NetIQ® AppManager® ResponseTime for Microsoft Windows

Management Guide

December 2018

Legal Notice

Contents

About this Book and the Library

The NetIQ AppManager product (AppManager) is a comprehensive solution for managing, diagnosing, and analyzing performance, availability, and health for a broad spectrum of operating environments, applications, services, and server hardware.

AppManager provides system administrators with a central, easy-to-use console to view critical server and application resources across the enterprise. With AppManager, administrative staff can monitor computer and application resources, check for potential problems, initiate responsive actions, automate routine tasks, and gather performance data for real-time and historical reporting and analysis.

Intended Audience

This guide provides information for individuals responsible for installing an AppManager module and monitoring specific applications with AppManager.

Other Information in the Library

The library provides the following information resources:

Installation Guide for AppManager

Provides complete information about AppManager pre-installation requirements and step-by-step installation procedures for all AppManager components.

User Guide for AppManager Control Center

Provides complete information about managing groups of computers, including running jobs, responding to events, creating reports, and working with Control Center. A separate guide is available for the AppManager Operator Console.

Administrator Guide for AppManager

Provides information about maintaining an AppManager management site, managing security, using scripts to handle AppManager tasks, and leveraging advanced configuration options.

Upgrade and Migration Guide for AppManager

Provides complete information about how to upgrade from a previous version of AppManager.

Management guides

Provide information about installing and monitoring specific applications with AppManager.

Help

Provides context-sensitive information and step-by-step guidance for common tasks, as well as definitions for each field on each window.

The AppManager library is available in Adobe Acrobat (PDF) format from the AppManager Documentation page of the NetIQ Web site.

About NetIQ Corporation

We are a global, enterprise software company, with a focus on the three persistent challenges in your environment: Change, complexity and risk—and how we can help you control them.

Our Viewpoint

Adapting to change and managing complexity and risk are nothing new

In fact, of all the challenges you face, these are perhaps the most prominent variables that deny you the control you need to securely measure, monitor, and manage your physical, virtual, and cloud computing environments.

Enabling critical business services, better and faster

We believe that providing as much control as possible to IT organizations is the only way to enable timelier and cost effective delivery of services. Persistent pressures like change and complexity will only continue to increase as organizations continue to change and the technologies needed to manage them become inherently more complex.

Our Philosophy

Selling intelligent solutions, not just software

In order to provide reliable control, we first make sure we understand the real-world scenarios in which IT organizations like yours operate — day in and day out. That's the only way we can develop practical, intelligent IT solutions that successfully yield proven, measurable results. And that's so much more rewarding than simply selling software.

Driving your success is our passion

We place your success at the heart of how we do business. From product inception to deployment, we understand that you need IT solutions that work well and integrate seamlessly with your existing investments; you need ongoing support and training post-deployment; and you need someone that is truly easy to work with — for a change. Ultimately, when you succeed, we all succeed.

Our Solutions

- Identity & Access Governance
- Access Management
- Security Management
- Systems & Application Management
- Workload Management
- Service Management

Contacting Sales Support

For questions about products, pricing, and capabilities, contact your local partner. If you cannot contact your partner, contact our Sales Support team.

Worldwide:	www.netiq.com/about_netiq/officelocations.asp
United States and Canada:	1-888-323-6768
Email:	info@netiq.com
Web Site:	www.netiq.com

Contacting Technical Support

For specific product issues, contact our Technical Support team.

Worldwide:	www.netiq.com/support/contactinfo.asp
North and South America:	1-713-418-5555
Europe, Middle East, and Africa:	+353 (0) 91-782 677
Email:	support@netiq.com
Web Site:	www.netiq.com/support

Contacting Documentation Support

Our goal is to provide documentation that meets your needs. If you have suggestions for improvements, click **Add Comment** at the bottom of any page in the HTML versions of the documentation posted at www.netiq.com/documentation. You can also email Documentation-Feedback@netiq.com. We value your input and look forward to hearing from you.

Contacting the Online User Community

Qmunity, the NetIQ online community, is a collaborative network connecting you to your peers and NetIQ experts. By providing more immediate information, useful links to helpful resources, and access to NetIQ experts, Qmunity helps ensure you are mastering the knowledge you need to realize the full potential of IT investments upon which you rely. For more information, visit http://community.netiq.com.

1 Understanding Windows-RT

AppManager (Windows-RT) helps you monitor your organization's Windows infrastructure to ensure the availability and performance of critical applications and services on a Microsoft Windows computer. Using the Windows-RT Designer, you can create Knowledge Scripts that can identify problems a user might encounter when accessing an application.

This chapter provides a brief introduction to the AppManager ResponseTime modules and a more detailed look at Windows-RT.

1.1 Why Do I Need to Measure Response Time?

With the AppManager ResponseTime modules, you can measure or test the response time and the availability of key applications and services. You can use this information for managing and reporting on the performance of a particular Windows computer. You can also use this information to identify the most effective configuration settings for a computer and the applications and services running on it.

The results you get from response-time testing with ResponseTime Knowledge Scripts are extremely accurate because AppManager times a transaction that acts just like a real transaction from the monitored computer. You can test your system the way end users use it every day and see the same results and performance that end users see.

1.2 How AppManager ResponseTime Modules Work

The AppManager ResponseTime modules measure the response time and availability of a client/server transaction from the client perspective. Therefore, you typically install ResponseTime modules on client computers and not on application servers.

The following modules make up the ResponseTime family:

Module Name	Knowledge Script Category	What Is Monitored
AppManager ResponseTime for Microsoft Active Directory	AD-RT	Microsoft Active Directory and DNS transactions
AppManager ResponseTime for Microsoft Exchange	Exchange-RT	Microsoft Outlook transactions
AppManager ResponseTime for Networks	Networks-RT	Simulated transactions for many popular applications to measure network performance
AppManager ResponseTime for Oracle Database	Oracle-RT	ODBC and ADO transactions to Oracle Servers
AppManager ResponseTime for Microsoft SQL Server	SQL-RT	ODBC and ADO transactions to Microsoft SQL Server

Module Name	Knowledge Script Category	What Is Monitored
AppManager ResponseTime for Web	Web-RT	Web, Internet Mail, and News (NNTP) transactions This module allows you to record a Web-browsing session and “play back” synthetic transactions to measure response time.
AppManager ResponseTime for Windows	Win-RT	Microsoft Windows transactions This module allows you to record and “play back” synthetic transactions from any 32-bit Windows or Citrix client.

1.3 How Windows-RT Works

The AppManager ResponseTime modules use **synthetic network transactions** to measure application and server response time and availability.

Whenever you run a job using one of the ResponseTime Knowledge Scripts, an agent performs a transaction on the client computer, known as the managed client, to test the application or server. Transactions performed for response-time testing are automated or synthetic in the sense that no actual user is involved.

The ResponseTime for Windows module, also called Windows-RT, measures the response time of virtually any Windows application, including Citrix terminal sessions, by using scripts to record and play back client operations.

The Windows-RT module includes the following elements:

- A **managed object** component, `qWinRT7.dll`, installed in the `NetIQ\AppManager\bin`. The managed object handles communication between the running scripts and the Windows-RT service.

  NOTE: This component requires the netiqmc agent process to run as Local System, which allows the agent to start engine processes as different users.

- The **ResponseTime for Windows Designer** (Windows-RT Designer) extension (`Designer.exe`) lets you create and customize Knowledge Scripts for AppManager that can collect data to help you track the performance of Windows applications and services over time, such as during peak usage periods. The Knowledge Scripts you create with the Windows-RT Designer are based on *transactions* you record while using a Windows application on a managed client. During *playback*, the Knowledge Scripts you recorded run on a client computer to measure *response time* and *availability*. The Windows-RT Designer helps you view a Windows application from the perspective of a user.
- The **Windows-RT Playe**r, a service that extracts the script and associated parameter settings from the local repository and executes the script. The Windows-RT Player also writes any data streams or event information to the Windows-RT local repository, and marks the job iteration as complete. The Player does not run unless a user is logged on.
- The **Windows-RT service**, a service that handles the communication between the Windows-RT module and AppManager. The Windows-RT service also handles the login to the managed client, using the data specified in Security Manager for AppManager. This service runs continually, and it must run as “local system” for desktop access. The display name of this service is NetIQ ResponseTime 7 for Windows Service, and the service name is NETIQWinRT7.

NOTE: When you install the module, the setup program also installs the Windows-RT Designer, the Player, and the service.

1.4 Counting AppManager Licenses

AppManager for Windows-RT licenses are counted based on the number of installed Windows-RT Player or service components, or per discovered Windows-RT managed objects.

Users of the previous version of AppManager for Windows-RT can use their existing licenses to upgrade to this version of the module.

2 Installing AppManager ResponseTime for Windows

This chapter describes system requirements and provides installation instructions for AppManager ResponseTime for Windows (Windows-RT).

This chapter assumes you have AppManager installed. For more information about installing AppManager or about AppManager system requirements, see the *Installation Guide for AppManager*, which is available on the AppManager Documentation page.

2.1 System Requirements

AppManager ResponseTime for Windows has the following system requirements:

Software/Hardware	Version
AppManager installed on the AppManager repository (QDB) computers, on the computers you want to monitor (agents), and on all console computers	8.0.3, 8.2, 9.1, 9.2, 9.5, or later One of the following AppManager agents are required: ◆ AppManager agent 7.0.4 with hotfix 72616 or later ◆ AppManager agent 8.0.3, 8.2, 9.1, 9.2, 9.5, or later
Microsoft Windows operating system on the agent computers	One of the following: ◆ Windows 10 (32-bit or 64-bit) ◆ Windows Server 2012 R2 ◆ Windows Server 2012 ◆ Windows 8 (64-bit) ◆ Windows Server 2008 R2 ◆ Windows Server 2008 (32-bit or 64-bit) ◆ Windows 7 (32-bit or 64-bit) ◆ Windows Vista (32-bit or 64-bit) **NOTE:** Support for Windows Server 2008 on AppManager 7.x requires AppManager Windows Agent hotfix 71704 or later. For more information, see the AppManager Suite Hotfixes page.
Microsoft .NET Framework on the agent computer	4.0 or later for Windows 7, Windows Vista, Windows Server 2008, and Windows Server 2008 R2. **NOTE:** For Windows 8, Windows 2012, Windows 2012 R2, and Windows 10, the .Net Framework 4.0 or later is in-built feature.

Software/Hardware	Version
Visual Studio 2010 C++ runtime support file	2010 **NOTE:** ◆ 64-bit Windows operating systems require both 32-bit and 64-bit VC++ runtime support file. ◆ VC++ runtime support files install automatically during the module installation.
Microsoft SQL Server Native Client 11.0 **(for TLS 1.2 support)**	11.3.6538.0 or later **NOTE:** The SQL Server Native client can be installed from this Microsoft download link.

NOTE: If you want TLS 1.2 support and are running AppManager 9.1 or 9.2, then you are required to perform some additional steps. To know about the steps, see the article.

2.2 Installing the Module

Run the module installer only once on any computer. The module installer automatically identifies and updates all relevant AppManager components on a computer.

All Windows-RT software elements are automatically installed as part of the installation process. These elements include the Windows-RT Designer, Player service, and managed object.

Access the `AM70-Win-RT-7.x.x.0.msi` module installer from the `AM70_Win-RT_7.x.x.0` self-extracting installation package on the AppManager Module Upgrades & Trials page.

If you are upgrading from the previous version of this module, version 7.5, you need to perform additional steps to install version 7.6 correctly. For more information, see “Upgrading from a previous version of the Module” on page 18.

You need Local Administrator Rights on the computer where you install Windows-RT, unless User Account Control (UAC) is enabled.

For Windows environments where User Account Control (UAC) is enabled, install the module using an account with administrative privileges. Use one of the following methods:

- Log in to the server using the account named Administrator. Then, run the module installer `.msi` file from a command prompt or by double-clicking it.
- Log in to the server as a user with administrative privileges and run the module installer `.msi` file as an administrator from a command prompt. To open a command-prompt window at the administrative level, right-click a command-prompt icon or a **Windows** menu item and select **Run as administrator**.

You can install the Knowledge Scripts into local or remote AppManager repositories (QDBs). Install these components only once per QDB.

The module installer now installs Knowledge Scripts for each module directly into the QDB instead of installing the scripts in the `\AppManager\qdb\kp` folder as in previous releases of AppManager.

IMPORTANT: After installing the module or upgrading from a previous version of this module, you must restart the agent computer so that the Win-RT auto logon feature works properly.

Manually Installing the Module

You can install the module manually, or you can use Control Center to deploy the module on a remote computer where an agent is installed. For more information, see Section 2.3, “Deploying the Module with Control Center,” on page 18. However, if you use Control Center to deploy the module, Control Center only installs the *agent* components of the module. The module installer installs the QDB and console components as well as the agent components on the agent computer.

To install the module manually:

1. Double-click the module installer `.msi` file.
2. Accept the license agreement.
3. Review the results of the pre-installation check. You can expect one of the following three scenarios:
   - **No AppManager agent is present:** In this scenario, the pre-installation check fails, and the installer does not install agent components.
   - **An AppManager agent is present, but some other prerequisite fails:** In this scenario, the default is to not install agent components because of one or more missing prerequisites. However, you can override the default by selecting Install agent component locally. A missing application server for this particular module often causes this scenario. For example, installing the AppManager for Microsoft SharePoint module requires the presence of a Microsoft SharePoint server on the selected computer.
   - **All prerequisites are met:** In this scenario, the installer installs the agent components.
4. To install the Knowledge Scripts into the QDB:
   - **4a** Select **Install Knowledge Scripts** to install the repository components, including the Knowledge Scripts, object types, and SQL stored procedures.
   - **4b** Specify the SQL Server name of the server hosting the QDB, as well as the case-sensitive QDB name.
5. (Conditional) If you use Control Center 7.x, run the module installer for each QDB attached to Control Center.
6. (Conditional) If you use Control Center 8.x or later, run the module installer only for the primary QDB. Control Center automatically replicates this module to secondary QDBs.
7. Run the module installer on all console computers to install the Help and console extensions.
8. Run the module installer on the Windows-RT computers you want to monitor (agents) to install the agent components.
9. Configure permissions information in AppManager Security Manager. For more information, see Section 2.5, “Configuring Windows-RT Credentials in Security Manager,” on page 20.
10. (Conditional) If you have not discovered Windows-RT resources, run the Discovery_Win-RT7 Knowledge Script on all agent computers where you installed the module. For more information, see Section 2.6, “Discovering Windows-RT Resources,” on page 20.
11. To get the updates provided in this release, upgrade any running Knowledge Script jobs. For more information, see Section 3.7, “Upgrading Existing Knowledge Scripts,” on page 26.

After the installation has completed, the `Win-RT7_Install_MoFw.log` and the `Win-RT7_Install.log`, located in the `\NetIQ\Temp\NetIQ_Debug\`*`ServerName`* folder, list any problems that occurred.

Upgrading from a previous version of the Module

If you are upgrading from a previous release to this version, 7.7, install version 7.7 on each Windows-RT agent, AppManager repository (QDB), and console

To upgrade from a previous version to the version 7.7:

1. Stop the previous version ad hoc jobs for this module and remove all Win-RT monitoring jobs.
2. Install version 7.7 of the module on all AppManager repositories (QDBs), consoles, and agents. For more information about running the installer, see Chapter 2, “Installing AppManager ResponseTime for Windows,” on page 15.
3. The module installer automatically runs the Discovery Knowledge Script. If it does not, manually run Discovery_WinRT. For more information about the Discovery Knowledge Script, see Section 2.6, “Discovering Windows-RT Resources,” on page 20.
4. Recreate the ad hoc jobs and create new Win-RT monitoring jobs.

2.3 Deploying the Module with Control Center

You can use Control Center to deploy the module on a remote computer where an agent is installed. This topic briefly describes the steps involved in deploying a module and provides instructions for checking in the module installation package. For more information, see the *Control Center User Guide for AppManager*, which is available on the AppManager Documentation page.

NOTE: When you do the remote deployment, stop any Win-RT Player sessions that are running for the installation to succeed. If any Player session is running, the installer does not update the `Player.exe` properly.

