

ATME COLLEGE OF ENGINEERING

13th KM Stone, Bannur Road, Mysore - 560 028



A T M E

College of Engineering

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN



LECTURE NOTES

**COURSE: ROBOTIC PROCESS AUTOMATION DESIGN &
DEVELOPMENT (UiPath)**

COURSE CODE: BCG701

SEMESTER: VII

COURSE COORDINATOR: Mr. YOGESH N

(ACADEMIC YEAR 2026-27)

INSTITUTIONAL MISSION AND VISION

Objectives

- To provide quality education and groom top-notch professionals, entrepreneurs and leaders for different fields of engineering, technology and management.
- To open a Training-R & D-Design-Consultancy cell in each department, gradually introduce doctoral and postdoctoral programs, encourage basic & applied research in areas of social relevance, and develop the institute as a center of excellence.
- To develop academic, professional and financial alliances with the industry as well as the academia at national and transnational levels.
- To develop academic, professional and financial alliances with the industry as well as the academia at national and transnational levels.
- To cultivate strong community relationships and involve the students and the staff in local community service.
- To constantly enhance the value of the educational inputs with the participation of students, faculty, parents and industry.

Vision

Development of academically excellent, culturally vibrant, socially responsible and globally competent human resources.

Mission

- To keep pace with advancements in knowledge and make the students competitive and capable at the global level.
- To create an environment for the students to acquire the right physical, intellectual, emotional and moral foundations and shine as torch bearers of tomorrow's society.
- To strive to attain ever-higher benchmarks of educational excellence.

Department of Computer Science & Design

Vision of the Department

- To develop highly talented individuals in Computer Science and Engineering to deal with real world challenges in industry, education, research and society.

Mission of the Department

- To inculcate professional behaviour, strong ethical values, innovative research capabilities and leadership abilities in the young minds & to provide a teaching environment that emphasizes depth, originality and critical thinking.
- Motivate students to put their thoughts and ideas adoptable by industry or to pursue higher studies leading to research

PROGRAM EDUCATIONAL OBJECTIVES (PEO'S):

1. Empower students with a strong basis in the mathematical, scientific and engineering fundamentals to solve computational problems and to prepare them for employment, higher learning and R&D.
2. Gain technical knowledge, skills and awareness of current technologies of computer science engineering and to develop an ability to design and provide novel engineering solutions for software/hardware problems through entrepreneurial skills.
3. Exposure to emerging technologies and work in teams on interdisciplinary projects with effective communication skills and leadership qualities.
4. Ability to function ethically and responsibly in a rapidly changing environment by applying innovative ideas in the latest technology, to become effective professionals in Computer Science to bear a life-long career in related areas.

Program Specific Outcomes (PSOs)

PSO1: Ability to apply skills in the field of algorithms, database design, web design, cloud computing and data analytics.

PSO2: Apply knowledge in the field of computer networks for building network and internet based applications.

Module 2: UI Automation

This module provides a comprehensive introduction to **UI Automation** using UiPath, focusing on automating interactions with web and desktop applications. The key learning objective is to **understand and apply UI automation activities** to manipulate and control user interface elements within business process automation.

Sample Application: Contacts Management

The sample application used throughout this module is Contacts Management, accessible via a browser at <https://therpabook.com/samples/contactsmanagement/>.

- Home Page:
- Header: Displays the title of the application.
- Buttons:
- Add Contact (opens dialog to input new contact details; not operational, no persistence).
- View Contact (displays pre-set contact details; application is non-functional beyond viewing).
- Download contacts list in Excel and CSV formats.
- Contacts Table: Shows a multi-page list of contacts.

Detailed UI Automation Activities

Use Application/Browser Activity

Purpose: Serves as a container to open, close, and perform all UI interactions with the target application (web or desktop).

Configuration steps include:

- Indicate Application: Identifies the target app to automate; StudioX determines app type (web or desktop).
- Browser URL (web apps): Specifies the targeted web page URL, defaulting from the indicated browser page.
- Application Path (desktop apps): Full path to the executable file, auto-detected upon indication.
- Application Arguments (desktop apps – optional): Arguments to launch the application (e.g., opening a specific file in Notepad).
- Match Exact Title (optional for desktop apps): Defines if a strict application window title match is required.
- Open/Close Settings:
- Open options: Never, Always, IfNotOpen (to specify when to launch).
- Close options: Never, Always, IfOpenedByAppBrowser (to define closing behaviour post-automation).

Go To URL Activity

Purpose: Directly navigates a browser to a specified URL.

