Avaya Interaction Center®

Release 6.1

Agent Script Workflow Reference

585-248-208
Issue 1.0
August 2003

Notice

While reasonable efforts were made to ensure that the information in this document was complete and accurate at the time of printing, Avaya Inc. can assume no liability for any errors. Changes and corrections to the information in this document may be incorporated in future releases.

Preventing toll fraud

"Toll fraud" is the unauthorized use of your telecommunications system by an unauthorized party (for example, anyone who is not a corporate employee, agent, subcontractor, or person working on your company's behalf). Be aware that there may be a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya fraud intervention

If you suspect that you are being victimized by toll fraud and you need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1-800-643-2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Web site:

http://www.avaya.com

Select **Support**, then select **Escalation Lists**. This Web site includes telephone numbers for escalation within the United States. For escalation telephone numbers outside the United States, select **Global Escalation List**.

Providing telecommunications security

Telecommunications security (of voice, data, and video communications) is the prevention of any type of intrusion to (that is, either unauthorized or malicious access to or use of) your company's telecommunications equipment by some party.

Your company's "telecommunications equipment" includes both this Avaya product and any other voice/data/video equipment that could be accessed via this Avaya product (that is, "networked equipment").

An "outside party" is anyone who is not a corporate employee, agent, subcontractor, or person working on your company's behalf. Whereas, a "malicious party" is anyone (including someone who may be otherwise authorized) who accesses your telecommunications equipment with either malicious or mischievous intent.

Such intrusions may be either to/through synchronous (time-multiplexed and/or circuit-based) or asynchronous (character-, message-, or packet-based) equipment or interfaces for reasons of:

- Use (of capabilities special to the accessed equipment)
- Theft (such as, of intellectual property, financial assets, or toll-facility access)
- Eavesdropping (privacy invasions to humans)
- Mischief (troubling, but apparently innocuous, tampering)
- Harm (such as harmful tampering, data loss or alteration, regardless of motive or intent)

Be aware that there may be a risk of unauthorized intrusions associated with your system and/or its networked equipment. Also realize that, if such an intrusion should occur, it could result in a variety of losses to your company (including, but not limited to, human and data privacy, intellectual property, material assets, financial resources, labor costs, and legal costs).

Your responsibility for your company's telecommunications security

The final responsibility for securing both this system and its networked equipment rests with you, an Avaya customer's system administrator, your telecommunications peers, and your managers. Base the fulfillment of your responsibility on acquired knowledge and resources from a variety of sources, including, but not limited to:

- Installation documents
- System administration documents
- Security documents
- Hardware-/software-based security tools
- Shared information between you and your peers
- Telecommunications security experts

To prevent intrusions to your telecommunications equipment, you and your peers should carefully program and configure:

- Your Avaya-provided telecommunications systems and their interfaces
- Your Avaya-provided software applications, as well as their underlying hardware/software platforms and interfaces
- Any other equipment networked to your Avaya products.

Warranty

Avaya Inc. provides a limited warranty on this product. Refer to your sales agreement to establish the terms of the limited warranty. In addition, Avaya's standard warranty language, as well as information regarding support for this product, while under warranty, is available through the following Web site:

http://www.avaya.com/support

Link disclaimer

Avaya Inc. is not responsible for the contents or reliability of any linked Web sites and does not necessarily endorse the products, services, or information described or offered within them. We cannot guarantee that these links will work all of the time and we have no control over the availability of the linked pages.

Trademarks

All trademarks identified by the ® or ™ are registered trademarks or trademarks, respectively, of Avaya Inc. All other trademarks are the property of their respective owners.

Avaya, Conversant, CustomerQ, Definity, DefinityOne, MultiVantage, Nabnasset, Quintus, and WebQ are registered trademarks or trademarks of Avaya Inc. in the United States or other countries or both.

Portions of Avaya Interaction Center include technology used under license as listed below, and are copyright of the respective companies and/or their licensors:

Internet Explorer, Windows, Visual Basic, and Visual Basic for Applications are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Siebel is a trademark of Siebel Systems, Inc.

Java and all Java-based trademarks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Ordering information: Avaya Publications Center

Voice: +1-207-866-6701
1-800-457-1764 (Toll-free, U.S. and Canada only)

Fax: +1-207-626-7269
1-800-457-1764 (Toll-free, U.S. and Canada only)

Write: Globalware Solutions
200 Ward Hill Avenue
Haverhill, MA 01835 USA
Attention: Avaya Account Manager

Web: http://www.avayadocs.com

E-mail: totalware@gwsmail.com

Order: Document No. 585-248-208, Issue 1.0
August 2003

Avaya support

Avaya provides a telephone number for you to use to report problems or to ask questions about your contact center. The support telephone number is 1-800-242-2121 in the United States. For additional support telephone numbers, see the Avaya Web site:

http://www.avaya.com

Select **Support**, then select **Escalation Lists**. This Web site includes telephone numbers for escalation within the United States. For escalation telephone numbers outside the United States, select **Global Escalation List**.

Avaya Training

Avaya provides training for Avaya Operational Analyst. For more information, contact Avaya University at:

Web site: http://www.avaya-learning.com/logon_form.asp
E-mail address: avaya.u.helpdesk@accenture.com
US telephone: 1-800-288-5327
Outside US telephone: +1-303-406-6089

Comments

To comment on this document, send e-mail to crminfodev@avaya.com.

Acknowledgment

This document was written by the CRM Information Development group.

Avaya Interaction Center®
Release 6.1
Agent Script Workflow Reference

Contents

Before You Begin

This section includes the following topics:

Typographical Conventions

This guide uses the following font conventions:

Font Type	Meaning
`command`	This font signifies commands, information that you enter into the computer, or information contained in a file on your computer.
`commandvariable`	This font indicates variables in a command string.
italics	This font is used to add emphasis to important words and for references to other chapter names and manual titles.
link	Blue underlined text in online documents indicates a hypertext jump to related information. To view the related material, click the blue underlined text.

Notes, Tips, and Cautions

Note:

A note calls attention to important information.

Important:

An important note calls attention to a situation that has the potential to cause serious inconvenience or other similar repercussions.

Tip:

A tip offers additional how-to advice.

CAUTION:

A caution points out actions that may lead to data loss or other serious problems.

Contacting Technical Support

If you are having trouble using Avaya software, you should:

1. Retry the action. Carefully follow the instructions in written or online documentation.
2. Check the documentation that came with your hardware for maintenance or hardware-related issues.
3. Note the sequence of events that led to the problem and the exact messages displayed. Have the Avaya documentation available.
4. If you continue to have a problem, contact Avaya Technical Support by:
   - Logging in to the Avaya Technical Support Web site http://www.avaya.com/support/qq
   - Calling or faxing one of the following numbers from 8:30 a.m. to 8:30 p.m. (Eastern Standard Time), Monday through Friday (excluding holidays):
     - Toll free in the U.S. and Canada: 1-888-TECH-SPT (1-888-832-4778)
     - Direct line for international and domestic calls: 1-512-425-2201
     - Direct line for faxes: 1-512-997-4330
   - Sending email with your question or problem to crmsupport@avaya.com. You may be asked to email one or more files to Technical Support for analysis of your application and its environment.

Note:

If you have difficulty reaching Avaya Technical Support through the above URL or email address, please go to http://www.avaya.com for further information.

Product Documentation

Most Avaya product documentation is available in both printed and online form. However, some reference material is available only online, and certain information is available only in printed form. A PDF document with detailed information about all of the documentation for the Avaya Interaction Center is included in the `Doc` directory on the product CD-ROM. This PDF document is also included on the separate documentation CD-ROM.

Readme File

The Readme file is a PDF file included on the Avaya Interaction Center software CD-ROM. This file contains important information that was collected too late for inclusion in the printed documentation. The Readme file can include installation instructions, system requirements, information on new product features and enhancements, suggested work-arounds to known problems, and other information critical to successfully installing and using your Avaya software. Avaya may also deliver an Addendum to the Readme, which will be posted on the Avaya Technical Support Website. The Readme Addendum will contain similar information uncovered after the manufacture of the product CD-ROM. Review the Readme file and the Readme Addendum before you install your new Avaya software.

Electronic Documentation

The electronic documentation (in PDF or HTML format) for each Avaya Interaction Center product is installed automatically with the program. Electronic documentation for the entire Avaya product suite is included on the product CD-ROM and the documentation CD-ROM.

You can also view the documentation set online at http://www.avayadocs.com.

Printed Documentation

You can purchase printed copies of these manuals separately. For details, see Ordering information: Avaya Publications Center on the back of this manual's title page.

License to Print the Electronic Documentation

Online copies of documentation are included on the CD-ROM that accompanies every software release. An Avaya customer who has licensed software (a "Licensee") is entitled to make this online documentation available on an internal network or "intranet" solely for the Licensee's use for internal business purposes. Licensees are granted the right to print the documentation corresponding to the software they have purchased solely for such purposes.

Right-To-Print License Terms

Documents must be printed "as-is" from the provided online versions. Making changes to documents is not permitted. Documents may be printed only by any employee or contractor of Licensee that has been given access to the online documentation versions solely for Licensee's internal business purposes and subject to all applicable license agreements with Avaya. Both online and printed versions of the documents may not be distributed outside of Licensee enterprise or used as part of commercial time-sharing, rental, outsourcing, or service bureau use, or to train persons other than Licensee's employees and contractors for Licensee's internal business purposes, unless previously agreed to in writing by Avaya. If Licensee reproduces copies of printed documents for Licensee's internal business purposes, then these copies should be marked "For internal use only within <Licensee> only." on the first page or cover (where <Licensee> is the name of Licensee). Licensee must fully and faithfully reproduce any proprietary notices contained in the documentation. The copyrights to all documentation provided by Avaya are owned by Avaya and its licensors. By printing any copy of online documentation Licensee indicates its acceptance of these terms and conditions. This license only governs terms and conditions of printing online documentation. Please reference the appropriate license agreement for terms and conditions applicable to any other use, reproduction, modification, distribution or display of Avaya software and documentation.

Educational Services

Avaya University provides excellent training courses on a variety of topics. For the latest course descriptions, schedules, and online registration, you can get in touch with us:

- Through the web at http://www.avaya-learning.com/logon_form.asp
- Over the telephone at 800-288-5327 (within the U.S.) +001 303-406-6089 (outside of the U.S.)
- Through email at Avaya.U.Helpdesk@accenture.com

Chapter 1: Introduction to Agent Script Workflows

With the Prompter palette in Workflow Designer, you can create agent script workflows that help to improve the quality of agent interaction with customers. Before working with agent script workflows, you should be familiar with the information contained in *Avaya Workflow Designer User Guide*.

To create an agent script workflow that uses database objects, you also need to be very familiar with the data model of your Avaya Business Application.

Note:

Certain Java scripts may not run if used with Prompter agent script workflows created in a previous release of Workflow Designer. To use a workflow created in a previous release, delete the Stop block, then add a new Stop block from the catalog.

Understanding Agent Script Workflows

Agents frequently refer to predetermined scripts to assist them in handling requests from customers, whether those contacts arrive by phone, email, web chat, or voice chat. When you implement Prompter in your contact center:

- Agents have immediate online access to agent scripts
- Agents can collect customer information through surveys and checklists
- Agents can automatically save the information to a database

Running an Agent Script Workflow

An agent script workflow is a three step process:

1. Start script:
   a. Avaya Agent calls the script from the database.
   b. The Prompting Engine inserts the required database information in the script.
   c. The first HTML page displays in the Prompter pane (in Avaya Agent or CallCenterQ).
2. Collect data:
   a. The agent steps through the script.
   b. Prompter displays the workflow pages.
   c. Depending on the answers to questions and other actions that the agent performs, Prompter accesses the database, displays and updates tables, and saves data.
3. End script:
   a. The agent completes all steps in the script.
   b. The script displays a final script page.
   c. If necessary, Prompter updates the database.
   d. Depending on the workflow design, the agent's browser can remain on the desktop and reset itself to re-start the agent script in preparation for a new customer.

Using an Agent Script Workflow

You use an agent script workflow to perform two primary types of activities:

- Prompting - Requires Prompter blocks from the Prompter palette. The script displays a set of survey questions or a qualification process. For example, the agent could ask the contact questions such as "Are you interested in today's special?" or "Are you a registered voter?" For more information, see Creating HTML Pages for Prompting Scripts on page 26.
- Application tasks - Requires App blocks from the Prompter palette. The script guides an agent through a series of steps to assist them with a task, such as taking an order from a customer or entering a complaint. For more information, see Creating HTML Pages for Application Scripts on page 39.

