solution configuration from the Debug toolbar.

2. Click the **Debug** button. Studio compiles the project and launches a design version of the project in the Pega Runtime application installed with Studio.

   The project should execute in this sequence:
   - The project is built and no build errors are reported.
   - Runtime loads the design project OSTextAdptrTraining.
   - Runtime loads the Search MCAT Topics Windows form. Note that the search term list box is disabled. This list box is only enabled when the MCAT query screen is displayed in the emulator.
   - Runtime launches the emulator (myEXTRA! Enterprise) and opens the UMICH.edp session.
   - The initial Database Selection Menu screen displays in the emulator.

3. Click the **Stop Debugging** button in Studio to unload the project and stop Runtime.
Exercise 6: Adding an automation to navigate the host application screens

This automation will navigate the host application from the main screen, Database Selection Menu, to the MCAT database.

Begin by highlighting the OSTextAdptrTraining project in Solution Explorer.

1. Add a new automation to the project. Assign this name to the automation:

   **Navigate_Apps.os**

   Your Solution Explorer should look similar to the following:

   ![Solution Explorer with project structure]

2. Add two Studio messageDialog components to the automation – Local tab. Select this overload option for both components:

   **(String message): DialogResult**

3. Select the messageDialog1 component and copy/paste the component to add another instance of the messageDialog to the automation.

4. Set the message parameters for the dialogs as follows:

<table>
<thead>
<tr>
<th>On this dialog</th>
<th>Set the message parameter to</th>
</tr>
</thead>
</table>
   | messageDialog1 | **DB Select screen not created.**  
                     This message appears if Studio cannot match the Database Selection Menu screen. |
   | messageDialog1 | **Search Cmd field not created.**  
                     This message appears if Studio cannot match the Search Command field on the MCAT Introduction screen. This indicates the project was unable to navigate from the Database Selection screen to the MCAT Introduction screen. |

5. Right-click on the automation and add an entry point.
6. Add these properties, methods, and events to the automation:
   - GenWinHllapi.scrDBSelectMenu.WaitForCreate method.
   - **GenWinHllapi.fldDbSelect.Text** property — set the text parameter to **MCAT**.
   - GenWinHllapi.fldDbSelect.Enter method — this method executes pressing the Enter key in the Database Selection field after the MCAT text is set.
   - messageDialog1.Show(message=DB Select Screen not created) method.
   - messageDialog1.Show(message=Search Command field not created) method.

7. Arrange the connection blocks and connect the event path as shown here:

8. Save all solution files.

9. Return to the StartSession automation and add the Navigate_Apps.Execute method. This method runs the Navigate_Apps automation once the session has started.

10. Connect the True output event node from the ScrDBSelectMenu.WaitForCreate method to the input event node of the Navigate_Apps.Execute method. Here is an example of the completed StartSession automation:
Exercise 7: Building and debugging the project

Make sure the UMICH_Large.bat file is running and then perform these steps:

1. Select the Debug solution configuration from the Debug toolbar.
2. Click the Debug button. Studio compiles the project and launches a design version of the project in the Runtime application installed with Studio.

   The project should execute in this sequence:
   • The project is built and no build errors are reported.
   • Runtime loads the design project OSTextAdptrTraining.
   • Runtime loads the Search MCAT Topics windows form. Note that the search term list box is disabled. This list box only becomes enabled when the MCAT query screen is displayed in the emulator.
   • Runtime launches the emulator (myEXTRA! Enterprise) and opens the UMICH.edp session.
   • The initial Database Selection Menu screen displays in the emulator.
   • The text MCAT appears in the Select field and the host is navigated to the MCAT Introduction screen.
3. Click the Stop Debugging button in Studio to unload the project and stop the Runtime application.

Exercise 8: Adding an automation to search the MCAT library

This automation will perform a keyword search based on the keyword selected on the Windows form. The search begins when the user clicks the Search Selected Keyword button. Note that keyword list box is disabled by default so users cannot initiate a search unless the UMICH session is running and the UMICH Online Catalog Introduction screen is matched (including the Command field).

The automation has these parts:
• Enabling the Keyword list box (lbxTerms) and clearing any titles listed on the Windows form.
• Searching for the keyword selected in the lbxTerms list box.
• Clearing the form results (Titles and Number of Records).