Deployment Overview

This section describes the tasks required to deploy the module on an agent computer.

To deploy the module on an agent computer:

1. Verify the default deployment credentials.
2. Check in an installation package. For more information, see “Checking In the Installation Package” on page 18.
3. Configure an e-mail address to receive notification of a deployment.
4. Create a deployment rule or modify an out-of-the-box deployment rule.
5. Approve the deployment task.
6. View the results.

Checking In the Installation Package

You must check in the installation package, `AM70-Win-RT-x.x.x.0.xml`, before you can deploy the module on an agent computer.

To check in a module installation package:

1. Log on to Control Center using an account that is a member of a user group with deployment permissions.

2 Navigate to the **Deployment** tab (for AppManager 8.x or later) or **Administration** tab (for AppManager 7.x).

3 In the Deployment folder, select **Packages**.

4 On the Tasks pane, click **Check in Deployment Packages** (for AppManager 8.x or later) or **Check in Packages** (for AppManager 7.x).

5 Navigate to the folder where you saved `AM70-Win-RT-x.x.x.0.xml` and select the file.

6 Click **Open**. The Deployment Package Check in Status dialog box displays the status of the package check in.

7 To get the updates provided in this release, upgrade any running Knowledge Script jobs. For more information, see Section 3.7, “Upgrading Existing Knowledge Scripts,” on page 26.

2.4 Silently Installing the Module

To silently (without user intervention) install a module using the default settings, run the following command from the folder in which you saved the module installer:

```
msiexec.exe /i "AM70-Win-RT-7.x.x.0.msi" /qn MO_B_CONFIGREQUIRED=0
```

where *x.x* is the actual version number of the module installer.

To get the updates provided in this release, upgrade any running Knowledge Script jobs. For more information, see Section 3.7, “Upgrading Existing Knowledge Scripts,” on page 26.

To create a log file that describes the operations of the module installer, add the following flag to the command noted above:

```
/L* "AM70-Win-RT-7.x.x.0.msi.log"
```

The log file is created in the folder in which you saved the module installer.

NOTE: To perform a silent install on an AppManager agent running Windows Server 2008 R2, Windows 8 or Windows Server 2012, open a command prompt at the administrative level and select **Run as administrator** before you run the silent install command listed above.

To silently install the module on a remote AppManager repository, you can use Windows authentication or SQL authentication.

Windows authentication:

```
AM70-Win-RT-7.x.x.0.msi /qn MO_B_QDBINSTALL=1 MO_B_MOINSTALL=0
MO_B_SQLSVR_WINAUTH=1 MO_SQLSVR_NAME=SQLServerName MO_QDBNAME=AM-RepositoryName
```

SQL authentication:

```
AM70-Win-RT-7.x.x.0.msi /qn MO_B_QDBINSTALL=1 MO_B_MOINSTALL=0
MO_B_SQLSVR_WINAUTH=0 MO_SQLSVR_USER=SQLLogin MO_SQLSVR_PWD=SQLLoginPassword
MO_SQLSVR_NAME=SQLServerName MO_QDBNAME=AM-RepositoryName
```

2.5 Configuring Windows-RT Credentials in Security Manager

The domain, name, and password of the account to be associated with the Windows-RT service is stored in AppManager Security Manager. AppManager uses these credentials to unlock or log in to a computer left in a locked state. AppManager Security Manager stores the information on a per-agent basis. AppManager raises an event if no Security Manager entry is found for an agent.

You must complete this step before running the Discovery_WinRT7 Knowledge Script.

On the **Custom** tab in Security Manager, complete the following fields:

Field	Description
Label	`Win-RT7`
Sub-label	`ServiceAccount`
Value 1	Service account domain name
Value 2	Service account user name
Value 3	Service account password
Extended application support	Required field. Encrypts the user name and password in Security Manager. This option must be selected.

NOTE

- When you change the credentials in the Security Manager, the new credentials are not synced automatically. As a result, run a new job on the agent so that the new credentials are synced to the Agent.
- You can configure the **CacheCredentials** field in Win-RT config file to cache the Security Manager credentials or not. If you set the value of this field to True, the Win-RT service caches the Security Manager credentials and this cached credentials are used to log in to Windows every time the agent machine gets locked. If the value is set to False, the Win-RT service picks up the credentials from Security Manager every time a script is executed. The credentials are used for the Win-RT auto logon feature.

2.6 Discovering Windows-RT Resources

Use the Discovery_WinRT Knowledge Script to discover applications installed on systems with the Windows-RT managed object and the Windows-RT service. This script also verifies that the user domain, name, and password associated with the agent's Windows-RT service have been entered in AppManager Security Manager. For more information, see Section 2.5, "Configuring Windows-RT Credentials in Security Manager," on page 20.

After you run the Discovery script on a server, any discovered applications display as individual objects under the Win-RT7 object in the Navigation pane or TreeView. By default, this script runs once for each computer.

Set the **Values** tab parameters as needed.

Description	How To Set It
General	
List of applications to discover, separated by commas	Specify the names of the applications you want to discover. To discover more than one application, separate the application names with commas.
Raise event if discovery succeeds?	Select **Yes** to raise an event when the application or applications listed above are successfully discovered. The default is selected.
Event severity when discovery succeeds	Specify a severity level, from 1 to 40, to indicate the importance of an event if discovery is successful. Default is 25.
Raise event if discovery fails?	Select **Yes** to raise an event when the application or applications listed above are not successfully discovered. The default is selected.
Event severity when discovery fails	Specify a severity level, from 1 to 40, to indicate the importance of an event if discovery is not successful. Default is 5.
Debug	
Continue?	Select this check box to ignore any run time error and proceed with the Knowledge Script execution. The default is selected.
Debug?	Select this check box to enable debugging. The default is unselected. **NOTE:** Debug creates a *scriptname_jobid_*A.log file to log the error messages for a job.

2.7 Upgrading Knowledge Script Jobs

If you are using AppManager 8.x or later, the module upgrade process now *retains* any changes you might have made to the parameter settings for the Knowledge Scripts in the previous version of this module. Before AppManager 8.x, the module upgrade process *overwrote* any settings you might have made, changing the settings back to the module defaults.

As a result, if this module includes any changes to the default values for any Knowledge Script parameter, the module upgrade process ignores those changes and retains all parameter values that you updated. Unless you review the management guide or the online Help for that Knowledge Script, you will not know about any changes to default parameter values that came with this release.

You can push the changes for updated scripts to running Knowledge Script jobs in one of the following ways:

- Use the AMAdmin_UpgradeJobs Knowledge Script.
- Use the Properties Propagation feature.

Running AMAdmin_UpgradeJobs

The AMAdmin_UpgradeJobs Knowledge Script can push changes to running Knowledge Script jobs. Your AppManager repository (QDB) must be at version 7.0 or later. Upgrading jobs to use the most recent script version allows the jobs to take advantage of the latest script logic while maintaining existing parameter values for the job.

For more information, see the **Help** for the AMAdmin_UpgradeJobs Knowledge Script.

Propagating Knowledge Script Changes

You can propagate script changes to jobs that are running and to Knowledge Script Groups, including recommended Knowledge Script Groups and renamed Knowledge Scripts.

Before propagating script changes, verify that the script parameters are set to your specifications. You might need to appropriately set new parameters for your environment or application.

If you are not using AppManager 8.x or later, customized script parameters might have reverted to default parameters during the installation of the module.

You can choose to propagate only properties (specified in the **Schedule** and **Values** tabs), only the script (which is the logic of the Knowledge Script), or both. Unless you know specifically that changes affect only the script logic, you should propagate the properties and the script.

For more information about propagating Knowledge Script changes, see the “Running Monitoring Jobs” chapter of the *Control Center User Guide for AppManager*.

Propagating Changes to Ad Hoc Jobs or Knowledge Script Groups

You can propagate the properties and the logic (script) of a Knowledge Script to ad hoc jobs started by that Knowledge Script. Corresponding jobs are stopped and restarted with the Knowledge Script changes.

You can also propagate the properties and logic of a Knowledge Script to corresponding Knowledge Script Group members. After you propagate script changes to Knowledge Script Group members, you can propagate the updated Knowledge Script Group members to associated running jobs. Any monitoring jobs started by a Knowledge Script Group member are restarted with the job properties of the Knowledge Script Group member.

To propagate changes to ad hoc Knowledge Script jobs or Knowledge Script Groups:

1. In the Knowledge Script view, select the Knowledge Script or Knowledge Script Group for which you want to propagate changes.
2. Right-click the script or group and select **Properties propagation** > **Ad Hoc Jobs**.
3. Select the components of the Knowledge Script that you want to propagate to associated ad hoc jobs or groups and click **OK**:

Select	To propagate
Script	The logic of the Knowledge Script.
Properties	Values from the Knowledge Script Schedule and Values tabs, such as schedule, monitoring values, actions, and advanced options. If you are using AppManager 8.x or later, the module upgrade process now *retains* any changes you might have made to the parameter settings for the Knowledge Scripts in the previous version of this module.

3 Getting Started

Before you begin working with the AppManager (Windows-RT) module or the Windows-RT Designer, review the following steps as needed to optimize your environment and ensure efficient recording and playback of scripts, including scripts made in a previous version of Windows-RT.

If you are a new Windows-RT user, see the following topics:

3.1 Using a Dedicated Computer as the Agent Computer

You should use a dedicated computer as the agent computer for Windows-RT. This computer should be used only to create and play back scripts, and to run Windows-RT Knowledge Script jobs. Use a dedicated computer because the Windows-RT Designer records and plays back mouse actions during the execution of scripts. If the organization of icons and shortcuts on the desktop changes, scripts that include the playback of mouse actions might fail or behave unexpectedly.

3.2 Activating the WinRT7 Display Theme

Use the WinRT7 display theme with the agent computer to get the best results with the Windows-RT OCR tools.

To use the WinRT7 theme with Windows:

1. Right-click the desktop and select **Personalize**.
2. From the Theme list, navigate to the `\Windows\Resources\Themes\` folder and select **WinRT7**.
3. Click **Display**.
4. Click **Adjust resolution**.
5. Select the recommended screen resolution of **1024 by 768** or **1280 by 1024**. For best results, use the same resolution on all client computers.

6 Click **OK**.

7 Click **Personalization**.

8 Click **Window Color**.

9 To ensure that the gradient effect for Windows title bars has been disabled, verify that only one color is used for the Active **Title Bar** and Inactive Title Bar options in the Item list.

NOTE: For the best results, use the same color quality setting on all client computers.

10 Click **OK**.

NOTE: The procedure to configure the display theme might vary for different operating systems. For more information, refer to the operating system specific help documents.

3.3 Changing Compatibility Settings

Before running Windows-RT, change the compatibility settings so that the Windows-RT Designer and Player run as the Administrator.

To change compatibility settings for the Designer and Player:

1 Navigate to `NetIQ\AppManager\bin\Win-RT7\Player` and right-click `NetIQ.AppMan.WinRT7.Player.exe`.

2 Select **Run as** and log in as the Administrator.

3 Navigate to `NetIQ\AppManager\bin\Win-RT7\Designer` and right-click `Designer.exe`.

4 Select **Run as** and log in as the Administrator.

3.4 Setting Windows Firewall Exceptions

If you are running Windows Firewall, you will need to add Windows Firewall exceptions for processes used by Windows-RT to prevent Windows Firewall from blocking those incoming processes.

To set up Windows Firewall exceptions:

1 From the Windows Control Panel, double-click **Windows Firewall**.

2 On the **General** tab, make sure the option for **Don't allow exceptions** is not selected.

3 On the **Exceptions** tab, click **Add Program** and navigate to the following executable files:

- `NetIQ\AppManager\bin\WinRT\Service\NetIQ.AppMan.WinRT7.Service.exe`
- `NetIQ\AppManager\bin\WinRT\Player\NetIQ.AppMan.WinRT7.Player.exe`
- `NetIQ\AppManager\bin\WinRT\Designer\Remote.exe`

4 After adding all three processes to the exception list, click **OK.**

3.5 Configuring Settings for the Windows-RT Service

When you play back a Windows-RT script that you created with the Designer, the Windows-RT service locks the desktop of the playback computer by default after the transaction completes. To unlock the desktop when you execute the script again, the service uses information stored in AppManager Security Manager. For more information, see Section 3.2, “Activating the WinRT7 Display Theme,” on page 23.

You can change the configuration settings by modifying the `NetIQ.AppMan.WinRT7.Service.exe.config` file, located in the `NetIQ\AppManager\bin\Win-RT7\Service` directory.

You must stop and restart the Windows-RT service after making changes to this file. You can do this from the Control Panel (**Administrative Tools > Services**) or from a command prompt using the following syntax: `net stop NetIQWinRT7` and `net start NetIQWinRT7`.

The following table describes configuration options stored in the `NetIQ.AppMan.WinRT7.Service.exe.config` file. You can edit some of the values associated with each option to modify the behavior of the Windows-RT service. If you experience problems, consult the Windows-RT service log file. For more information, see “Agent Playback Log Files” on page 77.

Configuration Option and Default	What It Does
CommunicationPort value=8090	Specifies the port used for the communication with the Windows-RT service. You should not change this value.
MaxTraceFiles value=100	Specifies the maximum number of trace files to generate.
MaxScreenShots value=100	Specifies the maximum number of screenshot files to generate, using the Screenshot scriptstep.
ScreenshotPath (no default path)	Specifies the directory to store screenshots taken with the Screenshot scriptstep.
TracePath (no default path)	Specifies the directory to store the script traces Windows-RT creates when it executes and saves a script using the Save Trace option. Windows-RT does not save traces automatically.
ThumbnailPath (no default path)	Specifies the directory to store thumbnail files.
WebServerPort value=8080	Specifies the port used for the Web server. The value of 8080 is new to this release; previously the value was 80. NetIQ Corporation recommends that you do not change this value, unless you encounter a communication error.
UseWebServer value=True	Specifies you want to use the Web server.
WebServerHostName (no default value)	Specifies the host name of the Web server.
PlayerInactivityTimeOutMinutes value=3	Specifies the inactivity timeout for the Player. If the service reaches the specified time, the Player stops.

3.6 Locking the Windows-RT Player

When the Windows-RT Player runs, it displays a “lock” icon on the screen to indicate that the computer is locked. However, anyone can double-click the lock icon and gain control of the computer. You can configure the Windows-RT Player to remain locked until the you run the ChangeLocking Knowledge Script. Locking prevents anyone from using the Player computer and potentially changing the organization of icons on the computer’s desktop. If the organization of icons and shortcuts on the desktop changes, scripts that include the playback of mouse actions might fail or behave unexpectedly.

You can change the locking option in one of the following ways:

- Configure the Player to automatically start in locked mode:
  1. In Windows Explorer, navigate to the **WinRT7Player** shortcut in the `C:\Documents and Settings\Administrator\Start Menu\Programs\Startup` folder.
  2. Right-click on the shortcut and select **Properties**.
  3. In the **Target** field, type `-locked`.
- Manually launch the Player to start in locked mode:
  1. From a command prompt, navigate to `\Program Files\NetIQ\AppManager\bin\WinRT7\Player`.
  2. Run the following command: `NetIQ.AppMan.WinRT7.Player.exe -locked`

3.7 Upgrading Existing Knowledge Scripts

If you have Knowledge Scripts from version 7.1 of Windows-RT already running in your environment, you should upgrade these Knowledge Scripts to the latest version. Before upgrading, you should assign a default value to every variable in the older version of the script.

Upgraded scriptsteps that do not have corresponding scriptsteps in this version display on the Compatibility category, with (Compatibility) added to their names. These scriptsteps display in yellow on the flowchart view of the Script pane.

To upgrade a script from a version of Windows-RT earlier than version 7.1:

1. In the Windows-RT Designer, open the project to which you want to import the script.
2. On the **Start** menu, click **Import**.
3. In the Import Script dialog box, click **WinRT Knowledge Scripts** from the **Files of type** list.
4. Browse to the location of the desired script file and select the file.
5. Click **Open**. The Windows-RT Designer upgrades the script as needed
6. On the **Start** menu, click **Save Project**.
7. Check the script into AppManager. For more information, see Section 5.4, “Checking Scripts into an AppManager Repository,” on page 60.