You can combine these activities and use both Prompter and App blocks in a workflow to assist agents with a combined set of tasks.

Both types of Prompter workflows typically require access to a database, so the agent can retrieve and save database information while following the script. You specify the name of the database network accessed by your workflow in the Start block.

You can create database networks (modules) with Database Designer. If you are unfamiliar with the networks in your database, consult your database administrator before you design a Prompter workflow.

Working with the Limitations in Agent Script Workflows

When creating Prompter agent script workflows, remember the following limitations:

- Limitations of Variables on page 13
- Limitations of Requests by Prompter Actions on page 14

Limitations of Variables

The Workflow server only transmits variables of type String, Long, and Float (specifically `SVstring`, `SVlong`, and `SVfloat`) when it pushes pages out in agent script workflows. This limitation prevents the Workflow server from flooding the agent's machines with large collections of useless data.

If you need to get a `SeqCouple` to a client machine, convert it to a string using `.tostring` and include that string in a global string variable. For more information on creating and using global variables, see.

Limitations of Requests by Prompter Actions

The following Prompter actions cannot request to branch to a block using a connection name that begins with a dot (`.`):

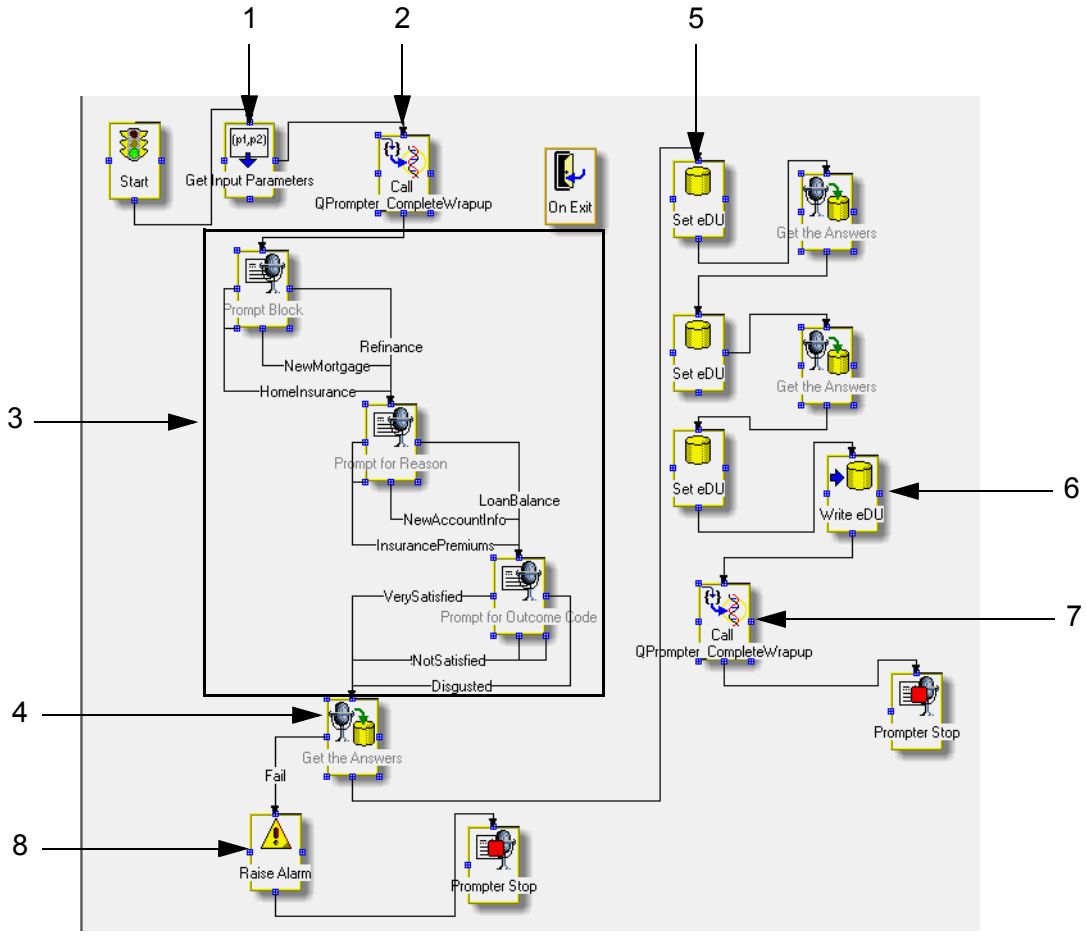
- `goto`
- `survey answer`

Sample Wrapup Workflow

Avaya IC installs the Sample Wrapup workflow in the following directory:

```
<install-dir>\IC61\design\IC\Flows\Avaya\Prompter\sample_wrapup.qfd
```

The Sample Wrapup workflow uses a banking or insurance model. The workflow includes a series of questions that an agent can ask a customer at the end of a voice contact or chat contact to enter information about the contact and the customer's reaction to the contact.



The Sample Wrapup workflow includes the steps described in the following table:

Step	Block Names	Description
	Start	Starts the workflow.
1	Get Input Parameters	Retrieves the following values: • EDUID • Task ID • Media type
2	Call Prompter Complete Wrapup	Creates a string representation of an IC Script call that runs after the workflow ends.
3	Prompter blocks	Displays a series of DHTML pages in Prompter for the agent to enter information about the contact, such as: • A category that relates to the customer contact, such as auto insurance or a new mortgage • A more specific reason for the customer contact, such as finding out information about insurance premiums or checking the current balance on a loan • An outcome code that describes the customer's opinion of the contact, such as Very Satisfied or Not Satisfied
4	Get the Answers	Returns the answer that an agent enters for the question specified on the Basic tab as a string. If the answer is returned, proceeds to Step 5, Set EDU. If the answer is not returned, proceeds to Step 8, Raise Alarm. You must include one Get the Answers block for each Prompter block. You need to follow each Get the Answers block with a Set EDU block.
5	Set EDU	Sets an EDU element to the value returned in the related Get the Answers block, then assigns these values in a script variable. You must include one Set EDU block for each Get the Answers block.
6	Write EDU	Writes the values assigned to the script variable by the Set EDU blocks as a sequence of couples to the EDU for the current contact. You need only one Write EDU block in the workflow, even if you have multiple Set EDU blocks.
7	Call Prompter Complete Wrapup	Creates a string representation of an IC Script call that runs after the workflow ends.

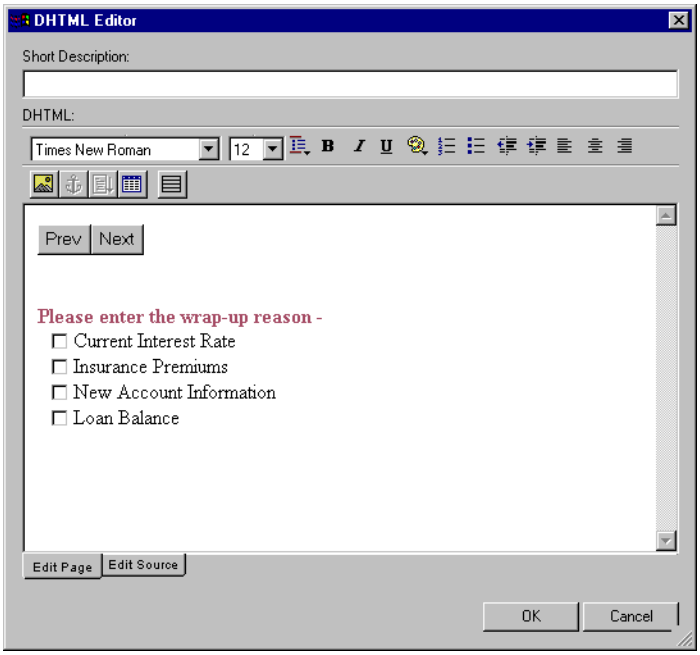
Step	Block Names	Description
8	Raise Alarm	Raises an alarm in the Alarm server. The alarm contains the information that you specify in the following properties on the **Basic** tab: ● AlarmName - Name of the alarm, such as *Prompter Wrapup* ● Description - Text to be included in the alarm, such as *Could not get answers to survey* ● Priority - Priority to be given to the alarm, such as *info* or *Emergency*
	Prompter Stop	Marks the exit point for the workflow. After the Prompter Stop block, the Workflow server executes the On Exit block.

Using the DHTML Editor

Whether you want to create an agent script workflow that prompts the agent to fill out fields and answer questions through the script or that finds and updates information in database fields, you use the DHTML Editor to create the workflow.

Prompter blocks and App blocks both contain one DHTML property, `Page`. The Page property is on the **Basic** tab. The Page property defines the HTML page that Prompter displays to the agent when the workflow is run. If you are unfamiliar with properties and with the basic principles of creating a workflow, see "Example: Creating a Simple Workflow" in *Avaya Workflow Designer User Guide*.

Create the HTML page in the DHTML Editor.



Even if you are proficient with HTML, do not use the Edit Source view in the DHTML Editor to work directly with the HTML source code. If HTML source code for a block contains an error, the workflow compiles, but the error only surfaces when the workflow is run. This results in very difficult troubleshooting to locate the error.

Note:

If you make changes directly in the HTML source code using the **Edit Source** tab, you must return to the **Edit Page** tab before you can select **OK**. If you select **Cancel** in the **Edit Source** tab, the DHTML Editor closes without saving and any changes are then lost.

To create an HTML page in the DHTML Editor:

1. Select a Prompter or App block in your workflow.
2. Select the **Basic** tab in the block's Property sheet to display the Page property.
3. Select the ellipsis (…) button next to the Page property to open the DHTML Editor.

   Note:

   The DHTML Editor cannot open if your workflow is not part of an open project.

4. In the **Short Description** text box, enter a description for the HTML page.

   This description is displayed on the **Basic** tab, so you can easily identify the HTML page.

5. Enter the questions and answers or database queries for the script in the main section of the DHTML Editor using one of the following methods:

   - Type the text in the main section of the **Edit Page** tab. When you create a new HTML page, the DHTML Editor inserts the cursor at the first available line. When you edit an existing page, the DHTML Editor inserts the cursor after the last line of text.
   - Use the Prompter Wizard to guide you through creating a prompting script. See Creating HTML Pages for Prompting Scripts on page 26 for more information.
   - Use the Database Wizard to guide you through creating an application script. See Creating Database Objects on page 40 for more information.

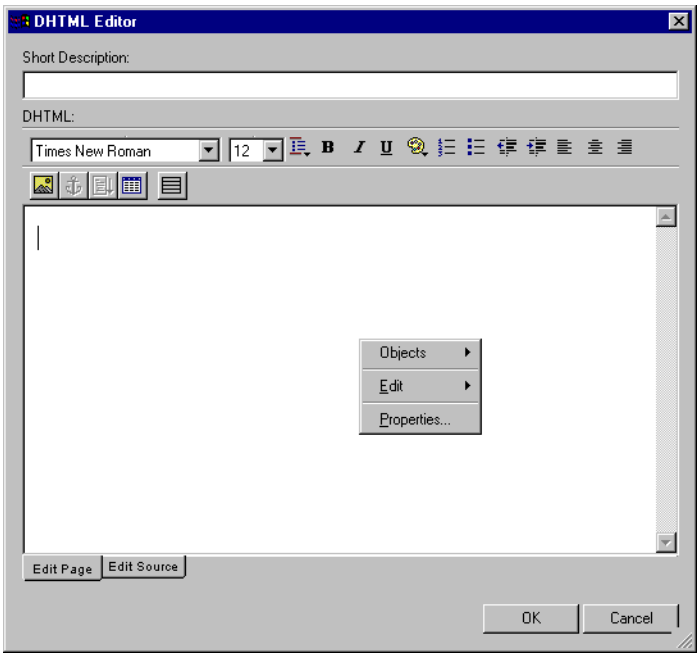
The DHTML Editor's right-click menu contains three options that you can use to create or modify objects on a page:

- **Objects** - Lets you create either a Prompter object (used in Prompting scripts) or a Database object (used in Application scripts).

  You can create a Prompter object or a Database object on a page. After you create one type of object, the other object type is no longer available for this page.

- **Edit** - Lets you add or delete rows in a table. This option is available only after you create a table.

- **Properties** - Displays the properties of the page in eXtensible Markup Language (XML) format.



You can delete, move, or change the information in the DHTML Editor with standard editing shortcuts and the toolbar icons. Use the toolbar icons to set the font size, weight, and style of the text, and to insert images, hyperlinks, and tables into the script.