Begin by highlighting the OSTextAdptrTraining project in Solution Explorer.

1. Add a new automation to the project. Assign this name to the automation:
   
   Search_records.os

   Your Solution Explorer should look similar to the following:
2. Add a Studio messageDialog component to the automation – Local tab. Select this overload option:

```(String message); DialogResult```

This message will be used to display an error if the UMich Online Catalog Keyword Index (Search Results) screen is not matched.

3. Assign this name to the messageDialog1 component:

`msgKeywrdScrn`

4. Set the message parameter to:

`Results Screen not Created`

5. Add a StringUtils component to the automation – Local tab. Assign this name to the component:

`strUtKEq`

This component formats the search command as required by the host application (such as k=baseball).

6. Right-click on the automation and add an entry point. The entry point will enable executing the automation only after the UMich Online Catalog Introduction (Welcome to MCAT), Next Command field is matched.

7. Add the following properties, methods, and events to the automation:

- `WindowsForm1.BringToFront` method.
- `WindowsForm1.lbxTerms.Enabled` property – set the `Enabled` property to True on the connection block.
- `WindowsForm1.lbxTitles.Items.Clear` method – add two instances of this method to the automation. To access this method, highlight the `lbxTitles` box in Object Explorer and click the `Explore Component Properties` button. Next, highlight the `Items` property and use Object Inspector to select the `Clear` method for the `Items` property. Return to the standard view of Object Explorer by clicking the `Explore Components` button.
- `WindowsForm1.btnSearch.Click` event – this event starts the search of MCAT library.
- `WindowsForm1.lbxTerms.SelectedItem` property – this sets the keyword used in the search.
Using Text Adapters

• **WindowsForm1.lblRecNo.Text** property – the label displays the number of titles returned from the search.
• WindowsForm1.btnClear.Click event – this event resets the results text shown on the Windows form.
• **GenWinHllapi.fldSrchCmd.Text** property – sets the keyword search command for MCAT.
• GenWinHllapi.fldSrchCmd.Enter method – simulates pressing Enter in the MCAT screen, search field.
• GenWinHllapi.scrKeyword.WaitForCreate method – indicates when the search results have been displayed. Add two instances of this method to the automation.
• **GenWinHllapi.rgnResults.Text** property – displays the number of titles found for the search.
• msgKeywrdScrn.Show(Results Screen Not Created) method – reports an error if the UMich Online Catalog Keyword Index (Search Results screen) Results text region is not matched.

8. Select the strUtKEq stringUtils component in the Local tray and use Object Inspector to select the **Format**(2 parameters – String formatString, String stringValue0) method. Add this method to the automation.

9. Set the formatString parameter for the strUtKEq.Format method to:

   \[ k=0 \]

   ![strUtKEq](image)

   **Note** Make sure to enter a zero (0) and not the letter “O” inside the braces of the format string.

10. Save all solution files.

**Exercise 9: Executing an automation and enabling the Keyword Terms list box**

Follow these steps.

1. On the Search_Records.os automation, arrange the connection blocks and connect the event path as shown here:

   ![Connect Blocks](image)

2. Open the Navigate_Apps.os automation and add the Search_Records.Execute method to the automation. Connect the event path for the method as follows:
Exercise 10: Executing a search

1. On the `Search_Records.os` automation, arrange the connection blocks and connect the event/data path as shown here:

```
<table>
<thead>
<tr>
<th>Windows Form1</th>
<th>Search_Records</th>
<th>GenWinHllapi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click</td>
<td>Execute</td>
<td>GenWinHllapi fldSrchCmd</td>
</tr>
<tr>
<td>Properties</td>
<td>Format</td>
<td>WaitForCreate</td>
</tr>
<tr>
<td>SelectedItem</td>
<td>k=40</td>
<td>Result</td>
</tr>
<tr>
<td>stringValue0</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>Result</td>
<td></td>
<td>False</td>
</tr>
</tbody>
</table>
```

2. Save all solution files.

Exercise 11: Resetting the Windows form for a new search

1. On the `Search_Records.os` automation, arrange the connection blocks and connect the event/data path, as shown here:

```
<table>
<thead>
<tr>
<th>Windows Form1</th>
<th>Search_Records</th>
<th>GenWinHllapi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click</td>
<td>ArgnResults</td>
<td>GenWinHllapi scrKeyword</td>
</tr>
<tr>
<td>Properties</td>
<td>Properties</td>
<td>GenWinHllapi fldSrchCmd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enter</td>
</tr>
<tr>
<td>Text</td>
<td>msgKeyWordScrn</td>
<td>WaitForCreate</td>
</tr>
<tr>
<td>Show</td>
<td></td>
<td>Result</td>
</tr>
<tr>
<td>Results Screen not Created</td>
<td></td>
<td>False</td>
</tr>
</tbody>
</table>
```

Enter dashes (`-----`) for the **Text** property.
2. Click on the **Text** property of the `lblRecNo` label and set the text to dashes (`-----`).

3. Save all solution files.

**Exercise 12: Building and debugging the project**

Make sure the `UMICH_Large.bat` file is running and then perform these steps.

1. Select the Debug solution configuration from the Debug toolbar.

2. Click the **Debug** button. Studio compiles the project and launches a design version of the project in the Runtime application installed with Studio.

   The project should execute in this sequence:
   - The project is built and no build errors are reported.
   - Runtime loads the design project `OSTextAdptrTraining`.
   - Runtime loads the Search MCAT Topics windows form. Note that the search term list box is disabled. This list box is only enabled when the MCAT query screen is displayed in the emulator.
   - Runtime launches the emulator (myEXTRA! Enterprise) and opens the UMICH.edp session.
   - The initial Database Selection Menu screen displays in the emulator.
   - The emulator navigates the host application to the UMich Online Catalog Introduction (Welcome to MCAT) screen.

3. Select a keyword from the list box and click the **Search Selected Keyword** button. The project should execute in this sequence:

   - This text displays in the **Next Command** field:
     
     k=your keyword selection
   - Search is initiated in the host application.
   - The UMich Online Catalog Keyword Index (Search Results) screen shows the number of records found in the Search Results field and a listing of the records (Date, Title, Author) in rows.

   The number of records found appears on the Windows form next to the Number of Records Found label. Here are examples of the session window and Windows form:
4. Click the **Stop Debugging** button in Studio to unload the project and stop the Runtime application.
Project 3: Using partial screens

In host/mainframe applications, the screen may repeat similar types of information in multiple locations. For example, using the solution you created in this training module, rows of titles and authors appear which represent books matching the query term (the example below shows the results for k=fishing):

<table>
<thead>
<tr>
<th>DATE</th>
<th>TITLE:</th>
<th>AUTHOR:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>All Fishermen are liars : true tales from Greenlaw, Linda</td>
<td>CU</td>
</tr>
<tr>
<td>2</td>
<td>Arrowood, National Wildlife Refuge Fishin</td>
<td>PD</td>
</tr>
<tr>
<td>3</td>
<td>Blue Ridge Parkway : fishing opportunities United States</td>
<td>PD</td>
</tr>
<tr>
<td>4</td>
<td>Buying a fishing rod for my grandfather : Gao, Xingjian</td>
<td>CU</td>
</tr>
<tr>
<td>5</td>
<td>Chicago Island adventure : recreation tra</td>
<td>PD</td>
</tr>
<tr>
<td>6</td>
<td>Erle National Wildlife Refuge : fishing</td>
<td>PD</td>
</tr>
<tr>
<td>7</td>
<td>A good day’s fishing</td>
<td>Prosek, James</td>
</tr>
<tr>
<td>8</td>
<td>Grand excursions on the upper Mississippi</td>
<td>Mittal, Anu K</td>
</tr>
<tr>
<td>9</td>
<td>Individual fishing quotas economic effects</td>
<td>United States</td>
</tr>
<tr>
<td>10</td>
<td>Individual fishing quotas methods for comm</td>
<td>United States</td>
</tr>
<tr>
<td>11</td>
<td>Luis Owens : literary reflections on his</td>
<td>CU</td>
</tr>
<tr>
<td>12</td>
<td>Native American Fish and Wildlife Resource United States</td>
<td>PD</td>
</tr>
<tr>
<td>13</td>
<td>Red drum : natural history and fishing</td>
<td>Wenner, Charles A</td>
</tr>
<tr>
<td>14</td>
<td>S. G37, Individual Fishing Quota Act of 20</td>
<td>United States</td>
</tr>
</tbody>
</table>

These rows occur on 14 lines on the screen. One for each book result. If there are less than 14 lines of results, the lines are blank. Each result row has a unique numeric identifier.

If you want to create an automation that retrieves the text from each row of results shown in a screen like the sample above, you have these options:

<table>
<thead>
<tr>
<th>Interrogate</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Each line as a region</td>
<td>This results in 14 region controls per screen.</td>
</tr>
</tbody>
</table>
| The line as a partial screen| You can define a partial screen as an area within an already interrogated parent screen.  
A partial screen can appear multiple times and in multiple places within the parent screen. This results in a single partial screen control required to return all of the rows of the results. |

This exercise describes how to create and use partial screens in Studio text adapter projects. The solution used in the previous exercises is continued to add logic for returning the titles of records which match the search keyword in the MCAT database.
Exercise 1: Interrogating the Results screen using partial screen matching

Follow these steps.

1. In Studio, start the emulator application by selecting the Windows adapter project item (WinEulator.os) and clicking the Start Interrogation button in the Designer. The emulator application launches and the UMICH.edp session loads.

2. Open the Text Adapter project item, GenWinHllapi.os, and select Start Interrogation from the Designer. The University of Michigan Libraries Database Selection Menu screen is loaded into the Designer. Interrogate the session using the screen displayed in the Designer for the text adapter.

3. Type MCAT in the Database Selection field and press Enter to go to the UMich Online Catalog Introduction screen.

4. Type k=baseball in the Next Command field and press Enter to perform a keyword search. The UMich Online Catalog Keyword Index (Search Results) screen appears. The screen lists 14 titles which match the search criteria.

5. Highlight the first row of results as shown here:

6. Right-click on the highlighted row to view the interrogation options.

7. Select Save as Partial Screen. The Select Parent dialog appears.

8. Use the Select Parent dialog to identify the main screen which must be matched before the partial screen is matched. If the partial screen can appear on any screen within the application, use the Emulator as the parent. Here is an example of the Select Parent dialog:

Note: Use the Set PartialScreen Options dialog to define where the partial screen appears within the application screens. A partial screen location is defined by row and column. For example, if the partial screen always appears at a certain row on the screen, you can set the row number. Likewise, if the partial screen always appears at a certain column, you can set the column number.

Depending on how the partial screen appears within the parent screen, you have the options to set:

- Specific Row – Any Column (Floating horizontally)
- Specific Column – Any Row (Floating vertically)
- Specific Row and Specific Column
- Any Row and Any Column

If the partial screen floats vertically or horizontally (starts at any row and/or any column), you must set the Floating Anchor Options to define whether the partial screen appears multiple times within the parent.
Using Text Adapters 28

9. For this training module, select `Screen:scrKeyword` as the parent and click OK. The Set PartialScreen Options dialog appears.

10. The result row always starts in Column 1 and can appear at any row position within the screen. Therefore, set the partial screen options as follows:

<table>
<thead>
<tr>
<th>Set this option</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row Anchor</td>
<td>Any</td>
</tr>
<tr>
<td>Column Anchor</td>
<td>1</td>
</tr>
<tr>
<td>Floating Anchor Options</td>
<td>“The partial screen can occur more than once in the presentation space. The entire presentation space is searched for all occurrences.”</td>
</tr>
</tbody>
</table>

Click OK to save these settings.
Exercise 2: Partial Screen Match Rule using the Complex Match Rule option

Note

Use the Complex Rule Designer to set the position and text matching options which Studio uses to match the selected Match Region text.
For a partial screen, the position options are already set through the Set PartialScreen Options dialog.
Use the Complex Rule dialog to set the way Studio matches the selected text.
In the Options group box, set the RegEx option to define the type of regex syntax to use.
In the Selected Text option, set the text to match for the Contains or Does Not Contain RegEx options.
If you select Free Form Regex for the Regex Option, the Selected Text option changes to Regular Expression for you to enter the regex syntax to match the text.