After you have imported a Windows 7.1 script to the Window-RT Designer, you can access that script on the Compatibility category on the Script Library pane. To view this category, click the **Application Settings** button on the **Tools** menu and select the **Show Compatibility Scriptsteps** check box on the **Projects** tab.

3.8 Disabling CTRL + ALT + DELETE and Legal Notice screens

The CTRL + ALT + DELETE screen and Legal Notice screen must be disabled so that the Win-RT auto logon feature works properly:

- To disable the local settings, run the TakeDesktopOwnership Knowledge Script. For more information, see Section 6.3, “TakeDesktopOwnership,” on page 73.
- To disable the group policies, contact your system administrator.

3.9 Using the Windows-RT Module for the First Time

This section provides a brief overview step-by-step example illustrating how you can use the Windows-RT module.

TIP: Windows-RT jobs run serially, which means that Windows-RT completes one job before starting another job. Scheduling jobs efficiently is the key to getting the most out of this module.

To begin recording scripts to monitor Windows response time:

1. In the Windows-RT Designer, create a new project and script. For more information, see Section 5.1, “Creating a Script Manually,” on page 55 or Section 5.2, “Creating a Script Using the Wizard,” on page 57.
2. Check the script you created into the AppManager repository. For more information, see Section 5.4, “Checking Scripts into an AppManager Repository,” on page 60.
3. In AppManager, run the new Windows-RT Knowledge Script, specifying which data streams should be collected, which events are enabled, and the values of the corresponding event severities.

TIP

- After you install the module and start to create scripts using the Windows-RT Designer, you should not make changes to the appearance and organization of the desktop on the dedicated computer.
- Record and play back scripts on computers using the same locale to ensure locale settings such as date formats, currency formats, and other region-based settings are maintained. Do not attempt to use scripts interchangeably on computers that use different locales.

You can use the Check In Startup Script and Check In Active Script buttons in the Windows-RT Designer only if the Designer is installed on a computer that also contains the AppManager Operator Console. Otherwise, save the script as a `.qml` file and copy it to a directory that can be accessed by the computer containing the Operator Console.

3.10 What Happens When You Run a Windows-RT Script

At run time, the following actions occur:

1. The Knowledge Script transmits the embedded script to the Windows-RT managed object on the agent computer.
2. The managed object communicates with the Windows-RT service and transfers the script to the service.
3. The service passes the job request to the Windows-RT Player, and transfers the script to the Player. If no user is logged in, or if the computer is locked, the service retrieves the encrypted user information from Security Manager and uses the credentials to login to or unlock the computer.
4. The Player changes desktops and executes the script.
5. The Player passes the results of the script back to the Windows-RT service.
6. The Windows-RT service informs the managed object that the execution is finished.
7. The Knowledge Script generates events and data streams.

3.11 Running Windows-RT and Web-RT Knowledge Scripts on the Same Computer

If you are running the AppManager ResponseTime for Web module (Web-RT) and install Windows-RT on the same computer, you might experience problems. Installing Web-RT and Windows-RT on the same computer is not recommended.

You cannot run Windows-RT Knowledge Scripts and Web Transaction scripts generated by the Web-RT Web Recorder extension simultaneously because only one module can take ownership of the desktop at any given time.

NOTE: Only Web-RT scripts generated by the Web Recorder are affected by this conflict. Other Web-RT Knowledge Scripts, including those generated by the URL Check Recorder extension (URLCheck Knowledge Scripts), do not require ownership of the desktop, and you can run these scripts at the same time as Windows-RT Knowledge Scripts.

The `Desktop` Registry value controls desktop ownership. You can find this registry value under the `HKEY_LOCAL_MACHINE\SOFTWARE\NetIQ\Response Time` Registry key. The module name listed as the value for this key (`Web-RT` or `Win-RT7`) determines which Knowledge Scripts can be run.

When you drop a Web-RT_WebTransaction or Win-RT7 Knowledge Script on an agent, that script or transaction checks whether the `Desktop` Registry value already exists, and performs the following actions:

- If the `Desktop` value does not exist or is blank, the script attempts to create or update the `Desktop` value with the associated module name. If the value is created (or updated) successfully, the Knowledge Script runs normally. If the script fails to create (or update) the "`Desktop`" value, AppManager raises an error event and the script stops running.
- If the `Desktop` value already exists and is set to the Knowledge Script's associated module name, the script runs normally.
- If the `Desktop` value already exists, and is set to any value other than the Knowledge Script's associated module name, AppManager raises an error event and the script stops running.

Both Windows-RT and Web-RT contain a Knowledge Script called TakeDesktopOwnership that you can use to take ownership of the desktop. Once you set the `Desktop` Registry value, you can run that script to overwrite any existing value and give control of the desktop to the relevant ResponseTime module. For more information, see Section 6.3, “TakeDesktopOwnership,” on page 73.

4 Using the Windows-RT Designer

You use the Windows-RT Designer to create and customize Knowledge Scripts that you can use to monitor the availability and performance of your Windows environment. This chapter defines the elements of the Windows-RT Designer, including the scriptsteps you use to create your customized scripts.

To launch the Windows-RT Designer in the AppManager Operator Console, click the **Extensions** menu and select **Win-RT 7 Designer**. If you do not use the Operator Console, add a shortcut to the `Designer.exe` file from `\NetIQ\AppManager\bin\Win-RT7\Designer` to your desktop.

To launch Windows-RT Designer in the AppManager Control Center Console, click **Win-RT Designer** in the **Tasks** pane, and then click the **URLs** tab. Enable the Link table cell and type the URL to the Win-RT Designer in the **Link** field, and then click **Enter**.

4.1 Understanding Script Elements

This section describes how the different elements that make up a script in Windows-RT.

Scripts and Scriptsteps

A script is composed of a set of instructions that explain how to run Windows transactions, such as launching an application or executing steps within an application. These instructions are recorded as a series of user actions called scriptsteps, which can include mouse movements, mouse clicks, and key strokes. These actions can be played back when the script is run.

NOTE

- A standard script, also called a main script, is different from a module script in that a module script can be embedded like a scriptstep into a main script. You can create a module script for a set of procedures that you use often. You create and edit both scripts in essentially the same manner.
- If you create a new parameter in a module script and assign it as **Is Input**, this parameter does not display in the **Value** tab of AppManager Knowledge Script. You can use the Is Input functionality for the standard script.
- If you create a module script that calculates any data stream, export it from the Win-RT7 Designer, and run the script multiple times as an AppManager job, only a single instance of the data stream is sent to AppManager.

After you check Windows-RT scripts into the AppManager repository, those scripts function just like other AppManager Knowledge Scripts. The Knowledge Scripts on the Win-RT7 tab of the AppManager Operator Console are typically used to measure response time for Windows-based applications, but they can perform many other tasks as well.

Script Parameters

Script parameters are the overall settings for a script that you can access on the Parameters pane. To view script parameters, select the script in the Project Files pane and click the **Display Script Parameters** button on the **Start** menu.

To rename an existing script, right-click that script in the Project Files pane and select **Rename** from the pop-up menu. You can also create new scripts, display the selected script in the Script pane, change the startup script, and delete scripts.

The following table describes the different script parameters. Note that numbers used as parameters must be integers: numbers that can be written without a fractional or decimal component.

Parameter	What it Does
Author	Lists the name of the creator of this script.
Title	Lists the name of this script. Default is Script (*n*), with n representing the number of existing scripts in the project, if a project contains more than one script.
Cleanup Processes	Stops all processes that are still running when this script has been completely executed. The default is checked.
Cleanup Processes Exceptions	Contains a comma-separated list of process that you do not want the Cleanup Process operation to stop. To view or edit a list of currently running processes, click the icon in the Value field. From the Process Selection dialog box, select the check boxes next to the running processes you do not want the Cleanup Process operation to stop.
Execution Timeout	Specifies the length of the timeout if a script takes too much time to open or to run. If the Timeout value is met, Windows-RT creates an entry in the Traces pane. The default is 900 seconds (15 minutes).
Run on default desktop	Plays back the script on the default desktop/session when the Win-RT7 script is run as an AppManager job, instead of using a virtual desktop for execution. **NOTE:** If you create a script that includes launching the Citrix ICA Client, select this option.
Use Windows Explorer	Opens a new instance of Windows Explorer when the script is executed.
Flowchart Size	Specifies the dimensions of the flowchart. The default is 2400 by 1800 pixels.
Move Step Large	Specifies the maximum number of pixels a flowchart element will move when you select it, hold down the `[Shift]` key, and then press an arrow key to move it. The default is 10 pixels.
Move Step Small	Specifies the minimum number of pixels a flowchart element will move when you select it and press an arrow key to move it. The default is 2 pixels.

Custom Parameters

You can create customized Script Parameters using the **Create Parameter** option available in the Parameters pane.

To create a new Parameter, perform the following:

1. In the Parameters pane, Click **Create Parameter.**

   The Parameter Settings page is displayed.

2. Specify the Parameter details on the Parameter Settings page.

   NOTE: If you select **This parameter is an input value of the script,** then this parameter is displayed in the **Value** tab of AppManager Knowledge Script.

3. Click **OK**.

   The new Parameter is displayed within **Custom Parameters** in Parameter Description.

4.2 Scriptstep Library Quick Reference

The tables in the following sections describe the functions and parameters available in the various scriptsteps found on the Scriptstep Library pane.

NOTE

- By default, every script contains a Start scriptstep, which serves as the initial step for every script. A Start scriptstep cannot be deleted.
- All scriptsteps have an **Enabled** check box. If this check box is cleared, the script skips this scriptstep during script execution.

- To make parameter changes for more than one scriptstep, press [Ctrl] while clicking the scriptsteps in the Script pane, and then make the changes in the Parameters pane. All selected scripts update with the new parameter data.
- All scriptsteps have a Style section that lets you change the color and other properties of that script's flowchart icons. You can change the style parameters of your flowchart icons to make your flowchart easier to read, especially if you have many scriptsteps in your script.

Compatibility Category

This category displays a list of scriptsteps that have been imported from a previous version of Windows-RT and upgraded as a result of the import. To view the Compatibility category, click **Application Settings** on the **Tools** tab and select **Show Compatibility Scriptsteps** on the **Projects** tab.

Scriptstep Name	What It Does
Arithmetic Allocation (Compatibility)	Performs an arithmetic operation on a pair of variables. The result of the operation is the Calculation Result output value. **Key Parameters:** ◆ **First Operand:** This value is a variable that serves as the first operand in the arithmetic operation. Default is 0. ◆ **Operation:** Specifies the type of operation performed on the two operands. Options include: Add, Subtract, Multiply, Divide, Exponentiate, Higher Of, and Lower Of. Default is Add. ◆ **Second Operand:** This value is a variable that serves as the second operand in the arithmetic operation. Default is 0. ◆ **Target Parameter:** Specifies a script parameter that will store the result of the calculation.
Extended OCR (Compatibility)	Provides the option for branching the script, performing an OCR scan, and adding a delay as well as a loop to the script. **Key Parameters:** ◆ **Decision Expression:** Specifies the criteria required for the decision, which is set up in the Decision Expression Editor dialog box. For more information, see Section 5.4, "Checking Scripts into an AppManager Repository," on page 60. ◆ **Delay:** Specifies the length of a pause during the execution of the script. Default is 1 second. ◆ **Loop Count:** Specifies the number of times the loop is executed. Default is 1. ◆ **Region:** Defines the region of the OCR scan using X and Y coordinates and a specified width and height. Defaults are: Fullscreen, X=100, Y=100, Width=100, Height=100**.** Target Parameter: Specifies a variable that will store the result of the OCR scan. ◆ **Method:** Specifies the method of analyzing the OCR scan, removing everything but the requested types of characters. Select Raw to not change anything in the scan results. ◆ **White Space:** Specifies how the white, or empty, space in the OCR scan should be used. The white space can be kept, compressed, or removed. Default is Keep.

Scriptstep Name	What It Does
OCR Allocation (Compatibility)	Scans the screen and stores the textual result of the OCR scan to the given variable. **Key Parameters:** ◆ **Region:** Defines the region of the OCR scan using X and Y coordinates and a specified width and height. Defaults are: Fullscreen**.** Note: Avoid using the Fullscreen option unless you absolutely need this data, because scanning a region this large is a CPU-intensive operation and may decrease performance significantly. ◆ **Target Parameter:** Specifies a variable that will store the result of the OCR scan. ◆ **Method:** Specifies the method of analyzing the OCR scan, removing everything but the requested types of characters. Select Raw to not change anything in the scan results. Default is Raw. ◆ **White Space:** Specifies how the white, or empty, space in the OCR scan should be used. The white space can be kept, compressed, or removed. Default is Keep.
Start Timer (Compatibility)	Starts a new timer for the specified variable. If the timer is stopped with a Stop Timer scriptstep, the variable will contain the amount of seconds elapsed between the execution of the Start Timer scriptstep and the execution of the Stop Timer scriptstep. If no Stop Timer scriptstep runs for the specified variable before the script ends, the timer value will be set to the Time Default Value parameter. **Key Parameters:** ◆ **Target Parameter:** Specifies a variable that will store the result of the timer. ◆ **Time Default Value:** Specifies the default value of the timer. Default is 0.
Stop Timer (Compatibility)	Stops the timer currently running for a specified variable. **Key Parameters:** ◆ **Start Timer:** Specifies the timer to be stopped, which you can choose from the Scriptstep Browser dialog box. ◆ **Target Parameter:** Specifies a variable that will store the timer results.
Window Allocation (Compatibility)	Searches for a specified window and assigns its handled to a parameter. **Key Parameters:** ◆ **Target Window:** Specifies the target window you want to locate. ◆ **Target Parameter:** Specifies a variable that will store the window handle for the target window.

Debug Category

These scriptsteps insert basic debugging operations into a script, along with tools to help you organize your script and take screenshots.

Scriptstep Name	What It Does
Debug Text	Creates a debug entry with your specified text in the Traces pane. **Key Parameters:** ◆ **Text:** The text that is displayed on the scriptstep icon in the flowchart view of the Script pane. The default text is the name of the scriptstep. ◆ **Debug Text:** The text you want to display in the Traces pane, under the Debug Text row of the trace log. To add the value of another parameter to this field, click the **Parameter Browser** button and select a parameter from the Parameter Browser dialog box.
Message Box	Displays a simple feedback or status message box with specified text on the playback computer. You can add variables based on scriptstep parameters or script parameters. **Key Parameters:** ◆ **Center Message Box**: Ensures the message box is centered when it is displayed. Default is checked. ◆ **Duration**: Specifies how long you want the message box to display. Default is three seconds. ◆ **Is Topmost Window**: Ensures the message box is displayed on top of all other windows so it can be read. Default is checked. ◆ **Message Text**: The text you want to display in the message box. To add the value of another parameter to this field, click the **Parameter Browser** button or select **Define Parameter Definition** from the menu and select a scriptstep parameter or a script parameter from the Parameter Browser dialog box. ◆ **Position**: Specifies the X and Y coordinates for the location of this message box on the screen. Default is the middle of the screen, or X=0 and Y=0. ◆ **Size**: Specifies the width and height of the message box. Default is 200 by 200.
Screenshot	You can take a screenshot during script execution and use this screenshot to specify a search image by defining where you clicked the mouse button. You can open and save this screenshot from the **Screenshot** menu on the Traces pane. **Key Parameter:** ◆ **Include Mouse Pointer:** Records the location of the mouse pointer when the screenshot is made. Default is checked.

Documentation Category

The scriptsteps in this category let you organize and label your scriptsteps in the flowchart view on the Script pane. These scriptsteps are used only to help clarify and organize the visual display of the script, and they perform no function when the script is executed.

Scriptstep Name	What It Does
Frame	Adds a resizable text box to the flowchart of the Script pane to help you organize the visual display of the script. You can place the frame around scriptsteps to sort similar scriptsteps and better arrange the appearance of your flowchart. **NOTE:** You must click the border of the Frame scriptstep to move it. **Key Parameter:** ◆ **Text:** This text is displayed on the text box of the Frame scriptstep.
Label	**Adds a resizable label to the** flowchart of the Script pane to help you organize the visual display of the script. **Key Parameter:** ◆ **Text:** Text for your label to help organize and explain the flowchart in the Script pane.