Configuration:

- URL field (required): Specifies destination web address for automation.
- Used within the Use Application/Browser scope to manage navigation.

Key Insights

- UiPath StudioX provides a structured framework to automate UI interactions efficiently.
- Configurations of activities must be precise to ensure correct targeting and execution, particularly distinguishing between desktop and web applications.
- The Use Application/Browser activity acts as a fundamental container, within which all UI actions occur, making it essential for successful automation.
- Use of indicate application feature aids automation by auto-determining application details for streamlined setup.
- Practical sample application enhances comprehension by providing hands-on experience, albeit with application functionality limited to demonstration only (non-persistent data).
- Browser extensions for automation are mandatory prerequisites before automating web applications.

Navigate Browser

Purpose: Enables automation scripts to simulate browser navigation commands.

Supported Actions:

- Go Back
- Go Forward
- Go Home
- Refresh
- Close Tab

Configuration:

- Must specify the **Action** to perform on the browser.
- Default action is set to **Go Back**.
- This activity is fundamental for workflows requiring browser interactions.

Highlight

Purpose: Creates a visible coloured box around UI elements to draw attention during automation execution.

Use Case: Useful for debugging, demonstrations, and emphasising interface components during automated runs.

Configurations:

- **Indicate Target on Screen:** Specifies the UI element to be highlighted (Required).
- **Duration:** Time (in seconds) for which the highlight is visible (Optional).
- **Colour:** Colour of the highlight box, with a default of **gold** (Optional).
- Exercises reinforce the configuration and practical application of the highlight activity.

Take Screenshot

Purpose: Captures an image snapshot of a specified UI region or element.

Application: Essential for audit trails, evidencing specific UI states, or documenting automated task outputs.

Configuration:

- **Indicate Target on Screen:** Select the UI element for screenshot (Required).
- **Output To:** Defines if the screenshot is saved to a file or clipboard (Required).
- **File Name and Location:** When saving as a file, specify the folder and filename (Required if Output to File).
- **Auto Increment:** Optionally appends timestamps or sequential indices to prevent file overwriting (Optional).
- Exercising this activity demonstrates saving multiple screenshots with unique filenames over repeated automation runs.

Exercises and Practical Application

- Multiple exercises were provided to apply the concepts of UI Automation activities.
- Students configured and executed:
 - Navigate Browser actions
 - Highlighting specific UI areas
 - Taking and saving screenshots in organised folders
- Final exercise outputs exhibit:
 - Effective UI element highlighting during process execution.
 - Proper screenshot naming convention utilising timestamps.
 - Demonstrated understanding of configuring activity properties.

Key Insights

- **UI Automation activities** are critical for simulating user interactions within both web and desktop applications in UiPath.
- Understanding the **configuration parameters** such as target element selection, action type, duration, and output settings is essential for successful automation.
- The **Highlight activity** not only aids in debugging but also improves process transparency during execution.
- The **Take Screenshot activity** serves as a reliable method to create automated documentation and audit evidence.
- Practical exercises reinforce understanding by allowing students to apply configurations and validate their outputs in real scenarios.
- Proper naming conventions and output handling, such as **auto incremented filenames**, are vital to avoid data overwrite and maintain record integrity in automated processes.

Check App State

This activity is used to **detect the presence or absence of specific UI elements** (e.g., loading icons, pop-up messages) to determine the current state of an application or screen.

Purpose: Ensures that the automation waits for the screen or element to fully load before proceeding.

Use cases:

- Waiting for spinners or loading icons to disappear.
- Handling message boxes, error pop-ups, or other conditional UI elements during automation.

Configuration settings:

Example use case: Waiting up to 15 seconds for a spinner to disappear before displaying a message box indicating the status.

Click Activity

This activity simulates a **mouse click on a specified UI element**, emulating human interaction for selecting buttons, links, or other clickable elements.

Configuration settings:

- **Click types** include: single click, double click, right click, and others (specific options not detailed in the source).
- **Mouse buttons** allow selecting left, right, or middle as appropriate.

- **Key modifiers** enable combining mouse clicks with keyboard keys to extend functionality.

Type Into Activity

This activity inputs text into a specified UI element, such as text boxes or editable dropdowns.

Utility: It is particularly useful for dropdown menus that do not support the native Select Item activity in UiPath.

Configuration settings:

- Clearing a field is done using keyboard shortcuts internally before typing.
- Clicking before typing helps position the cursor and ensure the field is active for input.