In this task, you set a matching condition for the partial screen. The matching condition specifies text that appears in all occurrences of the partial screen. In this case, the text that appears for each partial screen is a record number (1, 2, ...).

1. Highlight the result number in the partial screen (1) and the two spaces preceding the number — this allows for record numbers up to three digits.
   
   The easiest way to highlight this area is to left-click immediately following the number 1 and then use the left arrow key to highlight the number and two preceding spaces, as shown here:

2. Right-click on the highlighted area and select Save As Match Region > For Partial Screen > Complex.

3. Since the record number differs for each record, the Complex matching option is required. When you select this option, the Complex Rule Designer appears.

4. Use the Options group box to set these matching requirements:

<table>
<thead>
<tr>
<th>Set this option</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>RegEx Option</td>
<td>Free form Regex</td>
</tr>
<tr>
<td>Regular Expression</td>
<td>\d{1,3}? (match any three digits followed by any character)</td>
</tr>
</tbody>
</table>
Using Text Adapters 30

<table>
<thead>
<tr>
<th>Set this option</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ignore Case</td>
<td>Accept the default, which is checked.</td>
</tr>
</tbody>
</table>

Your Complex Rule Designer dialog should look like this example:

5. Click OK to apply the match rule. The Match Rules for Partial Screen displayed on the Designer should look like the following:

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Type</th>
<th>Position</th>
<th>Length</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>regionMatchRule4</td>
<td>Free form Regex</td>
<td>4</td>
<td>3</td>
<td>[\d(1,3)?]</td>
</tr>
</tbody>
</table>

6. Highlight the Partial Screen in the Object Explorer and use the Properties window to rename the control to:

*scrPartialSearch*

**Exercise 3: Interrogating the title text**

1. Highlight the title for the record in the partial screen, right-click and select **Save As Region > For Partial Screen**. A region control is added to Object Explorer under the *scrPartialSearch* screen.
2. Highlight the region in Object Explorer and use the Properties window to rename the control to:

   **rgnTitle**

   Your Object Explorer should look like the following:

   ![Object Explorer screenshot showing the renamed control](image)

3. Click Stop Interrogation for the *GenWinHllapi.os* project item.

4. Open the *WinEmulator.os* project item in the Designer and click the **Stop Interrogation** button.

5. Save all solution files.

**Exercise 4: Adding an automation to get titles for matching records**

This automation loops through the results which appear after the user searches on a keyword in the MCAT database and records are found. When the user clicks the *Get Titles* button on the Windows form, the automation reads each title in the rgnTitle control (interrogated in the previous section) and adds the title text to a list box on the Windows form. The automation then prompts the user as follows:

- If 15 or more titles are returned, the user is prompted:

  **Retrieve more titles?**
  
  If the user chooses Yes, the next screen of titles is read and added to the list box. If the user chooses No, the adapters are shut down and search disabled.

- If there are no more titles available or if the search returns fewer than 14 titles, the user is prompted:

  **No more records. New Search?**
  
  If the user chooses Yes, the Windows form results display is reset and a new search can be initiated. If the user chooses No, then the adapters are shut down and the search disabled.

This automation contains these main sections:

- Processing result rows and yielding title text on the first screen of results.
- Halting the row processing when there are less than 14 titles listed on a screen.
- Processing the next screen of results if more than 14 titles are returned.
- Ending the retrieval process.
Begin by highlighting the OSTextAdptrTraining project in the Solution Explorer.

1. Add a new automation to the project. Assign this name to the automation:

   **Get_Records.os**

   Your Solution Explorer should look similar to the following:

   ![Solution Explorer - Solution 'OSTextAdptrTraining' (1 project)](image)

   - Solution 'OSTextAdptrTraining' (1 project)
   - Solution Items
     - Solution Notes.txt
     - OSTextAdptrTraining
       - Properties
         - References
           - GenWinHlapi.os