I/O Category

The scriptsteps in this category let you read or write data from the files on the local system. All of these scriptsteps have Default Value parameters you can set to prevent errors if the wrong type of data is read.

Scriptstep Name	What it Does
Read Clipboard	Reads text from the Clipboard. If the Clipboard contains no content, or if the content is not text, the value in the Default Value (Text) parameter is used. If the value read from the Clipboard is convertible to a number or a Boolean value, that value is displayed with the Value (Numeric) or Value (Boolean) output parameter. **Key Parameters:** ◆ **Default Value (Boolean):** Select this check box if you want to allow a Boolean value to be used as the Clipboard content. ◆ **Default Value (Numeric):** This numeric value can be used as the Clipboard content if no content is added to the Clipboard during the execution of the script. Default is 0. ◆ **Default Value (Text):** This text is used as Clipboard content if no content is added to the Clipboard during the execution of the script, or if the content is not text.

Scriptstep Name	What it Does
Read INI File	Reads the specified data from .ini files, simple text files with a basic structure. **Key Parameters:** ◆ **Default Value (Boolean):** Select this check box if you want to allow a Boolean value to be used as the Clipboard content. ◆ **Default Value (Numeric):** This numeric value can be used as the INI file data if no content is read from the INI file during the execution of the script. Default is 0. ◆ **Default Value (Text):** This text is used as the INI file data if no content is read from the file during the execution of the script, or if the content read from the file is not text. ◆ **Entry Name:** Identifies the requested value in the INI file. ◆ **File Name**: Specifies the name and location of the INI file. You can browse for files by clicking the yellow folder icon, or you can choose from a list of commonly used folders by clicking the green icon. ◆ **Section Name:** Specifies the section name from the INI file, which is displayed in brackets.
Read Registry File	Reads the specified data from the registry file. **Key Parameters:** ◆ **Default Value (Boolean):** Select this check box if you want to allow a Boolean value to be used as the Clipboard content. ◆ **Default Value (Numeric):** This numeric value can be used as the registry file data if no content is read from the registry file during the execution of the script. Default is 0. **Default Value (Text):** This text is used as the registry file data if no content is read from the file during the execution of the script, or if the content read from the file is not text. ◆ **Registry Hive:** Specifies the section of the registry where you can locate the desired registry file. Default is Local Machine. ◆ **Registry Key:** The key and subkey location for the registry file, such as `\Software\Test`. ◆ **Registry Value Name:** The name/data pairs stored within the key for the registry file.
Read Text File	Reads the specified data from a text file or a related file type, such as .xml or .html. **Key Parameters:** ◆ **Default Value (Boolean):** Select this check box if you want to allow a Boolean value to be used as the Clipboard content. ◆ **Default Value (Numeric):** This numeric value can be used as the text file data if no content is read from the text file during the execution of the script. Default is 0. ◆ **Default Value (Text):** This text is used as the text file data if no content is read from the file during the execution of the script, or if the content read from the file is not text. ◆ **Encoding:** Specifies any special encoding the text may use, such as Unicode, ASCII, or UTF-7. Default is the system's current ANSI code page. ◆ **File Name**: Specifies the name and location of the text file. You can browse for files by clicking the yellow folder icon, or you can choose from a list of commonly used folders by clicking the green icon.

Scriptstep Name	What it Does
Read XML File	Reads the specified text from an XML file. **Key Parameters:** ◆ **Default Value (Boolean):** Select this check box if you want to allow a Boolean value to be used as the Clipboard content. ◆ **Default Value (Numeric):** This numeric value can be used as the XML file data if no content is read from the XML file during the execution of the script. Default is 0. ◆ **Default Value (Text):** This text is used as the XML file data if no content is read from the file during the execution of the script, or if the content read from the file is not text. ◆ **Attribute Name:** The attribute value from the XML code, which always is displayed in quotation marks, such as "5 mins". ◆ **Element Type**: Specifies the XML element type for the data you want to read from the file. Default is Node Inner Text. ◆ **File Name**: Specifies the name and location of the XML file. You can browse for files by clicking the yellow folder icon, or you can choose from a list of commonly used folders by clicking the green icon. ◆ **XPath Expression**: Refers to a specific section of an XML document.
Write Text File	Creates a new text file, or adds new text to an existing text file, and then adds a line of text you provide in the Content parameter to that file. **Key Parameters** ◆ **Content**: Type the text you want to include in the text file. ◆ **Encoding:** Select any special encoding the text may use, such as Unicode, ASCII, or UTF-7. Default is the system's current ANSI code page. ◆ **File Mode:** Select how you want the text added to the file if the text file already exists. You can add the text and overwrite any existing text in the file, add the new text at the end of the file, or add the text at the beginning of the file without overwriting any existing text in the file. ◆ **File Name**: Specifies the name and location of the text file. You can browse for files by clicking the yellow folder icon, or you can choose from a list of commonly used folders by clicking the green icon.

Input Simulation Category

The scriptsteps in this category allow you to simulate user actions, including pressing specific keys, and clicking or moving the mouse.

Scriptstep Name	What It Does
Hold Mouse Button	Simulates holding down of the mouse button, such as when dragging and dropping. **Key Parameters:** ◆ **Mouse Button:** Specifies which mouse button should be held down. Default is Left Mouse Button. ◆ **Position:** Specifies the screen coordinates for the location of the simulated mouse action. To set the coordinates, you can click a specific location, use a screen shot to determine the mouse click location, or use X and Y coordinates with a rectangle you can resize. Default is Undefined.
Keys	Plays a specified key combination, such as `Ctrl+s` or `Windows logo+r`. You can also simulate the typing of text in an application. Tip: You can create a shortcut to launch an application on your desktop by editing the **Shortcut** tab of the Properties dialog box for that application. Right-click the application's icon on the desktop and select **Properties**. In the Shortcut key field, press the key combination for launching the application and click **OK**. Add the same key combination to the parameters for the Keys scriptstep, below. For more information, see "Using the Keys Scriptstep with Windows Shortcuts" on page 76. **Key Parameters:** ◆ **Delay:** Specifies the length of a delay between each key that is pressed. Default is 50 Milliseconds. ◆ **Input Text:** Any additional keys that will simulate being pressed at the same time as the above keys in the preceding parameter settings, such as F4 used with Press `ALT` Key to close a file. For special keys, click the **Select a Key** icon to display a list of keys. You can also simulate the typing of text into an application by adding that text to this field. **NOTE:** Any letters you enter as part of a key combination should be lower-case, such as a "s" instead of "S" for `Ctrl+s`. ◆ **Press ALT Key:** Simulates the pressing of the `Alt` key. ◆ **Press CTRL Key:** Simulates the pressing of the `Ctrl` key. ◆ **Press SHIFT Key:** Simulates the pressing of the `Shift` key. ◆ **Press WINDOWS Key:** Simulates the pressing of the `Windows logo` key.

Scriptstep Name	What It Does
Mouse Click	Simulates a click or multiple clicks of the mouse. **Key Parameters:** ◆ **Click Count:** Specifies the number of mouse clicks to make. The default is 1. ◆ **Delay:** Specifies the length of the delay between mouse clicks. The default is 200 milliseconds, or 0.2 seconds. ◆ **Mouse Button:** Specifies which mouse button should be clicked. Default is Left Mouse Button. ◆ **Position:** Specifies the screen coordinates for the location of the simulated mouse click or clicks. To set the coordinates, you can click a specific location, use a screen shot to determine the mouse click location, or use X and Y coordinates. Default position is Undefined.
Mouse Move	Simulates the movement of the mouse. **Key Parameters:** ◆ **Delay:** Specifies the length of the delay between mouse movements. The default is 20 milliseconds. ◆ **Destination Position:** Specifies the end location for the mouse movement. Default position is Undefined. ◆ **Source Position:** Specifies the starting location for the mouse movement. Default position is Undefined. ◆ **Split Count:** Specifies the number of steps to take to move the mouse from the source location to the destination. The default is 10.
Release Mouse Button	Simulates the release of a mouse button. **Key Parameters:** ◆ **Mouse Button**: Specifies which mouse button should be released. Default is Left Mouse Button. ◆ **Position**: Specifies the screen coordinates for the location of the simulated mouse action. To set the coordinates, you can click a specific location, use a screen shot to determine the mouse click location, or use X and Y coordinates. Default position is Undefined.

Measurement Category

You can use the scriptsteps in this category to track availability as well as set timers for response time. Timers are automatically defined as data streams in the Knowledge Script when the script is exported from the Designer.

Scriptstep Name	What It Does
Availability	Tracks whether an application is functioning or “up” instead of “down” and creates a data stream. Availability values are 100 if the application is available, and 0 if application is not available. **Key Parameters:** ◆ **Availability Value**: Specifies the availability of the monitored process as a number. For example, if the application is up and running, Windows-RT sets the availability to 100. Default is 100. ◆ **Detail Information**: Text or a variable selected with the Parameter Browser that further explains the availability value. ◆ **Parameter Description**: Custom text for this parameter that will display on the **Properties** tab of the Knowledge Script in AppManager. Required. ◆ **Title:** Custom text label for the availability data stream. You can dynamically generate the text for this label using the Parameter Browser dialog. ◆ **Unit:** Specifies a label for the unit of measurement. Default is %.
Measurement	Creates a custom data stream with any value. **Key Parameters:** ◆ **Detail Information**: Text or a variable selected with the Parameter Browser that further explains the measurement value. ◆ **Parameter Description**: Custom text for this parameter that will display on the **Properties** tab of the Knowledge Script in AppManager. Required. ◆ **Title:** Custom text label for the measurement data stream. You can dynamically generate the text for this label using the Parameter Browser dialog. ◆ **Unit:** Specifies a label for the unit of measurement, if needed. ◆ **Value**: Specifies the measured value as a number. Default is 0. ◆ **Postpone Execution**: Select this check box to write the data stream after the script executes.
Pause Timer	Pauses a running performance measurement. This scriptstep should come after a Start Timer scriptstep in the flowchart. **Key Parameter:** **Start Timer:** Specifies the timer to be paused, from the Scriptstep Browser dialog box.
Resume Timer	Resumes a paused performance measurement. This scriptstep should come after a Start Timer scriptstep and a Pause Timer scriptstep. **Key Parameter:** **Start Timer:** Specifies the timer to be resumed, from the Scriptstep Browser dialog box.

Scriptstep Name	What It Does
Start Timer	Starts a new timer for the specified variable to measure how long it takes to open a program or run a process. This scriptstep also creates a data stream. When a Stop Timer scriptstep or Pause Timer scriptstep stops or pauses the timer, the Elapsed Time output parameter will contain the amount of seconds that have passed between the execution of the Start Timer scriptstep and the execution of the Stop Timer or Pause Timer scriptstep. **Key Parameters:** ◆ **Detail Information**: Text or a variable with Parameter Browser that further explains the availability value. ◆ **Parameter Description**: Custom text for this parameter that will display on the **Properties** tab of the Knowledge Script in AppManager. Required. ◆ **Title:** Custom text label for the timer data stream. You can dynamically generate the text for this label using the Parameter Browser dialog.
Stop Timer	Stops the timer that was begun by the Start Timer scriptstep. After script execution, the Measured Time output parameter contains the amount of seconds that have passed between the execution of the Start Timer scriptstep and the execution of the Stop Timer scriptstep. **Key Parameter:** ◆ **Start Timer:** Specifies the timer to be stopped, which you can choose from the Scriptstep Browser dialog box.

Notification Category

This scriptstep allows you to generate an event when a specified condition in the script is met, such as when an application fails to open in the specified time.

Scriptstep Name	What It Does
Event	When certain criteria are met, generates an event in AppManager using the specified data. **Key Parameters:** ◆ **Description**: The text entered here is used in the *Raise event?* parameter of the Knowledge Script when the script is checked into the AppManager repository. ◆ **Detail Message**: A detailed text message for the event. The text can include the value of a variable by using the Parameter Browser dialog box. ◆ **Raise Event**: Raises an event by default. Default is checked. ◆ **Severity: Specifies** the event severity as a number. The event severity can be changed at run time when you execute the script in AppManager. Default is 40. ◆ **Short Message**: Specifies the event title. This text can include variables, which you can add by using the Parameter Browser dialog box. ◆ **Take Screenshot**: Takes a screenshot of the application where the event occurred. Default is unchecked.

Parameters Category

The scriptsteps in this section let you perform mathematical equations and establish OCR settings, as well as set up a variety of parameters that are usually used for compatibility purposes with previous versions of Windows-RT.

Scriptstep Name	What It Does
Arithmetic (Date)	Performs an arithmetic operation based on date values, such as days, months, or years. You can use this scriptstep to automatically increment the date for weekly or monthly transactions The result of the operation is the Calculation Result output value, which is displayed in the Traces pane. You can use the result of your operation as a variable in other scriptsteps. **Key Parameters:** ◆ **Add Days:** Specifies a number of days to add to the current date. Default is 0. ◆ **Add Months:** Specifies a number of months to add to the current date. Default is 0. ◆ **Add Years:** Specifies a number of years to add to the current date. Default is 0. ◆ **Use Current Date As Value:** Select this check box to use today's date instead of the number in the Value parameter, below. ◆ **Value:** Type or select a date to use as your base value for your arithmetic operation. Click the **Calendar** icon to insert today's date, click the **Garbage Can** icon to clear the date, or click the down arrow to access a calendar.
Arithmetic (Number)	Performs an arithmetic operation on one or two variables from other scriptsteps. The result of the operation is the Calculation Result output value, which is displayed in the Traces pane. You can use the result of your operation as a variable in other scriptsteps. **Key Parameters:** ◆ **First Operand:** This value is a variable that serves as the first operand in the arithmetic operation. Default is 0. ◆ **Operation:** Specifies the type of operation performed on the two operands. Options include: Add, Subtract, Multiply, Divide, Exponentiate, Higher Of, and Lower Of. Default is Add. ◆ **Second Operand:** This value is a variable that serves as the second operand in the arithmetic operation. Default is 0.
Eval	Allows you to evaluate a string of text or other characters. **Key Parameter**: **Input:** Type the name of the string to be evaluated by this scriptstep. You can select a key by clicking the button icon, browse for commonly used files by clicking the green folder icon, or open the **Paramer Browser** button by clicking the **Parameter Browser** icon.

Scriptstep Name	What It Does
OCR	Performs an optical character recognition, or OCR, scan on a specified region. After the scriptstep runs, you can use the OCR Result output parameter in other scriptsteps, such as with the Measurement Scriptstep, as data detail. **Key Parameters:** ◆ **Screen Region:** Defines the region of the OCR scan using X and Y coordinates and a specified width and height. Defaults are: Fullscreen, X=100, Y=100, Width=100, Height=100**.** Note: Avoid using the Fullscreen option unless you absolutely need this data, because scanning a region this large is a CPU-intensive operation and may decrease performance significantly. ◆ **Method:** Specifies the method of analyzing the OCR scan, removing everything but the requested types of characters. Select Raw to not change anything in the scan results. ◆ **White Space:** Specifies how the white, or empty, space in the OCR scan should be used. The white space can be kept, compressed, or removed. Default is Keep.
Set Parameter (Boolean)	Assigns a Boolean value to the referenced parameter. **Key Parameters:** ◆ **Target Parameter**: Specifies a script parameter chosen from Parameter Browser dialog box. **Value:** Select this check box to allow a Boolean value to be used
Set Parameter (Date)	Assigns a date value to a referenced parameter. **Key Parameters:** ◆ **Default to Current Date:** Select this check box to use today's date if the date is not valid in the Value parameter, below. ◆ **Target Parameter**: Specifies a script parameter chosen from Parameter Browser dialog box. The script sets this parameter to the date in the Value parameter, below, and removes any parameter reference for this parameter. ◆ **Use Current Date**: Select this check box to use today's date instead of the date in the Value parameter, below. ◆ **Value**: The date you want to use for your parameter.
Set Parameter (Number)	Assigns **a numeric value to the referenced parameter.** **Key Parameters:** ◆ **Target Parameter**: Specifies a script parameter chosen from Parameter Browser dialog box. The script sets this parameter to the number in the Value parameter, below, and removes any parameter reference for this parameter. ◆ **Value**: The number you want to use for your parameter. Default is 0.
Set Parameter (Text)	Assigns **a text value to the referenced parameter.** **Key Parameters:** ◆ **Target Parameter**: Specifies a script parameter chosen from Parameter Browser dialog box. The script sets this parameter to the text in the Value parameter, below, and removes any parameter reference for this parameter. ◆ **Value**: The text you want to use for your parameter.