A range of **UI Automation activities** are critical for automating interactions with desktop or web applications, including:

- Use Application/Browser
- Go To URL, Navigate Browser
- Highlight, Take Screenshot
- Check App State
- Click, Type Into, Select Item
- Check/Uncheck
- Get Text, Get Attribute
- Extract Table Data
- Hover
- Keyboard Shortcuts
- Window management: Get Active Window, Maximize, Minimize, Hide, Restore, Move Window
- App/Web Recorder

The primary learning objective is to understand and configure these activities within UiPath workflows to automate UI tasks effectively.

Configuration Guidelines and Best Practices

- **Indicate Target on Screen:** All activities require the user to specify the UI element they want to interact with by visually indicating the target during configuration. This anchors the activity in the automation workflow.

Optional vs. Mandatory Fields:

- For Select Item, item selection is optional but recommended for precision once the dropdown is identified.
- For Check/Uncheck, action selection defaults to "Check" but can be tailored.
- For Get Attribute, storing the result is optional but highly useful for downstream automation.
- Handling Unsupported Scenarios: Some UI controls, such as dropdowns that are not standard combo boxes, may not support dedicated activities like Select Item. The suggested workaround is to use "Type Into" activities.

Exercises and Practical Engagement

- The module incorporates a series of sequential exercises for each activity, reinforcing concepts through practical implementation:
- Exercises for Select Item activity to practice dropdown selection.
- Exercises for Check/Uncheck to manage checkbox states dynamically.
- Multiple exercises on Get Text to extract and store UI text.
- Exercises on Get Attribute to retrieve and handle attribute values from UI elements.
- This hands-on approach ensures learners internalise activity configurations and operational contexts, supporting skill development in UiPath RPA design.

Key Insights

- **UI Automation is foundational for effective RPA** involving desktop and web applications, enabling bots to reliably interact with visual elements.
- Configuring activities such as **Select Item, Check/Uncheck, Get Text, and Get Attribute** requires understanding UI elements' types and states.
- The modular, exercise-based methodology facilitates a deep practical understanding, equipping learners with applicable UiPath skills.
- Attention to detail in target indication and choice of activities based on application control types is crucial for reliable automation.
- The course is structured progressively, with future sessions planned to cover more complex UI tasks, beneficial for comprehensive RPA proficiency.

Uncertain / Not Specified

- Specific software versions of UiPath used in this module are *not specified*.
- Detailed error handling and exception scenarios during UI automation were *not covered* in the provided content.
- Exact duration and total sessions of the course are *not indicated*.
- The content does not provide quantitative performance metrics or efficiency data for the activities discussed.

The module introduces a variety of UI automation activities commonly used within UiPath:

- Use Application/Browser
- Go To URL
- Navigate Browser
- Highlight
- Take Screenshot
- Check Application State
- Click
- Type Into
- Select Item
- Check/Uncheck
- Get Text
- Get Attribute
- Extract Table Data
- Hover
- Keyboard Shortcuts
- Get Active Window
- Window controls such as Maximise, Minimise, Hide, Restore, and Move Window
- App/Web Recorder

Among these, **Extract Table Data** and **Hover** are examined in detail.

Extract Table Data Activity

The **Extract Table Data** activity is designed to capture structured data in a tabular format from both web and desktop applications.

Key Features:

- Configured within a **Use Application/Browser** container.
- Can handle data spanning across multiple pages.
- Supports storing extracted data in various formats.
- Allows appending new data to previously extracted results or overwriting them.

Configuration Details:

*Conditional indicates the configuration becomes available only after specific prior configuration steps, such as target selection.

Use Cases and Practical Learning:

- Multiple exercises guide through setting up, configuring, testing, and optimising the Extract Table Data activity.
- Emphasis on navigating multi-page tables and handling delays in data loading.

- Ensures students gain proficiency in managing extraction workflows involving pagination and data integrity.
-

Hover Activity

The **Hover** activity simulates the mouse pointer hovering over specified UI elements.

Purpose:

- Frequently used in conjunction with the Click activity.
- Important for interacting with hover menus in modern web applications where options are only visible upon mouseover.

Configuration Summary:

Practical Learning:

- Multiple exercises reinforce proper identification of target UI elements.
- Demonstrates how hovering triggers UI changes required for further automation actions such as menu expansion.
- Users learn to balance hover duration for effective synchronisation without causing unnecessary delays.

Key Insights

- The **Extract Table Data** activity is essential for automating data extraction from structured tables on UI, accommodating complex scenarios such as multi-page navigation and delayed content loading.
- Configuration options allow flexibility in how data is extracted, stored, and managed, ensuring robustness in automation workflows.
- The **Hover** activity, though simple, is critical for interacting with UI elements that respond to mouse hover events and is often paired with clicking actions to access hidden menus.
- Proper configuration of activities using screen target indications is necessary to establish accurate automation sequences.
- Exercises are integral for mastering these activities, reinforcing configuration steps, troubleshooting, and effective automation design.
- The course implements a logical progression, building skills session by session, with upcoming lessons targeting keyboard actions and window handling to enhance UI automation capabilities further.