You can also use the following standard keyboard shortcuts in the DHTML Editor:

- **Ctrl+X** - Cut
- **Ctrl+C** - Copy
- **Ctrl+V** - Paste
- **Ctrl+Z** - Undo
- **Ctrl+Y** - Redo

Use the Table Border button to display borders around table elements in the DHTML Editor.

Creating Agent Script Workflows

Each prompting script contains a series of Prompter blocks. Each Prompter block represents a single HTML page in the script. Depending on the purpose of the script, each HTML page can contain one question with answer fields, or one piece of information the agent needs to give the customer.

After you design the script and know the order of the questions and whether the answer given to a question determines which step comes next, you can create the script workflow in Workflow Designer.

Tip:

Before creating an agent script workflow, you must be familiar with the data model of your application, including the table sets in focus modules, database tables, and the fields in the tables. For more information, see the Data Model Reference for your application. If your application has been customized with Database Designer, see your application designer for information on situations of which you should be aware.

To create a Prompter agent script workflow, complete the steps in the following sections:

1. Creating a New Project on page 21
2. Adding the Blocks on page 21
3. Connecting the Blocks on page 22
4. Specifying Block Properties on page 23
5. Create the HTML pages with one of the following:
   - Creating HTML Pages for Prompting Scripts on page 26
   - Creating HTML Pages for Application Scripts on page 39
6. Saving Workflows on page 50

After you create the agent script workflow, perform the following procedures that are found in *Avaya Workflow Designer User Guide*:

1. "Verifying Workflows"
2. "Building a Workflow"
3. "Loading Workflow in the Workflow Server"

Creating a New Project

A project contains a set of related workflows and information about how those workflows are built and stored. Each Prompter agent script workflow must be part of a project. If you do not create a project for your script, you cannot open the DHTML Editor.

By default, Workflow Designer saves projects in the same folder as the workflows.

When you open a read-only project, Workflow Designer lets you make it writable. For example, if you maintain your project (`.qfd` and `.prj`) files in a source-code control system, the system frequently makes the files read-only. When you re-open the project, Workflow Designer lets you change the permissions so you can save any changes.

To create a project:

1. Select **Project** > **New**.
2. Enter the project name.

   Note:
   Project names cannot contain spaces.
3. Select **OK**.

   Your project displays in the **Project** bar.

Adding the Blocks

Each block represents a question in the agent script. You place the blocks in the same order that the questions occur in the agent script.

To create a new workflow:

1. Select **File** > **New Flow**.

   A new workflow is made available in the Work Area.
2. In the **General** palette:
   a. Select the Start block.
   b. Drag it to the Work Area.
   c. Drop the block in the Work Area.
3. Drag all needed Prompter, Application, or other blocks from the Prompter palette to the Work Area and drop them below the Start block in the order that the corresponding questions should occur in the workflow.

   For information about the standard Workflow Designer blocks and the associated properties, see "Standard Blocks" in *Avaya Workflow Designer User Guide*.

4. Drag one or more Stop blocks to the Work Area:
   a. If you used Prompter blocks, drop a Prompter Stop block after the last Prompter block in the workflow.
   b. If you used Application blocks, drop an App Stop block after the last Application block in the workflow.
   c. For all workflows, drop the Stop block from the General palette below the last block in the workflow.

Connecting the Blocks

A workflow must indicate the order in which the server executes the blocks. You set the order by using connectors between the blocks. The direction that you draw the connector indicates the order.

The blocks in your workflow have blue squares around their edges. These squares are snap points. They are also known as anchor snaps or connection points. Snap points indicate where you can connect one block to another block.

On the drawing toolbar, you can select one of the following types of connectors:

Tool Icon	Description
Connector	Connects blocks with straight lines and 90 degree bends. If you use this tool, Workflow Designer displays labels in the center of the connector.
Line	Connects blocks with diagonal or straight lines. This tool does not create lines with angles.

To connect the blocks in your workflow:

1. From the Drawing bar, select a connector tool.

   If the Drawing bar is not displayed, select **View** > **Toolbars** > **Drawing bar**.

2. With the connector tool selected:
   a. Select the blue snap point of the Start block
   b. Drag your mouse down to a blue snap point on the Demo block and release

   Insert the connector in the direction you want the workflow to progress. If you drag the connector up to a preceding block in the workflow, your workflow moves backwards.

Specifying Block Properties

The Property sheet contains the properties of a block. The four tabs in the Property sheet provide different information:

- **General** - Contains the description of the block
- **Basic** - Contains properties that must be set for each block, including the Page property for Prompter and Application blocks
- **Advanced** - Contains properties designed for use only by block designers and advanced workflow designers
- **Expert** - Contains information about the graphical aspects of the block, such as its color and shading, and information about the snap points that connect the block to other blocks

The properties in the tabs depend on the type of block. You need to review the properties in each tab and modify them if necessary. For more information about modifying block properties, see "Advanced Concepts" in *Avaya Workflow Designer User Guide*.

Each property has a property type. The property type is a specific type of information that can be stored in that property.

Some value properties can hold a literal text string such as `demoString`, or a symbol. A symbol is a definition of a variable that shares information between blocks. Advanced users can learn more about symbols in "Using Properties and Symbols" in *Avaya Workflow Designer User Guide*.

Required Block Properties

You need to specify the properties shown in the following table before you can create the HTML pages for your Prompter agent script workflow.

For this block	Specify this property
Start block	In the `DBNetwork` property on the **Basic** tab, enter the name of the focus in the Interaction Center application that contains the table set used by this script. For example, enter `q_emailanalysis` to specify that the workflow uses the `q_emailanalysis` focus of the Interaction Center application. You can find the Interaction Center application in the `ccq.adl` file.
	In the `ToolkitTimeout` property, if desired, change the default timeout of 30000 seconds (5 minutes). This default setting lets an agent wait for up to 5 minutes before navigating to the next page in a Prompter agent script workflow.

For this block	Specify this property
Prompter blocks	In the Page property on the **Basic** tab, create the HTML page for this block in the Prompter script. For more information, see Creating HTML Pages for Prompting Scripts on page 26.
Application blocks	In the Page property on the **Basic** tab, create the HTML page for this block in the Application script. For more information, see Creating HTML Pages for Application Scripts on page 39.

Saving Answers to the Database

By default, Prompting scripts save customer answers in the `qw_response` table. If you select another database table in the Prompter Script wizard while creating the questions for your prompting script, the workflow runs but does not save the answer in the database.

You can, however, save answers to any database table by including Add Record and Commit Record blocks from the Prompter palette in your Prompting script. For more information about these blocks, see "Prompter Palette" in *Avaya Workflow Designer User Guide*.

Note:

When answers are saved to the database, the questions are also saved. Weighted scores cannot be saved to the database.

To save answers in a database table other than `qw_response`:

1. Before the Prompter block where you want to save answers to the database, insert an Add Record block into your workflow.
2. Enter the name of the table where you want to save the answer in the `Table` property of the Add Record block.
3. Insert a Prompter block after the Add record block in your workflow.
4. Use the DHTML Editor to configure the question for this step in the Prompting script. See Configuring Prompter Questions on page 27 for more information. You must add the name of the table and table field where you want to save the answer.
5. Connect the Add Record block down to the Prompter block.
6. Insert a Commit Record block after the Prompter block in your workflow.
7. Enter the name of the table where you want to save the answer in the `Table` property of the Commit Record block.
8. Connect the Prompter block down to the Add Record block.

Creating HTML Pages for Prompting Scripts

A Prompting script is an agent script with a set of questions and answers. Agents can display prompting scripts in one of the following locations:

- Browser on the agent's desktop
- Prompter pane of Avaya Agent
- Prompter form in CallCenterQ applications

You create a Prompting script by including Prompter blocks in a workflow and using the Prompter Wizard in the DHTML Editor to create questions and answers in agent scripts. For more information, see Using the DHTML Editor on page 17.

Each prompting script contains a series of Prompter blocks. Each Prompter block represents a single HTML page in the script. Depending on the purpose of the script, each HTML page can contain one question with answer fields, or one piece of information the agent needs to give the customer.

After you design the script and know the order of the questions and whether the answer given to a question determines which step comes next, you can set the Page property for the Prompter blocks. For each step in the script, the HTML pages that form the Page property display one or more of the following:

- Information for the agent to give to the customer
- Questions for the agent to ask the customer
- If desired, a series of pre-determined answers for the customer to select

To create an HTML page for a Prompting script perform the steps in the following sections:

1. Setting the Block Text and Page Property on page 26.
2. Configuring Prompter Questions on page 27.
3. Creating HTML Pages for Application Scripts on page 39.

Note:

If you add text to agent script workflows with the DHTML Editor, an agent who views the workflow through a browser must use a supported browser. For information on supported browsers, see *IC Installation Planning and Prerequisites*.

Setting the Block Text and Page Property

The Page property is a short description in the **Basic** tab that identifies the HTML page. The block text is the title that displays on the block in the Work Area to identify the purpose of the block in the workflow.

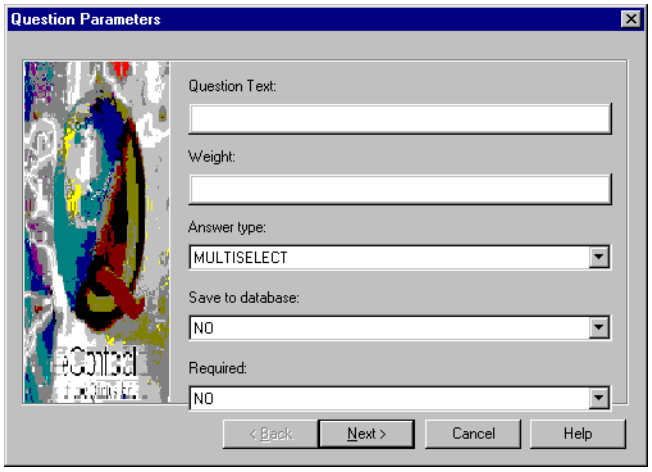
To set the block text and Page property for the prompting script page:

1. Select a Prompter block in your workflow.
2. Add the block text:
   a. Select the **Expert** tab in the Property sheet.
   b. Select **Text** and enter a description of the block's purpose. For example, Prompt for User Name.
3. Select the **Basic** tab in the block's Property sheet to display the Page property.
4. Select the ellipsis (...) button next to the Page property to bring up the DHTML Editor.

   Note:
   If your workflow is not part of an open project, Workflow Designer prompts you to add the workflow to a project before the DHTML Editor opens.
5. In the **Short Description** field, enter a description for the HTML page.

Configuring Prompter Questions

After you set the Page property, the Question Parameters dialog box is displayed.



How you specify the question parameters depends on which answer type you select for the question in the HTML page:

- Combo Box Answer Type - Agent selects an answer from a predetermined set of answers in a drop-down list
- Date Answer Type - Agent enters a date in a field
- Long Text Answer Type - Agent types the answer in a text box with no restriction on the length of the answer

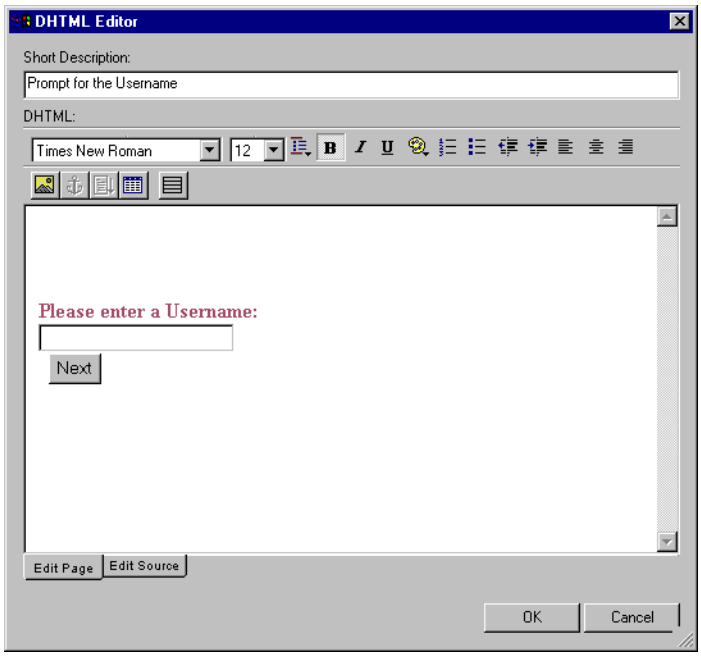
- Multiselect Answer Type - Agent selects one or more answers from a predetermined set of answers that are displayed with check boxes
- None Answer Type - Agent cannot enter an answer
- Radio Answer Type - Agent selects one answer from a predetermined set of answers that are displayed with radio buttons
- Text Box Answer Type - Agent types the answer in a text box restricted to a limited number of characters

If you create multiple questions for a Prompter block, the answers to only one question in the block can cause the workflow to branch to another block.