Processes Category

These scriptsteps start and monitor a process, such as launching an application and noting when that application ends.

Scriptstep Name	What It Does
Start Process	Runs a process from a file on the local system, usually an executable file that opens an application. **Key Parameters:** ◆ **Arguments**: These values specify any additional command-line arguments that may be needed for command-line parameters. ◆ **File Name**: Specifies the name and location of the executable file for the process. You can browse for files by clicking the yellow folder icon, or you can choose from a list of commonly used folders by clicking the green icon, and then specify the file name from that folder. ◆ **Wait for Termination**: Select this check box to force the script to wait until the process terminates before continuing. Default is unchecked.
Wait for Process Exit	After a process has been started with the Start Process scriptstep, this scriptstep pauses the script until the process has been closed. The green arrow points to the scriptstep to run after the process has exited, and the red arrow points to the scriptstep to run if the value in the Wait Time parameter is met, signifying that the process never completed. **Key Parameters:** ◆ **Process Handle**: Specifies the process for which you want the script to wait. This parameter should use the File Name variable from the Start Process scriptstep. ◆ **Wait Time**: Specifies the length of time the system should wait before timing out. Default is 30 seconds. ◆ **Wait For Dependent Processes**: Makes the script wait for all processes to finish before continuing.

Script Flow Category

These scriptsteps control the flow, organization, and branching of your script.

Scriptstep Name	What It Does
Decision	Branches a script according to criteria specified in the Decision Expression Editor dialog box. The criteria you compare must be of the same type, such as text, numeric, or Boolean. The green arrow connects to the scriptstep to run if the decision is yes or true. The red arrow connects to the scriptstep to run if the decision is no or false. Because you are branching your script, add an End scriptstep at the end of each branch. **Key Parameter:** ◆ **Decision Expression:** Specifies the criteria required for the decision, which is set up in the Decision Expression Editor dialog box. For more information, see Section 5.4, “Checking Scripts into an AppManager Repository,” on page 60.
Delay	Creates a pause during the execution of the script. **Key Parameter:** ◆ **Duration:** Specifies the length of your delay. Default is 1 second.
End	Marks the endpoint of the script flow, when the script execution ends. Required for all branches in a script. You do not need to edit any of the default parameters.
Loop	Creates a standard loop, allowing repeated execution of a transaction or series of transactions. The scriptstep connected with the green arrow is run each time the loop is executed. After the loop count has reached the specified limit, the scriptstep connected with the red arrow is run. This branching is the exact opposite behavior to how loops were branched in previous versions of Windows-RT. **Key Parameter:** ◆ **Loop Count:** Specifies the number of times the loop is executed. Default is 1.

Verification Category

These scriptsteps force the script to wait for specific criteria, such as the appearance of a particular graphic or text, before the script can continue.

Scriptstep Name	What It Does
Wait for Clipboard	Branches the script according to whether specified text is displayed in the Clipboard as a result of a copy or cut procedure. If the text is added to the Clipboard in the specified wait time, the script runs the scriptstep connected to the green arrow. If the Clipboard text is not found in the allotted time, the script runs the scriptstep connected by the red arrow. **Key Parameters:** ◆ **Text**: Specifies the text or variable that you want to wait to be added to the clipboard when the script is executed. ◆ **Wait Time**: Specifies the length of time the system should wait for the Clipboard text before timing out. Default is 30 seconds.
Wait for Graphic	Branches a script according to whether a specific graphic is displayed. If Windows-RT finds the image in the specified wait time, the script runs the scriptstep connected to the green arrow. If Windows-RT does not find the image in the allotted time, the script runs the scriptstep connected by the red arrow. **Key Parameters:** ◆ **Image**: Specifies the graphic you want to display, which can be added from a graphic file or from a screenshot. ◆ **Search Area**: The screen coordinates for the location of the graphic. To set the coordinates, you can click a specific location, use a screen shot to determine the graphic location, or use X and Y coordinates. Default position is Undefined. ◆ **Wait Time**: Specifies the length of time you want to wait for the graphic. Default is 30 seconds.
Wait for Graphic disappear	Specifies how long to wait for a specific graphic to go away, such as a splash screen or a message box. You can connect this scriptstep with a scriptstep that starts when the graphic disappears, with the green arrow, and you can use the red arrow to connect this scriptstep with a scriptstep that starts if the graphic does not disappear in the specified wait time. **Key Parameters**: ◆ **Image**: Specifies the graphic you want to use. The graphic can be added from a file or from a screenshot. ◆ **Search Area**: The screen coordinates for the location of the graphic. To set the coordinates, you can click a specific location, use a screen shot to determine the graphic location, or use X and Y coordinates. Default position is Undefined. ◆ **Wait Time**: Specifies the length of time you want to wait for the graphic to go away. Default is 30 seconds.

Scriptstep Name	What It Does
Wait for Text	Branches a script according to whether a set of specified text is displayed as a result of an OCR scan. If the text is found in the specified wait time, the script runs the scriptstep connected to the green arrow. If the text is not found in the allotted time, the script runs the scriptstep connected by the red arrow. **Key Parameters:** ◆ **Region**: Specifies the OCR location of the text. To set the coordinates, you can click a specific location, use a screen shot to determine the text location, or use X and Y coordinates. Default position is Undefined. ◆ **Target Text**: Specifies the text you want to wait to display. ◆ **Wait Time**: Specifies the length of time you want to wait for the text to display. Default is 30 seconds. ◆ **Method**: Specifies the method of analyzing the OCR scan, removing everything but the requested kinds of characters, like text or numbers. You can select Raw to not change anything in the original scan. Default is Raw. ◆ **White Space**: Specifies how the white, or empty, space in the OCR scan should be used. The white space can be kept, compressed, or removed. Default is Keep.

Windows Category

These scriptsteps insert operations that control the behavior of the agent computer's desktop, windows, and processes.

NOTE: Many of the scriptsteps in this category use the window handle, a unique value that Windows assigns to a window each time it is created. The handle is contained inside the code of a window and is not visible to the user.

Scriptstep Name	What It Does
Close Window	Closes a particular window as part of the script process. This scriptstep usually refers to the window opened by the Wait for Window scriptstep. Tip: If an aspect of the open application has been changed, you can send a Yes or No using the Keys scriptstep to save or not save that application item (such as a text file), or you can specify the Save dialog box graphic with the Wait for Graphic scriptstep and run the Mouse Click scriptstep to click the Yes or No button. Be sure to place these scriptsteps before the Close Window scriptstep. **Key Parameters:** ◆ **Close All:** Closes all open windows. ◆ **Target Window:** Specifies the target window you want to close. Use the Parameter Browser dialog box to find the target window parameter instead of using the Pick-a-Window or Window Browser feature.

Scriptstep Name	What It Does
Find Window	Searches all windows for the target window and stores its window handle in the specified variable. Some applications have dynamic names for their windows, so you can use a wildcard character, such as “* - Notepad” to find Notepad windows. If multiple windows are found, the first window to be found is stored. **Key Parameter**: **Target Window:** Specifies the target window you want to locate. You can type the Window Class Name and Window Caption for the target window, use the Window Browser, or use the Pick-a-Window option to help you identify the target window. The application must be open before you can see it in the Window Browser dialog box.
Move Window	Moves the target window to a specified location. **Key Parameters**: ◆ **Enable Resize**: Lets you resize the window when you move it. ◆ **Position**: The X and Y coordinates, for the location of this window on the screen. The Default position of the window is X=0 and Y=0. ◆ **Size:** Specifies the width and height of the window. Default is 640 by 480. ◆ **Target Window:** Specifies the target window you want to move. You can type the Window Class Name and Window Caption for the target window, use the Window Browser, or use the Pick-a-Window option to help you identify the target window. The application must be open to see it in Window Browser dialog box.
Set Window State	Specifies the state of an open window: default, maximized, minimized, restore, minimize (forced), show, or hide. **Key Parameters**: ◆ **Target State:** Specifies the state of that window. Default is Maximized. ◆ **Target Window:** Specifies the target window you want to access. You can type the Window Class Name and Window Caption for the target window, use the Window Browser, or use the Pick-a-Window option to help you identify the target window. The application must be open to see it in Window Browser dialog box.
Show Desktop	Minimizes all windows and displays the desktop so that other scriptsteps can utilize it.
System Information	Gathers system information for the specified system and makes that information available in a series of output parameters that can be used as variables by another scriptstep. These output parameters include current date, current time, machine name, operating system, memory information, screen resolution, and more.

Scriptstep Name	What It Does
Wait for Window	Specifies a particular window to open, and pauses the script until that window has been opened. When the window is found, the script runs the scriptstep connected by the green arrow. If the window is not found in the specified wait time, the script runs the scriptstep connected by the red arrow. **Key Parameters**: ◆ **Target Window:** Specifies the target window you want to monitor. You can type the Window Class Name and Window Caption for the target window, use the Window Browser, or use the Pick-a-Window wizard to identify the target window. The application must be open to see it in the Window Browser dialog box and the Pick-a-Window wizard. ◆ **Wait Time:** Specifies the length of time you want to wait for the window to open. Default is 30 seconds.
Wait for Window Exit	Specifies a particular window to close, and pauses the script for the specified wait time. When the window closes, the script runs the scriptstep connected by the green arrow. If the window does not close in the specified wait time, the script runs the scriptstep connected by the red arrow. **Key Parameters**: ◆ **Target Window:** Specifies the target window you want to monitor. You can type the Window Class Name and Window Caption for the target window, use the Window Browser, or use the Pick-a-Window option to help you identify the target window. The application must be open to see it in the Window Browser dialog box. ◆ **Wait Time:** Specifies the length of time you want to pause the script while waiting for the window to display. Default is 30 seconds. ◆ **Wait For All Windows:** Makes the script wait for all processes to finish before continuing.
Window to Front	Searches all windows for the specified window title and brings up all found windows. You can use an asterisk (*) as a wildcard character to help with your search. All found windows are placed in the foreground. The order of the windows is not predictable. **TIP:** One possible use of this scriptstep is to make the Target Window the same window found in the Wait for Window scriptstep. **Key Parameter**: **Target Window:** Specifies the target window you want to bring to the front. You can type the Window Class Name and Window Caption for the target window, use the Window Browser, or use the Pick-a-Window option to help you identify the target window. The application must be open to see it in the Window Browser dialog box.

Project Modules Category

These scriptsteps enable you to drag and drop a module script into other scripts. A module script can include a set of commonly used scriptsteps which you have used in other scripts and projects, and embedding a module script can save you time reproducing those scriptsteps all over again.

The **Project Modules** tab is displayed only when you create a new module script. After you embed a module script into a main script, you can double-click the module script to edit it.

Scriptstep Name	What It Does
ScriptName	Contains one or more module scripts that can be dragged into another script in the Script pane, embedding that module script into that selected script.

4.3 Setting up the OCR Scanning Tools

The **Test OCR** button on the **Tools** tab of the ribbon performs an Optical Character Recognition scan of a selected area and displays the results in the Test OCR dialog box.

To test OCR:

1. On the **Tools** tab of the ribbon, click **Test OCR**.
2. Select one of the following test options:
   - **Test OCR without Details**: To view the scan results as texts without any fonts details.
   - **Test OCR with Details**: To view the scan results with additional details on the **Output** tab and the **Statistics** tab.

     The **Output** tab displays the scanned characters with the font type details and the **Statistics** tab displays the list of fonts recognized by the scan and its percentage of accuracy.

     NOTE: The WinRT OCR engine uses non specific general OCR recognitions, so the font type returned by WinRT OCR might not always be accurate.
3. Move and resize the target box to specify the area of the OCR test.
4. Right-click and select **Save**. The relevant data is displayed in the Test OCR dialog box.

4.4 Updating the font database

The **Rebuild Font Database** button on the **Tools** tab allows you to update the Windows-RT OCR font database. If you have added or deleted any fonts on your system, you must rebuild the Windows-RT OCR font database.

To rebuild Windows-RT OCR font database, click **Rebuild Font Database** on the **Tools** tab.

4.5 Customizing Application Settings

You can customize the various settings for Window-RT to change the way files are saved, for example, as well as to edit the hotkeys used when recording scripts, among other options.

To access these settings, click **Application Settings** on the **Tools** tab of the ribbon, or click the **Windows-RT** application button in the upper left corner of the window, above the ribbon, and then click **Application Settings** on the menu that is displayed.

Editing Script Execution Settings

The **Script Execution** tab lets you edit warnings about invalid scripts, choose how scripts are saved, and edit the key or key combination used to interrupt script execution. You can edit the following options on this tab:

- **Warn When Executing Invalid Script:** Select this check box to receive a warning if a script contains at least one invalid step while that script is being executed. Default is checked.
- **Save Project Before Executing Script:** Select this check box to save the project file before executing a script to prevent any accidental loss of data. Default is checked.
- **Interrupt:** Select a key or key combination that you can use to interrupt script execution. Default is `CTRL+Q`.

Editing Hotkeys for the Wizard

The **Smart Recording Wizard** tab lets you choose a variety of key or key combinations that will affect the Smart Recording Wizard process. You can edit the following options on this tab:

- **Finish:** Select a key or key combination that you can use to stop the recording of the script and display the generated script. The default is **Escape**.
- **Menu:** Select a key or key combination that you can use to open a menu at any time that contains options for image or OCR verification. The default is `CTRL+F12`.
- **Pause:** Select a key or key combination that you can use to temporarily stop the recording of the script. To continue recording, you must press the Resume hotkey. The default is `CTRL+F10`.
- **Resume:** Select a key or key combination that you can use to resume script recording after it has been paused using the Pause hotkey. The default is `CTRL+F11`.
- **Screenshot:** Select a key or key combination that you can use to take a screenshot of the current screen during the recording process of the wizard. You can use this screenshot when defining a click location. If you take multiple screenshots in the wizard, you can move through the different screenshots by pressing the left or right arrow buttons, which enables you to choose from different versions of a particular screen to specify a search image. The default is `CTRL+F9`.
- **Show Message Before Recording Starts:** Select this check box to view a dialog box containing the hotkey settings before the Smart Recording Wizard begins. The default is checked.

Editing Project Settings

The **Projects** tab lets you control how your project files are saved as well as determine how the Windows-RT Designers behaves on startup. You can check or uncheck the following options on this tab:

- **Use File Compression:** Select this check box if you want to use file compression to decrease the size of the QML file that results when you export or check in a script. The default is checked.
- **Open Last Project File:** Select this check box to open the project you were last working on the next time you launch the Designer. The default is unchecked.
- **Open Empty Project:** Select this check box to open a new project with an empty script the next time you launch the Designer. The default is checked.
- **Show Compatibility Scriptsteps:** Select this check box to add the Compatibility category to the Scriptstep Library pane so you can access scriptsteps that have been upgraded from a previous version of Windows-RT. The default is unchecked.

Changing Flowchart Settings

The **Flowchart** tab lets you customize the appearance of the scriptstep elements that display on the Script pane. You can select or clear the following options on this tab:

- **Draw Direct Lines:** Select this check box to draw straight lines between the steps of a flowchart. The default is checked.
- **Draw Shadows:** Select this check box to include shadows for the steps and arrows on the flowchart. The default is checked.
- **Draw Grid:** Select this check box to include a grid on the background of the flowchart, behind the steps and arrows. The default is checked.

4.6 Customizing Windows-RT Templates

The QML Template Editor comes with a default template, but you can modify the default template to create new templates that you can use when exporting a script. The benefit of this customizing is that you can set specific values in the template, instead of recreating those settings as a module in the script.

