CHAPTER 4

# UI Automation

Interacting with several desktop and web applications is an essential part of our daily work routine. We use these applications to accomplish tasks like logging timesheets in a time tracking system, storing, organizing, and retrieving documents in a document management system, or tracking projects in a project management software. To interact with these various applications, we use their user interface (UI). Similarly, UiPath StudioX UI automation interacts with the user interfaces to automate desired tasks by replicating the user interaction like checking boxes, selecting dropdowns, entering data, and clicking buttons.

## Learning Objectives

At the end of this chapter, you will learn how to

- Load web and desktop apps
- Enter data in forms
- Extract data from forms and tables
- Perform miscellaneous activities to interact with user interfaces
- Use the App/Web Recorder to generate an automation


A. Javed et al., *Robotic Process Automation using UiPath StudioX*,
https://doi.org/10.1007/978-1-4842-6794-3_4

# Sample Overview

The sample application used for all exercises of this chapter is Contacts Management. The application is accessible via a browser from `https://therpabook.com/samples/contactsmanagement/` location. This section provides a quick overview of the sample application.

**Contacts List**: This is the default page that loads when you open this web application. As shown in Figure 4-1, this page contains the following sections:

1. Header: To show the title of the application
2. Buttons: To open the add contact details dialog, view contact details dialog, and download to Excel and CSV
3. Table: With a list of contacts spanning multiple pages

**Add Contact**: This dialog box, shown in Figure 4-2, is displayed when you click the Add Contact button from the home page. This screen allows you to enter new contact details. The application is not operational, so this does not persist with any new contacts.

**View Contact**: This dialog box, shown in Figure 4-3, is displayed when you click the View Contact button from the home page. This screen allows you to view contact details. The application is not operational, so this always shows the same contact information.

**Download**: This button allows you to download the contacts list in Excel and CSV formats.

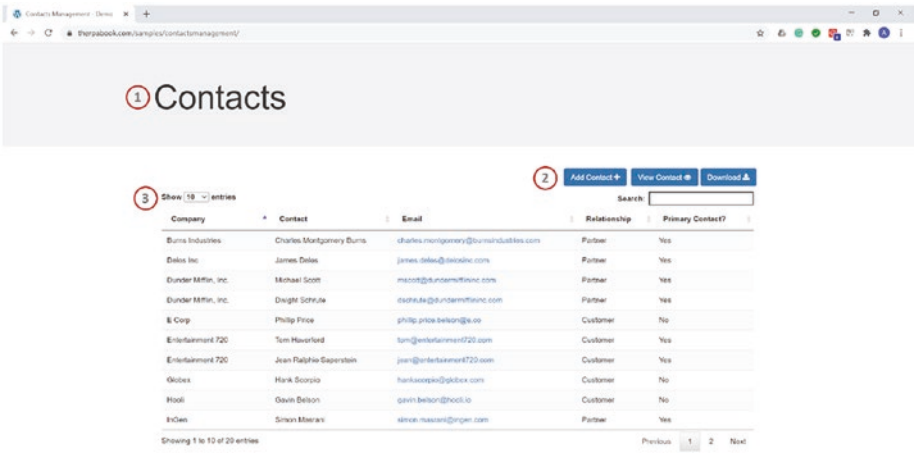


***Figure 4-1.*** *Contacts Management web application home page*

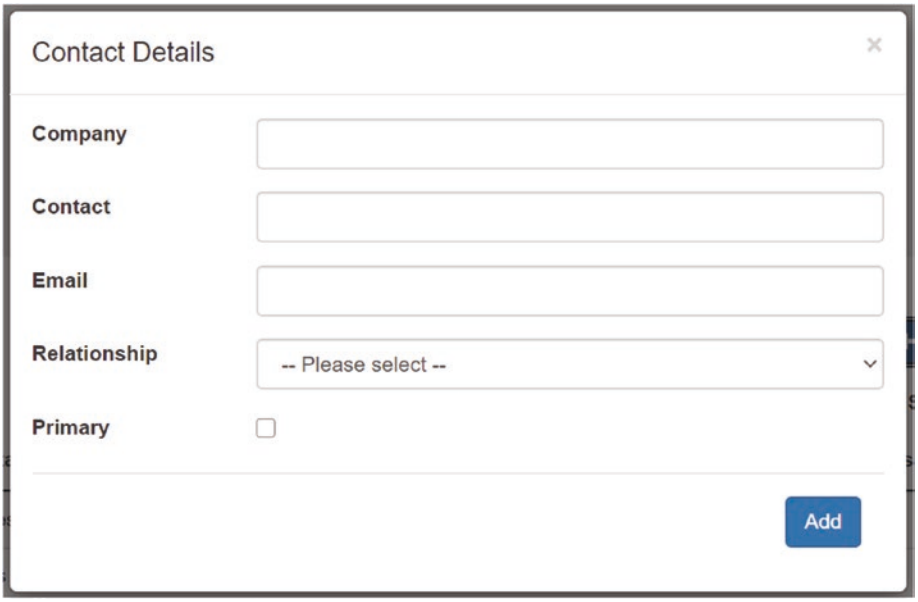


***Figure 4-2.*** *Add Contact dialog*

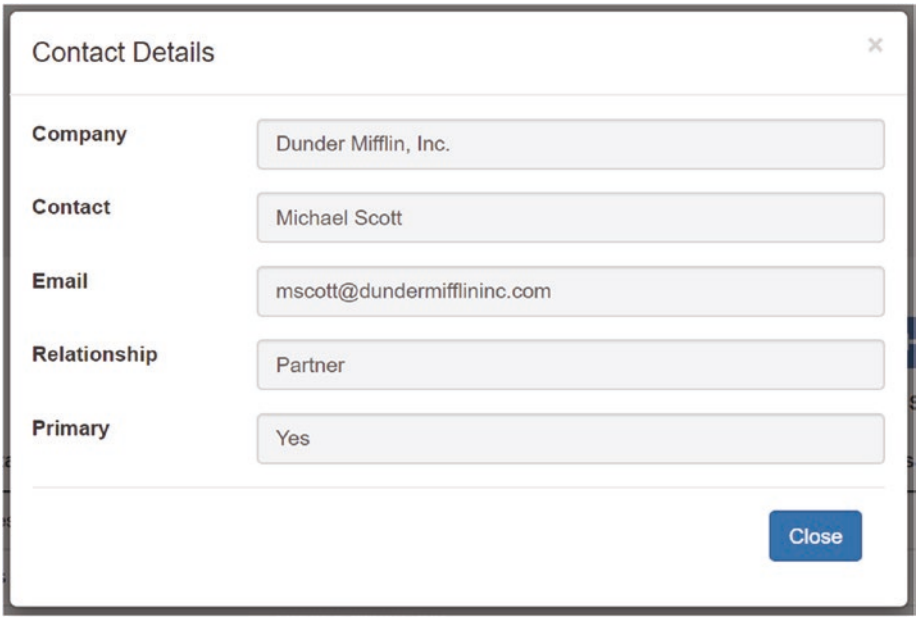


***Figure 4-3.*** *View Contact dialog*

# Activities Reference

UI automation activities are available from the App/Web category shown in Figure 4-4. The following sections provide instructions on how to configure and use each activity.

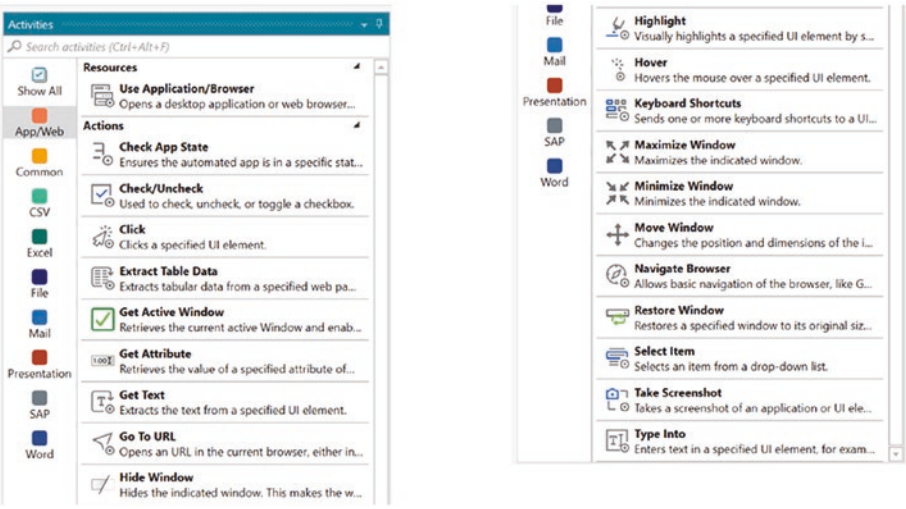


***Figure 4-4.*** *Activities for App/Web automation*

# Use Application/Browser

The **Use Application/Browser** activity allows you to open, close, and interact with a web or a desktop application.

---

**Note** The **Use Application/Browser** activity will contain all the actions that you want to perform on the UI of a target application. For example, if you want to click a button on the screen, the Click activity will have to be added to the body of this activity.

---

## Configuration

This section provides instructions on how to configure a, shown in Figure 4-5.

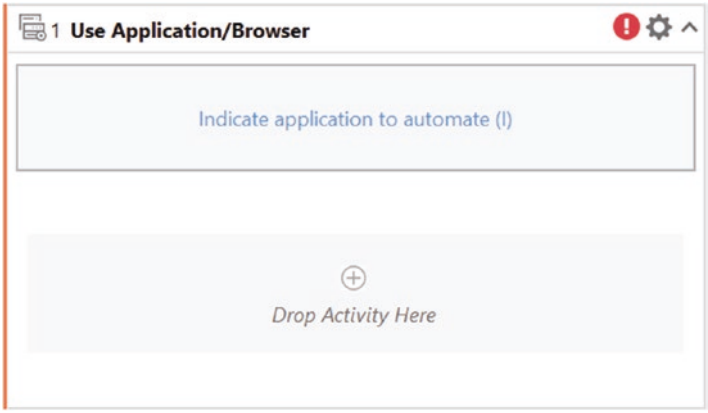


***Figure 4-5.*** *Activity card for Use Application/Browser*

---

**Note** Before you can automate a web application, you will need to install the relevant web browser extension from Home ➤ `Tools` ➤ `UiPath Extensions`.

---

**Indicate application to automate (I)**: This is a required configuration available on the activity card. This configuration allows you to indicate the target application that you are going to automate. To configure this activity, make sure your target application is already open. Once you have indicated your target application, StudioX will automatically identify if it is a desktop application or a web application. Figure 4-6 shows the activity card for a web application, while Figure 4-7 shows the activity card for a desktop application.

**Browser URL**: This is a required configuration available on the activity card. This configuration is only available when the target is a web application. This configuration allows you to specify the URL of the web application. By default, this is populated with the browser URL identified in the indicate application configuration.

**Application path**: This is a required configuration available on the activity card. This configuration is only available when the target is a desktop application. This configuration allows you to specify the complete path of the executable file on a local system. By default, this is populated with the application path identified in the indicate application configuration.

**Application arguments**: This is an optional configuration available on the activity card. This configuration is only available when the target is a desktop application. This configuration allows you to specify arguments for the application to execute. For example, when launching Notepad, you can specify what file to load when Notepad opens.

**Match exact title**: This is an optional configuration available on the activity card. This configuration is only available when the target is a desktop application. This configuration allows you to specify if the automation should exactly match the application title. By default, this option is not checked, that is, automation will not use exact title matching.

**Open**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify if the automation should open this application when executing. There are three options: `Never` (never open the application, this is useful when the application might already be open), `Always` (always open the application), and `IfNotOpen` (only open the application if it is not already open).

**Close**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify if the automation should close the target application after completion. There are three options: `Never` (never close the application, this is useful when a person or another program might have already opened the application), `Always` (always close the application), and `IfOpenedByAppBrowser` (only close the application if the automation opened it).

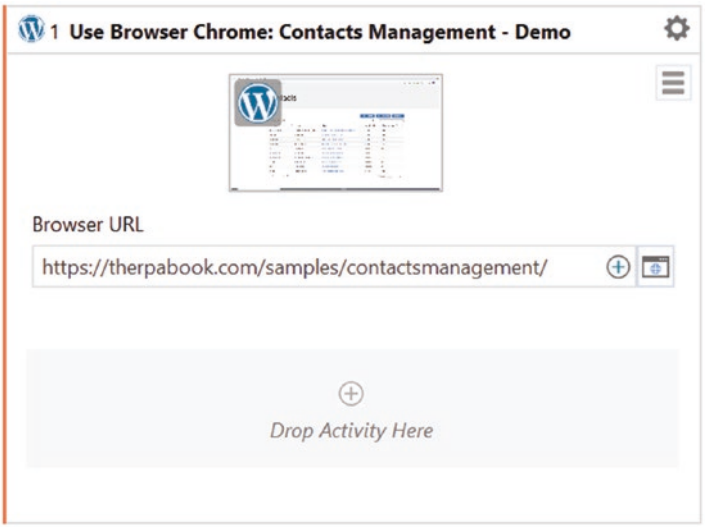


***Figure 4-6.*** *Activity card for Use Browser*

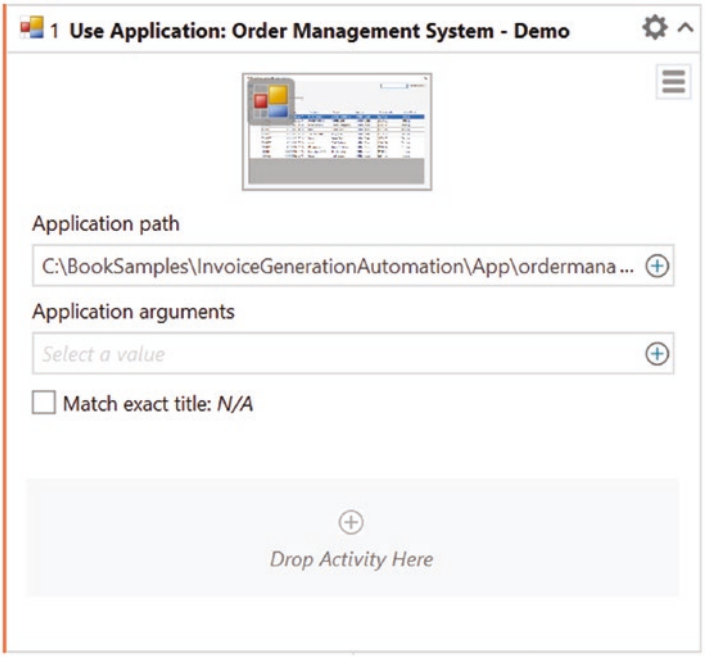


***Figure 4-7.*** *Activity card for Use Application*

# Go To URL

The **Go To URL** activity allows you to open a specified URL in a browser.

## Configuration

This section provides instructions on how to configure a **Go To URL** activity, shown in Figure 4-8.

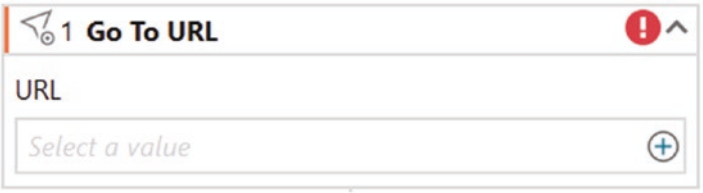


***Figure 4-8.*** *Activity card for Go To URL*

**URL**: This is a required configuration available on the activity card. This configuration allows you to specify the URL that you want to open in the browser.

**EXERCISE**

**Goal**: Use the `Go To URL` activity to navigate to `https://therpabook.com/samples/contactsmanagement/`.

**Source Code**: Chapter_4_MiscellaneousActivitiesExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter `about:blank` in the address bar. This will ensure that by default, no page is loaded.
2. In `StudioX`, add the `Use Application/Browser` activity to a blank process.

3. Next, click Indicate application in the Use Application/Browser activity card. You will notice a blue shade over the screen; this is to point your mouse to the browser to select the window. Clicking the browser will automatically populate the activity card.

4. Next, select the Use Application/Browser activity card, and from Properties, set the Options ➤ Open property to Always. This will ensure that the automation always opens a new browser.

5. Next, add a Go To URL activity in the body of Use Application/Browser activity.

6. Next, in the URL field of Go To URL activity, click the Plus icon, select the Text option, and type https://therpabook.com/samples/contactsmanagement/. Click Save.

Once you have completed the exercise, the final configuration of the **Go To URL** activity should resemble Figure 4-9. Figure 4-10 shows the state of the target web application once the automation has completed its run. In this case, the browser has been redirected to a new URL.