Keyboard Shortcuts Activity

Purpose:

This activity enables sending one or more keyboard shortcuts to a specified UI element. It is particularly useful when:

- Automating legacy applications where clicking UI elements is difficult
- Alternative navigation is required when standard clicks are ineffective

Important Note: Keyboard Shortcuts should not be used as the default navigation method.

Configuration Options:

Additional Properties:

- **Click before typing:**
Defines whether a Single or Double click occurs on the UI element before sending the shortcut(s). Default is **None**.
- **Delay between shortcuts:**
Specifies delay in seconds between sending multiple shortcuts. If left blank, no delay is applied.

Workflow for Configuring Keyboard Shortcuts:

- Start by indicating or recording the target UI element.
- Select one of the methods for shortcut input (record single, record multiple, or manual entry).
- Optionally configure click actions and delay properties.

Practical Exercises

Several exercises are mentioned which reinforce configuring and using Keyboard Shortcuts effectively, though specific details of exercises are not specified.

Get Active Window Activity

Purpose: The Get Active Window activity captures a reference to the currently active window on the machine. This is useful when subsequent actions need to manipulate the window such as minimising or maximising it during automation.

Technical Note:

- Does not require nesting inside Use Application/Browser activity.
- Can be used independently for referencing the active window.

Configuration:

Use Case Highlights:

- Obtain window for dynamic automation scenarios
- Control window states (maximise/minimise) in future automation steps

Key Insights

- **Keyboard Shortcuts activity** is a powerful but specialised tool to interact with UI where conventional clicks fail. Proper configuration, including specifying target elements and configuring delays, is critical for reliable automation.
- **Get Active Window activity** offers dynamic referencing to the UI elements by capturing the active window at runtime, enabling flexible control over window-based actions.
- Both activities emphasise precision in configuration and the importance of understanding UI context for robust automation designs.
- Practical exercises are integrated to solidify the theoretical knowledge through hands-on experience.
- The course structure is progressive, ensuring foundational skills before advancing to window manipulation commands.

The following topics emphasises understanding and utilizing window management activities crucial for interactive RPA workflows, particularly for automating desktop and web applications.

Maximize Window

Purpose: To maximise a specified window.

Usage: Typically nested inside Use Application/Browser or Get Active Window activities.

Configuration:

- Window property is optional.
- The window reference can be captured using Get Active Window.
- Practical exercise included to reinforce configuration.

Minimize Window

Purpose: To minimise the specified window.

Usage: Does not require nesting in Use Application/Browser or Get Active Window.

Configuration:

- Similar optional Window property available.
- Window reference can be obtained with Get Active Window.
- Practical exercise assigned.

Hide Window

Purpose: Allows hiding of the specified window.

Usage: Independent and doesn't require nesting.

Configuration:

- The Window property optionally specifies which window to hide.
- Window object acquired using Get Active Window.
- Included exercise for hands-on practice.

Restore Window

Purpose: Restores a window that has been minimised, hidden, or altered.

Usage: Standalone activity, no nesting needed.

Configuration mirrors previous activities, with focus on a specific Window.

Move Window

Purpose: Resize and reposition a window on the screen.

Usage: Does not need to be nested within other activities.

Configuration properties include:

- Window: Optional reference to the window.
- Height and Width: Set new dimensions.
- X and Y: Specify new screen coordinates.
- Exercise encourages application of these configurations.

App/Web Recorder

Purpose: Automates the creation of simple UI automation sequences by recording user actions on applications or web browsers.

Utilisation: Facilitates rapid automation development by capturing UI interactions visually.

An exercise reinforces learning via practical implementation.

Key Insights

- **Window Management is Fundamental:** Efficient control over application windows forms the backbone of reliable UI automation workflows.
- **Modular Activities with Optional Parameters:** Activities like Maximize Window often require being within higher-level scopes, whereas others can be applied independently offering modular design flexibility.
- **References are Central:** The use of Get Active Window to dynamically obtain window references mediates efficient and dynamic automation.
- **App/Web Recorder Boosts Productivity:** Automation recording eases the development process for simple workflows, reducing manual configuration overhead.
- **Hands-on Exercises Essential:** Practical application cements theoretical knowledge especially in UI automation where visual and interactive elements require trial-based learning.