To open the QML Template Editor:

1. On the **Tools** tab of the ribbon, click **QML Template Editor**.
2. Click **Open Template** and select the template you want to modify or use as the basis for a new template. Templates are located in the `Designer/Templates` directory.
3. On each of the tabs, you can edit the QML code to customize the template and create new parameters.
4. When you are done editing the default template, click **Save As** to save it as a new template to the `Designer/Templates` directory.
5. To return to the Windows-RT Designer, click **Close Template Editor**.

5 Working with Scripts

This chapter describes how to create, edit, save, import, and export scripts using the Windows-RT Designer. You can create scripts manually or with the wizard.

5.1 Creating a Script Manually

This section describes how to create a script using the Scriptstep Library pane and the Script pane of the Windows-RT Designer.

You can create a wide range of scripts in the Designer, but for the purposes of this example, assume you want to measure the total response time of a user transaction and generate an event as needed. In this example, you will track how long it takes to open Microsoft Outlook, generating an event if Outlook does not open in the specified time.

NOTE: Before you can create a new script, you must first create a new project, which then is displayed in the Project Files pane with a Scripts folder and a blank Script file. A project can contain multiple scripts. Windows-RT project files are stored in XML format.

To manually create a Windows-RT script:

1. Open the Windows-RT Designer.
2. Select a project to contain your new script by completing one of the following steps:
   - To create a new project, click **New** on the **Start** tab and select **Create New Project**.
   - To load an existing project and its scripts, click **Open Project**.
3. Select a script type by completing one of the following steps:
   - To create a new script, click **New** and select **Create New Script**.
   - To create a new module script, click **New** and select **Create New Module Script**.

   NOTE: A module script is a set of scriptsteps that can be dragged and dropped from the Scriptstep Library pane into another script on the Script pane.

   ---

4. From the **Measurement** category of the Scriptstep Library pane, drag a Start Timer scriptstep to the Script pane. This scriptstep allows you to monitor the response time for a scripts.

5 Without releasing the mouse button, hold the Start Timer scriptstep close to the Start scriptstep, until both scriptsteps are highlighted in green. The two highlighted scriptsteps will be connected in the flowchart when you release the mouse button.

6 Release the mouse button. The two scriptsteps in the flowchart are connected by an arrow.

NOTE: A scriptstep is not valid if the border for that scriptstep's icon is displayed as a broken red line. You must either add data to the Parameters pane for that scriptstep or ensure that the arrows going to and from the scriptstep connect with another scriptstep.

---

7 In the Parameters pane for the Start Timer scriptstep, type `Outlook Startup`.

8 From the Processes category of the Scriptstep Library pane, drag and drop a Start Process scriptstep to the Script pane, connecting it to the Start Timer scriptstep. This scriptstep launches an application or process.

9 In the Parameters pane of the Start Process scriptstep, use the File Name parameter to specify the application or process you want to monitor. In this example, the application is Microsoft Outlook.

NOTE: You can browse for the executable file by clicking the yellow folder icon, or you can choose from a list of commonly used folders by clicking the green icon, and then specifying the file name from that folder.

---

10 If needed, add any additional command-line arguments to the Arguments parameter for the Start Process scriptstep.

11 From the Windows category, drag and drop the Wait for Window scriptstep to the Script pane, connecting it to the Start Process scriptstep. This scriptstep specifies a particular window to open, and pauses the script until that window has been opened.

12 In the Parameters pane for the Wait for Window scriptstep, click **Undefined** on the **Target Window** parameter to select Outlook, the application you are waiting to open. You can select the application in one of the following ways:

- Click **Window Browser** and select the application from the Window Selection dialog box. The application must already be open for it to be displayed in this dialog box.
- Click the **Pick-A-Window** button to find the application you want to monitor with the Pick-A-Window wizard. The application must already be open for the wizard to work properly.
- Type the Window Class Name and Window Caption for Outlook, such as `rctrl_renwnd32` for the Window Class Name and `Inbox - Microsoft Outlook` for the Window Caption.

13 In the Parameters pane for the Wait for Window scriptstep, edit the Wait Time parameter to specify the maximum length of time you want to wait for the application to open. Use the default setting of 30 seconds.

NOTE: When the Outlook window opens successfully in the specified amount of time, the script runs the scriptstep connected by the green arrow. If the window is not opened in the specified wait time of 30 seconds, the script runs the scriptstep connected by the red arrow. For more information, see the Wait for Window entry in the "Windows Category" on page 49.

---

14 From the **Measurement** category of the Scriptstep Library pane, drag a Stop Timer scriptstep to the Script pane and connect it to the green (true) arrow of the Wait for Window scriptstep. This scriptstep stops the timer started earlier with the Start Timer parameter, and give you the elapsed time of the events in the script.

15 In the Parameters pane for the Stop Timer scriptstep, click **Undefined** in the Start Timer parameter and double-click Start Timer (Outlook Startup) from the Scriptstep Browser dialog box.

16 Drag and drop an Event scriptstep from the Notification category of the Scriptstep Library pane and connect it to the red (false) arrow of the Wait for Window scriptstep. This scriptstep creates an event in AppManager.

17 In the Parameters pane for the Event scriptstep, edit the text that will be displayed if the process failed. For example, you could edit the fields to contain the following messages:

Detail Message: `Application did not open in specified wait time.` **Severity**: `10` **Short Message**: `Application failed.`

For more information, see the Event entry in the "Notification Category" on page 43.

18 From the Script Flow category of the Scriptstep Library pane, drag and drop an End scriptstep to the Script pane, connecting it to the Stop Timer scriptstep. Connect the Event scriptstep to the same End scriptstep.

NOTE: All scripts must terminate with an End scriptstep. If you create branches in your script, you must terminate all paths with an End scriptstep. You can connect more than one arrow to a single End block.

---

19 If you want to clean up the display of the flowchart, click the **Auto Layout** button on the **Start** menu at any time. The flowchart is realigned in linear format

20 You can now check the new script into an AppManager repository for use with the Windows-RT module. For more information, see Section 5.4, "Checking Scripts into an AppManager Repository," on page 60.

5.2 Creating a Script Using the Wizard

The Create Script with Wizard function allows you to use the Smart Recording Wizard to capture and record a series of operations and use them in a script. These operations include regular mouse clicks, right-clicks, and double-clicks, as well as keystrokes and key combinations. After the wizard is complete, you can edit the resulting scriptsteps to in the Windows-RT Designer to refine your script as needed.

NOTE: The wizard replaces the action blocks and the **Record Actions** button used in previous versions of Windows-RT.

The following table lists the keys and key combinations that you can use during the Record Actions process:

Keystroke/Key Combination	Result
ESC	Ends the recording of the script and displays the recorded script in the Designer
CTRL+F9	Takes a screenshot during the recording that you can access later. Use the arrow buttons to navigate through the screenshots you have created.
CTRL+F10	Pauses the recording
CTRL+F11	Resumes the recording
CTRL+F12	Opens the **Recording** menu, which contains options for image or OCR verification that you can use to improve script recording

The key combinations in this table can be edited in the Application Settings dialog box. For more information, see "Editing Hotkeys for the Wizard" on page 53.

NOTE: If the wizard does not record a specific key combination, use the Keys scriptstep on the Input Simulation category of the Scriptstep Library pane. For more information, see "Input Simulation Category" on page 40.

For the purposes of this example, assume you want to measure the total response time of a user transaction and generate an event upon completion.

To create a script using the wizard:

1. Open the Windows-RT Designer.
2. Select a project to contain your new script by completing one of the following steps:
   - To create a new project, click **New** on the **Start** tab and select **Create New Project**.
   - To load an existing project and its scripts, click **Open Project**.
3. Select a script type by completing one of the following steps:
   - To create a new standard script with the wizard, click **New** and select **Create Script With Wizard**.
   - To create a new module script with the wizard, click **New** and select **Create Module Script With Wizard**.

   NOTE: A module script is a set of scriptsteps that can be dragged and dropped from the Scriptstep Library pane into another script on the Script pane.
4. On the first Smart Recording Wizard dialog box, click **OK** to begin.

   TIP: Select **Do not show this message again** to skip the initial dialog box the next time you create a script with the wizard.
5. If you want to measure the overall time the script takes to execute, select the Monitor overall script execution time check box and click **Next**. This adds a Start Timer scriptstep for the overall script execution time as well as its corresponding Stop Timer scriptstep to the script.
6. Select Start Windows Application Wizard and click **Next**.
7. In the File name text box, type the file name or click the folder icon to browse to the executable (`.exe`) file for the application you want to monitor.
8. If you need to type any command-line text to run the .exe file, type that text in the Arguments text box.
9. Click **Next**.
10. If you want to measure how long the application takes to start, select the Create Startup Time Measurement check box. This adds a Start Timer scriptstep for the application startup as well as its corresponding Stop Time scriptstep to the script.
11. Click **Finish**. The wizard launches the application you specified earlier. The wizard records your mouse clicks and keystrokes.
12. When you press a key or key combination, the Keyboard Input dialog box offers the following options:
    - Type a letter or a sentence into the Keyboard Input field.
    - Click the Key icon in the Keyboard Input field and select a key from the pop-up menu.

- To create a key combination such as `CTRL+o`, select one or more options in the Hold Key check boxes. Be sure to type any corresponding keys in the Keyboard Input field. Use lower-case for all letters in a key combination.
- If needed, type a title for the keyboard input as well as a description. This information is added to the Keys scriptstep that is created for this input.

13 When you click the left mouse button, click the right mouse button, or double-click the left mouse button, a pop-up menu offers the following options:

- **Edit Click**: Gives you the option of clicking on the exact, absolute screen coordinates for an image, clicking on the image, or creating a relative click with the image.
- **Record Simple Click**: Saves the mouse click.
- **Skip Event**: Ignores the mouse click. You can also ignore this click by simply moving the mouse again.

When you have specified your mouse click location, right-click and click **Save**.

NOTE: Only mouse clicks are recorded by the wizard, so if you want to record mouse movements, you will need to edit the script after finishing the wizard.

---

14 To cancel a mouse click, simply move the mouse away from its location at the time of the click. The pop-up menu will disappear.

15 At any time during the wizard, you can press `Ctrl+F12` to gain access to the following options:

- **Time menu**: Allows you to start or stop a time measurement.
- **Verification menu**: Allows you to verify contents of the Clipboard, an Image, an OCR scan, or the contents of a window.

16 Execute the transactions you want to record in that application for your script.

17 To stop recording, press the Esc button, or the hotkey you set up in the Application Settings. The Windows-RT Designer window is displayed, and the new script (or set of scripts) is displayed at the top of the list in the Script pane with default names (such as "`Script,`""`Script(2),`" etc.).

18 As needed, edit and rearrange the scriptsteps that were created by the wizard. For more information, see Section 5.1, "Creating a Script Manually," on page 55 or "Analyzing a Sample Script" on page 63.

19 If you want to clean up the display of the flowchart, click the **Auto Layout** button on the **Start** menu at any time. The flowchart is realigned in linear format

20 On the **Start** tab, click the **Save Project** button.

21 You can now check the new script into an AppManager repository for use with the Windows-RT module. For more information, see Section 5.4, "Checking Scripts into an AppManager Repository," on page 60.

5.3 Executing a Script

After you create a script, you should execute it to see how the script plays back for testing and debugging purposes. When you execute a script, Windows-RT minimizes the Designer window and runs the script file in the Project Files panes. Upon completion, Windows-RT reopens the Designer window. In the Traces pane, you can view output parameters and other information that results from running the script.

To select a script to execute, double-click that script in the Project Files pane and click **Execute Active Script** from the **Start** tab. If the script is a startup script, click **Execute Startup Script** from the **Start** tab.

TIP: To interrupt script execution, press `Ctrl+Q` or the key combination you set up in the Application Settings dialog box.

5.4 Checking Scripts into an AppManager Repository

After you finish creating a script, you can use the Windows-RT Designer to fine-tune it, or you can check it into the AppManager repository and start running jobs with it.

NOTE: You can use the **Check In Startup Script** and **Check In Active Script** buttons in the Windows-RT Designer only if the Designer is installed on a computer that also contains the AppManager Operator Console. Otherwise, save the script as a `.qml` file and copy it to a directory that can be accessed by the computer containing the Operator Console.

To check a script into the repository:

1. From the Windows-RT Designer, select the script and click **Check In Startup Script** or **Check In Active Script**.
2. If you used the QML Template Editor to create a new template, select a template for the script and click **Open**.
3. In the Knowledge Script Check-In Information dialog box, type a name for the new script in the **Name** field.

   NOTE: By default, the Knowledge Script is assigned the same name as the project filename, and the script folder is set to Win-RT7. With that prefix, the checked-in Knowledge Script will appear on the **Win-RT7** tab of AppManager.
4. Select **Overwrite existing Knowledge Script** to check in a new version of an existing Knowledge Script.

   NOTE: If you check in an updated version of a script for a job that is currently running in AppManager, you can update the job automatically. To do this, right-click on the script in AppManager select Properties propagation.
5. Complete the User logon fields to designate the repository where the Knowledge Script will be checked in and to grant Designer write access to it.
   - Type the name of the repository server in the **Server** field.
   - Type the full path to the repository server in the **Repository** field.
   - By default, SQL Server authentication is used. To use Windows NT authentication, select **Use Windows Integrated authentication**.
   - For SQL Server authentication, type the user **Name** and **Password**.
6. If desired, edit the **Script description** to change the text that is displayed in the Properties dialog box for this script.
7. Click **OK** to check in the Knowledge Script.

5.5 Exporting and Importing Scripts

If you want to use a script in more than one project, you must export that script from the original project, and then import it into the new project.

Exporting a Script

You can export a script in a variety of formats:

- As a Knowledge Script, or a `QML` file, which allows you to edit the file in XML without using the Windows-RT Designer
- As a Windows-RT script, which allows you to edit the file in the Windows-RT Designer

An exported script is not removed from the original project.

To export a script:

1. Open the project that contains the script you want to export.
2. In the Project Files pane, double-click the script you want to export.
3. On the **Start** menu, click **Export**.
4. Select the format for your export file by completing one of the following steps:
   - To export the script as a QML script, click **Export as Knowledge Script**.
   - To export the script as a Windows-RT script, click **Export as WinRT Script**.
5. If you used the QML Template Editor to create a new template, select a template for the script and click **Open**.
6. In the Export Script dialog box, specify a unique name and a directory location for the exported script and click **Save**.

NOTE: Encryption is not maintained if you export an encrypted script. A warning message is displayed if you attempt to export an encrypted script.

Importing a Script

You can import scripts as well as entire projects into your current project.

To import a script or project:

1. Open the project to which you want to import the script.
2. On the **Start** menu, click **Import**.
3. In the Import Script dialog box, browse to the location of the desired script file and double-click the file name.

NOTE: To import Windows-RT projects or Windows-RT scripts from version 7.1, click the relevant option from the Files of type list. For more information, see Section 3.2, “Activating the WinRT7 Display Theme,” on page 23

---

4. On the **Start** menu, click **Save Project**.

5.6 Printing a Script

To print the active script, click the Windows-RT application menu button and select **Print**, or press `Ctrl+P`. On the Print Preview dialog box you can set up the page before printing, change the scale of the printout, and make the layout of the printout portrait or landscape.

To set the size, source, orientation, and margins of your printout, and to select a specific printer, click the **Page Setup** button.

5.7 Using the Decision Scriptstep for Branching

The Decision scriptstep lets you branch a script according to criteria specified in the Decision Expression Editor dialog box.

For example, if you want to trigger separate AppManager events based on the value of a timer variable, such as a success event if the timer's value does not exceed the threshold, and a failure event if the timer's value is exceeded, you must create a decision tree.

To use a Decision scriptstep in a script:

1. From the Script Flow category of the Scriptstep Library pane, drag and drop the Decision scriptstep into the Script pane.

   NOTE: The Decision scriptstep has two connecting arrows attached. The green, or "true," arrow indicates the branch to follow if the specified condition is met, and should be connected to the next scriptstep in the "true" branch. The red, or "false," arrow indicates that the condition was not met, and should be connected to the next scriptstep in the "false" branch.