Note:

You must select a database connection to specify question parameters. Do not name the database connection `default`.

After you complete the question parameters, the DHTML Editor displays the questions and answers seen by the agent:



Note:

If you make changes directly in the HTML source code using the **Edit Source** tab, you must return to the **Edit Page** tab before you are allowed to select **OK** and save those changes.

Combo Box Answer Type

In combo box format, the answers display in a drop-down list. The agent can select only one answer from the list.

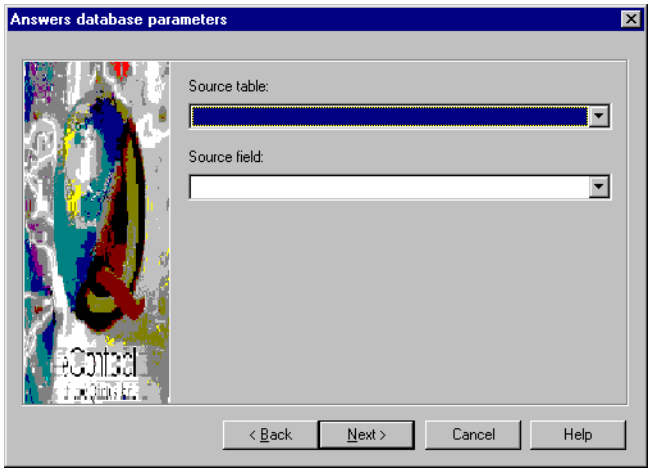
To create an HTML page with the answers in a combo box:

1. Right-click in the DHTML Editor and select **Objects** > **Prompter**.
2. In the **Question Text** field, enter the text of the question.
3. In the **Weight** field, enter the weight to be given to the answer to this question.

   A weight can be between 0 and 1, inclusive of all decimal values. The weight is a relative value that calculates the importance of a question to aid in analyzing customer responses.
4. From the **Answer type** drop-down list, select **Combo box**.
5. From the **Save to Database** drop-down list, select:
   - **Yes** - If you want the answer to be saved in the database when the workflow is completed
   - **No** - If you do not want the answer saved in the database
6. From the **Required** drop-down list, select:
   - **Yes** - To require the agent to enter an answer for the question before continuing the workflow
   - **No** - If the answer is optional
7. Select **Next**.
8. If you selected **Yes** in the **Save to Database** field:
   a. Select the database table where the workflow saves the answers.
   b. Select a field in the database table that matches the data type of the answer to be saved by the agent.
   c. Select whether to save the answer in the `qw_response` table.
   d. Select **Next**.

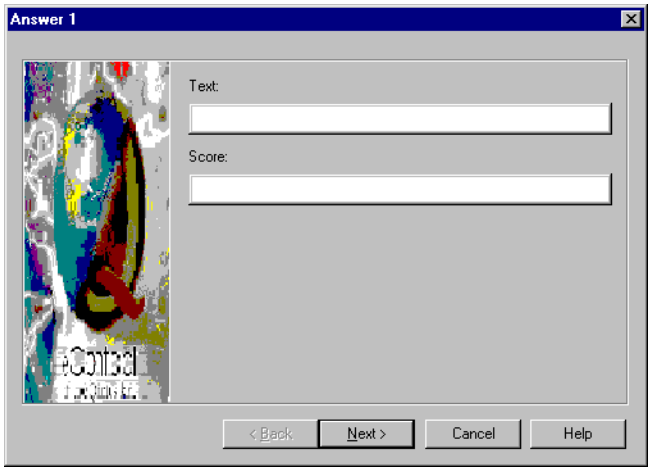
   If you do not want to store the answer in a database, Workflow Designer does not display this dialog box.
9. In the **Answers Datasource** dialog box, select the location where the workflow will save the answers:
   - Database (go to Step 10)
   - Text (go to Step 11).

10. If you selected **Database** in Step 9, the **Answers Database Parameters** dialog box is displayed. Select the following to identify where the workflow gets the answers:
    a. The name of the source database table from the **Source Table** drop-down list.
    b. The name of the database field in the selected table from the **Source Field** drop-down list.



11. If you selected **Text** in Step 9, enter the number of possible answers. Workflow Designer opens a series of dialog boxes. In those dialog boxes, repeated for each possible answer:
    a. In the first dialog box, enter the text of the first answer and its score.

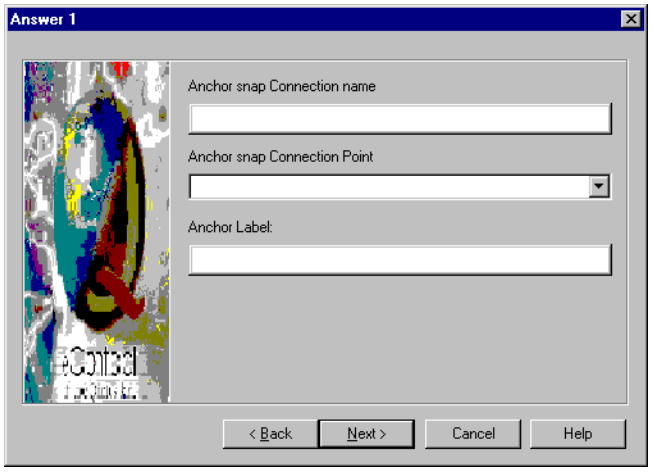
    A score is a number between 0 and 100. You can assign a different score for each answer. If you are using answers from a database, you cannot associate a score with your answers.



b. In the second dialog box, enter the following information to identify the snap point that takes an agent to the next block if the agent selects this answer:

- Anchor Snap Connection Name
- Anchor Snap Connection Point - Select an available anchor snap from the drop-down list
- Anchor label - Limited to four characters. This text is displayed on the specified anchor snap in your workflow

If you need to add more snaps to the block, see "Creating Snap Points" in *Workflow Designer User Guide*.



12. When you have completed the series of dialog boxes for each answer, select **Finish**.

    If you correctly completed the question and answer screens, you receive a message indicating success and Workflow Designer prompts you to select **Finish**.

Date Answer Type

The date answer type is used by an agent to answer a question by entering a date.

To create an HTML page with a date entry field:

1. Right-click in the DHTML Editor and select **Objects** > **Prompter**.
2. In the **Question Text** field, enter the text of the question.
3. In the **Weight** field, enter the weight to be given to the answer to this question.

   A weight can be between 0 and 1, inclusive of all decimal values. The weight is a relative value that calculates the importance of a question to aid in analyzing customer responses.
4. From the **Answer type** drop-down list, select **Date**.

5. From the **Save to Database** drop-down list, select:
   - **Yes** - If you want the answer to be saved in the database when the workflow is completed
   - **No** - If you do not want the answer saved in the database
6. From the **Required** drop-down list, select:
   - **Yes** - To require the agent to enter an answer for the question before continuing the workflow
   - **No** - If the answer is optional
7. Select **Next**.
8. If you selected **Yes** in the **Save to Database** field, in the next dialog box:
   a. Select the database table where the workflow saves the answer.
   b. Select a field in the database table with data type of `date` or `date time` to match the data type of the answer the agent will save.
   c. Select whether to save the answer in the `qw_response` table.
   d. Select **Next**.

   If you do not want to store the answer in a database, Workflow Designer does not display this dialog box.
9. Select **Finish**.

Long Text Answer Type

The long text answer type opens a dialog box so that an agent can enter an answer with no restriction on the length.

To create an HTML page with a long text entry field:

1. Right-click in the DHTML Editor and select **Objects** > **Prompter**.
2. In the **Question Text** field, enter the text of the question.
3. In the **Weight** field, enter the weight to be given to the answer to this question.

   A weight can be between 0 and 1, inclusive of all decimal values. The weight is a relative value that calculates the importance of a question to aid in analyzing customer responses.
4. From the **Answer type** drop-down list, select **Long Text**.
5. From the **Save to Database** drop-down list, select:
   - **Yes** - If you want the answer to be saved in the database when the workflow is completed

- **No** - If you do not want the answer saved in the database

6. From the **Required** drop-down list, select:
   - **Yes** - To require the agent to enter an answer for the question before continuing the workflow
   - **No** - If the answer is optional
7. Select **Next**.
8. If you selected **Yes** in the **Save to Database** field, in the next dialog box:
   a. Select the database table where the workflow saves the answer.
   b. Select a field in the database table that matches the data type of the answer to be saved by the agent.
   c. Select whether to save the answer in the `qw_response` table.
   d. Select **Next**.

   If you do not want to store the answer in a database, Workflow Designer does not display this dialog box.
9. Select **Finish**.

Multiselect Answer Type

In the multiselect answer type, the agent can select one or more answers by selecting the associated check boxes.

To create an HTML page with the answers next to check boxes:

1. Right-click in the DHTML Editor and select **Objects** > **Prompter**.
2. In the **Question Text** field, enter the text of the question.
3. In the **Weight** field, enter the weight to be given to the answer to this question.

   A weight can be between 0 and 1, inclusive of all decimal values. The weight is a relative value that calculates the importance of a question to aid in analyzing customer responses.
4. From the **Answer type** drop-down list, select **Multiselect**.
5. From the **Save to Database** drop-down list, select:
   - **Yes** - If you want the answer to be saved in the database when the workflow is completed
   - **No** - If you do not want the answer saved in the database
6. From the **Required** drop-down list, select:

- **Yes** - Select this option if the agent must supply an answer before the workflow is allowed to continue.
- **No** - Select this option if answering this question is optional for the agent.

7. Select **Next**.
8. If you selected **Yes** in the **Save to Database** field, in the next dialog box:
   a. Select the database table where the workflow saves the answer.
   b. Select a field in the database table that matches the data type of the answer to be saved by the agent, for example `varchar` or `text`.
   c. Select whether to save the answer in the `qw_response` table.
   d. Select **Next**.

   If you do not want to store the answer in a database, Workflow Designer does not display this dialog box.
9. In the **Answers Datasource** dialog box, select the location where the workflow will save the answers:
   - Database (go to Step 10)
   - Text (go to Step 11)
10. If you selected **Database** in Step 9, the **Answers Database Parameters** dialog box opens. Select the following to identify where the workflow gets the answers:
    a. The name of the source database table from the **Source Table** drop-down list.
    b. The name of the database field in the selected table from the **Source Field** drop-down list.
11. If you selected **Text** in Step 9, enter the number of possible answers. Workflow Designer opens a series of dialog boxes. In those dialog boxes, repeated for each possible answer:
    a. In the first dialog box, enter the text of the first answer and its score.

       A score is a number between 0 and 100. You can assign a different score for each answer. If you are using answers from a database, you cannot associate a score with your answers.
    b. In the second dialog box, enter the following information to identify the snap point that takes an agent to the next block if the agent selects this answer:
       - Anchor Snap Connection Name
       - Anchor Snap Connection Point - Select an available anchor snap from the drop-down list
       - Anchor label - Limited to four characters. This text is displayed on the specified anchor snap in your workflow

       If you need to add more snaps to the block, see "Creating Snap Points" in *Workflow Designer User Guide*.

12. When you have completed the series of dialog boxes for each answer, select **Finish**.

    If you correctly completed the question and answer screens, you receive a message indicating success and Workflow Designer prompts you to select **Finish**.

None Answer Type

The None answer type lets you provide information needed by the agent to complete the task in the workflow, but that does not require an answer from the agent or contact.

To create an HTML page with no answer field:

1. Right-click in the DHTML Editor and select **Objects** > **Prompter**.
2. In the **Question Text** field, enter the text of the information for the agent.
3. In the **Weight** field, enter 0.

   A weight can be between 0 and 1, inclusive of all decimal values. The weight is a relative value that calculates the importance of a question to aid in analyzing customer responses.
4. From the **Answer type** drop-down list, select **None**.
5. From the **Save to Database** drop-down list, select **No**.
6. From the **Required** drop-down list, select **No**.
7. Select **Next**.
8. Select **Finish**.

   The DHTML Editor displays the text as the agents see it in their browsers.

Radio Answer Type

With the radio answer type, the answers display next to radio buttons. The agent can select only one answer by selecting the associated radio button.

To create an HTML page with the answers next to radio buttons:

1. Right-click in the DHTML Editor and select **Objects** > **Prompter**.
2. In the **Question Text** field, enter the text of the question.
3. In the **Weight** field, enter the weight to be given to the answer to this question.