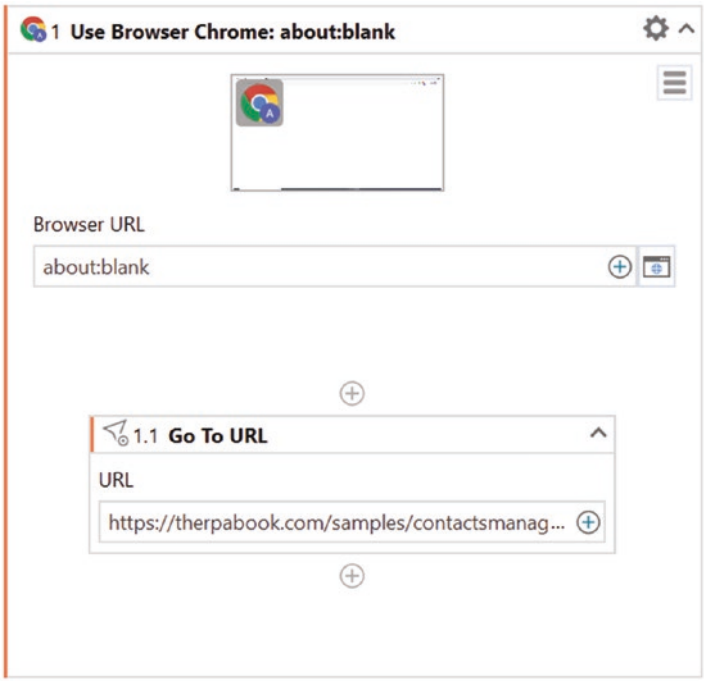


***Figure 4-9.*** *Final configuration of the Go To URL activity exercise*

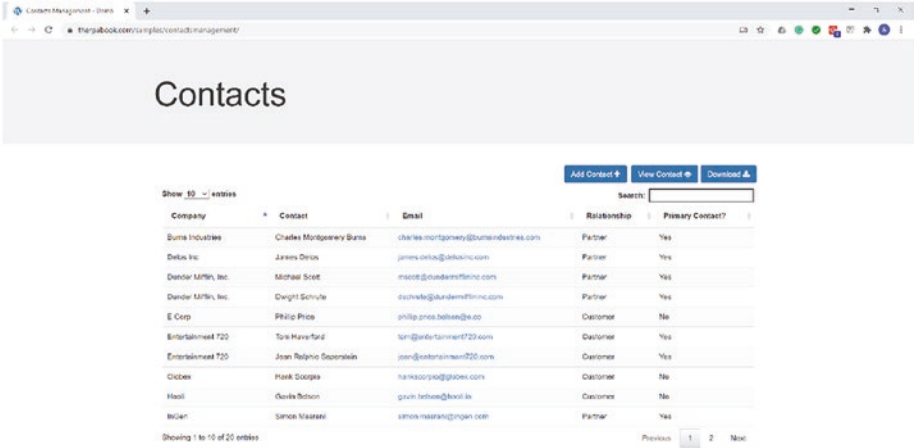


***Figure 4-10.*** *Result of the Go To URL activity exercise*

# Navigate Browser

The **Navigate Browser** activity allows you to perform basic browser navigations such as Go Back, Go Forward, Go Home, Refresh, and Close Tab.

## Configuration

This section provides instructions on how to configure a **Navigate Browser** activity, shown in Figure 4-11.

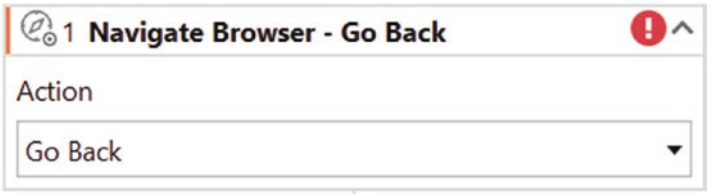


***Figure 4-11.*** *Activity card for Navigate Browser*

**Action**: This is a required configuration available on the activity card. This configuration allows you to specify the navigation action that you want to perform on the browser. Table 4-1 provides a quick description of each action. By default, the action is set to `Go Back`.

***Table 4-1.*** *Actions for Navigate Browser activity*

| Action | Description |
|---|---|
| Go Back | Opens the previous page in the browser; this action only works if another page was open in the browser earlier. |
| Go Forward | Opens the next page in the browser; this action only works if the Go Back action was used earlier. |
| Go Home | Opens the default home page of the browser. |
| Refresh | Refreshes the page. |
| Close Tab | Closes the currently active tab. |

## EXERCISE

**Goal**: Use the `Navigate Browser` activity to refresh the already open web page. This exercise builds upon the `Go To URL` exercise. Figure 4-10 shows the state of the target web application before this exercise.

**Source Code**: Chapter_4_MiscellaneousActivitiesExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Navigate Browser` activity in the body of `Use Application/Browser` activity after the `Go To URL` activity.
2. Next, from the `Action` dropdown, select `Refresh`.

Once you have completed the exercise, the final configuration of the **Navigate Browser** activity should resemble Figure 4-12.

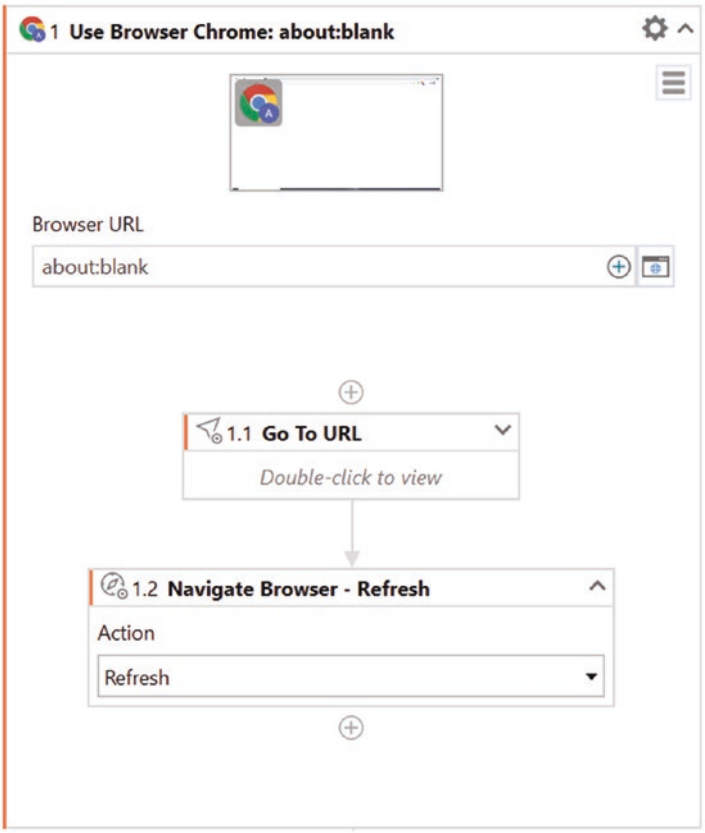


***Figure 4-12.*** *Final configuration of the Navigate Browser activity exercise*

## Highlight

The **Highlight** activity allows you to visually highlight a specified area while the automation is running. The automation creates a box around the specified element.

**Tip** For tasks where you want to visually emphasize specific UI elements while a process is executing, the Highlight activity is a great way to support such requirements.

## Configuration

This section provides instructions on how to configure a **Highlight** activity, shown in Figure 4-13.

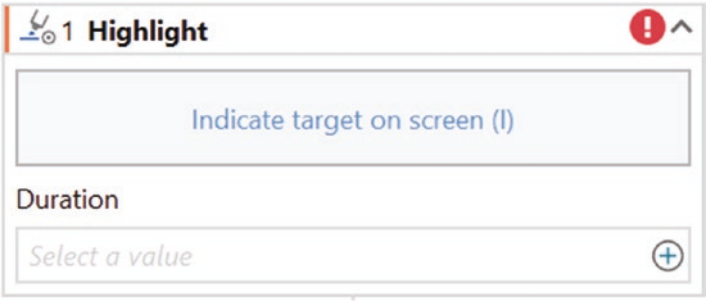


***Figure 4-13.*** *Activity card for Highlight*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify the element or area on UI that you want to highlight.

**Duration**: This is an optional configuration available on the activity card. This configuration allows you to specify (in seconds) how long the automation should highlight the specified element.

**Color**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the color of the highlight box and is set to gold by default.

## EXERCISE

**Goal**: Use the `Highlight` activity to highlight the number of records in the Contacts table visually. This exercise builds upon the `Go To URL` and `Navigate Browser` exercises. Figure 4-10 shows the state of the target web application before this exercise.

**Source Code**: Chapter_4_MiscellaneousActivitiesExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Highlight` activity in the body of `Use Application/Browser` activity after the `Navigate Browser` activity.
2. Next, in the `Highlight` activity, click `Indicate target on screen (I)` link. You'll notice a green highlight as you point your mouse to specify the target element; select the bottom-left area of the table. Then, you'll notice a blue highlight as you point your mouse to specify an anchor; indicate the `Previous` button. At this point, your selection should resemble Figure 4-14. Click the `Confirm` button.

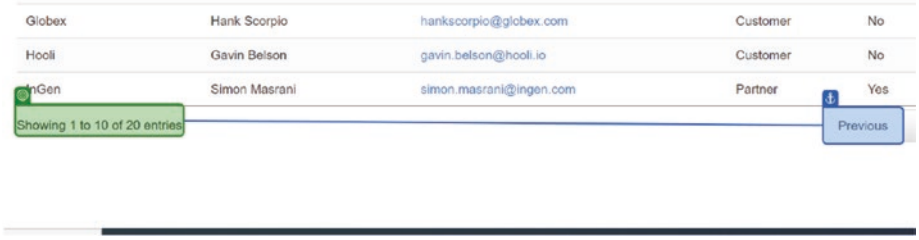


***Figure 4-14.*** *Selection of element to be highlighted*

3. Next, in the `Duration` field of `Highlight` activity, click the `Plus` icon, select the `Number` option, and enter `5` seconds. This will highlight the specified UI element for 5 seconds.

Once you have completed the exercise, the final configuration of the **Highlight** activity should resemble Figure 4-15. Figure 4-16 shows the state of the target web application once the automation has completed its run. In this case, the total number of entries area is highlighted.

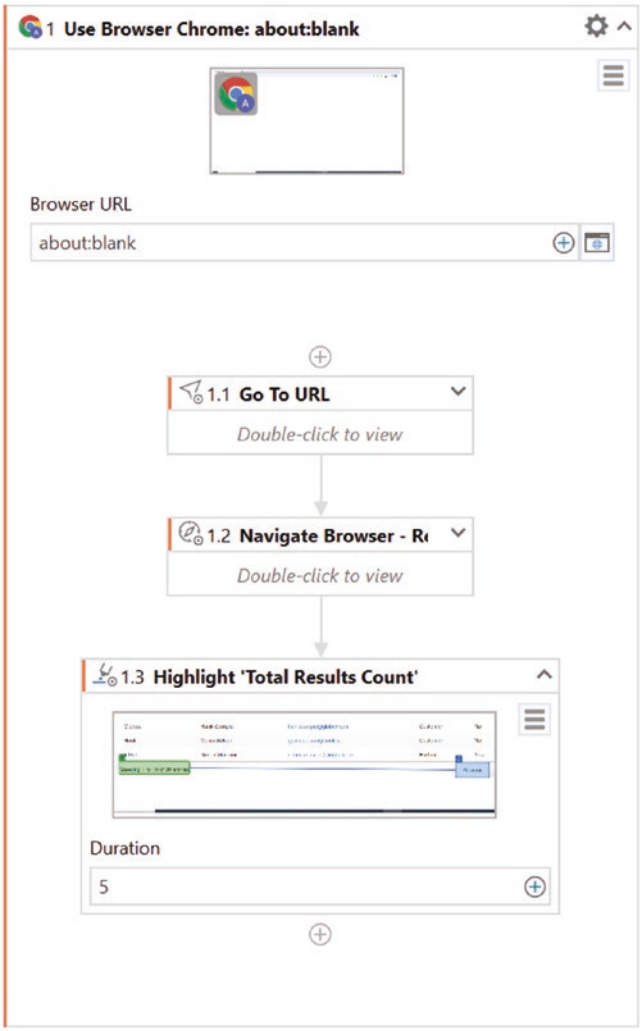


***Figure 4-15.*** *Final configuration of the Highlight activity exercise*

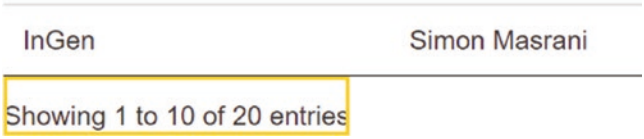


***Figure 4-16.*** *Result of the Highlight activity exercise*

# Take Screenshot

The **Take Screenshot** activity allows you to capture a screenshot of a specified area.

---

**Tip** For tasks where you need to provide evidence for audit purposes that a particular action was taken, the `Take Screenshot` activity is a great way to support such requirements.

---

## Configuration

This section provides instructions on how to configure a **Take Screenshot** activity, shown in Figure 4-17.

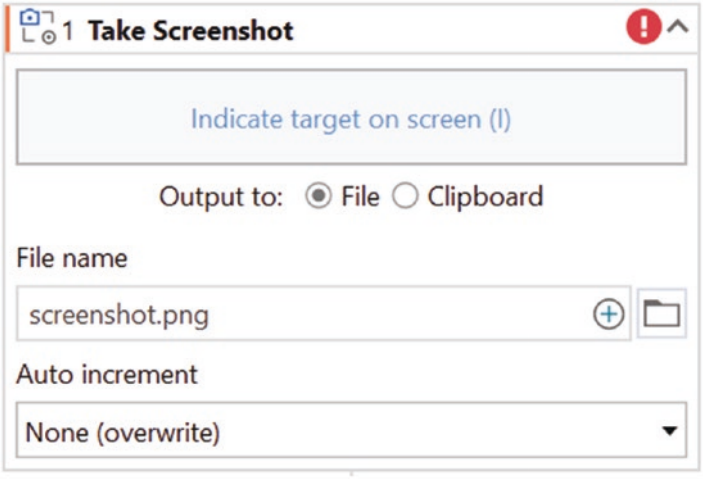


***Figure 4-17.*** *Activity card for Take Screenshot*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify the element on UI that you want to capture in your screenshot.

**Output to**: This is a required configuration available on the activity card. This configuration allows you to specify if the screenshot should be physically saved as a file or just saved in the Clipboard memory.

**File name**: This is a required configuration available on the activity card. This configuration is only available when you select `File` from the `Output to` field. This configuration allows you to specify the folder location where screenshots should be stored and the name of the screenshot.

**Auto increment**: This is an optional configuration available on the activity card. This configuration is only available when you select `File` from the `Output to` field. This configuration automatically appends an index or a timestamp, shown in Figure 4-18, to the file name only in case the same file already exists in the `Save to` location. This field is helpful in avoiding overwriting over an existing file.

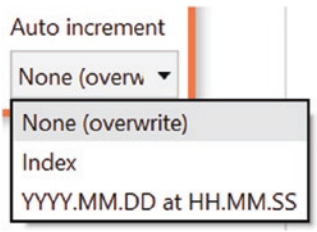


***Figure 4-18.*** *Available options for Auto increment*

## EXERCISE

**Goal**: Use the `Take Screenshot` activity to take a screenshot of the Contacts table. This exercise builds upon the `Go To URL`, `Navigate Browser`, and `Highlight` exercises. Figure 4-10 shows the state of the target web application before this exercise.

**Source Code**: Chapter_4_MiscellaneousActivitiesExercise

**Setup**: Here are step-by-step implementation instructions:

1. In StudioX, add the `Take Screenshot` activity in the body of `Use Application/Browser` activity after the `Highlight` activity.
2. Next, in the `Take Screenshot` activity, click `Indicate target on screen (I)` link and point your mouse to the contacts list table. Once you have specified the target element, you'll notice a red highlight over the table. This is to prompt you to specify an anchor; in this case, use the page header. At this point, your selection should look like Figure 4-19. Click the `Confirm` button.


***Figure 4-19.*** *Selection of element to be captured in the screenshot*

3. Next, in the `Save to folder` field of the `Take Screenshot` activity, click the `Browse for folder` icon and select `C:\BookSamples\Chapter_04` folder.
4. The name of the screenshot file will be automatically appended to the folder path. The default name is `screenshot.png`.
5. From the `Auto increment` dropdown, select the `YYYY.MM.DD at HH.MM.SS` option. This will append a timestamp at the end of the screenshot file name.

Once you have completed the exercise, the final configuration of the **Take Screenshot** activity should resemble Figure 4-20. Figure 4-21 shows the folder with multiple screenshots. If you run the automation multiple times, it will keep adding screenshots to the specified folder with a timestamp appended to the name.