2. To set up the criteria for branching, click **Modify Expression** in the Decision Expression field of the Parameters pane.
3. In the Decision Expression Editor dialog box, click **Add Expression** to define an expression to use as your criteria for branching
4. In the Edit Expression dialog box, select the values you want to compare for your branching criteria. In the Left Value field, click **Click here to open the parameter browser**.
5. In the Parameter Browser dialog box, select the parameter that will be your first value from the **Scriptstep Parameters** tab or the **Script Parameters** tab and click **OK**.
6. In the Edit Expression dialog box, select the Comparison type, such as Greater Than, Less Than, or Equals. Each parameter type has its own Comparison types in this list.

   NOTE: To set up a negative comparison, select **Negation Modifier (NOT)**. Text values must be set to EQUALS.

7. In the Right Value text box, click the **Click here to open the parameter browser** link, select the parameter that will be your second value, and click **OK**.

   NOTE: If you want to use a specific value for the second value, select the Use Constant Value check box and type that value into the text box above the check box.

8. In the Edit Expression dialog box, click **OK**. The Expression is added to the Decision Expression Editor dialog box.
9. Add more expressions as needed.
10. If you want to add additional criteria to an existing expression, click **Add Sequence** and create the new expression criteria to add to the existing expression.
11. If you have more than one expression, you can change the Boolean state of the list of expressions by selecting an expression and clicking And or Or as needed.

    NOTE: You can also change the order of the expressions, by using the arrow buttons on the toolbar.

12 Click **OK** to save the settings and close the Decision Expression Editor dialog box. When this Decision scriptstep is executed, Windows-RT will determine if the criteria set up in the Decision Expression Editor have been met. If so, the condition is met and the script proceeds along the "True" branch to the next step, which is indicated by the green (true) connecting arrow. If not, the branch indicated by the red (false) connecting arrow is followed.

13 To set up the two branches, drag and drop the scriptstep for the green arrow, and drag and drop the scriptstep for the red arrow.

14 Be sure to connect all branches of your flowchart with an End scriptstep from the Scriptstep Library pane.

15 Edit and save your script.

5.8 Analyzing a Sample Script

This section examines a sample script that performs the following activities:

- Open an application (Notepad)
- Type text in the application and exit the application
- Verify that the application has been closed
- Create AppManager events for any process that doesn't run properly
- Track overall response time for the script

The following steps describe the creation of this sample script, which we will call Sample5_Win-RT7. This script is an updated version of the Sample5 Knowledge Script from previous versions of Windows-RT.

Step 1: Set up the Script Parameters

You can set up some global parameters for the new script by using the options on the Parameters pane.

To set up the script parameters:

1 In the Windows-RT Designer, click **New** and **Create New Project** from the **Start** tab of the ribbon. A new project containing a blank script is created.

2 In the Project Files pane, click the name of the script you just created. A list of global parameters for the script is displayed in the Parameters pane.

3 If you want to view a list of all parameters in the script, click the **Show All Parameters** button in the Parameters pane.

NOTE: The blue parameter icon identifies an input parameter that you can edit, the red parameter icon identifies an output parameter that contains data after the script executes, and the green parameter icon identifies a parameter that is both an input and output parameter.

---

4 Take note of the following script parameters in the Parameters pane:

- **Cleanup Processes:** Stops all processes that are still running when this script has been completely executed. Keep this option selected.
- **Cleanup Processes Exceptions**: Lets you list any processes, separated by a comma, that you do not want the Cleanup Process operation to stop. To view or edit a list of currently running processes, click the icon in the Value field. Leave this field blank.

NOTE: Setting the Cleanup Processes parameters ensure the computer is left in a stable state after the script executes.

5 In the Title parameter field, type `Sample5_Win-RT7` as the name of this script. The name is updated in the Project Files pane as well as on the tab in the Script pane.

Step 2: Start Timer for Response Time

The Start Timer scriptstep invokes a timer to track response time for the script. The timer value is stored in the Elapsed Time output parameter, and this variable returns a data stream.

To start a timer for response time:

1 From the Measurement category on the Scriptstep Library pane, drag a Start Timer scriptstep to the Script pane and connect it to the Start scriptstep.

2 In the Title parameter field for the Start Timer scriptstep, type `Script Response Time`.

TIP: In addition to measuring the total response time for a script, you can use the Start Timer scriptstep to measure how long it takes to open a program or run a process.

Step 3: Start the Notepad Process

The Start Process scriptstep runs a process from a file on the local system, usually an executable file that opens an application.

1 From the Processes category on the Scriptstep Library pane, drag a Start Process scriptstep to the Script panel and connect it to the Start Timer scriptstep.

2 On the Parameters pane for the Start Process scriptstep, type `Notepad` in the Text parameter field.

3 In the File Name parameter, click the yellow folder icon and navigate to the Notepad executable file. You can also navigate to a shortcut icon for the application, and the relevant executable file location will be added to the field.

4 If you want to create this script for yourself, launch Notepad and keep Notepad open as you set up the sample script.

Step 4: Locate the Notepad Window

The Wait for Window scriptstep is a branching scriptstep that checks to make sure the Notepad window opens in a specified time.

If the Notepad window before the specified time passes, the script goes to the scriptstep connected by the green, or true, arrow. If the Notepad window does not open in time, the script follows the red, or false, arrow.

To locate the Notepad window:

1 From the Windows category of the Scriptstep Library pane, drag a Wait for Window scriptstep to the Script pane and connect it to the Start Process scriptstep.

2 In the Parameters pane for the Wait for Window scriptstep, click **Undefined** in the Target Window parameter.

3 Use one of the following methods to select the Notepad window as your target window:
   - Type the Window Class Name (Notepad) and Window Caption (Untitled - Notepad) data in the Window Handle dialog box
   - Click Window Browser and select the Untitled - Notepad window from the list of running applications
   - Click Pick-a-Window to launch a wizard that lets you navigate through the various applications currently running on your computer until you find the Untitled - Notepad window. Click the Notepad title bar to add Notepad to the Target Window parameter.
4 In the Wait Time parameter, type `15` to decrease the maximum length of time the script will pause as Windows-RT tries to find the Notepad window.
5 If Windows-RT does not find the Notepad window in 15 seconds, the script runs the scriptstep connected by the red arrow. For more information, see "Step 5a: Notepad Window Not Found, Event Created" on page 65.
6 If Windows-RT finds the Notepad window in less than 15 seconds, the script runs the scriptstep connected by the green arrow. For more information, see "Step 5b: Notepad Window is Found, Text Typed" on page 66.

Step 5a: Notepad Window Not Found, Event Created

If Windows-RT does not find the Notepad window in the specified wait time of 15 seconds, the script runs the scriptstep connected by the red arrow. The scriptsteps in this "false" branch display a message box, create an AppManager event, stop the timer, and end the script.

To create scriptsteps for when Notepad is not found:

1 From the Debug category in the Scriptstep Library pane, drag a Message Box scriptstep to the Script pane and connect it to the red arrow from the Wait for Window scriptstep.
2 In the Parameters pane for the Message Box scriptstep, type the following text into the Message Text parameter field: `Could not find Notepad Window.`
3 Change the length of time the popup message is displayed by entering a `5` in the Duration parameter field.
4 From the Notification category in the Scriptstep Library pane, drag an Event scriptstep to the Script pane and connect it to the Message Box scriptstep.
5 In the Detail Message field, type `Event 1: Notepad failed to start`.
6 In the Short Message field, type `Notepad failed to start`.

NOTE: You can also edit the severity of the event and take a screenshot of the event.

---

7 From the Measurement category of the Scriptstep Library pane, drag a Stop Timer scriptstep to the Script pane and connect it to the Event scriptstep.
8 In the Parameters pane for the Stop Timer scriptstep, click **Undefined** in the Start Timer parameter field.
9 From the Scriptstep Browser dialog box, select the Start Timer variable and click **OK**.
10 From the Script Flow category of the Scriptstep Library, drag an End scriptstep to the Script pane and connect it to the Stop Timer scriptstep.

NOTE: Because the script uses the Cleanup Processes script parameter, Windows-RT stops any processes started but not closed by the script, and the script ends.

Step 5b: Notepad Window is Found, Text Typed

If Windows-RT finds the Notepad window, the script proceeds along the true branch, which is indicated by the green arrow. The scriptstep attached to the green arrow is a Keys scriptstep, which simulates the typing of text in the Notepad window.

To enter sample text:

1. From the Input Simulation category of the Scriptstep Library pane, drag a Keys scriptstep to the Script pane and connect it to the Wait for Window scriptstep.
2. In the Parameters pane for the Keys scriptstep, type the following text in the Input Text parameter: `This is sample text.`
3. Open Notepad and type the text from step 2 into the Untitled - Notepad window. Do not save or close the file.
4. The script continues in "Step 6: Close the Notepad File" on page 66.

Step 6: Close the Notepad File

The script uses a series of Mouse Click scriptsteps to close the Notepad file without saving it.

To close the Notepad file:

1. From the Input Simulation category of the Scriptstep Library pane, drag a Mouse Click scriptstep to the Script pane and connect it to the first Keys scriptstep in the true branch of the script.
2. In the Parameters pane for the Mouse Click scriptstep, click **Undefined** in the Position parameter.
3. Click **Specify Point** on the Point dialog box. The Designer window is minimized, and the Notepad window is displayed.
4. Drag the small target box that is displayed next to the Notepad window and position it on top of the Close button in the upper right-hand corner of the Notepad window.
5. Right-click to make the menu display and select Image Location.
6. Move and resize the larger rectangle that replaces the original target box until the rectangle fits exactly over the Close button on the Notepad dialog box.
7. Right-click again and click **Save**. The Windows-RT Designer window is displayed again, with the image location saved in the Position parameter.
8. In Notepad, click the **Close** button. A Notepad dialog box is displayed.
9. In the Windows-RT Designer, drag another Mouse Click scriptstep to the Script pane and connect it to the first Mouse Click scriptstep.
10. In the Parameters pane for the Mouse Click scriptstep, click Undefined in the Position parameter.
11. Click Specify Point on the Point dialog box.
12. Drag the small target box that is displayed next to the Notepad window and drop it on top of the **No** button on the Notepad dialog box.
13. Right-click again and click **Save**. The Windows-RT Designer window is displayed again, with the image location saved in the Position parameter.

Step 7: Wait for the Notepad Window to Close

After the script uses the Keys scriptstep and Mouse Click scriptstep to save the Notepad file, it uses the Wait for Window Exit scriptstep to close Notepad and verify the application closed properly.

To verify the Notepad window closes:

1. From the Windows category of the Scriptstep Library pane, drag a Wait for Window Exit scriptstep to the Script pane and connect it to the last Mouse Click scriptstep.
2. On the Parameters pane for the Wait for Windows Exit scriptstep, click the red and green Parameter Browser icon for the Target Window parameter.
3. Click the **ScriptStep Parameters** tab and select Wait for Window in the ScriptSteps with Output Parameters list.
4. In the Parameter list, select the Found Window variable, which should be set to the Notepad window, and click **OK**.
5. Leave the Wait Time parameter at its default setting of 30 seconds.
6. If the Notepad window does not close in 30 seconds, the script runs the scriptstep connected by the red arrow. For more information, see “Step 8a: Notepad Did Not Close in Time, Event Created” on page 67.
7. If the Notepad window closes within 30 seconds, the script runs the scriptstep connected by the green arrow. For more information, see “Step 8b: Notepad Closed in Time, Timer Stopped” on page 68.

Step 8a: Notepad Did Not Close in Time, Event Created

If Notepad did not close in 30 seconds, the script runs the scriptstep connected by the red arrow. The scriptsteps in this “false” branch display a message box, create an AppManager event, stop the timer, and end the script.

To create scriptsteps for when Notepad does not close:

1. From the Debug category in the Scriptstep Library pane, drag a Message Box scriptstep to the Script pane and connect it to the red arrow from the Wait for Window scriptstep.
2. In the Parameters pane for the Message Box scriptstep, type the following in the Message Text parameter field: `Notepad window did not close in time.`

   TIP: You can configure the length of time the message box is displayed by editing the Duration parameter.
3. From the Notification category in the Scriptstep Library pane, drag an Event scriptstep to the Script pane and connect it to the Message Box scriptstep.
4. In the Detail Message field, type `Event 2: Notepad failed to close.`
5. In the Short Message field, type `Notepad failed to close.`

   TIP: You can also edit the severity of the event and take a screenshot of the event.
6. From the Measurement category of the Scriptstep Library pane, drag a Stop Timer scriptstep to the Script pane and connect it to the Event scriptstep.
7. In the Parameters pane for the Stop Timer scriptstep, click **Undefined** in the Start Timer parameter field.

8 From the Scriptstep Browser dialog box, select the Start Timer variable and click **OK**.

9 From the Script Flow category of the Scriptstep Library, drag an End scriptstep to the Script pane and connect it to the Stop Timer scriptstep.

NOTE: Because the script uses the Cleanup Processes script parameter, Windows-RT stops any processes started but not closed by the script, and the script ends.

Step 8b: Notepad Closed in Time, Timer Stopped

If Notepad closed successfully, the script stops the timer and stores the overall script response time in the Measured Time variable for the Stop Timer scriptstep, which is returned as a data stream to AppManager. Response time is also stored in the Elapsed Time output parameter for the Start Timer scriptstep. Both variables return a data stream.

To stop the response time timer:

1 From the Measurement category of the Scriptstep Library pane, drag a Stop Timer scriptstep to the Script pane and connect it to the green arrow of the Wait for Window Exit scriptstep.

2 On the Parameters pane for the Stop Time scriptstep, click **Undefined** in the Start Timer parameter.

3 From the Scriptstep Browser dialog box, select Start Timer and click **OK**.

The script continues in "Step 9: Raise Event if Response Time is Too High" on page 68.

Step 9: Raise Event if Response Time is Too High

After the script stops the timer, the script uses a Decision scriptstep to establish a threshold event based on the response time, which is stored in the Elapsed Time variable for the Start Timer scriptstep.

In this script, if the total response time is greater than 15 seconds, AppManager raises an event.

To branch the script using the Decision scriptstep:

1 From the Script Flow category of the Scriptstep Library, drag the Decision scriptstep to the Script pane and connect it to the Stop Timer scriptstep.

2 In the Parameters pane for the Decision scriptstep, click **Modify Expression** in the Decision Expression parameter field.

3 In the Decision Expression Editor dialog box, click **Add Expression**.

4 In the Left Value field, click **Click here to open the parameter browser**.

5 In the Parameter Browser dialog box, click the **ScriptStep Parameters** tab.

6 Select Start Timer and Elapsed Time and click **OK**.

7 In the Comparison field of the Edit Expression dialog box, select > for greater than.

8 In the Right Value section, select Use Constant Value so you can type a number in the field instead of using a variable.

9 Type 15 in the **Right Value** field and click **OK**.

10 Click **OK** to close the Decision Expression Editor dialog box.

11 From the Notification category of the Scriptstep Library pane, drag an Event scriptstep to the Script pane and attach it to the green arrow of the Decision scriptstep.

12 In the Parameters pane for the Event scriptstep, type the following text in the Detail Message field: `Event 3: Transaction took too long to complete.`

13 In the Parameters pane for the Event scriptstep, type the following text in the Short Message field: `Transaction took too long to complete.`

NOTE: After the event is raised, the script moves to "Step 10: Check Availability and Report on Response Time" on page 69.

Step 10: Check Availability and Report on Response Time

If the total response time is less than 15 seconds, the script proceeds directly to the Availability scriptstep that stores the availability value. The Availability scriptstep also generates a data stream in AppManager. The Availability scriptstep connects to a Message Box scriptstep that displays the total script response time.

1 From the Measurement category, drag an Availability scriptstep to the Script pane and connect it to the red arrow of the Decision scriptstep as well as the blue arrow from the Event scriptstep.

2 In the Title parameter, type `Sample5_Win-RT7`.

3 After the script has run, verify that the Availability Value output parameter is set to 100 in the Parameters pane, indicating that the Notepad application was available and successfully accessed during this iteration of the script.