   A weight can be between 0 and 1, inclusive of all decimal values. The weight is a relative value that calculates the importance of a question to aid in analyzing customer responses.
4. From the **Answer type** drop-down list, select **Radio**.

5. From the **Save to Database** drop-down list, select:
   - **Yes** - If you want the answer to be saved in the database when the workflow is completed
   - **No** - If you do not want the answer saved in the database
6. From the **Required** drop-down list, select:
   - **Yes** - To require the agent to enter an answer for the question before continuing the workflow
   - **No** - If the answer is optional
7. Select **Next**.
8. If you selected **Yes** in the Save to Database field, in the next dialog box:
   a. Select the database table where the workflow saves the answer.
   b. Select a field in the database table that matches the data type of the answer to be saved by the agent, for example `varchar` or `text`.
   c. Select whether to save the answer in the `qw_response` table.
   d. Select **Next**.

   If you do not want to store the answer in a database, Workflow Designer does not display this dialog box.
9. In the **Answers Datasource** dialog box, select the location where the workflow will save the answers:
   - Database (go to Step 10)
   - Text (go to Step 11).
10. If you selected **Database** in Step 9, the **Answers Database Parameters** dialog box, opens. Select the following to identify where the workflow gets the answers:
    a. The name of the source database table from the **Source Table** drop-down list.
    b. The name of the database field in the selected table from the **Source Field** drop-down list.
11. If you selected **Text** in Step 9, enter the number of possible answers. Workflow Designer opens a series of dialog boxes. In those dialog boxes, repeated for each possible answer:
    a. In the first dialog box, enter the text of the first answer and its score.

       A score is a number between 0 and 100. You can assign a different score for each answer. If you are using answers from a database, you cannot associate a score with your answers.
    b. In the second dialog box, enter the following information to identify the snap point that takes an agent to the next block if the agent selects this answer:
       - Anchor Snap Connection Name

- Anchor Snap Connection Point - Select an available anchor snap from the drop-down list
- Anchor label - Limited to four characters. This text is displayed on the specified anchor snap in your workflow

If you need to add more snaps to the block, see "Creating Snap Points" in *Workflow Designer User Guide*.

12. When you have completed the series of dialog boxes for each answer, select **Finish**.

    If you correctly completed the question and answer screens, you receive a message indicating success and Workflow Designer prompts you to select **Finish**.

Text Box Answer Type

The text box answer type opens a dialog box that lets an agent type in an answer that is restricted to a limited number of characters. If you save the answer to the database, the text box is limited to the number of characters in the field.

To create an HTML page with a text box entry field:

1. Right-click in the DHTML Editor and select **Objects** > **Prompter**.
2. In the **Question Text** field, enter the text of the question.
3. In the **Weight** field, enter the weight to be given to the answer to this question.

   A weight can be between 0 and 1, inclusive of all decimal values. The weight is a relative value that calculates the importance of a question to aid in analyzing customer responses.
4. From the **Answer type** drop-down list, select **Text Box**.
5. From the **Save to Database** drop-down list, select:
   - **Yes** - If you want the answer to be saved in the database when the workflow is completed
   - **No** - If you do not want the answer saved in the database
6. From the **Required** drop-down list, select:
   - **Yes** - To require the agent to enter an answer for the question before continuing the workflow
   - **No** - If the answer is optional
7. Select **Next**.
8. If you selected **Yes** in the **Save to Database** field, in the next dialog box:
   a. Select the database table where the workflow saves the answer.

b. Select a field in the database table that matches the data type of the answer to be saved by the agent.

c. Select whether to save the answer in the `qw_response` table.

d. Select **Next**.

9. If you selected **No** in the **Save to Database** field, in the next dialog box select one of the following formats for the agent to input data into the text box:
   - **Email** - `x@y.z` (for example, `johndoe@anyisp.com`)
   - **Literal** - Only the following characters: (_) (underscore), lower and uppercase alphabetical characters
   - **None** - Any alpha-numeric or standard keyboard character
   - **Numeric** - Only numbers
   - **Phone - US** - Any of the following telephone formats: (111) 222-333, 111 222 3333, 111-222-3333, or 1-111-222-3333
10. Select **Finish**.

Creating HTML Pages for Application Scripts

Application scripts guide agents through a series of steps to assist them with a task, such as taking an order or entering a complaint from a customer.

Application tasks typically require an agent to access information in a database:

- To add to the data. For example, an agent can add a new order for a customer.
- To modify data. For example, an agent can change a customer's address.

To access database information, an application script includes database objects, such as:

- A form that Prompter fills with information from the database
- A browser that displays the results of a search of the database

The Prompter Database Wizard guides you through the process of inserting database objects into your scripts.

After you design the script and know which steps use database objects, you can set the Page property and create the database objects for the Application blocks.

Note:

You cannot create Prompter questions and answers in HTML pages attached to Application blocks. A workflow can contain both Application and Prompter scripts, but they must display on different HTML pages.

To set the Page property and create an HTML page for an Application script perform the steps in the following sections:

1. Setting the Block Text and Page Property on page 39.
2. .Creating Database Objects on page 40

Setting the Block Text and Page Property

The Page property in the **Basic** tab includes a short description that identifies the HTML page. The block text is the title that displays on the block in the Work Area to identify the purpose of the block in the workflow.

To set the block text and Page property for the prompting script page:

1. Select a Prompter block in your workflow.
2. Add the block text:
   a. Select the **Expert** tab in the Property sheet.
   b. Select **Text** and enter a description of the block's purpose. For example, Contact Status.

3. Select the **Basic** tab in the Property sheet to display the Page property.
4. Select the ellipsis (…) button next to the Page property to open the DHTML Editor.

   If your workflow is not part of an open project, Workflow Designer prompts you to add it to a project before the DHTML Editor opens.
5. In the **Short Description** text box, enter a description for the HTML page.

Creating Database Objects

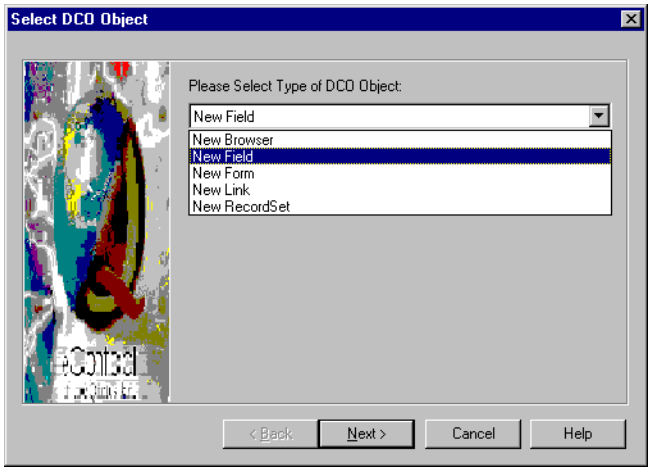
The Database Wizard assists you in creating the following database objects in agent scripts:

- Browser Object - Displays the results of a database search in a scrolling list box
- Field Object - Displays the contents of a field from a database table in a text box
- Form Object - Lets an agent collect and search for information in an application page
- Link Object - Causes an agent script to branch to another block in the workflow
- Record Set Object - Lets an agent view or update records in the database

Note:

To create a database object, you must be familiar with the structure and content of your database and with the Database Designer description of your database.

The database objects are displayed in a drop-down list in the **Select DCO Object** dialog box.



Note:

If you make changes directly in the HTML source code using the **Edit Source** tab, you must return to the **Edit Page** tab before you can select **OK** to save your changes. If you select **Cancel** in the **Edit Source** tab, the DHTML Editor closes without saving and any changes are then lost.

For more information about database objects, and the structure and data model of your Avaya agent application, see *IC Database Designer User Guide* and the Data Model Reference for your application.

Browser Object

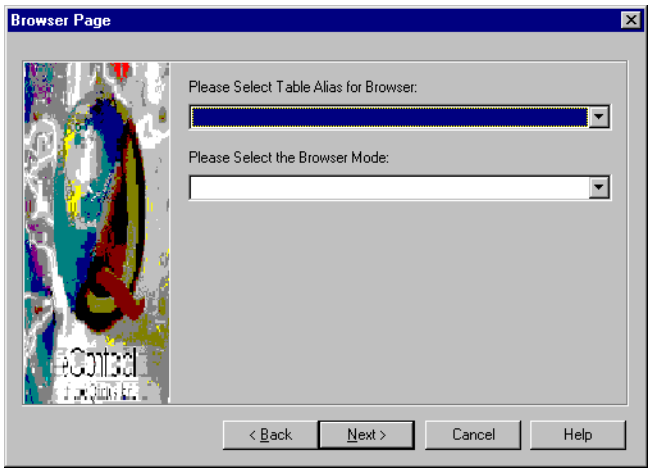
A browser object is a scrolling list box that displays the results of a database search. Before you create a browser object for your script, the browser must be in the ADL file. If the browser does not exist, see *IC Database Designer User Guide.*

To create a browser object

1. Right-click in the DHTML Editor:
   a. Select **Objects** > **Database**.
   b. Select **New Browser** from the drop-down list.
2. In the **Browser Page** dialog box, do the following actions:
   a. From the **Table Alias** drop-down list. select the alias for the database table that contains the data to be displayed by the browser.

      The database connection determines the list of available tables. See "Specifying Project Settings" in *Avaya Workflow Designer User Guide* for more information.
   b. In the **Browser Mode** field, select one of the following:
      - **None** - If the agent can only view the data.
      - **Select** - If the agent can select a set of records for a detailed view in a Record Set object. For more information, see Record Set Object on page 48.
      - **Update** - If the agent can modify the data. Browsers let agents modify one field of information for multiple records simultaneously.

c. Select **Next**.



3. In the second **Browser Page** dialog box, do the following actions:
   a. Select the browser from the **Browser** drop-down list.

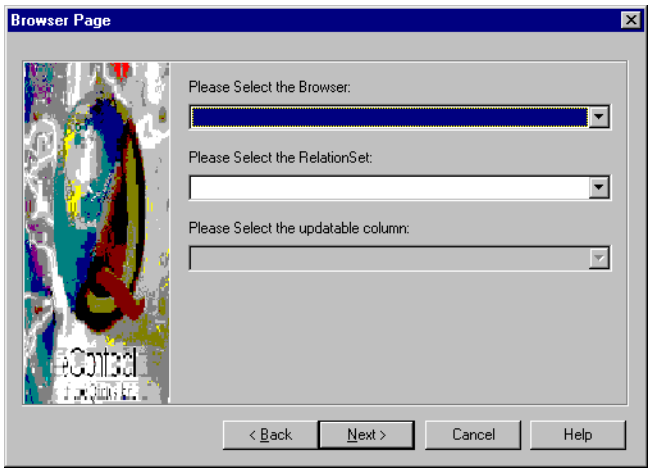
      The list contains all browsers that are associated with the table alias that you selected in the previous dialog box.

   b. Select the relation set associated with the browser from the **Relation Set** drop-down list.

      A relation set is a group of relations. A relation is an association between two tables. The selected relation set constrains the information to be displayed in this browser, by restricting the search to the database tables associated with the selected relation set.

   c. If you selected **Update** mode in the previous dialog box, enter the name of the column that agents can update.

d. Select **Next**.



4. Select **Finish**.

   The DHTML Editor displays the Web page as it is displayed in the Web browser of an agent. You cannot alter the order of columns in the Web page, nor can you split the columns to display in different places within the page. If you move a Web page, you must move all the columns of the Web page.

Field Object

Field objects display the contents of a database field in a text box.

To create a field object:

1. Right-click in the DHTML Editor:
   a. Select **Objects** > **Database**.
   b. Select **New Field** from the drop-down list.
2. In the **Field Page** dialog box, do the following actions:
   a. Select the table alias for the table that contains the field from the drop-down list.
   b. Select **Next**.
3. In the **Select Field** dialog box, do the following actions:
   a. Select the field from the drop-down list.
   b. Select **Next**.
4. Select **Finish**.

Form Object

Form objects let agents do the following tasks:

- Collect information - For example, you can create a form that lists products from a database record. You can give each product in the list a check box that an agent can use to indicate a customer response.
- Search for information - The forms display the results of database searches in a browser.

The agent application displays the contents of forms in a table format. Each value is displayed in two connected cells. One cell contains the label and the other contains the value. When you select a cell, you select both the label and the value. You can add or delete rows from the table in the DHTML Editor.