***Figure 4-20.*** *Final configuration of the Take Screenshot activity exercise*

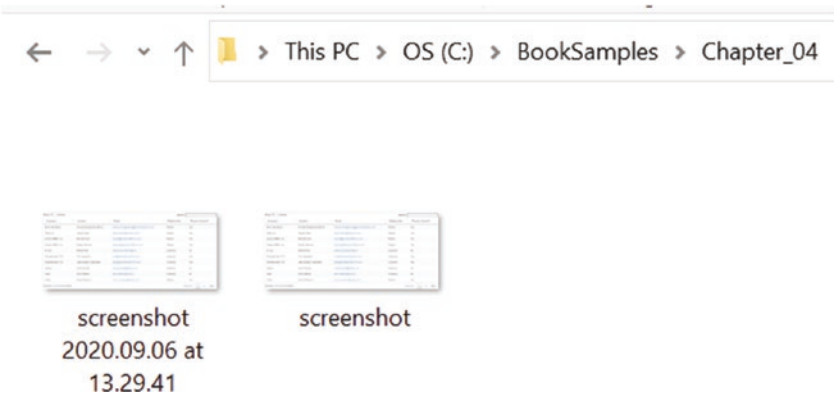


***Figure 4-21.*** *Result of the Take Screenshot activity exercise*

## Check App State

When needing to perform an action based on a particular UI element such as a loading icon on an application, the Check App State activity can be used. We, as humans, understand that this means the screen is still loading, but the automation does not. The **Check App State** activity is a great way to ensure that a screen has completely loaded before the automation starts interacting with it.

---

**Tip** For tasks where you need to perform different actions based on UI elements such as a message box or error pop-up window, the Check App State activity is a great way to support such requirements.

---

### Configuration

This section provides instructions on how to configure a **Check App State** activity, shown in Figure 4-22.

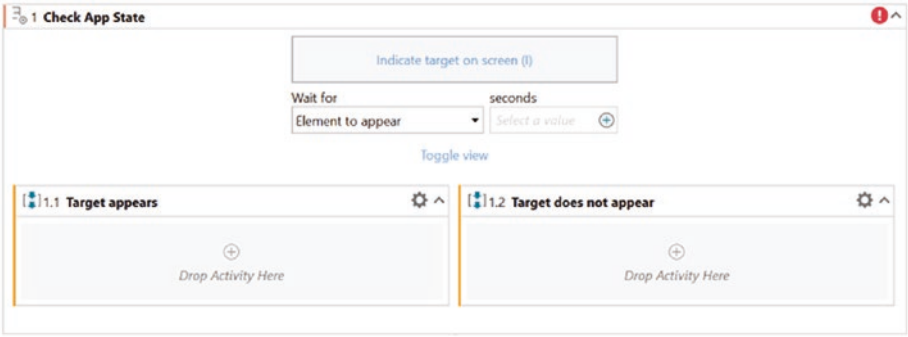


***Figure 4-22.*** *Activity card for Check App State*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify a UI element that the automation will wait for either to appear or to disappear before proceeding.

**Wait for**: This is a required configuration available on the activity card. This configuration allows you to specify if the automation needs to wait for the target element to appear or disappear before proceeding. By default, this is set to `Element to appear`.

**Seconds**: This is an optional configuration available on the activity card. This configuration allows you to specify the amount of time (in seconds) that the automation needs to wait for the target element to appear or disappear before proceeding. By default, this is set to 5 seconds.

**Target appears/disappears**: A set of activities you want to run if the target successfully appears or disappears will be added to this block.

**Target does not appear/disappear**: A set of activities you want to run if the target does not appear or disappear will be added to this block.

**Result**: This is an optional configuration available on the `Properties` panel. This configuration allows you to save the result (`True` or `False`) for later use.

## EXERCISE

**Goal**: Use the `Check App State` activity to wait for the spinner to disappear before proceeding.

**Source Code**: Chapter_4_CheckAppStateExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter `https://therpabook.com/samples/contactsmanagement/home_load.html` in the URL field.
2. In `StudioX`, add the `Use Application/Browser` activity to a blank process.
3. Next, in the `Use Application/Browser` activity card, click `Indicate application` and point your mouse to the browser. This will automatically populate the activity card.

---

**Note** Because the Use Application/Browser card is capturing the URL and it's not changing while it's loading, you can indicate the browser while the page is loading or has loaded.

---

4. Next, select `Use Application/Browser` activity card, and in the `Properties`, set the `Options` ➤ `Open` property to `Always`. This will ensure that the automation always opens a new browser.
5. Next, add a `Check App State` activity in the body of `Use Application/Browser` activity.

6. Next, in the Check App State activity, click the Indicate target on screen (I) link and point your mouse to the spinner. The sample app shows the spinner for 10 seconds, so you will need to indicate the element within that timeframe. If that timeframe is too short, use the pause option (pressing F2 from your keyboard) while indicating the target. Additionally, you can hit refresh (F5 or Ctrl + R) as the pause is about to end, so that you have enough time to indicate the spinner.
7. Next, from the Wait for dropdown, select the Element to disappear option.
8. In the Seconds field, click the Plus icon, select the Number option, and type 15 seconds (this is adding a 5-second buffer).

---

**Tip** When there is a potential lag of a UI element appearing or disappearing, adding a buffer to the seconds field of the Check App State activity is a great way to support latency.

---

9. Next, in the Target disappears block, add a Message Box activity.
10. Next, in the Message Box activity, click the Plus icon, select the Text option, and type Target disappeared. Click Save.
11. Next, in the Target does not disappear block, add a Message Box activity.
12. Next, in the Message Box activity, click the Plus icon, select the Text option, and type Target did not disappear. Click Save.

Once you have completed the exercise, the final configuration of the **Check App State** activity should resemble Figure 4-23. Once you run the automation, it will wait for 15 seconds and then check if the spinner has disappeared. Depending on the state of the target application, a message box will appear with the appropriate message.

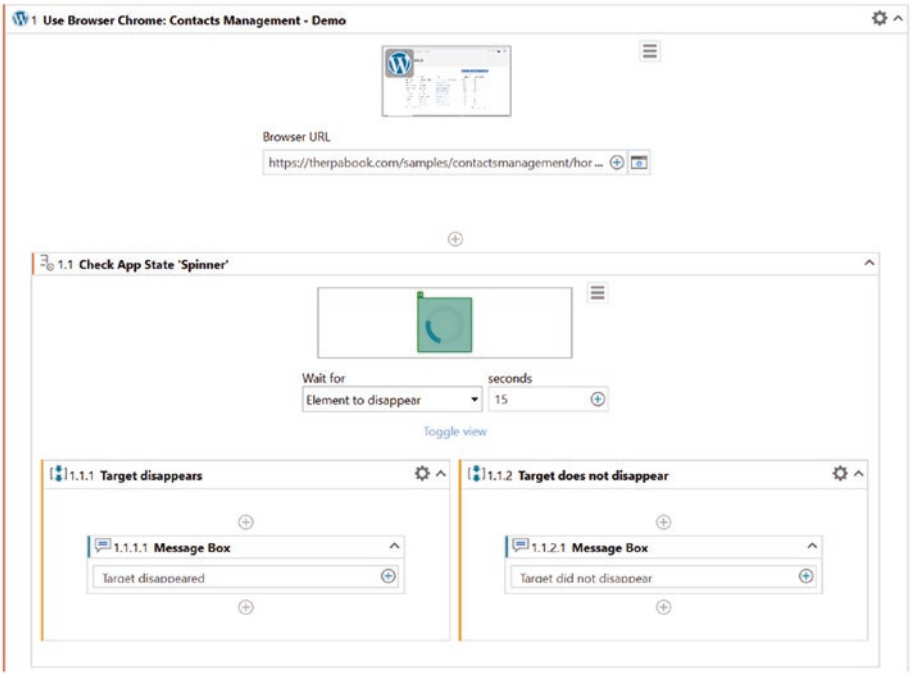


***Figure 4-23.*** *Final configuration of the Check App State activity to wait for spinner to disappear*

## Click

The **Click** activity allows you to perform a mouse click on a specified UI element.

## Configuration

This section provides instructions on how to configure a **Click** activity, shown in Figure 4-24.

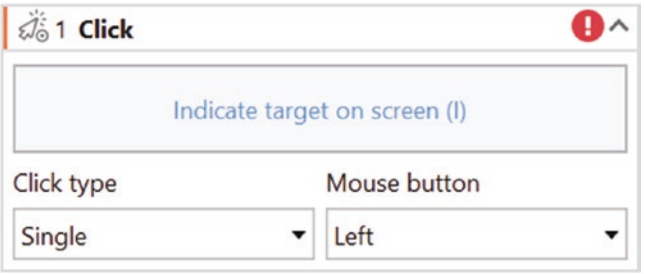


***Figure 4-24.*** *Activity card for Click*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify a UI element that you want the automation to click.

**Click type**: This is a required configuration available on the activity card. This configuration allows you to specify the type of mouse click you want to perform. Figure 4-25 shows all the available options. By default, this is set to `Single` click.

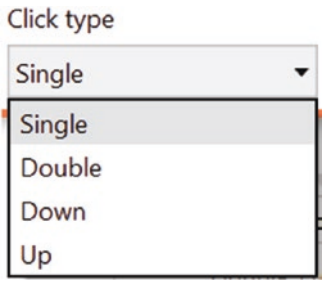


***Figure 4-25.*** *Available options for click types*

**Mouse button**: This is a required configuration available on the activity card. This configuration allows you to specify which mouse button to click. Figure 4-26 shows all the available options. By default, this is set to `Left` click.

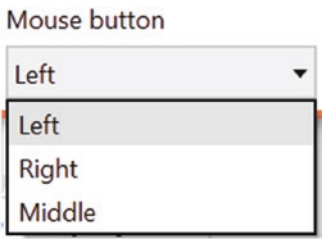


***Figure 4-26.*** *Available options for mouse button click*

**Key modifiers**: This is an optional configuration available on the `Properties` panel. This configuration allows you to use keys in combination with the mouse click action. Figure 4-27 shows all available options. By default, this is set to `None`.

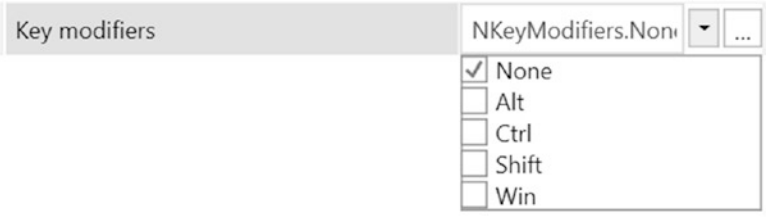


***Figure 4-27.*** *Available options for key modifiers*

## EXERCISE

**Goal**: Use the Click activity to click the Add Contact + button to open the Contact Details dialog.

**Source Code**: Chapter_4_FormDataEntryExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter https://therpabook.com/samples/contactsmanagement/ in the address bar.
2. In StudioX, add the Use Application/Browser activity to a blank process.
3. Next, click Indicate application in the Use Application/Browser activity card, and point your mouse to the browser. This will automatically populate the activity card.
4. Next, select the Use Application/Browser activity card, and from Properties, set the Options ➤ Open property to Always. This will ensure that the automation always opens a new browser.
5. Next, add a Click activity within the Use Application/Browser activity.
6. Next, in the Click activity, click the Indicate target on screen (I) link and you'll notice a green highlight as you point your mouse to the Add Contact button. Then, you'll see a blue highlight as you point your mouse to specify an anchor; indicate the View Contact button. If you are unable to detect any elements, click F4 to change the detection mechanism. Once you have selected the target element, use the View Contact button as an anchor. At this point, your selection should look like Figure 4-28. Click the Confirm button.

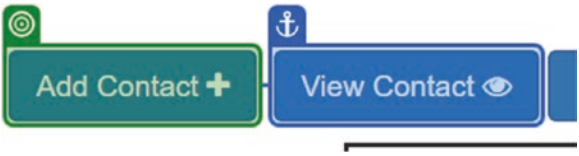


***Figure 4-28.*** *Target and anchor for the Add Contact button*

Once you have completed the exercise, the final configuration of the **Click** activity should resemble Figure 4-29.

---

**Tip** When needing to click within an application that is minimized, using the Maximize Window activity can be used to maximize the application for the bot to click the correct UI element. Additionally, setting the Input mode to Simulate within the properties pane of the Click activity can have the bot click the application even when minimized.

---

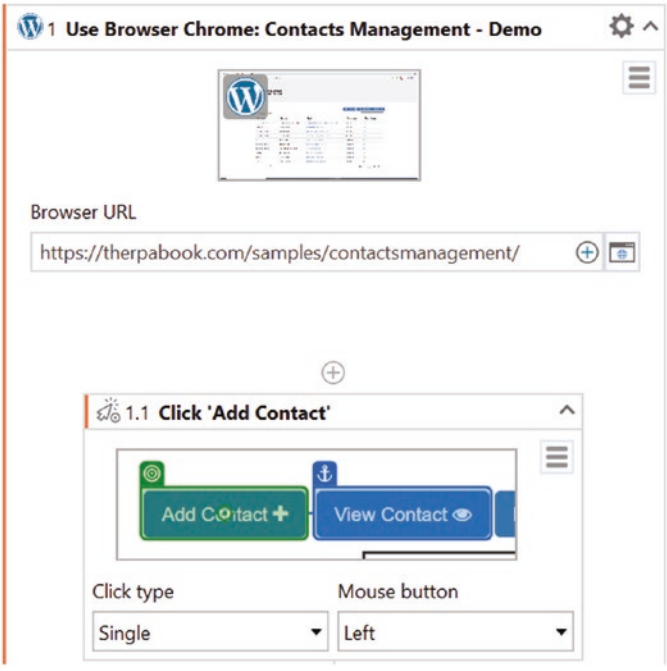


***Figure 4-29.*** *Final configuration of the Click activity exercise*

Figure 4-30 shows the state of the target web application once the automation has completed its run. In this case, an empty Contact Details dialog is displayed.

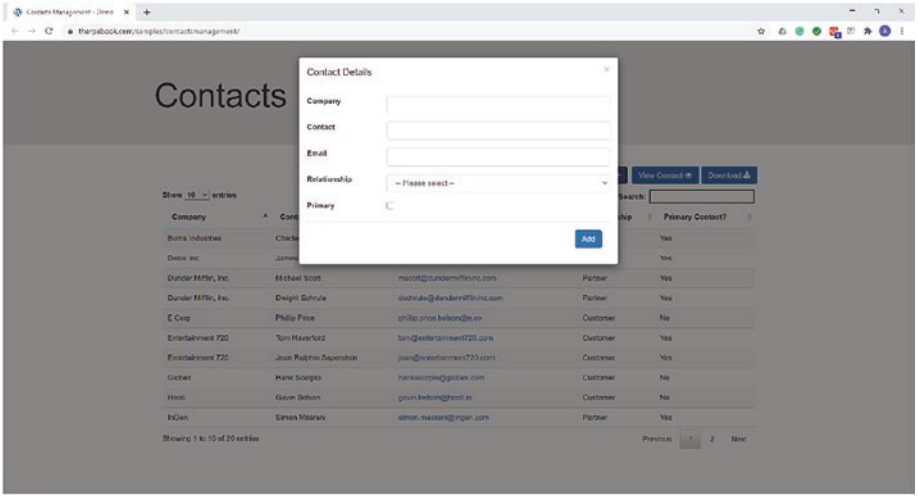


***Figure 4-30.*** *Result of the Click activity exercise*

# Type Into

The **Type Into** activity allows you to enter text in a specified element on the UI.

---

**Tip** When a dropdown menu of an application does not support the Select Item activity (discussed next), using a Type Into activity is a great way to support selecting a specific dropdown option.

---

## Configuration

This section provides instructions on how to configure a **Type Into** activity, shown in Figure 4-31.

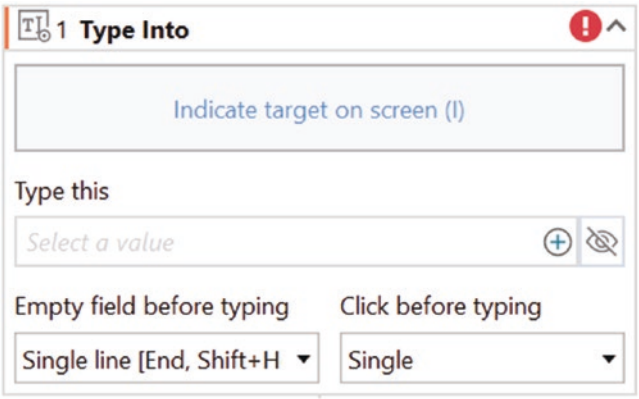


***Figure 4-31.*** *Activity card for Type Into*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify which UI element you want to enter data in.

**Type this**: This is an optional configuration available on the activity card. This configuration allows you to specify the text that should be entered in the target element. You also have the option to include special keys in the text or toggle the password mode to type the text securely. This text can be static or dynamic.