TIP: You can also find this information in the Traces pane.

---

4 From the Debug category of the Scriptstep Library, drag a Message Box scriptstep to the Script pane.

5 Type the following text in the Message Text parameter: `Transaction Successful. Overall Response Time= seconds.`

NOTE: Leave a blank space after "=" so you can insert the Elapsed Time variable in the message box in the next step.

---

6 Put the insertion point of the mouse to the right of the "=" in the Message Text parameter field.

7 Click the Parameter Browser icon in the Message Text parameter.

8 On the Parameter Browser dialog box, click the **ScriptStep Parameters** tab.

9 Select Start Timer and the Elapsed Time variable.

10 Click **OK**.

11 From the Script Flow category of the Scriptstep Library, drag an End scriptstep to the Script pane and connect it to the Stop Timer scriptstep.

6 Windows-RT Knowledge Scripts

The Windows-RT category provides the following Knowledge Scripts you can use with AppManager. From the Knowledge Script view of Control Center, you can access more information about any NetIQ-supported Knowledge Script by selecting it and clicking **Help**. In the Operator Console, click any Knowledge Script in the Knowledge Script pane and press `F1`.

Knowledge Script	What It Does
ChangeLocking	Lets you turn on or off the locking of a workstation so you can control access to that workstation.
ClosePlayer	Lets you shut down the Player that runs a script made by the Windows-RT Designer, such as when a running script is not working properly or a process is looping incorrectly.
TakeDesktopOwnership	Updates the Registry key indicating ownership of the desktop so that Windows-RT can run on the specified computer. Also lets you disable the Legal Notice screen and the CTRL + ALT + DELETE screen.

6.1 ChangeLocking

The ChangeLocking Knowledge Script allows you to turn on or off the locking of the keyboard and mouse used by a computer during the playback of a script. By default, when the Player executes a Windows-RT script on a system, Windows-RT locks all input from the mouse and keyboard for that system.

You can also change the default setting for the `InputLockOnStartup` setting in the `NetIQ.AppMan.WinRT7.Player.exe.config` file, located in the `NetIQ\AppManager\bin\Win-RT7\Player` directory.

Resource Object

WinRT7 Folder

Default Schedule

By default, this script runs once.

Setting Parameter Values

Set the following parameters as needed:

Parameter	How To Set It
General Settings	
Enable the ability to lock mouse and keyboard input?	Select this check box to prevent input from the mouse and keyboard of the specified computer during script playback.

Parameter	How To Set It
Raise event if the locking of the mouse and keyboard input has been successful?	Select this check box to raise an event when the input locking has been successful. The default is selected.
Event severity when the locking of the mouse and keyboard inputs has been completed successfully	Specify a severity level, from 1 to 40, to indicate the importance of the event related to discovery. Default is 35.
Debug	
Continue?	Select this check box to ignore any run time error and proceed with the Knowledge Script execution. The default is selected.
Debug?	Select this check box to enable debugging. The default is unselected. **NOTE:** Debug creates a *scriptname_jobid*_A.log file to log the error messages for a job.

6.2 ClosePlayer

The ClosePlayer Knowledge Script lets you shut down the Player that runs a script made by the Windows-RT Designer.

Resource Object

WinRT7 Folder

Default Schedule

By default, this script runs once.

Setting Parameter Values

Set the following parameters as needed:

Parameter	How To Set It
General Settings	
Raise event if player closed successfully?	Select this check box to raise an event when the script player closes successfully. The default is selected.
Event severity when player closed successfully	Specify a severity level, from 1 to 40, to indicate the importance of the event in which the script player closes. The default is 35.
Debug	
Continue?	Select this check box to ignore any run time error and proceed with the Knowledge Script execution. The default is selected.
Debug?	Select this check box to enable debugging. The default is unselected. **NOTE:** Debug creates a *scriptname_jobid*_A.log file to log the error messages for a job.

6.3 TakeDesktopOwnership

Use this Knowledge Script to check and update the shared registry value used to indicate which ResponseTime module is permitted to run its associated Knowledge Scripts. This Knowledge Script is also used to disable the Legal Notice screen and the CTRL + ALT + DELETE screen.

IMPORTANT: The Win-RT auto logon feature works properly when the Legal Notice screen and the CTRL + ALT + DELETE screen are disabled.

This script checks and updates the DisableCAD, LegalNoticeText, and LegalNoticeCaption registry values that are located under `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon`.

This script attempts to update the `Desktop` registry value (located under the `HKEY_LOCAL_MACHINE\SOFTWARE\NetIQ\Response Time` Registry key) to indicate that Windows-RT Knowledge Scripts are allowed to run. For more information, see Section 3.11, “Running Windows-RT and Web-RT Knowledge Scripts on the Same Computer,” on page 28.

IMPORTANT: Because this script does not contain an embedded Windows-RT script, you cannot modify it using the Windows-RT Designer.

Resource Object

WinRT7 Folder

Default Schedule

By default, this script runs once.

Setting Parameter Values

Set the following parameters as needed:

Parameter	How To Set It
General Settings	
Desktop Ownership Settings	
Event severity when take desktop ownership fails	Set the event severity level, from 1 to 40, to indicate the importance of an event in which an error prevents the script from setting the Registry value to indicate that Windows-RT has ownership of the desktop. The default is 5.
Raise event if take desktop ownership successful?	Select **Yes** to raise an event when the script updates the Registry value to indicate that Windows-RT has ownership of the desktop. The default is unselected.
Event severity when take desktop ownership successful?	Set the event severity level, from 1 to 40, to indicate the importance of an event in which the script updates the Registry value indicating Windows-RT has ownership of the desktop. The default is 35.
DisableCAD Settings	

Parameter	How To Set It
Disable CTRL + ALT + DELETE and Legal Notice screens?	Select **Yes** to disable the CTRL + ALT + DELETE screen and the Legal Notice screen that appear before you log in to an agent computer. The default is selected. **IMPORTANT:** This option disables only the local settings for the screens. To disable the group policies associated with CTRL + ALT + DELETE and the Legal Notice screen, contact your system administrator.
Event severity when the job failed to disable CTRL + ALT+ DELETE or Legal Notice screens	Set the event severity level from 1 to 40, to indicate the importance of an event in which the job failed to disable the CTRL + ALT+ DELETE screen or the Legal Notice screen. The default is 5.
Raise event if CTRL + ALT + DELETE and Legal Notice screens are disabled successfully?	Select **Yes** to raise an event if the CTRL + ALT+ DELETE screen and the Legal Notice screen are disabled successfully. The default is unselected.
Event severity when CTRL + ALT + DELETE and Legal Notice screens are disabled successfully	Set the event severity level from 1 to 40, to indicate the importance of an event in which the CTRL + ALT+ DELETE screen and the Legal Notice screen are disabled successfully. The default is 35.

7 Best Practices for Using Windows-RT

This chapter provides best practices as well as tips and troubleshooting advice for the Windows-RT module.

7.1 Improving Scripts

The following sections will help you create more effective scripts with the Windows-RT Designer.

Recommended Display and System Settings

The following tips describe the recommended display and system settings for computers running Windows-RT:

- Maintain uniform screen resolution and color quality on the agent computer. Changes in screen resolution impact the processing of all mouse actions, and if the color quality settings are not the same, the image search will not work properly. Recommended screen resolutions are 1024 by 768 or 1280 by 1024, but for best results, use the same resolution on all client computers. The recommended color quality is Medium (16 bit). For more information, see Section 3.2, “Activating the WinRT7 Display Theme,” on page 23.
- Use the WinRT display theme on the agent computer. This theme will enable you to get the best results with the Windows-RT OCR tools. The WinRT theme was installed when you installed the Windows-RT module.
- Disable font smoothing in Microsoft Office 2007. If you want to monitor Office 2007 applications, clear the **Always use ClearType** check box in the Options dialog box of the Office application. Some images with text might not be found in Office applications if ClearType is in use.
- Disable a ClearType text tuner for the Windows system by deselecting the **Turn on ClearType** check box in the computer.
- Use consistent color depth when saving bitmaps for image verification. Use 16-bit images to match the color quality settings. Images cannot be recognized if their color depth and color quality do not match, such as images with color gradients or images with many colors.
- Be aware of the difference between selected and unselected images. In some applications, bitmaps are different depending on whether the item is selected. The difference between a selected image and an unselected image might cause the image search to fail, or if an image is highlighted when the mouse moves over it. To address these situations, use the `Ctrl+F12` option in the wizard to take a screenshot of the image when it is not highlighted, and use that image for the image location. If you press `Ctrl+F12` more than once, you can view the different screenshots by using the left or right arrow keys.
- Before executing a script, ensure the agent computer is not running non-essential applications or processes. Use the Set Window State and Show Desktop scriptsteps to create the desired startup environment. In addition, always try to close all open applications when the script concludes. You can force application termination with the Cleanup Processes scriptstep.

Using the Keys Scriptstep with Windows Shortcuts

You can create a Windows shortcut key to launch most applications, and then invoke the shortcut key with a Keys scriptstep.

NOTE: Any letters you enter as part of a key combination should be lower-case, such as a “s” instead of “S” for Ctrl+s.

To specify a shortcut key for a Windows application:

1. Create a shortcut to the application and place it on the Windows desktop.
2. Right-click the shortcut and select **Properties**.
3. In the **Shortcut key** field on the **Shortcut** tab, press the shortcut key or keys you want to use to launch that application, and then click **OK**.
4. In the Windows-RT Designer, drag and drop a Keys scriptstep onto the Script pane.
5. Edit the parameters for the Keys scriptstep to give that scriptstep the relevant key combinations to run your new Windows shortcut, resulting in the launch of that application. For more information, see “Input Simulation Category” on page 40.

Monitoring Script Flow Performance

To improve script performance, always report unexpected behavior in the script flow with an error Event scriptstep. Examples of unexpected behavior include:

- Application does not open
- Login failed
- Login takes longer than allotted playback time
- Unexpected dialog or message box (such as error messages or login results) interrupts the script

7.2 Using Tracing and Logging to Improve Scripts

For information about debugging problems, refer to the following topics.

Install Log Files

The following files contain data about installation:

- `Maint.log` and `maint.err` in `\NetIQ\Temp\NetIQ_debug`
- `WinRTSetupExtraWrapper_Install.log`

  This is the log of the post installer. It is invoked by the maintenance installer to launch the MSI for additional setup. This file is located in `\Netiq\temp\netiq_debug\`<*computer*>
- `WinRTSetupExtra_Install.log`

  This is the log for the MSI that performs additional setup. It is located in `\Netiq\temp\netiq_debug\`<*computer*>

Editor Log Files

Any problems encountered while using the Windows-RT Designer are written to the `WinRTEditor.log` file located in `\Netiq\temp\netiq_debug\<computer>`

Agent Playback Log Files

The following files contain data about tracing the Windows-RT managed object, service, and Player:

- Tracing of the Windows-RT managed object (`qWinRT7.dll`) is handled by the Knowledge Script parameter *Raise event with managed object trace results?* If this parameter is set to **Yes**, an event is raised that contains the trace entries from the modules.
- Tracing of the Windows-RT service is enabled by default. The trace file is written to `\Netiq\AppManager\bin\Win-RT7\Service\Traces\`*GUID*`.wrttrace.log.` This log file wraps at 10M, and a backup is kept: `wrttrace.log.bkup`.
- Tracing of the Windows-RT Player is enabled by default. This trace file is written to `\Netiq\AppManager\bin\Win-RT7\Service\Traces\`*GUID*`.wrttrace.log.`

7.3 Troubleshooting Windows-RT

This section describes issues you might encounter while using Windows-RT with AppManager and with the Windows-RT Designer.

Failure to Import Security Manager Login Credentials

If the installation process fails to import login credentials into AppManager Security Manager, you must manually configure the information in Security Manager. For more information, see Section 2.5, “Configuring Windows-RT Credentials in Security Manager,” on page 20.

Player Does Not Run Unless User is Logged On

The Win-RT7 Knowledge Scripts scripts do not work if the Player is not running. The Player does not run unless there is a user logged on to the computer on which the Player is running. If no user is logged on, an event is raised stating that the “Player is not started.”

Maintaining a physical server in an active, logged-in state is simple enough because you can walk up to the server, log in, and then lock it. The Player can run without incident.

However, if you installed the ResponseTime for Windows module on a virtual server, maintaining a logged-in-and-locked state is a bit more complicated. If you use Remote Desktop Protocol to log into the virtual computer, and then disconnect the remote session, the session is no longer active, even if you are still logged in. The Win-RT7 Knowledge Scripts will fail when they access the remote server, because the screen will be black, which prevents the scripts from performing their optical character recognition (OCR) functions.

A workaround is to set up the virtual server to automatically log on after it is rebooted, and ensure the Windows-RT Player is listed in the Startup folder. Then, use the virtual server’s console, such as VMware Infrastructure Client, to reboot the virtual server. Do not use RDP to access the virtual server. The operating system will automatically log in and the Windows-RT Player will start because it is listed in the Startup folder. The virtual server will be in an active, logged-in state that permits Win-RT7 Knowledge Scripts to run.

For more information about automatic logon, see the Microsoft Support Web site. The following link provides instructions for automating logon in Windows XP: http://support.microsoft.com/kb/315231.

To add the Windows-RT Player to the Startup folder:

1. In Windows Explorer, create a shortcut of the Windows-RT Player service in the `NetIQ\AppManager\bin\Win-RT7\Service` folder.
2. Drag the shortcut icon to the `\Startup` folder at `C:\Documents and Settings\All Users\Start Menu\Programs`.

Running Discovery Twice on the Same Computer

If you run the Discovery_Win-RT7 Knowledge Script on a computer a second time, and you have a different set of applications listed in the parameters, your previous set of applications will be removed. To avoid this situation, make sure you list all applications that you previously discovered and still want to monitor in the parameter of the Discovery script before you run discovery again. For more information, see Section 2.6, “Discovering Windows-RT Resources,” on page 20.

Scripts Fail to Execute if Player not Connected to Service

If you start a job and the Windows-RT Player cannot connect to the Windows-RT service, the managed object will request the job to play back the script. The script will not run, however, because there is no Player connected to execute the script. To avoid this situation, ensure the Player and the service are connected. For more information, see Section 3.10, “What Happens When You Run a Windows-RT Script,” on page 28.

Running Windows-RT in Conjunction with Web-RT

You can install AppManager ResponseTime for Windows and AppManager ResponseTime for Web, version 6.3 (and later), on a single computer. However, you cannot run Windows-RT Knowledge Scripts and Web Transaction scripts generated by the Web-RT Web Recorder extension simultaneously, because only one module can take ownership of the desktop at any given time. For more information, see Section 3.11, “Running Windows-RT and Web-RT Knowledge Scripts on the Same Computer,” on page 28.

Modified Windows Messages

Avoid products that modify the default behavior of Windows messages. Custom modifications of these message boxes may cause errors in the recording and playback of mouse actions and keystrokes. For more information, see Section 5.2, “Creating a Script Using the Wizard,” on page 57.

Knowledge script size increases after importing from the version 7.1 to a later version

When you import the Knowledge scripts from Windows-RT version 7.1 to a later version, the sizes of the Knowledge scripts increase significantly. To avoid this situation, install Microsoft .NET Framework 4.5 before you start importing the Knowledge Scripts.

Knowledge Script fails to auto login

If the Knowledge Script fails to auto login, the CTRL+ ALT+ DELETE screen and Legal notice screens might be enabled in your machine. Do the following to disable the screens:

- To disable the local system settings for the CTRL+ ALT+ DELETE screen and Legal notice screen, run the TakeDesktopOwnership Knowledge Script.
- To disable the group policies associated with CTRL + ALT + DELETE and the Legal Notice screens, contact your system administrator.

Also, if you use a Knowledge Script, which was created with a previous version of Windows-RT Designer, the Knowledge Scrip fails to auto login.

To avoid this failure, do the following:

1. Open the Windows-RT Designer.
2. Import the old Knowledge Script and then export it using the Windows- RT Designer. For information about exporting and importing Knowledge Script, refer to Section 5.5, “Exporting and Importing Scripts,” on page 60.
3. Run the exported Knowledge Script.