Note:

For form objects, the DHTML Editor does not support drag-and-drop editing. To move items in a table, cut the row, insert a new row, and paste the cut row. The DHTML Editor also inserts an empty row for each <SPAN> tag in the HTML. The application does not display the empty rows. Do not delete the extra <SPAN> row because this deletes the row associated with the tag.

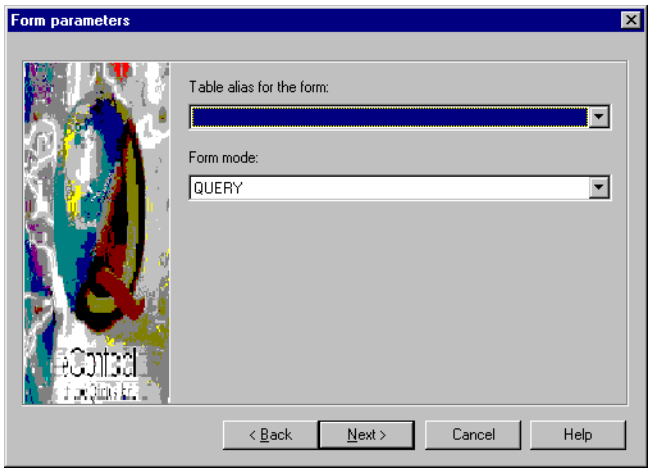
To create a form object:

1. Right-click in the DHTML Editor:
   a. Select **Objects** > **Database**.
   b. Select **New Form** from the drop-down list.
2. In the first **Form Parameters** dialog box, do the following actions:
   a. Select a table alias from the **Table Alias** drop-down list.

      The table alias determines which table of database information is available on the form.

   b. From the **Form Mode** list, select one of the following:
      - **Query** - Creates a form that searches for a match to text that an agent enters into a field
      - **Update** - Create a form that lets an agent enter or update information

c. Select **Next**.



3. In the second **Form Parameters** dialog box, do the following actions:

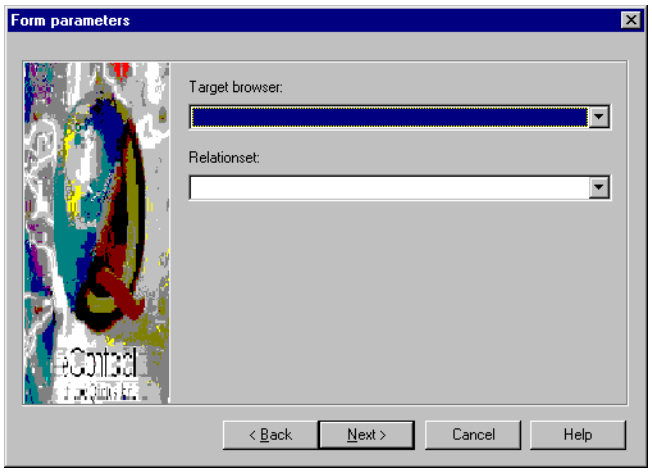
   a. From the **Target Browser** list, select an available browser from the drop-down list.

      The table alias determines which browsers are available. For more information about how to create browsers, see *IC Database Designer User Guide.*

   b. From the **Relation Set** list, select a relation set.

      A relation is an association between two tables. A relation set is a group of relations. The selected relation set constrains the information to be displayed in this browser, by restricting the search to the database tables associated with the selected relation set.

   c. Select **Next**.

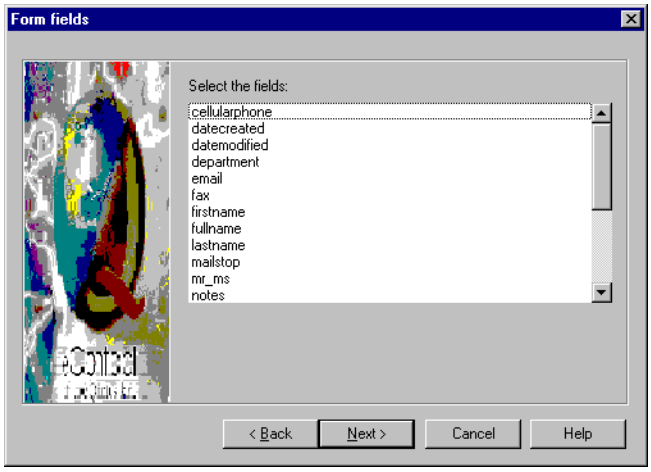


4. In the **Form Fields** dialog box, do the following actions:

a. Select the fields that you want to include in the form.

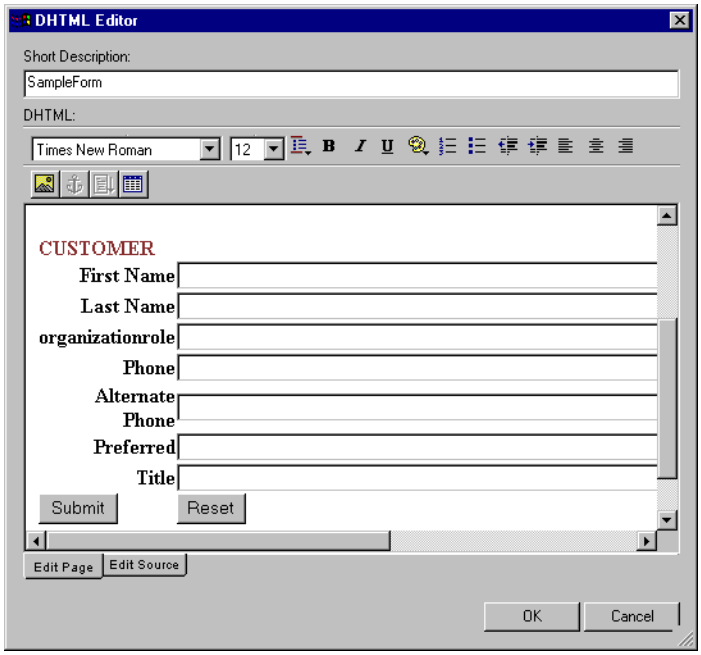
   Each field that you select adds a field on your form where an agent can enter text. Select a highlighted field to remove the field from the form.

b. Select **Next**.



5. Select **Finish**.

   The DHTML editor displays your form.

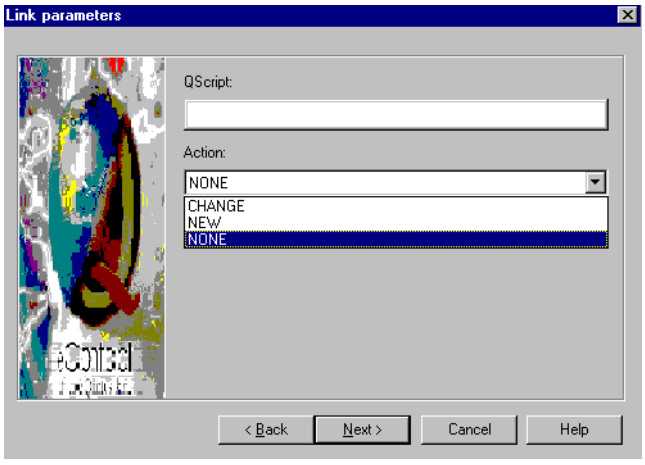


Link Object

Link objects let agents move the agent script to another block in the workflow.

To create a link object:

1. Right-click in the DHTML Editor:
   a. Select **Objects** > **Database**.
   b. Select **New Link** from the drop-down list.
2. In the first **Link Parameters** dialog box, do the following actions:
   a. Enter the name of an IC Script that the application runs before branching in the IC Script field. This field is optional.
   b. From the **Action** list, select the action to be performed after branching:
      - **Change** - Links a Record Set page to an Update Form page, and lets agents change records
      - **New** - Links to an Update Form page, and lets agents create new records
      - **None** - Specifies that no pre-set action be performed
   c. Select **Next**.



3. In the second **Link Parameters** dialog box, do the following actions:
   a. In the **Anchor Snap Connection Name** field, enter the name of the anchor snap on the block where this link takes the agent.

      For more information about anchor snaps, see "Connecting Blocks" in *Avaya Workflow Designer User Guide*.

b. From the **Anchor Snap Connection Point** drop-down list, select the location of the anchor snap on the current block that this link uses.

   The drop-down list includes all unused connection points on the block. If you have already assigned a connection point to an anchor snap, that connection point is not in the list.

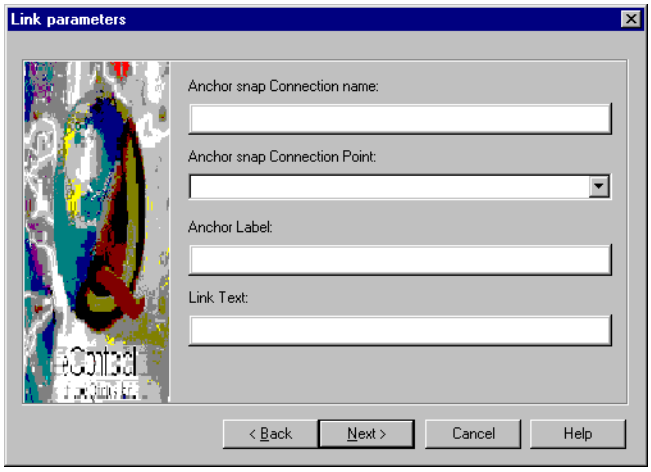
c. In the **Anchor Label** field, enter the anchor name displayed on the anchor snap.

   The label can be a maximum of four characters.

d. Enter the text that the agent sees in the script in the **Link Text** field.

   For example, you can enter `Click here to enter a new customer record`.

e. Select **Next**.



4. Select **Finish**.

Record Set Object

Record set objects let agents view and update records in the database.

To create a record set object:

1. Right-click in the DHTML Editor:
   a. Select **Objects** > **Database**.
   b. Select **New Record Set** from the drop-down list.
2. In the **Record Set Page** dialog box:
   a. Select the table alias containing the records to be displayed from the drop-down list.
   b. Select **Next**.

3. Select **Finish**.

The DHTML Editor displays all of the fields which make up a complete record. The agent can use the buttons in the HTML page to display the previous or next record. You can move or delete entries, but other editing features of the DHTML Editor are not available for Record sets.

Saving Workflows

Use alphanumeric characters in workflow names. If you use a non-alphanumeric character, such as a space or a punctuation mark, Workflow Designer converts the character to an underscore when you compile the workflow. For example, `my flow` and `my.flow` would both become `my_flow`.

To save the workflow:

1. Select **File** > **Save**.
2. Enter the workflow name in the field.
3. Select **Save**.

   Workflow Designer automatically adds the extension `.qfd`.
4. Select **Yes** when Workflow Designer prompts you to insert the workflow into the flowset.

   Workflow Designer saves your workflow in the current project. You can expand the project in the Project bar and see the workflow.

After saving your agent script workflow, complete the steps in the following sections of *Avaya Workflow Designer User Guide*:

1. "Verifying Workflows"
2. "Building a Workflow"

Example: Integrating a Prompting Script and an Application

After you create your agent script workflow, you need to integrate the workflow with an Avaya Business Application or Avaya Agent. This integration lets you control what happens when the workflow ends to the information gathered by an agent. For example, you can ensure that the application saves a customer order in the database, then automatically takes the agent to a Fulfillment focus with the list of coupons filled in as values for the Fulfillment order.

To fully integrate a workflow with an application:

- Design the workflow in Workflow Designer
- Write one or more custom IC Scripts in the Script Editor in Database Designer
- Add a Prompter control to the Business Application or Avaya Agent
- Associate the workflow and flowset with the Prompter control in the application's Administration focus

After you complete the integration, the Prompter control launches the agent script workflow with a generic application IC Script. The IC Script passes the required parameters to the workflow. The IC Script sets these parameters as arguments in a custom block at the end of the workflow.

The arguments are variable symbols that Workflow Designer adds to the Symbol Dictionary when you compile the workflow. If you give an IC Script access to the Symbol Dictionary and survey history at the end of a workflow, the IC Script can perform a tangible function, such as creating a new fulfillment request or order entry within the application.

Note:

The following example uses a prompting script. You perform similar steps to integrate an application script with an application.

In the following example, you integrate a prompting script as follows:

1. Plan the integration by looking at what you need for the application. See Planning the Integration on page 52 for more information.
2. Create a prompting script to collect information from a customer and save the information to the database. See Creating a Prompting Script on page 53 for more information.
3. Create a custom block named IC Script *Settings*. See Creating a Custom IC Script Settings Block on page 55 for more information.
4. Add the IC Script Settings block at the end of the prompting script just before the Stop block. See Adding the Custom Block to the Agent Script Workflow on page 57 for more information.