**Empty field before typing**: This is an optional configuration available on the activity card. This configuration allows you to specify if you want to keep the existing text in the target element or clear it. Set to `None` if you want to leave the existing text as is. If you want to clear the existing text, then use either the single line or multiline options. If you are dealing with a single line Text field, then using `Single line` will clear the text, and if you are dealing with a multiline component like a Text Area, then the `Multi line` option will work the best. As a note, field data is cleared using keyboard shortcuts.

**Click before typing**: This is an optional configuration available on the activity card. This configuration allows you to specify if the automation should click the target element before it starts typing. You can either do a `Single` click or a `Double` click. By default, this is set to `Single`.

**Tip** Click before typing is useful in scenarios where text fields do not become editable until you click them once or twice.

**Delay between keys**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify a delay between each keystroke. By default, there is no delay set. This field is only applicable for Hardware Events input methods.

**Deselect at end**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify if, at the end of the activity, the automation should remove move from the UI element. This field is only applicable for Simulate input methods.

## EXERCISE

**Goal**: Use the `Type Into` activity to fill out text fields of the Contact Details dialog. This exercise builds upon the previous exercise of the `Click` activity. Figure 4-30 shows the state of the target web application prior to this exercise.

**Source Code**: Chapter_4_FormDataEntryExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Type Into` activity within the `Use Application/Browser` activity right after the `Click 'Add Contact'` activity.
2. Next, in the `Type Into` activity, click `Indicate target on screen (I)` link and point your mouse to the `Company` field. The field label is auto-detected as an anchor. At this point, your selection should look like Figure 4-32. Click the `Confirm` button.


***Figure 4-32.*** *Target and anchor for the Company text field*

3. Next, in the `Type this` field, click the `Plus` icon, select the `Text` option, and type `Stark Industries`. At this point, your `Type Into` activity configuration should resemble Figure 4-33.

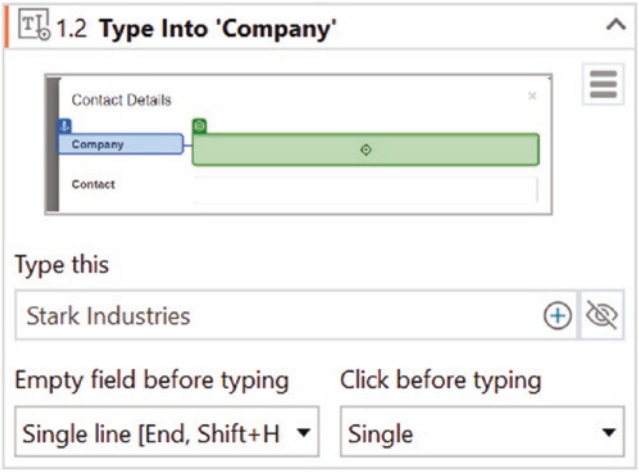


***Figure 4-33.*** *Configuration of Type Into activity*

4. Next, repeat steps 2–4 for the `Contact` field.
5. Next, In the `Type this` field, click the `Plus` icon, select the `Text` option, and type `Tony Stark`.
6. Next, repeat steps 2–4 for the `Email` field.
7. Next, in the `Type this` field, click the `Plus` icon, select the `Text` option, and type tstark@starkindustries.com.

Once you have completed the exercise, the final configuration of the **Type Into** activities should resemble Figure 4-34. Figure 4-35 shows the state of the target web application once the automation has completed its run. In this case, data has been entered into Company, Contact, and Email fields on Contact Details dialog.


***Figure 4-34.*** *Final configuration of the Type Into activity exercise*

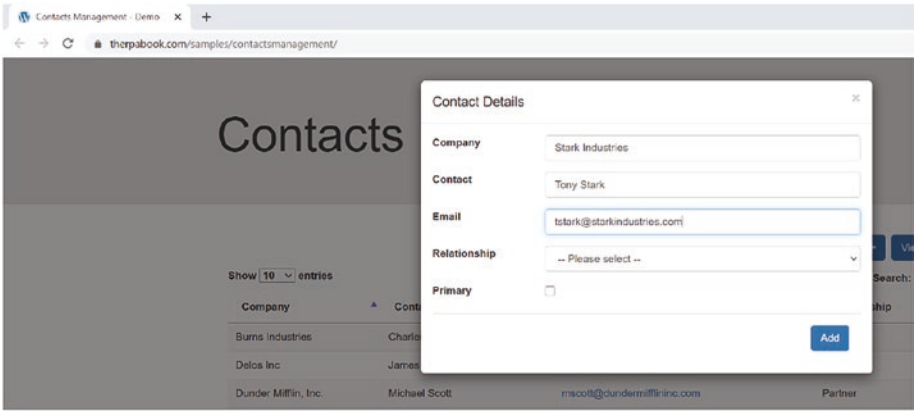


***Figure 4-35.*** *Result of the Type Into activity exercise*

## Select Item

The **Select Item** activity allows you to select an item in a specified dropdown list.

---

**Note** If the application dropdown is not a combo box or list box, the Select Item activity may not be supported. In this case, it's advised to use a Type Into activity.

---

### Configuration

This section provides instructions on how to configure a **Select Item** activity, shown in Figure 4-36.


***Figure 4-36.*** *Activity card for Select Item*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify a dropdown on the UI in which you want to select an item.

**Item to select**: This is an optional configuration available on the activity card. By default, this list is empty. Once you have indicated the dropdown element on the UI, this list will be populated with all options available in the dropdown. Figure 4-37 shows an example when this list is populated.

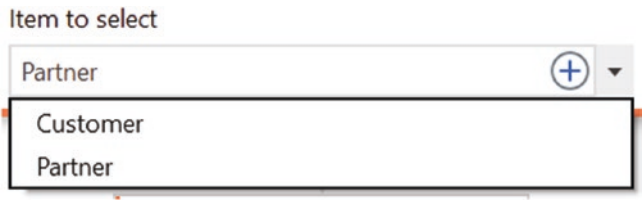


***Figure 4-37.*** *Available options in the dropdown*

## EXERCISE

**Goal**: Use the `Select Item` activity to choose Partner from Relationship dropdown on the Contact Details dialog. This exercise builds upon the previous exercises for `Click` and `Type Into` activities. Figure 4-35 shows the state of the target web application before this exercise.

**Source Code**: Chapter_4_FormDataEntryExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Select Item` activity in the body of `Use Application/Browser` activity after the three `Type Into` activities.
2. In the `Select Item` activity, click the `Indicate target on screen (I)` link and point your mouse to the `Relationship` dropdown. The field label, Relationship, is auto-detected as an anchor. At this point, your selection should resemble Figure 4-38. Click the `Confirm` button.

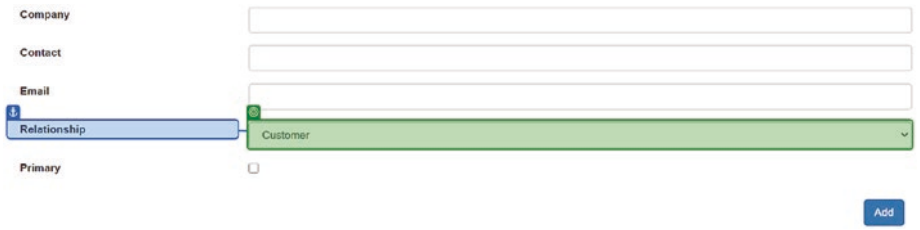


***Figure 4-38.*** *Target and anchor for Relationship dropdown*

3. At this point, the `Item to select` field will be populated with all options available in the `Relationship` dropdown. Select `Partner` as the relationship from the list.

Once you have completed the exercise, the final configuration of the **Select Item** activity should resemble Figure 4-39. Figure 4-40 shows the state of the target web application once the automation has completed its run. In this case, an item has been selected from the Relationship field.

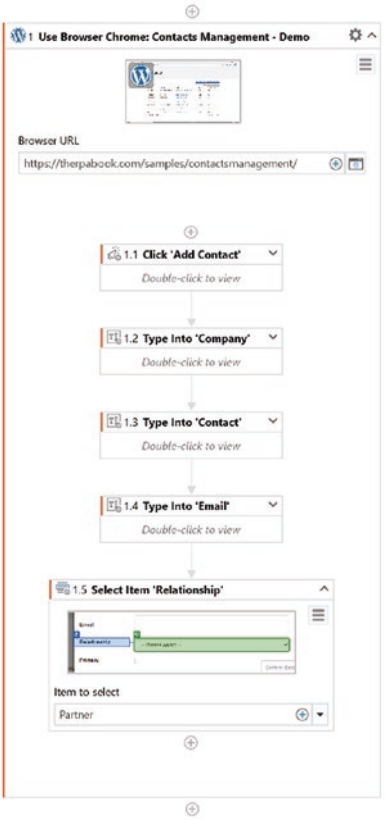


***Figure 4-39.*** *Final configuration of the Select Item activity exercise*

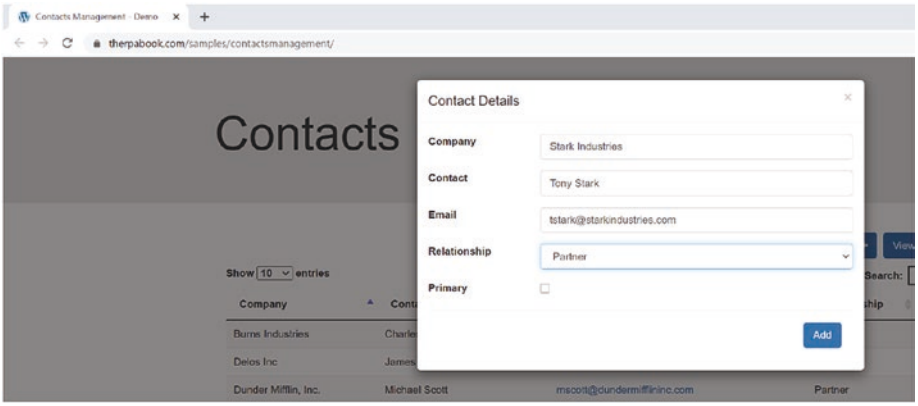


***Figure 4-40.*** *Result of the Select Item activity exercise*

## Check/Uncheck

The **Check/Uncheck** activity allows you to interact with checkboxes on a user interface.

---

**Tip** In place of using a Click activity to interact with a checkbox within an application, using the Check/Uncheck activity gives the option to toggle, check, or uncheck a field. This way, if the application may already have a checkmark marked or unmarked, it won't unnecessarily change it.

---

### Configuration

This section provides instructions on how to configure a **Check/Uncheck** activity, shown in Figure 4-41.

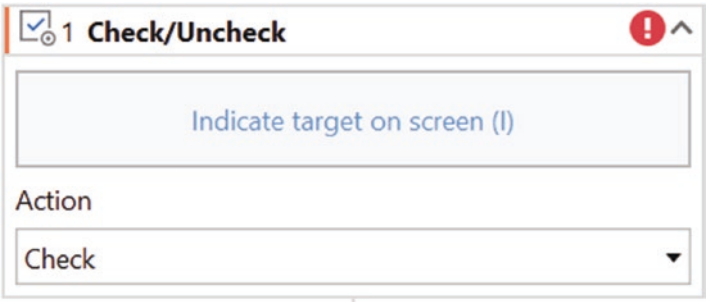


***Figure 4-41.*** *Activity card for Check/Uncheck*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify a checkbox on the UI which you want to check or uncheck.

**Action**: This is a required configuration available on the activity card. This configuration allows you to specify the state of the checkbox. Figure 4-42 shows all available actions. By default, this is set to `Check`.

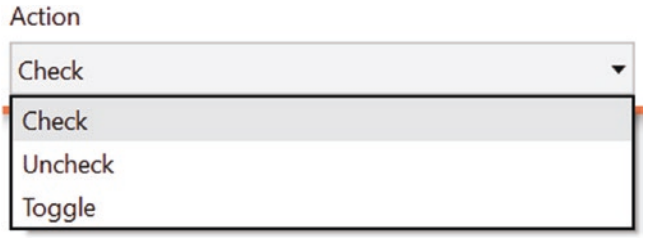


***Figure 4-42.*** *Available action for Check/Uncheck activity*

## EXERCISE

**Goal**: Use the Check/Uncheck activity to specify that the contact being added is primary by checking the Primary option in the Contact Details dialog. This exercise builds upon the previous exercises for Click, Type Into, and Select Item activities. Figure 4-40 shows the state of the target web application before this exercise.

**Source Code**: Chapter_4_FormDataEntryExercise

**Setup**: Here are step-by-step implementation instructions:

1. In StudioX, add the Check/Uncheck activity in the body of Use Application/Browser activity after the Select Item activity.
2. Next, in the Check/Uncheck activity, click Indicate target on screen (I) link and point your mouse to the Primary checkbox. Select the field label, Primary, as an anchor. At this point, your selection should look like Figure 4-43. Click the Confirm button.


***Figure 4-43.*** *Target and anchor for Primary checkbox*

3. Next, select Check from the Action dropdown.

Once you have completed the exercise, the final configuration of the **Check/Uncheck** activity should resemble Figure 4-44. Figure 4-45 shows the state of the target web application once the automation has completed its run. In this case, the Primary checkbox has been checked.


***Figure 4-44.*** *Final configuration of the Check/Uncheck activity exercise*

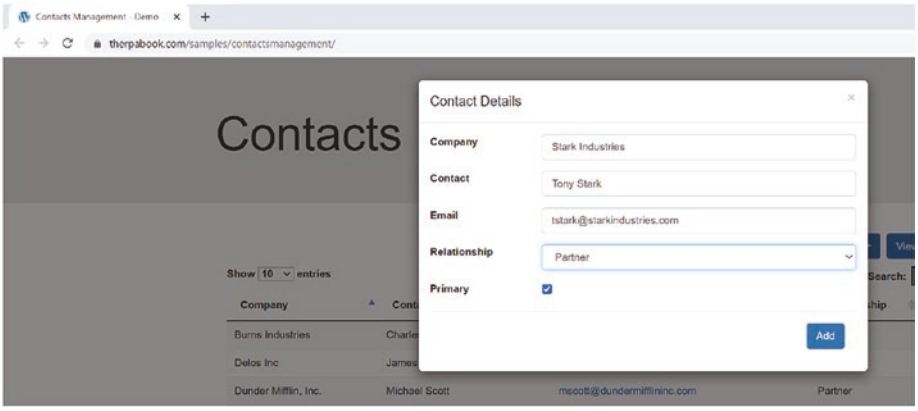


***Figure 4-45.*** *Result of the Check/Uncheck activity exercise*

## Get Text

The **Get Text** activity allows you to extract text from a specified element on the UI.

### Configuration

This section provides instructions on how to configure a **Get Text** activity, shown in Figure 4-46.


***Figure 4-46.*** *Activity card for Get Text*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify an element on the UI from which you want to retrieve text.

**Save to**: This is a required configuration available on the activity card. This configuration allows you to specify how you want to store retrieved text for use later.

## EXERCISE

**Goal**: Use the `Get Text` activity to read all fields of the View Contact Details dialog.

**Source Code**: Chapter_4_FormDataExtractionExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter `https://therpabook.com/samples/contactsmanagement/` in the address bar.
2. In StudioX, add the `Use Application/Browser` activity to a blank process.
3. Next, click `Indicate application` in the `Use Application/Browser` activity card, and point your mouse to the browser. This will automatically populate the activity card.
4. Next, select the `Use Application/Browser` activity card, and from `Properties`, set the `Options` ➤ `Open` property to `Always`. This will ensure that the automation always opens a new browser.
5. Add a `Click` activity within the `Use Application/Browser` activity and configure it to click the `View Contact` button.
6. Next, make sure the `View Contact` dialog is open.

7. Add the `Get Text` activity within the `Use Application/Browser` activity right after the `Click` activity.

8. In the `Get Text` activity, click the `Indicate target on screen (I)` link and point your mouse to the `Company` field. The field label is auto-detected as an anchor. At this point, your selection should resemble Figure 4-47. Click the `Confirm` button.

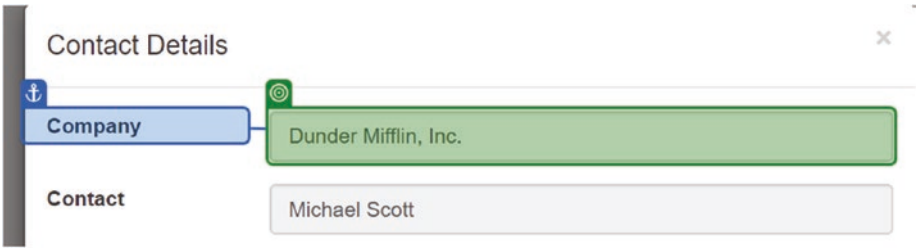


***Figure 4-47.*** *Target and anchor for the Company field*

9. Next, click the `Plus` icon in the `Save to` the field. Select the `Save for Later Use` option and name your saved value as `CompanyText`. Click Ok. At this point, your activity `Get Text` activity configuration should resemble Figure 4-48.