           - Get_Records.os
           - Navigate_App.os
           - Search_records.os
           - StartSession.os
           - Windows_Form1.os
           - WinEmulator.os

2. Add two Studio messageDialog components to the automation - Local tab. Select this overload option for both components:

   (String message): DialogResult

3. Apply these names to the messageDialog components:

   - **msgNoMoreRec**
   - **msgGetMoreRec**

4. Set the properties for the messageDialogs as follows:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgNoMoreRec</td>
<td>&quot;No more records. New Search?&quot; This message appears if Studio detects there are no more titles to retrieve. Set the Buttons property to: YesNo.</td>
</tr>
<tr>
<td>msgGetMoreRec</td>
<td>&quot;Retrieve more titles?&quot; This message appears if Studio detects that an additional screen of titles is available. Set the Buttons property to: YesNo.</td>
</tr>
</tbody>
</table>

5. Add a forLoop component to the automation. Set the Limit parameter on the connection block to 14 (the maximum number of results returned by the search per screen).

6. Add a Jump label to the automation and assign the label this name:

   **EndRetry**

   This Jump control is used to route the event path in the automation.
7. Add a stringUtils component to the automation. Use Object Explorer to select the Contains method for the component. Set the seekString parameter to F8. This method determines whether additional screens of search results are available.

8. Add these properties, methods, and events to the automation:

<table>
<thead>
<tr>
<th>Add this item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WindowsForm1.btnGetRecs.Click</td>
<td>This event initiates the execution of the automation logic.</td>
</tr>
<tr>
<td>WindowsForm1.lbxTitles.Items.Add(1 param obj)</td>
<td>This method adds each title to the list box on the Windows form.</td>
</tr>
<tr>
<td>WindowsForm1.btnGetRecs.PerformClick()</td>
<td>This method re-initiates the reading of the result records.</td>
</tr>
<tr>
<td>GenWinHllapi.rgnTitle.Text</td>
<td>This property populates the list box on the Windows form.</td>
</tr>
<tr>
<td>GenWinHllapi.rgnF8Forward.Text</td>
<td>This property makes sure the additional screen of results exists.</td>
</tr>
<tr>
<td>GenWinHllapi.rgnF8Forward.Clear()</td>
<td>This method deletes the previous keys/commands sent to the field.</td>
</tr>
</tbody>
</table>

9. Save all solution files.

**Exercise 5: Reading screen rows and getting titles**

1. Use Studio’s forLoop component to read each line of result text and return the title (in the rgnTitle.Text property). Arrange the connection blocks and connect the event path as shown here:

   ![Connection diagram](image)

   **Note** When you connect the event path for the rgnTitle control, a Key parameter appears. This is because multiple instances of this control can occur (clones) and you must specify which instance to use. Your options are: Last, Index, None, First, and Current. For this exercise, the default key type (Index) applies since it can be linked to the record number/forLoop Index.

2. Connect the Index output from the forLoop to the Value for the Index Key of the rgnTitle.Text property. Complete the connections as shown here:
Exercise 6: Stopping processing when less than 14 titles are listed

Add logic to handle the case where the index of the forLoop does not correspond to the index of any of the rgnTitle controls. For example, suppose only five results are returned for the search. The forLoop would cause an exception error when the loop index is six (6) and the automation attempts to get the text for the rgnTitle having an index of six since this control does not exist.

1. Click the rgnTitle.Text node and click the Wrap In a Try...Catch Block icon from the Automation toolbar.

   This will add a Try block before GenWinHllapi.rgnTitle and a Catch block right after. This will catch any exceptions that come up while trying to access an index that does not exist.

   When the index of the loop is greater than the number of results returned, the forLoop needs to stop processing.

2. Add the forLoop Break method by selecting the forLoop node in the automation and then choosing the Break method from the Component Inspector in Object Explorer.

3. Connect the CATCH node Exception event output to the input of the forLoop1 Break node, as shown here:
The `msgNoMoreRec` message displays this message:

\[
\text{No more records. New Search?}
\]