5. Associate the workflow with the Prompter control in CallCenterQ or Avaya Agent. See Associating the Workflow with a Prompter Client Control on page 58 for more information.

After you complete these steps, you should debug your workflow and then test it in the application.

You can find more information about how to integrate agent script workflows with agent applications in the following documents:

Topic	Document
Writing IC Scripts	*IC Script Language Reference*
Adding the Prompter control to an Avaya Business Application	*IC Designer Application Reference*
Adding the Prompter control to Avaya Agent	Avaya Agent *Integrator's Guide*
Setting up a workflow to run with CallCenterQ	*CallCenterQ Administration Guide*

Planning the Integration

When you are integrating workflows, most of the necessary work is done during the planning phase.

First, you need to ask the right questions and determine which components are available on an agent's desktop. For example, if the agents use CallCenterQ, you must run the workflow in the Prompter form. If the agents use CallCenterQ with Avaya Agent, you can use the Prompter form in CallCenterQ or the Prompter pane in Avaya Agent. Your decision affects how you integrate the workflow with the application.

The following general questions are a good place to start. Depending on your system and your needs, you may need to answer more specific questions before you can start. After

you have the answers to your questions, you can create your prompting script. See Creating a Prompting Script on page 53 for more information.

Question	Answer for this Example
Why will the agent run the workflow?	The agent runs the workflow because a customer calls to cancel membership in Coffee of the Month club.
What information does the company need to get from the customer and to give to the customer?	The Coffee of the Month club needs to know: • Can the customer be persuaded not to cancel? • If not, does the customer want to remain on the mailing list? • If yes, what information might change the customer's mind? • If we change the customer's mind, what coupons can we give the customer for discounts on other items? • What items will the customer buy?
When will the agent run the workflow?	Immediately upon receiving the call from the customer.
Where will the agent be when starting the workflow?	In the Contact form of the application.
How will the agent start the workflow?	From **Run Script** in the Contact form.
Where will the workflow run?	In the Prompter form of CallCenterQ.
Where will the agent go in the application when the workflow ends?	If the customer wants to remain in the club and wants coupons for other products, the agent goes to the Order Fulfillment form where the In-Form Browser displays the selected coupons. The agent can then complete the order form. If the customer does not want to remain in the club, return to the Contact form.

Creating a Prompting Script

When you have determined the steps in your agent script workflow, you can open Workflow Designer and create the agent script workflow. For specific instructions, see Creating HTML Pages for Prompting Scripts on page 26.

The following figure shows a sample coffee survey prompting script for an agent to use when a customer calls to cancel membership in the Coffee of the Month club. The script

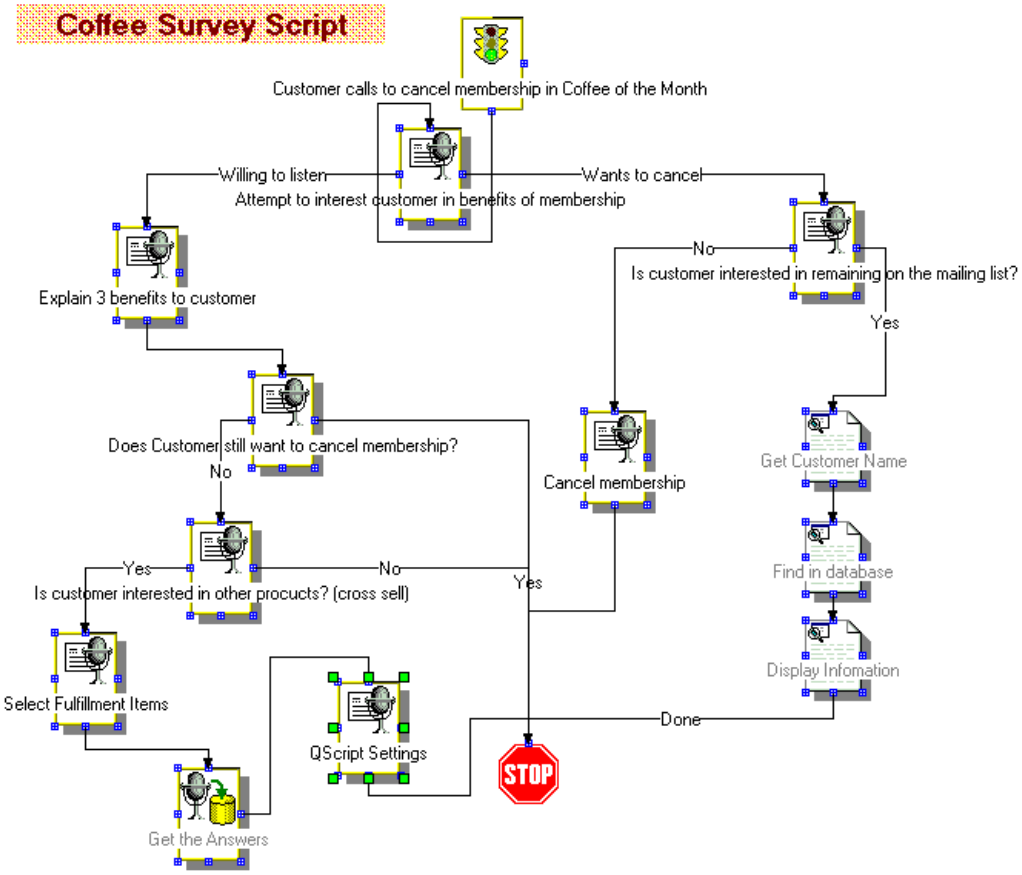
branches at each decision point, depending on whether the customer answers *yes* or *no*. If the customer wants to receive coupons for other products, the script saves that information to the database.

In this example coffee survey workflow, the script branches each time a customer makes a choice. In the first Prompter block, the script branches based on whether the customer listens to the agent and considers keeping the membership or the customer wants to continue with canceling the membership.

If the customer does not want to stay in the club nor stay on the mailing list, the workflow invokes a dummy IC Script. This IC Script displays a message to let the agent know that the workflow ran successfully.

If the customer wants to stay in the club and receive product coupons, fulfillment items are created and the discount coupons selected by the customer are saved as symbols in the workflow.

The highlighted IC Script Settings block is the custom block you create in the next step of the integration.



Creating a Custom IC Script Settings Block

The sample coffee survey script shown in the previous topic presents a series of questions for an agent to ask a customer.

The script uses two custom blocks that are not in the default Workflow Designer palettes. These custom blocks are based on the Prompter block that is located on the Prompter palette. The custom blocks are:

- Get the Answers - This block parses through the survey history maintained by Prompter and extracts the answers entered by the agent on behalf of the customer.
- IC Script Settings - This block invokes the `QprompterClient_FlowComplete` IC Script at the end of the workflow and sets the arguments required for that IC Script. The IC Script then calls the `FulfillmentQPrompter` IC Script, which creates a fulfillment record using the items recorded by the Get the Answers block.

The IC Script Settings block sets the following arguments from workflow input parameters received from the Get the Answers block:

Argument	Description
ScriptName	Name of the IC Script called at the end of the workflow.
ScriptParamCount	Number of parameters to get from the Symbol Dictionary.
ScriptParam1, ScriptParam2, and so on, to ScriptParamN	Arguments for the IC Script, set by the workflow parameters passed to the workflow.

To create the Setup IC Script Settings block:

1. Drag the Prompter block onto the Work Area.
2. Select the block to open the Property sheet.
3. Delete the Page property on the **Basic** tab.
4. Add the following properties on the **Advanced** tab. The block obtains the values for these properties from its IC Script.
   - `Answers` - sequence of strings
   - `ScriptName` - string
   - `ScriptParam1` - string
   - `ScriptParam2` - string
   - `ScriptParam3` - string
   - `ScriptParamCount` - string

5. Change the text that describes the block to IC Script **Settings** on the **Expert** tab.
6. Select the ellipsis (…) button in the `Start` property on the **Advanced** tab.
7. Replace the text of the default IC Script with the text of the IC Script for IC Script Settings block on page 56.
8. Move the new IC Script Settings block to the Custom palette.

For more detailed information about creating custom blocks, see "Creating New Blocks" in *Avaya Workflow Designer User Guide*.

IC Script for IC Script Settings block

The following is a sample IC Script for the IC Script Settings block.

Note:

This script is only an example and is not intended for use in a production environment.

```
'************************************************************************
'  Copyright (c) 2002 Avaya Inc. USA
'                All rights Reserved
'************************************************************************
' Block:  Setup IC Script Settings
' Version:Beta_1
' Description:This block sets up values to be used for running an
              'IC Script after a Prompter flow ends.
'
'************************************************************************
Dim nNumberOfAnswers As Integer
Dim n As Integer
Dim sAnswer As String
Dim sDelimiter As String

'Name of the IScript that gets called when the flow ends
Script.Variable.QScriptName = "FulfillmentQPrompter"

'Number of parameters that get passed to the QScript that gets called when the flow ends
Script.Variable.QScriptParamCount = "3"

'Name of Focus and Form in which the fulfillment record will be created/updated
'can be passed in as input parameters to the flow. These values can then be accessed from
'within any block in the flow in the following manner.
Script.Variable.QScriptParam1 = Script.InputArg.data.FindValue("Focus")
Script.Variable.QScriptParam2 = Script.InputArg.data.FindValue("Form")

'Name of Focus and Form in which the fulfillment record will be created/updated
'Script.Variable.QScriptParam1 = "Agent"
'Script.Variable.QScriptParam2 = "fulfillment_request"
```

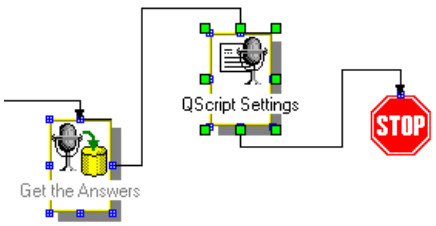
```
'Fulfillment items selected by the customer in 'Select Items' question block
'are sent back to 'FulfillmentQPrompter' qscript as a delimited string
'which will be added to 'Fullfillment' record
sDelimiter = "|"
sAnswer = ""
nNumberOfAnswers = {_#Answers}.length

For n = 0 To (nNumberOfAnswers-1)
    sAnswer = sAnswer & {_#Answers}(n) & sDelimiter
Next n

Script.Variable.QScriptParam3 = sAnswer
Print "*************************************************************************"
Print "* QScriptName: " & {_#QScriptName}
Print "*************************************************************************"
Print "* QScript ParamCount: " & {_#QScriptParamCount}
Print "* QScript Param1: " & {_#QScriptParam1}
Print "* QScript Param2: " & {_#QScriptParam2}
Print "* QScript Param3: " & {_#QScriptParam3}
Print "*************************************************************************"

Script.SetNextConnection "out"

#########################################################################################
```

Adding the Custom Block to the Agent Script Workflow

The IC Script Settings block must be the last block before the Stop block.

Set the connections of the IC Script Settings block as follows:

- Input connection from the Get the Answers block
- Output connection to the Stop block



Associating the Workflow with a Prompter Client Control

With an Avaya Business Application's Request Administration and Order Administration focuses, you can link agent scripts workflows to the following records through the `qw_flowest` and `qw_flow` tables:

- Category
- Product category
- Marketing event

Agents can launch and run the linked workflows in one of the following:

- Prompter form in CallCenterQ - For more information, see *CallCenterQ Administration Guide*.
- Prompter pane of Avaya Agent for all applications except CallCenterQ - For more information, see *Avaya Agent Integrator's Guide*.

Typically, an agent selects **Run Script** in the application to start a Prompter agent script workflow. **Run Script** uses the `FindAndRunScript` IC Script to launch the workflow. The `FindAndRunScript` IC Script determines the appropriate Prompter workflow according to which category, product category, or marketing event the agent selected in the form.