***Figure 4-48.*** *Configuration of Get Text activity*

10. Next, repeat steps 7–9 for the remaining fields, that is, `Contact`, `Email`, `Relationship`, and `Primary`. Name the saved values as `ContactText`, `EmailText`, `RelationshipText`, and `PrimaryText`, respectively.

11. Next, add a `Write Line` activity at the end. Click the `Plus` icon in the `Text` field, select the `Text` option, and enter the text as shown in Figure 4-49. To reference the values saved, click the `Plus` icon in the `Text Builder`, hover over `Use Saved Value`, and select `CompanyText`, `ContactText`, `EmailText`, `RelationshipText`, and `PrimaryText`, respectively.

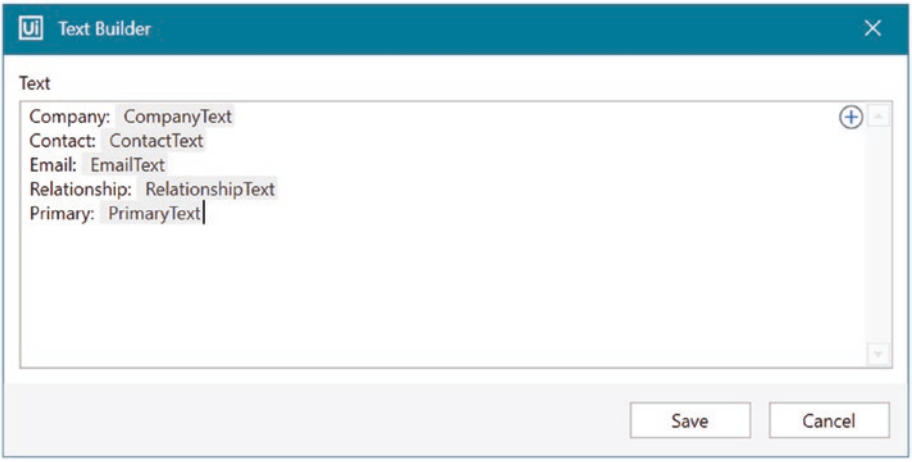


***Figure 4-49.*** *Write Line activity configuration to print all form data*

Once you have completed the exercise, the final configuration of the **Get Text** activity should resemble Figure 4-50. Figure 4-51 shows the data from the Write Line activity printed in the Output panel.

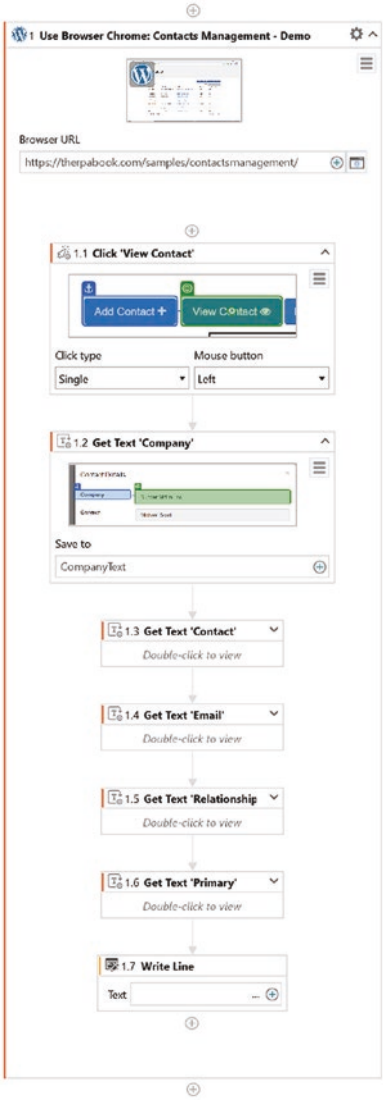


***Figure 4-50.*** *Final configuration of automation with Get Text activities*

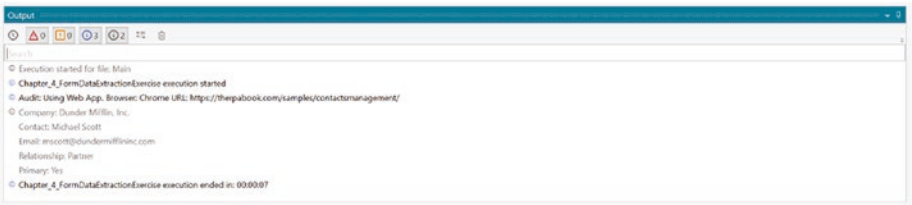


***Figure 4-51.*** *Output of the Get Text activity exercise*

# Get Attribute

The **Get Attribute** activity allows you to retrieve the value of an attribute of a specified element on the UI.

## Configuration

This section provides instructions on how to configure a **Get Attribute** activity, shown in Figure 4-52.

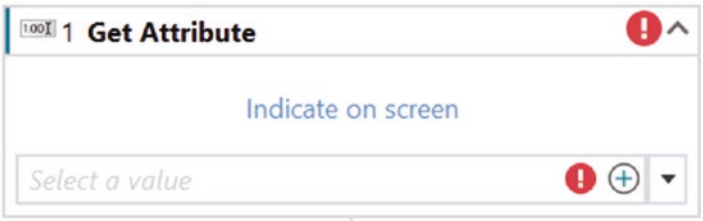


***Figure 4-52.*** *Activity card for Get Attribute*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify an element on the UI from which you want to retrieve the attribute value.

**Attribute**: This is required configuration available on the activity card. This configuration allows you to specify the attribute whose value you want to retrieve.

**Result**: This is an optional configuration available on the activity card. This configuration allows you to specify how you want to store retrieved value of the attribute for use later.

## EXERCISE

**Goal**: Use the `Get Attribute` activity to read value of the Company field from the View Contact Details dialog. This exercise builds upon the previous exercise for the `Get Text` activity.

**Source Code**: Chapter_4_FormDataExtractionExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Get Attribute` activity within the `Use Application/Browser` activity right after the `Write Line` activity.
2. In the `Get Attribute` activity, click the `Indicate on screen` link and point your mouse to the `Company` field.
3. Next, click the `Plus` icon in the `Attribute` field, select the `Text` option, and type value. This is the attribute that we are going to retrieve. Click `Save`.
4. Next, from the `Properties` panel, click the `Plus` icon in the `Result` field. Select the `Save for Later Use` option and name your saved value as `AttributeValue`. Click Ok.
5. Next, add the `Write Line` activity at the end. Click the `Plus` icon in the `Text` field, select the `Text` option, and enter the text as shown in Figure 4-53.

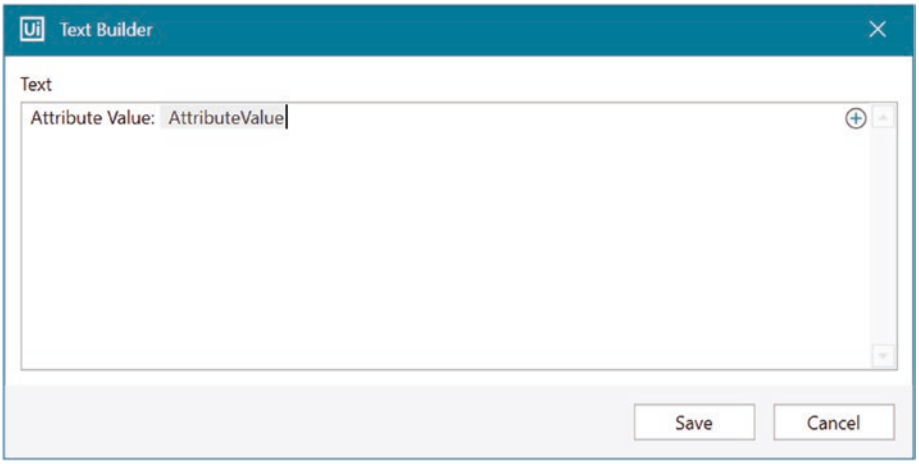


***Figure 4-53.*** *Write Line activity configuration to print all form data*

Once you have completed the exercise, the final configuration of the **Get Attribute** activity should resemble Figure 4-54. Figure 4-55 shows the data from the Write Line activity printed in the Output panel.


***Figure 4-54.*** *Final configuration of automation with Get Attribute activities*

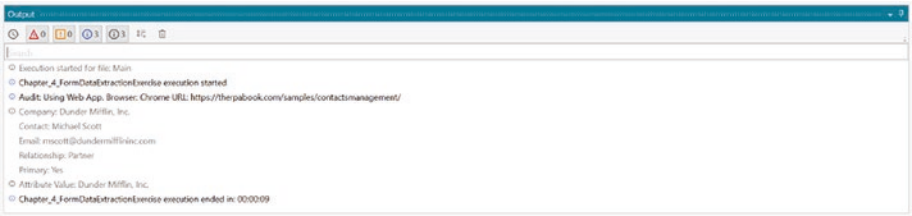


***Figure 4-55.*** *Output of the Get Attribute activity exercise*

# Extract Table Data

The **Extract Table Data** activity allows you to extract data in tabular format from web and desktop applications.

---

**Note** The **Table Extraction** menu item in the top ribbon also follows similar configuration steps. It creates a separate container inside a **Use Application/Browser** activity.

---

## Configuration

This section provides instructions on how to configure an **Extract Table Data** activity, shown in Figure 4-56.

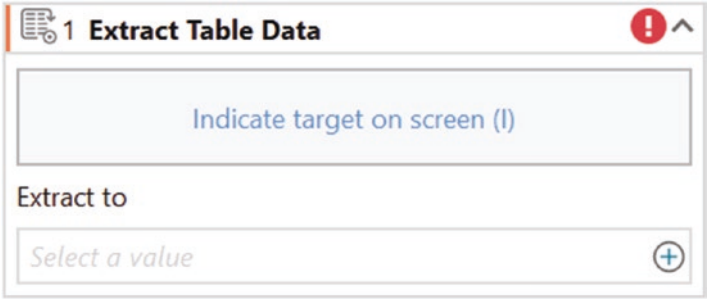


***Figure 4-56.*** *Activity card for Extract Table Data*

**Indicate target on screen (I)**: This is a required configuration available on the activity card. This configuration allows you to specify the table from which you want to extract data. If the table data spans multiple pages, this configuration will also allow you to specify the element that automation can use to navigate between pages.

**Extract to**: This is an optional configuration available on the activity card. This configuration allows you to specify how you want to store extracted data.

**Append results**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify if the automation should append any new extractions at the end of the previously extracted data or overwrite it.

**Edit Next Link**: This configuration becomes available from the activity menu only after you have indicated the target on the screen. This configuration, shown in Figure 4-57, allows you to edit the UI element you have specified to navigate to the next page of data.

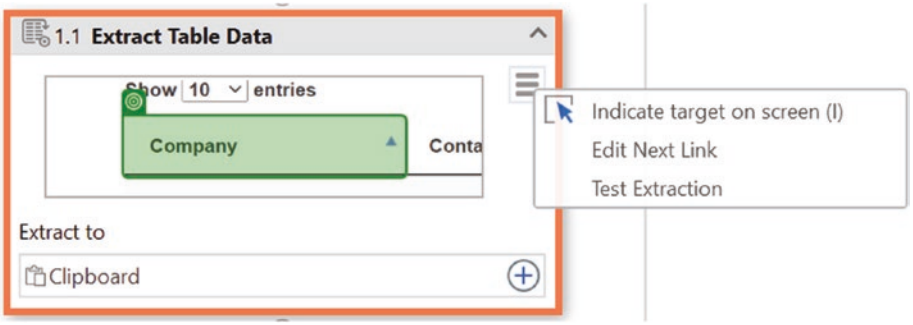


***Figure 4-57.*** *Options in the activity menu*

**Test Extraction**: This configuration becomes available from the activity menu only after you have indicated the target on the screen. This configuration, shown in Figure 4-57, allows you to test your data extraction. Once you click the `Test Extraction` option from the menu options, a popup shown in Figure 4-58 will appear containing extracted data from first page indicated.

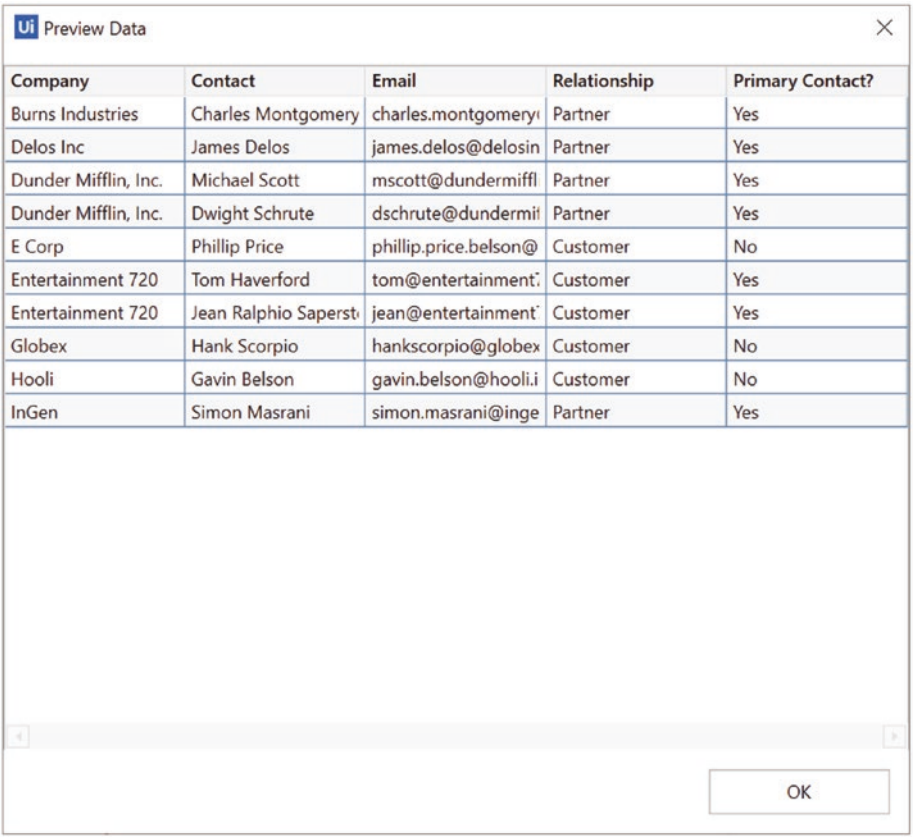


***Figure 4-58.*** *Preview table data using the Test Extraction option*

**Delay between pages**: This is an optional configuration available on the `Properties` panel. This configuration is useful when data spans multiple pages, and after opening the next page, data takes some time to load. You can specify how long (in seconds) you want to wait before extracting data from the next page.

**Maximum results**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the maximum number of results you want to extract. By default, this is `100`.

## EXERCISE

**Goal**: Use the `Extract Table Data` activity to read all contacts from all pages of the table.

**Source Code**: Chapter_4_TableDataExtractionExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter https://therpabook.com/samples/contactsmanagement/ in the address bar.
2. In StudioX, add the `Use Application/Browser` activity to a blank process.
3. Next, click `Indicate application` in the `Use Application/Browser` activity card, and point your mouse to the browser. This will automatically populate the activity card.
4. Next, select the `Use Application/Browser` activity card, and from `Properties`, set the `Options` ➤ `Open` property to `Always`. This will ensure that the automation always opens a new browser.
5. Next, add an `Extract Table Data` activity to the `Use Application/Browser` activity.
6. Next, in the `Extract Table Data` activity, click the `Indicate target on screen (I)` link. This will start the table data extraction wizard, shown in Figure 4-59.

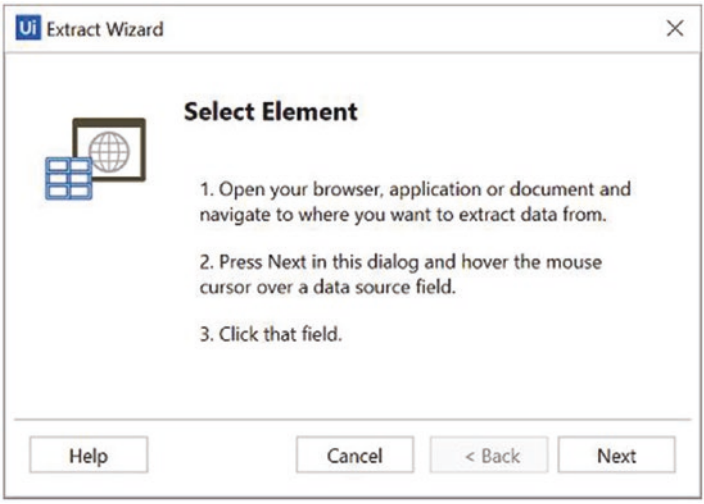


***Figure 4-59.*** *Extract Wizard – initial window*

7. Next, on the `Extract Wizard` dialog, click the `Next` button.
8. At this point, the `Extract Wizard` will prompt you to select a column of the table. Click the `Company` column, as shown in Figure 4-60.