The options are Yes and No. If the user selects Yes – New Search, then the automation executes the logic to reset the host application to the initial search input screen (UMich Online Catalog Introduction (Welcome to MCAT)).

4. Connect the `forLoop1` Break node to the `msgNoMoreRec`, as shown here:

5. Save all solution files.

**Exercise 7: Ending the retrieval process**

When the automation has retrieved all available records for the search, the user is notified via the `msgNoMoreRec` messageDialog:

\[
\text{No more records. New Search?}
\]

If the user selects Yes to start a new search, the automation must navigate the host application back to the UMich Online Catalog Introduction (Welcome to MCAT) screen so new search criteria can be entered. For this application, the MCAT command returns the application to the MCAT Introduction screen. You can use the text adapter’s `SendKeyStringToHostPlusEnter` method to navigate the application to the MCAT Intro screen.

Begin by adding the `SendKeyStringToHostPlusEnter` method to the automation:

1. Highlight the `GenWinHllapi` adapter in Object Explorer. Using Object Inspector, select:

   `SendKeyStringToHostPlusEnter(String key)`
2. Arrange the connection blocks and connect the event path as shown here:

3. Save all solution files.
Exercise 8: Processing additional Results screens

The host application retrieves all of the titles that match the search criteria and displays up to 14 results on the screen. If more than 14 results are found, the text <F8> FORward displays at the bottom right of the screen (you interrogated this text – the rgnF8Forward region). On the last screen of results, the text in this area changes to: <F7> BACK page.

When the rgnF8Forward text no longer contains the string “F8,” the application is displaying the last screen of results. Using the forLoop.Completed event and the text property from the rgnF8Forward, you can add logic that lets the user continue retrieving titles and then stop processing on the last screen of search results. Once the forLoop processes 14 rows and adds the titles to the Window form lbx.Titles list box, the loop raises the Completed event. At this point, you can give the user the option to retrieve more titles or end the processing.

In this part of the automation, add logic to prompt the user to continue retrieving titles and re-start the forLoop to retrieve the data. If the user chooses to retrieve more data but there are no more screens of results data (the rgnF8Forward text does not contain the string F8), a message appears that indicates there are no more records.

1. Right-click the automation and select Jump to > EndRetrv. A Jump To label is added. Connect the Yes output event from the msgNoMoreRec to the EndRetrv Jump To label:

   ![Jump To EndRetrv](image)

   **Note** If the users answers No to the “No more records. New search” message, the Text and Windows adapters are stopped. The automation for this process will be added later in this exercise.

2. Begin by connecting the Completed output event from the forLoop to the Show method for the msgGetMore messageDialog. This will cause this message to appear once the first 14 results have been retrieved:

   

   **Retrieve more titles?**

   3. If the user selects Yes, the automation must check to see if an additional results screen exists. If another results screen exists, the automation must navigate to the results screen and start the forLoop again to retrieve data from the screen. Add the following to the automation:
<table>
<thead>
<tr>
<th>Add</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GenWinHllapi.rgnTitle.TextChanged</td>
<td>After adding this event, right-click on the event and select: <strong>Wait For This Event</strong> This will be used to raise the TextChanged event after the PF8 (Forward) command occurs. When the event is raised, the <em>PerformClick</em> method is called on the Windows form <em>GetRecs</em> button which starts the forLoop again.</td>
</tr>
<tr>
<td>GenWinHllapi.scrKey.word_PF8()</td>
<td>This method sends the PF8 command to screen.</td>
</tr>
</tbody>
</table>

4. Arrange the connection blocks and connect the event path as shown here:

![Diagram]

5. If the rgnF8Forward region does not contain the string F8, the host application is on the last screen of results. In this case, the StringUtils.Contains method yields a False result and the False event is raised. Connect this event to the msgNoMoreRec messageDialog *Show* method. This tells the user there are no more records to retrieve.

![Diagram]

6. The msgNoMoreRec message dialog displays this message:

```
No more records. New Search?
```

The options are Yes and No. If the user selects Yes – New Search, then the automation executes the logic to reset the host application to the initial search input screen (UMich Online Catalog Introduction (Welcome to MCAT)). Right-click the automation and select **Jump to > EndRetrv**. A Jump To label is added. Connect the Yes output event from the msgNoMoreRec messageDialog to the EndRetrv Jump To label.
7. Save all solution files. Your automation should look like the following:
Exercise 9: Adding an automation to stop the adapters

The user has the option to stop searching when either of these messages appears:

<table>
<thead>
<tr>
<th>Message</th>
<th>Property Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>No more records. New Search?</td>
<td>msgNoMoreRec</td>
</tr>
<tr>
<td>Retrieve more titles?</td>
<td>msgGetMore</td>
</tr>
</tbody>
</table>

In either case, if the user selects No, then the automation should disable the **Search** button on the Windows form and shut down the adapters. Create an automation to end the search processing as follows:

1. Add an automation to the project. Assign this name to the automation:

   **EndSession.os**

   The new automation appears in Solution Explorer:

2. Right-click the automation and add an entry point. An Execute block appears on the automation.

3. Add these properties and methods to the automation:

<table>
<thead>
<tr>
<th>Add</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WindowsForm1.btnSearch.</td>
<td>Set this property to False on the connection block.</td>
</tr>
<tr>
<td>Enabled</td>
<td></td>
</tr>
<tr>
<td>GenWinHllapi.Stop()</td>
<td>This method stops the text adapter which ends the session.</td>
</tr>
<tr>
<td>WinEmulator.Stop()</td>
<td>This method stops the Windows adapter, which closes the emulator</td>
</tr>
<tr>
<td></td>
<td>application.</td>
</tr>
</tbody>
</table>

4. Arrange the connection blocks and connect the event path as shown here:

5. Save all solution files.
6. Return to the Get_Records automation and add the EndSession.Execute method as follows:

7. Save all solution files.

**Exercise 10: Building and debugging the solution**

Make sure the `UMICH_Large.bat` file is running and then perform these steps.

1. Select the Debug solution configuration from the Debug toolbar.

2. Click the **Debug** button. Studio compiles the project and launches a design version of the project in the Runtime application installed with Studio.

   The project should execute in this sequence:
   - The project is built and no build errors are reported.
   - Runtime loads the design project OSTextAdptrTraining.
   - Runtime loads the Search MCAT Topics Windows form. Note that the search term list box is disabled. This list box only becomes enabled when the MCAT query screen is displayed in the emulator.
   - Runtime launches the emulator (myEXTRA! Enterprise) and opens the UMICH.edp session.
   - The initial Database Selection Menu screen displays in the emulator.
   - The emulator navigates the host application to the UMich Online Catalog Introduction (Welcome to MCAT) screen.
   - The keyword list box on the Windows form becomes enabled.

3. Select a keyword from the list box and click the **Search Selected Keyword** button. The project should execute in this sequence:
   - This text appears in the **Next Command** field:
     
     ```
     k=your keyword selection
     ```
   - Search is initiated in host application.
• The UMich Online Catalog Keyword Index (Search Results) screen shows the number of records found in the Search Results field and lists of records (Date, Title, Author) in rows on the screen.
• The number of records found appears on the Windows form next to the Number of Records Found label.

Here are examples of the Session window and Windows form:

![Session window](image1)

4. Click the Get Titles button in the Windows form. The list of titles from the host application displays in the list box on the Windows form.
   • If another screen of results is available, this message appears:
     
     **Retrieve more titles?**
   • If there are no more results, this message appears:
5. Click **Yes** until you receive the *No More Records* message:

6. Click **Yes** and then click the **Clear Titles** button on the Windows form.

7. Select Windsurfing from the **Keyword** list box and click **Search Selected Keyword**. The number of search results shows eight entries.

8. Click the **Get Titles** button. Titles populate the list box and this message appears:

   **No More Records. New Search?**

   Click **No**. The emulator closes.

9. Click the **Stop Debugging** button in Studio to unload the project and stop the Runtime application.
Project 4: Using the Traceplayer application

A trace of a recorded host session rather than a live host session is used for this exercise.

To start the recorded host session for the University of Michigan Libraries system, run the `UMICH_Large.bat` file.

Note: If you are using a VM provided by Pega for this training module, the .bat file is located in the `C:\Traceplayer-1` folder.

Here is an example of the Traceplayer output:

![Example of Traceplayer output](image-url)
Summary

By performing the exercises in this training module you have learned how to:

• Configure the Windows adapter to integrate a terminal emulator application.
• Configure and using the generic WinHllapi adapter to integrate screens and fields from a mainframe application.
• Interrogate screens, fields, and regions within a mainframe application.
• Create automations to navigate a mainframe application and return data from it.