After an agent selects **Run Script**, the `FindAndRunScript` IC Script:

1. Checks to see if a category, product category, or marketing event record links to a Prompter agent script workflow.
2. Gets the workflow set and workflow name if this condition is true.
3. Checks to see if the `RunInQConsole` flag is set and if the workflow can run in Avaya Agent:
   a. If the `RunInQConsole` flag is set, the IC Script:
      - Gets the PrompterClient object from Avaya Agent
      - Gets the server and port information from the Avaya Agent CDL
      - Runs the workflow in the Prompter pane
   b. If the `RunInQConsole` flag is not set, the IC Script:
      - Gets the PrompterClient object from the application
      - Gets the server and port information from the application profile
      - Runs the workflow in a Prompter form within the Agent focus

To pass parameters into the workflow, use the `FindAndRunScript` before your workflow calls the `QPrompter_StartFlow` script as follows:

- Declare `FlowParameters(x) As String`
- Set all the workflow parameter values in the script

For example:

```
FlowParameters(0) = "Focus=" & iFocus.Name
FlowParameters(1) = "Form=fulfillment_request"
```

The following string array is then passed to the `QPrompter_StartFlow` script:

```
QPrompter_StartFlow(iQPrompterClient As Object, sFlowsetName As
String, sFlowName As String, FlowParameters() As String, sLabel
As String)
```

Chapter 2: Block reference

This section provides descriptions and other details for the blocks used in agent script workflows. These blocks are grouped into the following topic areas:

Application Palette

The Application palette contains the following blocks that retrieve and store data in a relational database:

Create Avaya Contact Label

Description Creates a record with the particulars of the contact to be stored in a database table, using the contact name (if any) and the EDU data.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Create Record

Description Creates a database record and populates the database table.

Alarms Generates no alarms.

Connections Has the following unnamed connections:

- 1 or more inputs
- 1 output

Get Customer Using Chat EDU

Description Obtains the Customer Key from the `wc_auth` table using chat information.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Get Record Value(s)

Description Retrieves the values of the fields from one record fetched from a database table using the Table Search block. You can specify the record number for which the values must be retrieved. By default, the values of the first record are retrieved, because the default value for the `RecordNumber` property of this block is 0.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Set Required Agent Desktop EDU Fields

Description Assigns values to EDU couples. The values provide information about the contact that is later stored in the EDU record. This sequence may be written to the EDU server at a later time by a Write EDU or Create EDU block.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Table Clear

Description Clears the buffer of the previous database search.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Table Search

Description Searches a relational database table using the Query by Example type of search. For details on the types of searches that are available, see *IC Script Language Reference*.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Prompter Palette

The Prompter palette contains the following blocks typically used in agent script workflows:

Add Record

Description Adds a record object to a database table, but does not commit the record to the database. This block is typically effective only when followed by a Commit Record block.

To use the Add Record block, complete the Table value in the Basic properties of the block by entering the name of the table where the block adds a record object.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

App Block

Description Performs application scripting. See Creating HTML Pages for Application Scripts on page 39 for more information.

To use the App Block, complete the Page value in the Basic properties of the block. The Page value is a DHTML page with the information for this step in the workflow. You create the page in the DHTML Editor.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

App Stop

Description Terminates an Application workflow in a workflow that includes at least one App Block. This block must be the last block in a script that contains Application blocks. If a workflow contains a combination of Application and Prompter blocks, you must include an App Stop block after the last App Block.

To use the App Stop block, complete the Page value in the Basic properties of the block. The Page value is a DHTML page with the information for this step in the workflow. You create the page in the DHTML Editor. Typically, this DHTML page lets the agent know that the workflow is complete.

App Stop invokes no external methods.

If the workflow contains a combination of App and Prompter blocks, and you want to display the interaction history, you can also use the Prompter Stop block.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Commit Record

Description Commits a record object from a database table to the database. This block is typically effective only when preceded by an Add Record block.

To use the Commit Record block, complete the Table value in the Basic properties of the block by entering the name of the table where the block commits the record object.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more

- Output - 1

On Exit

Description Executes when the workflow ends, whether the workflow completes normally or is interrupted. This block has no connections to other blocks. This block can terminate the EDU and provide extra debugging capability when needed.

On Exit invokes `VDU.Terminate` on the current EDUID if other blocks in the workflow have touched the EDU. By default the request is sent to server `EDU`, passing Script variable `vdu_id`. If `vdu_id` is empty, the block tries to fill it from the data that arrived with the event or method that started this workflow. You can modify these settings in the advanced properties.

In debug mode, this block prints the name of the block that executed just prior to On Exit. This feature is useful for debugging workflows.

For more information about this block, see On Exit Block on page 71.

Alarms Generates no alarms.

Connections Has no connections to any other blocks.

Prompter Block

Description Creates information, and question and answer scripts. See Creating HTML Pages for Prompting Scripts on page 26 for more information.

To use the Prompter Block, complete the Page value in the Basic properties of the block. The Page value is a DHTML page with the information for this step in the workflow. You create the page in the DHTML Editor.

Alarms Generates no alarms.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Prompter Stop

Description

Terminates a Prompter workflow in a workflow that includes at least one Prompter Block. This block must be the last block in a script that contains Prompter blocks. If a workflow contains a combination of Prompter and App blocks, you must include a Prompter Stop block after the last Prompter Block.

The Prompter Stop block lets you display interaction history.

To use the Prompter Stop block, complete the Page value in the Basic properties of the block. The Page value is a DHTML page with the information for this step in the workflow. You create the page in the DHTML Editor. Typically, this DHTML page lets the agent know that the workflow is complete.

Prompter Stop invokes no external methods.

Alarms

Generates no alarms.

Connections

Accepts the following connections:

- Input - 1 or more
- Output - 1

Raise Alarm

Description

Raises an alarm explicitly by issuing the method `Alarm.Alarm`. Specify the name, priority, and descriptive test that accompany the alarm.

At runtime the Workflow server prepends the specified alarm name with the text `QWorkflow`. For example, an alarm named MyAlarm has a designation of `QWorkflow.MyAlarm` at runtime.

To use this block, specify:

- Alarm Name
- Description
- Priority
- Name of the alarm
- Description for the alarm
- Priority of the alarm

Alarms

The purpose of this block is to generate an alarm.

Connections Accepts the following connections:

- Input - 1 or more
- Output - 1

Special Blocks

The following blocks deserve special attention:

Start Block

The Start block marks the beginning point of every workflow. Each workflow must contain one Start block.

The Start block's `DBNetwork` property specifies the focus that contains the table set used by the workflow. This property must be completed for blocks in a workflow to access the database. All workflows must use focuses from the Interaction Center application of the `ccq.adl` file. For more information, see "Accessing a Database from a Workflow" in *Avaya Workflow Designer User Guide*.

This block provides the ideal place to auto-define variables that have workflow-wide significance. These variables have been discussed previously in various contexts, but are summarized in the table below.

Property	Type	Auto-defined Script variable	Description
flowDebug	string	flowDebug	The flowDebug variable determines the debugging level for the workflow. If no global debug level is specified, this variable is set to the value of the flowDebug property. The default value of the property is `custom`. For more information, see "Workflow Debug Property" in *Avaya Workflow Designer User Guide*.

Property	Type	Auto-defined Script variable	Description
flowFullName	string	flowFullName	The Start block sets the `flowFullName` Script variable to the flowset identification, in the format `flowsetName.flowName`. All blocks use this string to help identify themselves when making log file entries. The property exists only to auto-define the variable, and does not allow a value to be entered.
lastBlock	string	lastBlock	The `lastBlock` Script variable contains the full identifying name of the block. All blocks use this string to identify themselves in log file entries. The On Exit Block uses it to log the exit point of the workflow. The property exists only to auto-define the variable, and does not allow a value to be entered.

On Exit Block

A workflow may contain an On Exit block. This block has no anchorsnaps, and is not connected to another block in the workflow. The On Exit block provides a common exit point, and contains code for cleanup and error handling. Whenever a workflow exits, whether through a Stop block, or because of an error, the Workflow server executes the On Exit block if one exists in the workflow.

The On Exit block typically:

- Logs the name of the last block run in the workflow
- Terminates an EDU, if needed
- Auto-defines the Script variable `vduTouched`, which other EDU-related blocks can set to `true` or `false`.
- Executes the `VDU.Terminate` method when a block in the workflow touches the EDU, if the `vduTerminate` property on the On Exit block is set to `on`

Although all EDU-related blocks can auto-define `vduTouched` and terminate the EDU, use the On Exit block to perform these functions to prevent EDU records from being orphaned if workflows exit unexpectedly.

In debug mode, On Exit also prints a log entry that includes the name of the last block that ran (from the `lastBlock` Script variable). You can use this entry to trace an unexpected exit, even if the block that exited leaves no debug trail.

Workflow Designer includes the following On Exit blocks:

- **General palette** - see On Exit on page 67
- **Prompter palette** - see On Exit on page 67

Chapter 3: Testing and monitoring workflows

This section provides information and procedures to help you test your workflows and ensure that workflow operations are occurring as expected.

This section contains the following topics:

Monitoring Workflow Performance

You can trace the execution of the workflow in the Workflow server and monitor how long the server takes to execute the workflow. Problems can arise if the execution of the workflow takes too long or if the workflow sends events too rapidly to an assigned client.

⚠ CAUTION:

If a workflow sends events faster than the assigned client can process them, the connection between the workflow and the client eventually breaks down. In this event, the workflow reacts as if the client de-assigned and the workflow eventually encounters a fatal error as it continues to issue events.

Analyzing the Timer information in the Workflow server log file tells you how long the server took to execute the following:

- Flow
- Block
- Vesp method
- Sleep statement

You should monitor workflow performance in a development environment. If you use the monitor feature with the Workflow Debugger, the breakpoints and other features of the Workflow Debugger cause the workflow to run more slowly and can stop the workflow during execution.

Note:

Do not enable Trace Execution of Flows in a production Workflow server unless you are looking for a performance problem. The server logs concise strings to minimize the impact, but the logging of timing data inevitably slows down the execution of a workflow.

The Workflow server logs two kinds of timing entries: Start and End. Start and End tags for a given thread pair and nest, unless an error occurs and the workflow stops.

Start entries use the form `//TIMER:Sx tid @info` where:

- `x` indicates what is being timed:
  - F - workflow
  - B - block
  - M - vesp method
  - S - sleep statement
- `tid` is a unique integer for each existing thread. Timer entries for various threads can overlap, so the `tid` becomes important when more than one thread runs at a time.
- `info` depends on the kind of timer designated for `x`:

- flow - workflow name (foo.bar)
- block - block name (Start)
- vesp method - the first few characters of the method request (`[DS.GetFewRecords("","type=srv"`)
- sleep statement - number of ms requested (120)

For example, the start entry of a workflow could be similar to the following example:

```
//TIMER:SF 11769532 @t4.code
```

End entries use the form `//TIMER:Ex tid timems {@info}` where:

- `x` and `tid` have the same meaning as in Start entries
- `time` is a floating point number of ms
- `@info` is optional and gives additional information:
  - flow - the workflow name
  - block - the block name

The following is an example from the log of a workflow with one block that performs a vesp request, a sleep, then another vesp request:

```
//TIMER:SF 11708256 @t4.code
//TIMER:SB 11708256 @Start
//TIMER:SM 11708256 @[DS.GetFewRecords("","type=srv",
//TIMER:EM 11708256     78.00ms
//TIMER:SS 11708256 @120
//TIMER:ES 11708256    125.00ms
//TIMER:SM 11708256 @[DS.GetFewRecords("","type=per",
//TIMER:EM 11708256     93.00ms
//TIMER:EB 11708256    609.00ms @Start
//TIMER:EF 11708256    609.00ms @t4.code
```

The granularity of the timers varies by platform. You cannot retrieve precise values under the best of circumstances. However, in the above example, the two vesp requests consumed 78+93=172 ms (of 609 ms), and the sleep statement consumed another 125 ms. These three requests account for half of the running time of the block.

Note:

The timer does not include workflow loading time. Workflows are loaded only once, regardless of how often they are used. Workflows rarely require an extended period of time to be loaded.

To monitor workflow performance:

1. In IC Manager:
   a. Select the **Servers** tab and double-click the Workflow server.

b. Select the **Workflow** tab in the **Workflow server settings** dialog box.

c. Check **Trace Execution of Flows**.

d. Select **OK**.

2. Run the workflow.
3. After the workflow has stopped running, open the Workflow server log and review the entries under `//TIMER:`.

   The Workflow server log is located with all other Avaya logs in `<install-dir>\IC61\logs`.

Testing Prompter Agent Script Workflows

You can test a Prompter agent script workflow by running it from an HTTP prompt. Make any necessary corrections to the workflow in Workflow Designer and the DHTML Editor.

For information on supported browsers, see *IC Installation and Configuration.*

To run a Prompter agent script workflow from an HTTP prompt in a browser:

1. In Internet Explorer, enter the following syntax in the Address or Location box:

   `http://<machine>:<httpConnectorServer_httpport>/<projectname>_shell.htm`

   For example, `http://dev1:9503/listsurvey_shell.htm`

   where:

Variable	Description
`<machine>`	The name of the machine on which the workflow resides.
`<httpport>`	The port number.
`<projectname>`	The name of the project containing this workflow.

2. Select the workflow name to run the agent script workflow.

Index

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W