***Figure 4-60.*** *Extract Wizard – column selection*

9. Next, the `Extract Wizard` will prompt you to confirm if you want to extract all columns of the specified table or just the column that you selected, shown in Figure 4-61. Click `Yes`.

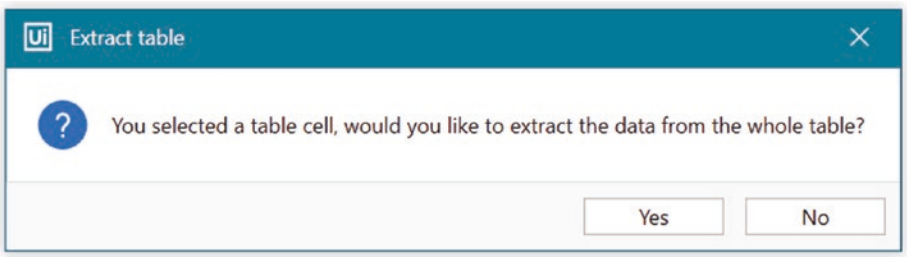


***Figure 4-61.*** *Extract Wizard – prompt to confirm data extraction from a single column or the entire table*

10. Next, the `Extract Wizard` will show you the data it was able to extract, shown in Figure 4-62. If the data looks accurate, that is, you are not missing any columns, then click `Finish`.

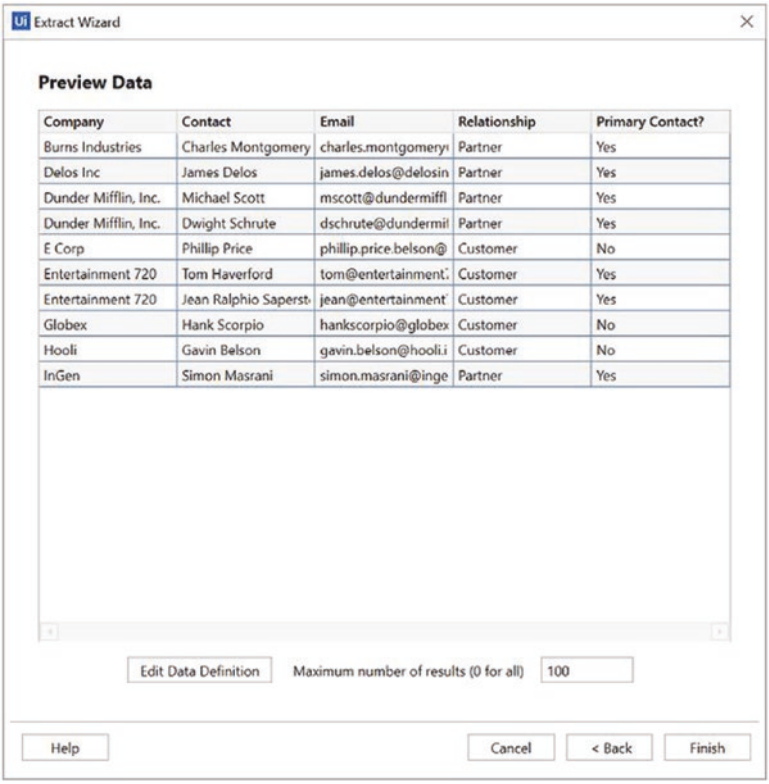


***Figure 4-62.*** *Extract Wizard – preview extracted data*

11. Next, the `Extract Wizard` will prompt another message, shown in Figure 4-63, asking you to confirm if the data spans multiple pages. In the case of this example, it does, so click `Yes`.

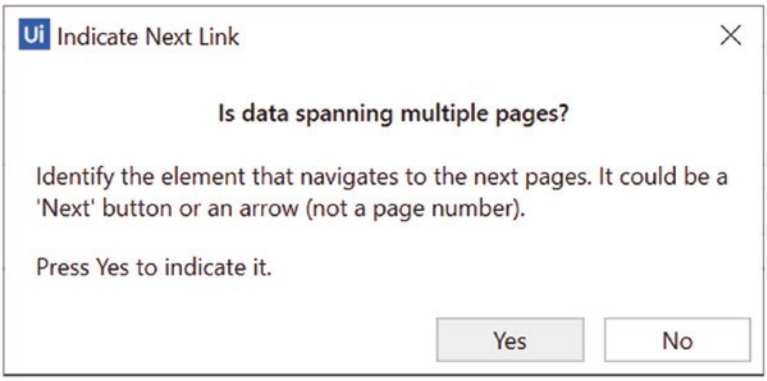


***Figure 4-63.*** *Extract Wizard – confirm if data spans multiple pages*

12. At this point, you will select the `Next` button on the Contacts List screen, shown in Figure 4-64. This button will allow the automation to move to the next page of data. This completes the extraction configuration.


***Figure 4-64.*** *Extract Wizard – select the data navigation UI element*

13. Next, in the `Extract to` field, select `Copy to clipboard` option.

14. Next, add a `Write Line` activity at the end. In the `Text` field, click the `Plus` icon, and select `Paste from clipboard` option.

Once you have completed the exercise, the final configuration of the **Extract Table Data** activity should resemble Figure 4-65. Figure 4-66 shows the data extracted from the table by using the Write Line activity to print to the Output panel.

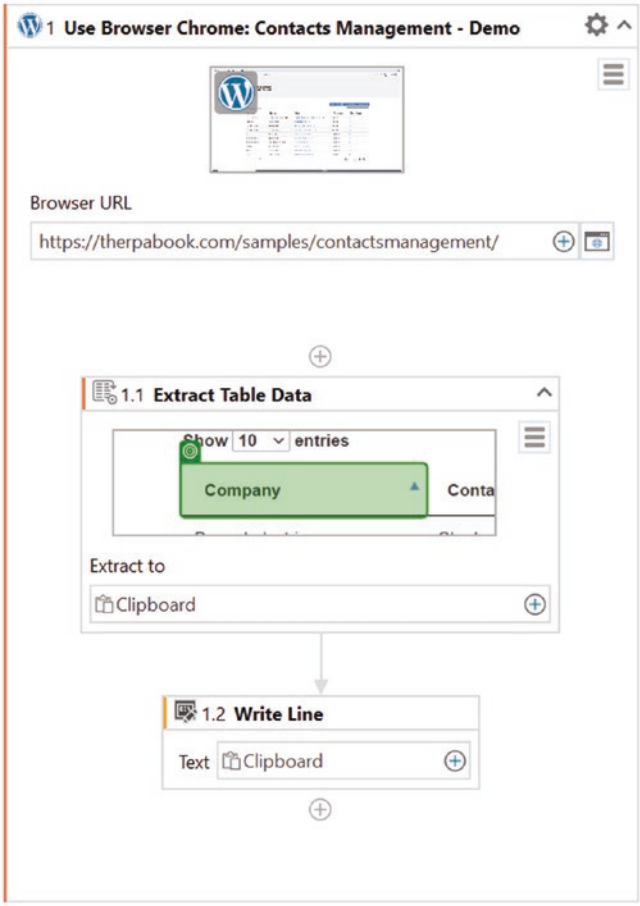


***Figure 4-65.*** *Final configuration of Extract Table Data activity exercise*

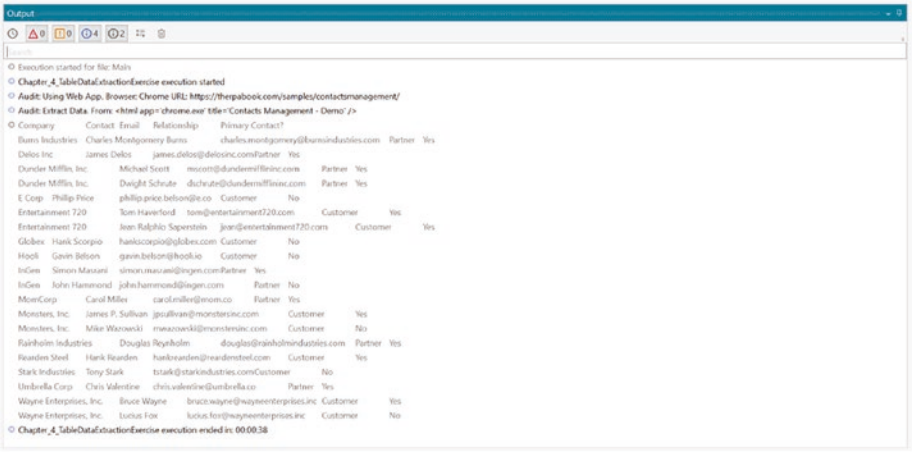


***Figure 4-66.*** *Output of the Extract Table Data activity exercise*

# Hover

The **Hover** activity allows you to hover over a specified element on the UI.

---

**Tip** The Hover activity is typically going to be used with a Click activity. For example, most modern web applications use hover menus, but you need to click a menu item to open it.

---

## Configuration

This section provides instructions on how to configure a **Hover** activity, shown in Figure 4-67.


***Figure 4-67.*** *Activity card for Hover*

**Indicate target on screen (I)**: This is a mandatory configuration available from the activity card. This configuration allows you to specify an element on the UI that you want to hover over.

**Duration**: This is an optional configuration available from the activity card. This configuration allows you to specify how long (in seconds) you want to hover over the specified element.

## EXERCISE

**Goal**: Use the `Hover` activity to open the `Download` menu, and use the `Click` activity to initiate the download of contacts data in Excel format. Use a `Wait for Download` activity to monitor file download.

**Source Code**: Chapter_4_MenuHoverExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter `https://therpabook.com/samples/contactsmanagement/` in the address bar.
2. In StudioX, add the `Use Application/Browser` activity to a blank process.

3. Next, click `Indicate application` in the `Use Application/Browser` activity card, and point your mouse to the browser. This will automatically populate the activity card.

4. Next, select the `Use Application/Browser` activity card, and from `Properties`, set the `Options` ➤ `Open` property to `Always`. This will ensure that the automation always opens a new browser.

5. Next, add the `Hover` activity within the `Use Application/Browser` activity.

6. Next, in the `Hover` activity, click `Indicate target on screen (I)` link to select the `Download` button. Once you have selected the target element, use the `View Contact` button as an anchor. At this point, your selection should resemble Figure 4-68. Click the `Confirm` button.

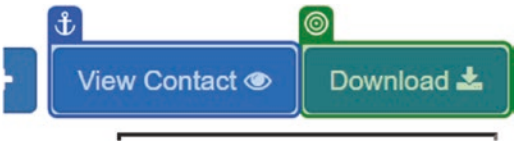


***Figure 4-68.*** *Target and anchor for the Company field*

7. Next, in the `Duration` field, click the `Plus` icon, select the `Number` option, and type `3` seconds.

8. Next, add a `Wait for Download` activity to the body of `Use Application/Browser` activity right after the `Hover` activity.

9. Next, in the `Downloaded file` field, click the `Plus` icon, select `Save for Later Use`, and name your saved value as `DownloadedFile`.

10. Leave the `Monitored folder` field as is.

11. Next, add a `Click` activity to the body of `Wait for Download` activity.

12. Next, in the `Click` activity, click `Indicate target on screen (I)` link to specify the `Excel` option from the `Download` menu. This is not going to be straightforward, because when you are in element selection mode, hover will not work, and you will not be able to select the `Excel` download option. To make this work, after you click Indicate, you must pause the selection process by pressing `F2`. While the selection process is paused, hover over the `Download` button, and make sure you are pointing your mouse on the `Excel` option. Once the selection process resumes, you will be able to select the `Excel` option and use the CSV option as an anchor. At this point, your selection should look like Figure 4-69. Click the `Confirm` button.

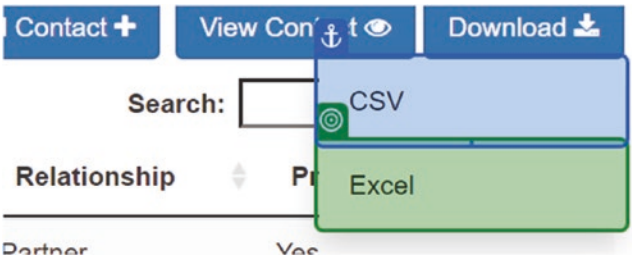


***Figure 4-69.*** *Hover menu selection*

13. Next, add a `Write Line` activity to the body of `Use Application/Browser` activity after the `Wait for Download` activity.

14. Next, in the `Text` field of the `Write Line` activity, click the `Plus` icon, select `Text` option, and type `File downloaded at path: DownloadedFile` ➤ `Full Name`. You will need to use the saved value `DownloadedFile` to display the path of the downloaded file.

Once you have completed the exercise, the final configuration of all activities should resemble Figure 4-70. Figure 4-71 shows the full path of the downloaded file in the Output panel.

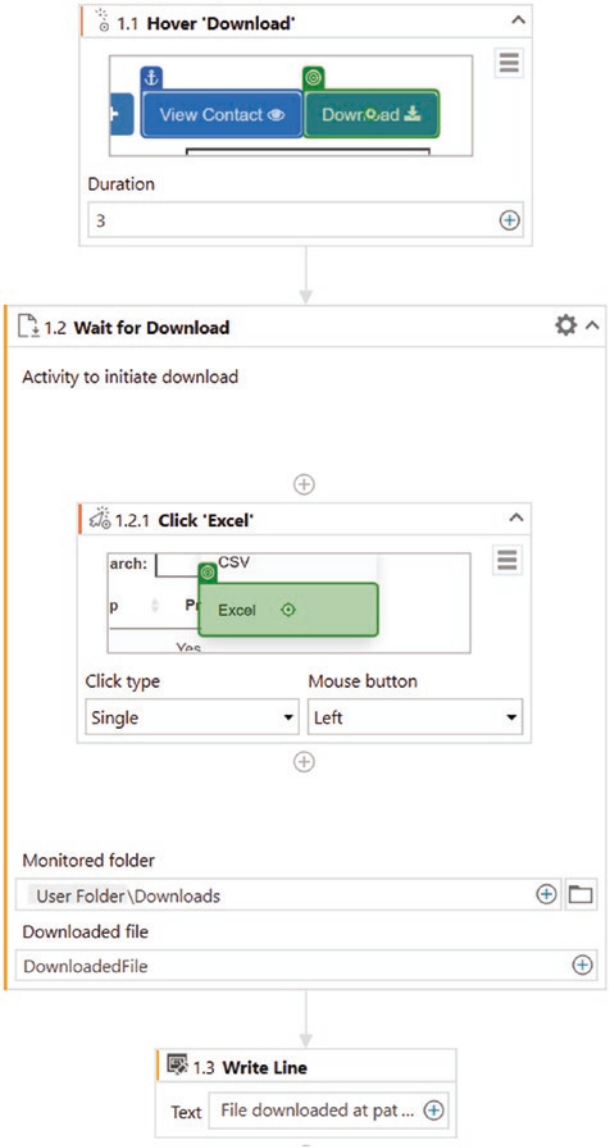


***Figure 4-70.*** *Final configuration of Hover activity exercise*

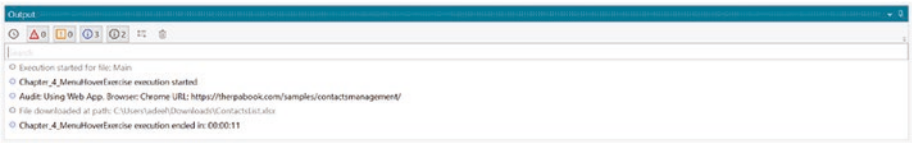


***Figure 4-71.*** *Output of the Hover activity exercise*

# Keyboard Shortcuts

The **Keyboard Shortcuts** activity allows you to send one or multiple keyboard shortcuts to a UI element.

---

**Tip** Keyboard shortcuts can be used while automating legacy applications or in scenarios where you are unable to click a UI element. This activity should not be used as the default navigation mechanism.

---

## Configuration

This section provides instructions on how to configure a **Keyboard Shortcuts** activity, shown in Figure 4-72.

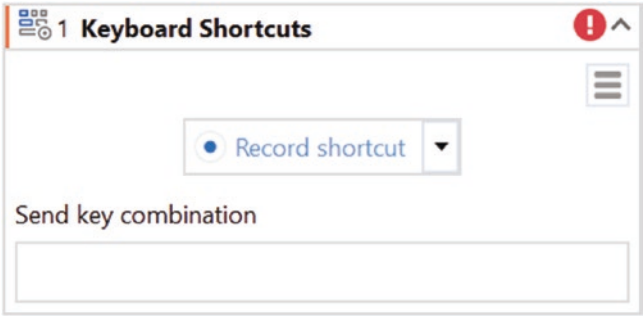


***Figure 4-72.*** *Activity card for Keyboard Shortcuts*

**Indicate target on screen (I)**: This is an optional configuration available from the menu of the activity. Your automation needs to be already interacting with a UI element to send keyboard shortcuts. You can either use an activity like `Click` or specify a UI element using this `Indicate target on screen (I)` link.

**Record shortcut**: This is a required configuration available on the activity card. This configuration allows you to specify the actual shortcuts that you want to send. Figure 4-73 shows the three options for specifying keyboard shortcuts.

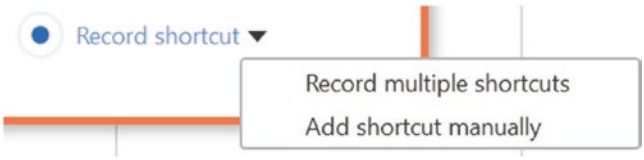


***Figure 4-73.*** *Available options for recording shortcuts*

First, you can use the `Record shortcut` option to record a single keyboard shortcut. As soon as you click this option, the recording starts, as shown in Figure 4-74.

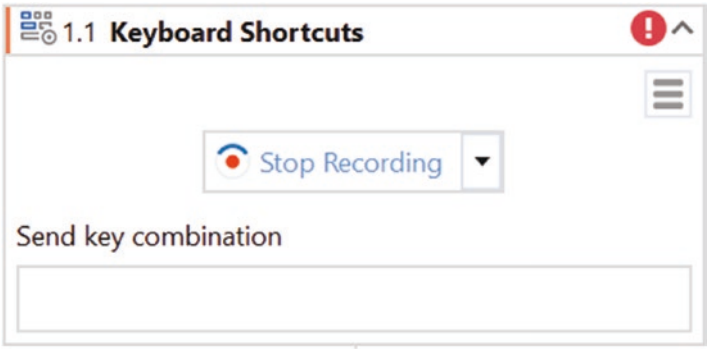


***Figure 4-74.*** *Recording in the progress indicator*

The next keystroke you type will be recorded as your shortcut. Figure 4-75 shows the result of the recording.

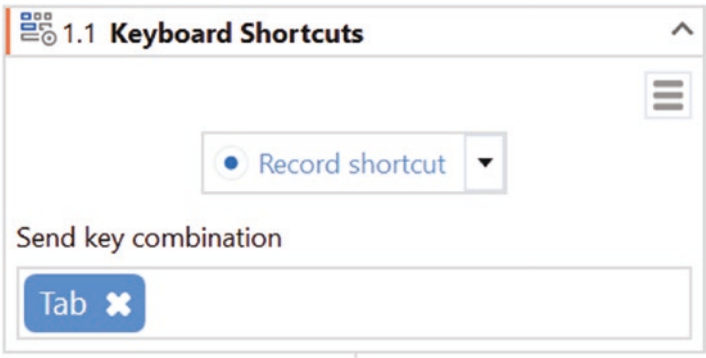


***Figure 4-75.*** *Results of single keyboard shortcut recording*

Second, you can use `Record multiple shortcuts` to record one or multiple keyboard shortcuts. The recording process is the same as the previous option. In this option, the recording does not stop after a single keyboard shortcut. Figure 4-76 shows how the activity looks after multiple shortcuts are captured.


***Figure 4-76.*** *Results of multiple keyboard shortcuts recording*

Finally, if you are having difficulty recording your shortcuts, you can use the `Add a shortcut manually` option to send one or multiple shortcuts to the specified UI element. Figure 4-77 shows all the options available to create shortcuts, while Figure 4-78 shows how the activity looks after a single shortcut has been added.

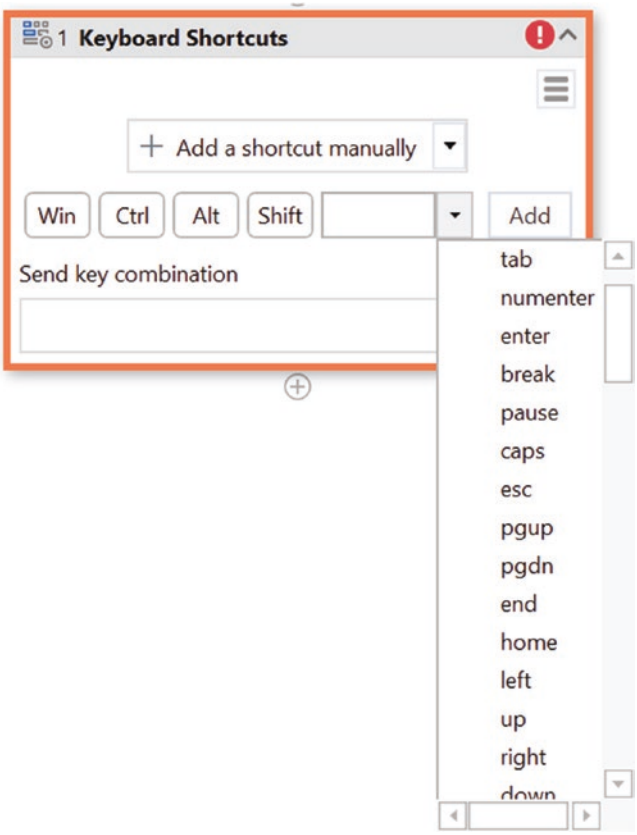


***Figure 4-77.*** *Available options for manually adding shortcuts*

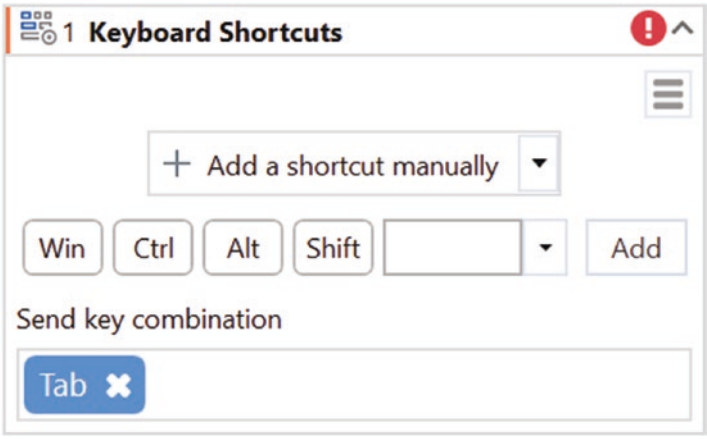


***Figure 4-78.*** *Results of manually adding keyboard shortcuts*

**Click before typing**: This is an optional configuration available on the `Properties` panel. This configuration allows you to perform a `Single` or `Double` click on the UI element before sending keyboard shortcuts. By default, this is set to `None`.

**Delay between shortcuts**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the delay (in seconds) between multiple keyboard shortcuts. By default, this is empty, that is, no delay is set.

## EXERCISE

**Goal**: Use the `Keyboard Shortcuts` activity to navigate between text fields of Contact Details dialog and enter data.

**Source Code**: Chapter_4_FormDataEntryKeyShortcutsExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter `https://therpabook.com/samples/contactsmanagement/` in the address bar.

2. In StudioX, add the Use Application/Browser activity to a blank process.
3. Next, click Indicate application in the Use Application/Browser activity card, and point your mouse to the browser. This will automatically populate the activity card.
4. Next, select the Use Application/Browser activity card, and from Properties, set the Options ➤ Open property to Always. This will ensure that the automation always opens a new browser.
5. Next, add a Click activity within the Use Application/Browser activity and configure it to click the Add Contact button (see Click activity exercise).
6. Before proceeding, make sure the Add Contact dialog is open.
7. When the dialog is opened, by default, none of the elements are active, so you cannot be sure to which element the automation will send keyboard shortcuts. To ensure that there is a known starting point, add another Click activity within the Use Application/Browser activity. Configure it to click the title of the dialog, that is, Contact Details.
8. Next, add a Keyboard Shortcuts activity right after the Click activity. This will make the automation move to the first field in the form. Use the Record shortcut feature to record a single Tab.
9. Next, add a Type Into activity. Click the Plus icon, select the Text option, and type Delos Inc. Click Save.
10. Next, add another Keyboard Shortcuts activity. This will make the automation move to the next field in the form. Use the Record shortcut feature to record a single Tab.

11. Next, add a `Type Into` activity. Click the `Plus` icon, select the `Text` option, and type `James Delos`. Click `Save`.

Once you have completed the exercise, the final configuration of the **Keyboard Shortcuts** activity should resemble Figure 4-79. Figure 4-80 shows the state of the target web application once the automation has completed its run. In this case, data in the Company and Contact fields have been entered.


***Figure 4-79.*** *Final configuration of Keyboard Shortcuts activity exercise*

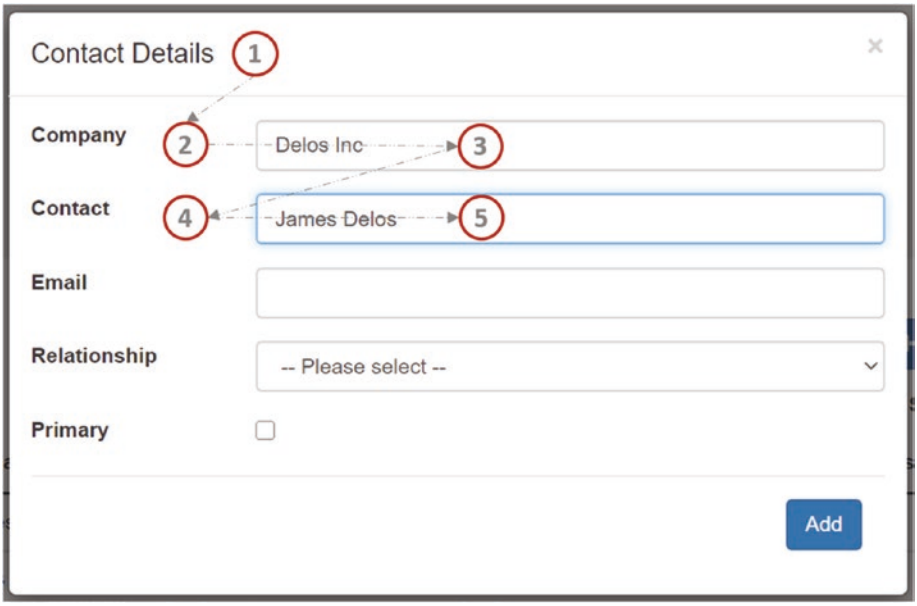


***Figure 4-80.*** *Result of the Keyboard Shortcuts activity exercise*

The next group of activities of this chapter are focused on window manipulation. The Get Active Window activity can be used to reference the current active window to enable you to perform multiple actions. For example, once you have the active window, you can choose to maximize, hide, restore, or move that window within your automation.

## Get Active Window

The **Get Active Window** activity allows you to get a reference to the window currently active on the machine where the automation is running. This activity can be used in instance you need to get the active window to later minimize or maximize it within your automation.

---

**Note** This activity does not need to be nested inside a Use Application/Browser activity.

---

## Configuration

This section provides instructions on how to configure a **Get Active Window** activity, shown in Figure 4-81.

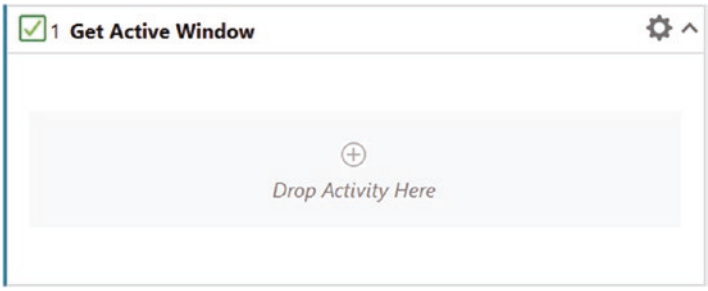


***Figure 4-81.*** *Activity card for Get Active Window*

**ApplicationWindow**: This is an optional configuration available on the `Properties` panel. This configuration allows you to save a reference to the active window for use later.

**EXERCISE**

**Goal**: Use the `Get Active Window` activity to get a reference of the Contacts Management application window.

**Source Code**: Chapter_4_WindowOperationsExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter `https://therpabook.com/samples/contactsmanagement/` in the address bar.
2. In `StudioX`, add the `Use Application/Browser` activity to the `Designer` panel.

3. Next, click `Indicate application` in the `Use Application/Browser` activity card, and point your mouse to the browser. This will automatically populate the activity card.
4. Next, select the `Use Application/Browser` activity card, and from `Properties`, set the `Options` ➤ `Open` property to `Always`. This will ensure that the automation always opens a new browser.
5. Next, add the `Get Active Window` activity after the `Use Application/Browser` activity.
6. Next, in the `Properties` panel of `Get Active Window` activity, click the `Plus` icon in the `ApplicationWindow` field. Click `Save for Later Use` and name the variable as `ActiveWindow`.

Once you have completed the exercise, the final configuration of the **Get Active Window** activity should resemble Figure 4-82.

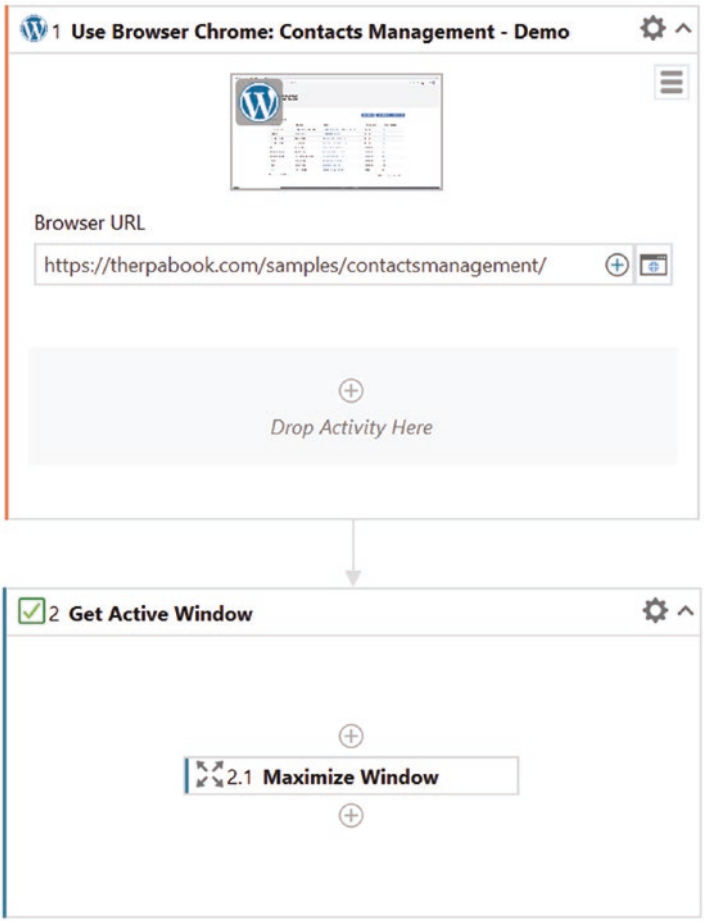


***Figure 4-82.*** *Final configuration of the Get Active Window exercise*

# Maximize Window

The **Maximize Window** activity allows you to maximize the specified window.

---

**Note** This activity is usually nested inside a Use Application/ Browser activity or a Get Active Window activity.

---

## Configuration

This section provides instructions on how to configure a **Maximize Window** activity, shown in Figure 4-83.


***Figure 4-83.*** *Activity card for Maximize Window*

**Window**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the window that you want to maximize. The reference to window can be obtained using `Get Active Window` activity.

**EXERCISE**

**Goal**: Use the `Maximize Window` activity to maximize the Contacts Management application window. This exercise builds upon the previous exercise for `Get Active Window` activity.

**Source Code**: Chapter_4_WindowOperationsExercise

**Setup**: Here are step-by-step implementation instructions:

1. In StudioX, add the Maximize Window activity inside the Get Active Window activity.
2. Next, in the Properties panel of Maximize Window activity, click the Plus icon in the Window field. Hover over Use Saved Value and select ActiveWindow.

Once you have completed the exercise, the final configuration of the **Maximize Window** activity should resemble Figure 4-84.

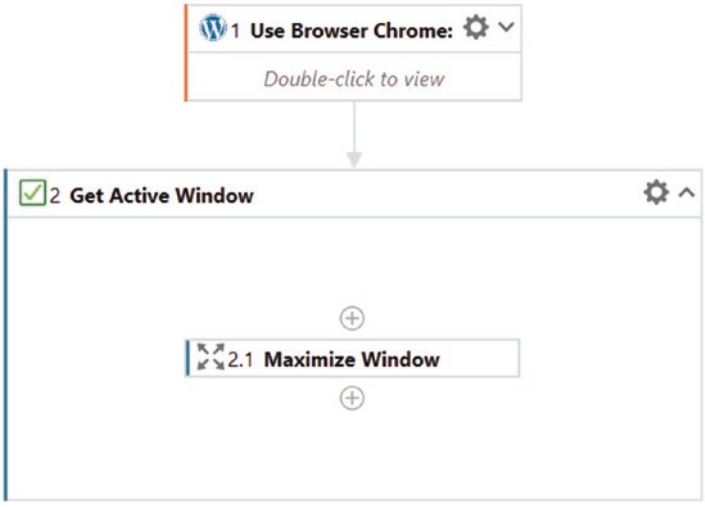


***Figure 4-84.*** *Final configuration of the Maximize Window activity exercise*

## Minimize Window

The **Minimize Window** activity allows you to minimize the specified window.

---

**Note** This activity does not need to be nested inside a Use Application/Browser activity or a Get Active Window activity.

---

## Configuration

This section provides instructions on how to configure a **Minimize Window** activity, shown in Figure 4-85.


***Figure 4-85.*** *Activity card for Minimize Window*

**Window**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the window that you want to minimize. The reference to window can be obtained using `Get Active Window` activity.

### EXERCISE

**Goal**: Use the `Minimize Window` activity to minimize the Contacts Management application window. This exercise builds upon the previous exercise for `Maximize Window` activity.

**Source Code**: Chapter_4_WindowOperationsExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Delay` activity inside the `Get Active Window` activity after the `Maximize Window` activity. Update the `Duration` field to `3` seconds. We are adding a slight delay just so that we can see the operations happening; otherwise, this is not needed.
2. Next, add the `Minimize Window` activity inside the `Get Active Window` activity after the `Delay` activity.

3. In the `Properties` panel of `Minimize Window` activity, click the `Plus` icon in the `Window` field. Hover over `Use Saved Value` and select `ActiveWindow`.

Once you have completed the exercise, the final configuration of the **Minimize Window** activity should resemble Figure 4-86.

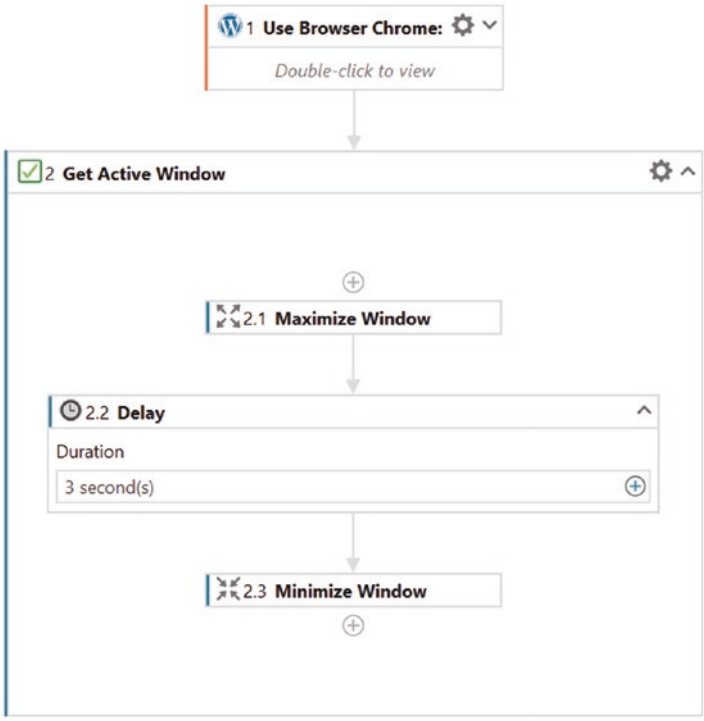


***Figure 4-86.*** *Final configuration of the Minimize Window activity exercise*

## Hide Window

The **Hide Window** activity allows you to hide the specified window.

**Note** This activity does not need to be nested inside a Use Application/Browser activity or a Get Active Window activity.

## Configuration

This section provides instructions on how to configure a **Hide Window** activity, shown in Figure 4-87.


***Figure 4-87.*** *Activity card for Hide Window*

**Window**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the window that you want to hide. The reference to window can be obtained using `Get Active Window` activity.

**EXERCISE**

**Goal**: Use the `Hide Window` activity to hide the Contacts Management application window. This exercise builds upon the previous exercise for `Minimize Window` activity.

**Source Code**: Chapter_4_WindowOperationsExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Delay` activity inside the `Get Active Window` activity after the `Minimize Window` activity. Update the `Duration` field to `3` seconds. We are adding a slight delay just so that we can see the operations happening; otherwise, this is not needed.

2. Next, add the `Hide Window` activity inside the `Get Active Window` activity after the `Delay` activity.
3. In the `Properties` panel of `Hide Window` activity, click the `Plus` icon in the `Window` field. Hover over `Use Saved Value` and select `ActiveWindow`.

Once you have completed the exercise, the final configuration of the **Hide Window** activity should resemble Figure 4-88.

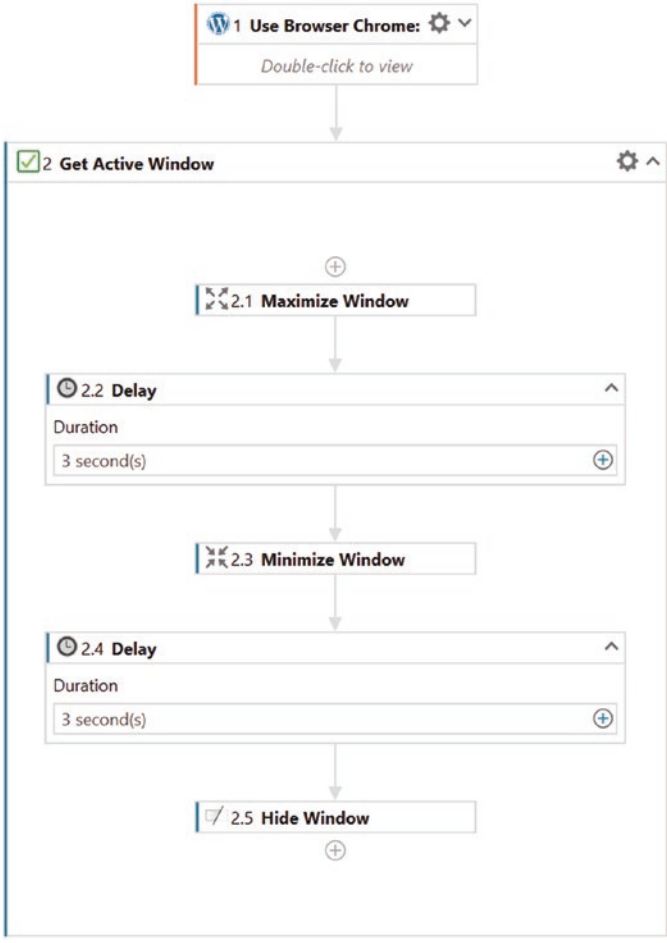


***Figure 4-88.*** *Final configuration of the Hide Window activity exercise*

# Restore Window

The **Restore Window** activity allows you to restore the specified window.

---

**Note** This activity does not need to be nested inside a Use Application/Browser activity or a Get Active Window activity.

---

## Configuration

This section provides instructions on how to configure a **Restore Window** activity, shown in Figure 4-89.


***Figure 4-89.*** *Activity card for Restore Window*

**Window**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the window that you want to restore. The reference to window can be obtained using `Get Active Window` activity.

## EXERCISE

**Goal**: Use the `Restore Window` activity to restore the Contacts Management application window. This exercise builds upon the previous exercise for `Hide Window` activity.

**Source Code**: Chapter_4_WindowOperationsExercise

**Setup**: Here are step-by-step implementation instructions:

1. In `StudioX`, add the `Delay` activity inside the `Get Active Window` activity after the `Minimize Window` activity. Update the `Duration` field to `3` seconds. We are adding a slight delay just so that we can see the operations happening; otherwise, this is not needed.
2. Next, add the `Restore Window` activity inside the `Get Active Window` activity after the `Delay` activity.
3. In the `Properties` panel of `Minimize Window` activity, click the `Plus` icon in the `Window` field. Hover over `Use Saved Value` and select `ActiveWindow`.

Once you have completed the exercise, the final configuration of the **Restore Window** activity should resemble Figure 4-90.

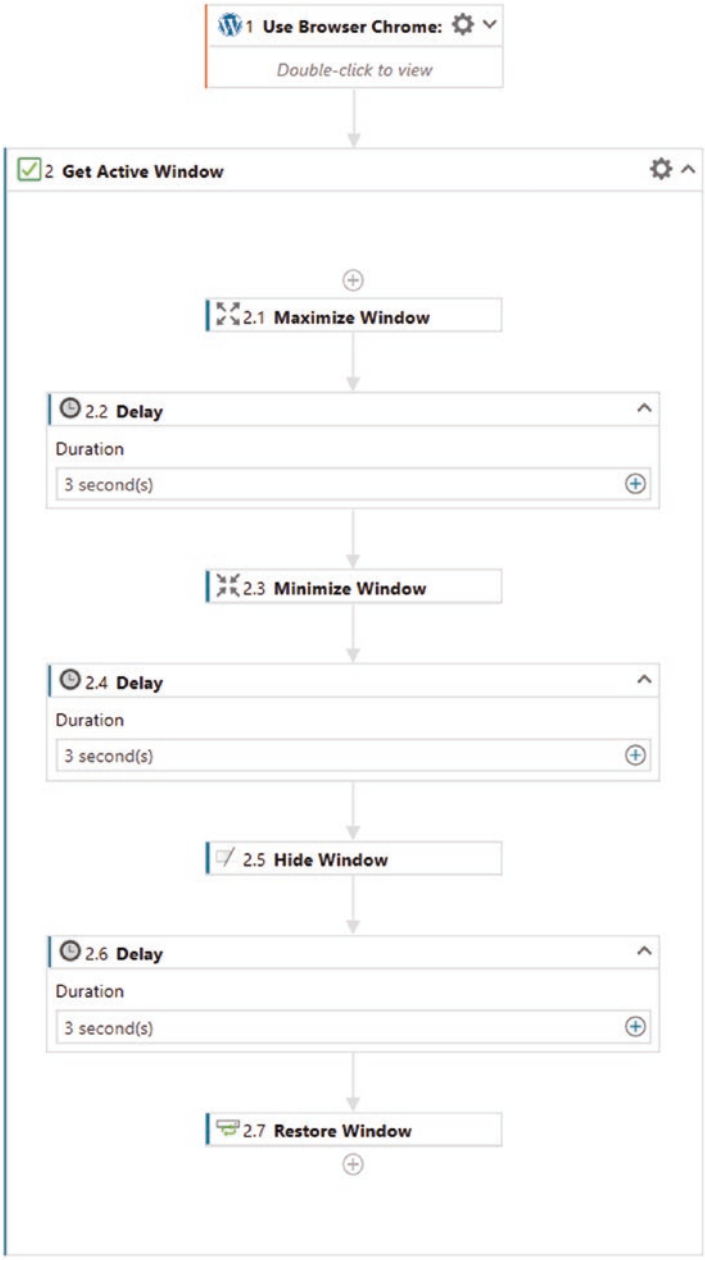


***Figure 4-90.*** *Final configuration of the Restore Window activity exercise*

# Move Window

The **Mode Window** activity allows you to change the size of the specified and move the specified window to a new position on the screen.

---

**Note** This activity does not need to be nested inside a Use Application/Browser activity or a Get Active Window activity.

---

## Configuration

This section provides instructions on how to configure a **Move Window** activity, shown in Figure 4-91.


***Figure 4-91.*** *Activity card for Move Window*

**Window**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the window that you want to minimize. The reference to window can be obtained using `Get Active Window` activity.

**Height**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the new height of the window.

**Width**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the new width of the window.

**X**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the new X position of the window.

**Y**: This is an optional configuration available on the `Properties` panel. This configuration allows you to specify the new Y position of the window.

## EXERCISE

**Goal**: Use the Move Window activity to reduce the screen size of the Contacts Management application to 500 x 500 and move it to X: 100 and Y: 100 location.

**Source Code**: Chapter_4_WindowOperationsExercise

**Setup**: Here are step-by-step implementation instructions:

1. In StudioX, add the Delay activity inside the Get Active Window activity after the Restore Window activity. Update the Duration field to 3 seconds. We are adding a slight delay just so that we can see the operations happening; otherwise, this is not needed.
2. Next, add the Move Window activity inside the Get Active Window activity after the Delay activity.
3. In the Properties panel of Move Window activity, click the Plus icon in the Window field. Hover over Use Saved Value and select ActiveWindow.
4. In the Height field, click the Plus icon, select the Number option, and type 500.
5. In the Width field, click the Plus icon, select the Number option, and type 500.
6. In the X field, click the Plus icon, select the Number option, and type 100.
7. In the Y field, click the Plus icon, select the Number option, and type 100.

Once you have completed the exercise, the final configuration of the **Move Window** activity should resemble Figure 4-92.

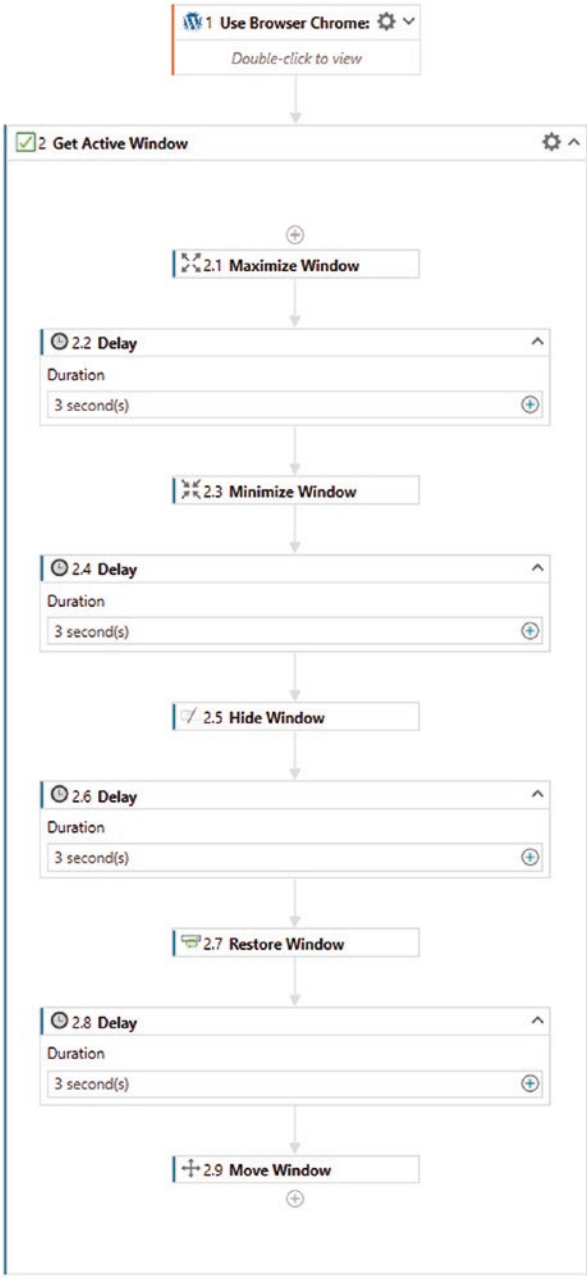


***Figure 4-92.*** *Final configuration of the Move Window activity exercise*

# App/Web Recorder

In this section, we are going to generate simple automation using the **App/Web Recorder** feature.

**EXERCISE**

**Goal**: Use the `App/Web Recorder` feature to generate automation for adding a new contact.

**Source Code**: Chapter_4_FormDataEntryRecorderExercise

**Setup**: Here are step-by-step implementation instructions:

1. Open a browser of your choice and enter `https://therpabook.com/samples/contactsmanagement/` in the address bar.
2. Next, click `App/Web Recorder` menu from the ribbon on top. This will launch the `App/Web Recorder` menu, shown in Figure 4-93.

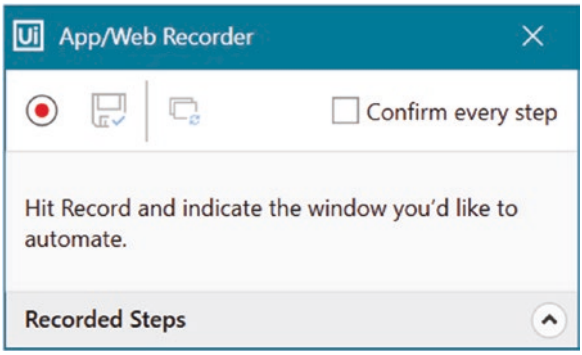


***Figure 4-93.*** *App/Web Recorder menu*