Script Editor Step Reference

Last Updated: April 28, 2010

This chapter lists all the steps available for use in creating auto attendant (AA) scripts and Interactive Voice Response (IVR) scripts. As indicated, some of the steps are for IVR scripts only. All of the steps are accessed using the Palette pane (see the "Palette Pane" section on page 10). This chapter includes the following sections:

Call Contact Steps

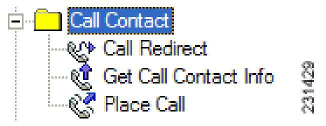
The steps in the **Call Contact** palette provide script designers with a way to manage calls.

The **Call Contact** palette contains the following steps:

- Call Redirect, page 62
- Get Call Contact Info, page 63
- Place Call (IVR Only), page 65

Figure 42 shows the steps in the **Call Contact** palette as they appear in the **Palette** pane of the Cisco Unity Express Script Editor.

Figure 42 Call Contact Palette Steps



Call Redirect

Use the **Call Redirect** step to redirect a call to another extension.

The **Call Redirect** step is often used in applications to transfer a call after a desired extension has been specified.

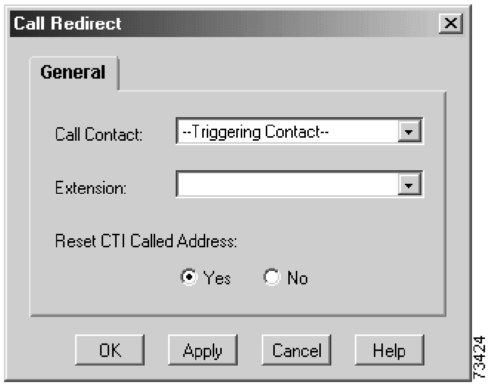
The **Call Redirect** step produces four output branches:

- **Successful**: The call is ringing at the specified extension.
- **Busy**: The specified extension is busy and the call cannot be transferred.
- **Invalid**: The specified extension does not exist.
- **Unsuccessful**: The redirect step fails internally.

Configure script steps after each of the four branches to handle the possible outcomes of a redirected call.

Figure 43 shows the customizer window for the **Call Redirect** step.

Figure 43 Call Redirect Customizer Window



Call Contact: --Triggering Contact--

Extension:

Reset CTI Called Address: [✓ Yes] No

Table 6 describes the fields of the **Call Redirect** customizer window.

Table 6 Call Redirect Customizer Window Fields

Field	Description
Call Contact	Contact that you want to redirect. Default is Triggering Contact, unless another contact is defined. The names of the selected **Call Contact** and extension variables appear next to the **Call Redirect** step icon in the Design pane.

Table 6 Call Redirect Customizer Window Fields (continued)

Field	Description
Extension	Variable that holds the extension where the call is to be redirected. (See Table 7 for supported extensions.)
Reset CTI Called Address	If **Yes**, the script resets the original destination of the call to the redirected destination. If **No**, the script preserves the original call destination even after the **Call Redirect** step executes. The information associated with the call gives no indication that the Route point or CTI port was ever involved with the call. Default is **Yes**. Set this field according to the requirements of the redirected destination.

Table 7 describes the extensions supported by the **Call Redirect** step.

Table 7 Call Redirect: Supported Extensions

Extension	Description
Extensions starting with "#" or "*"	Extensions that trigger a network take-back and transfer where the specified string is outpulsed as is. The redirect is successful if a hang-up event occurs within a maximum of 5 seconds. **Note** You can use a comma (,) in the string to insert a pause of 1 second before the next digit is outpulsed.
Extensions ending with ".wav"	Extensions that trigger a network announcement type of redirect. The system simulates a ring-back tone, plays back the specified .wav file 4 times, and finally simulates a fastbusy tone. The redirect is successful if at any time the caller hangs up or the end of the fastbusy tone is reached, at which point the call is disconnected.
Extensions equal to "PROBLEMS"	Extensions that trigger a network announcement type of redirect with a system problem announcement. The redirect is successful if at any time the caller hangs up or the end of the audio is reached. The call will be reported as disconnected, not redirected.
Extensions equal to "BUSY", "RNA[1]", "FASTBUSY" or "DIALTONE"	The specified audio treatment is generated before the call is disconnected. The redirect is successful if at any time the caller hangs up or the end of the audio is reached. The call will be reported as disconnected, not redirected.

1. RNA = Ring No Answer

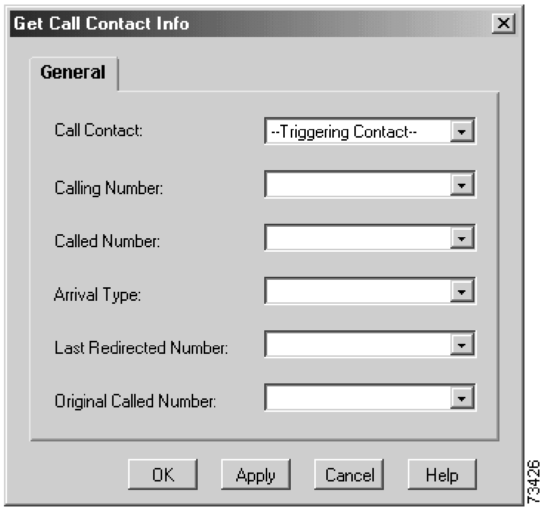
Get Call Contact Info

Use the **Get Call Contact Info** step to access call-specific information and to store values in specified variables.

You can use this step to handle a call in a variety of ways depending on the source of the call and other properties associated with the session. For example, use this step with a **Call Redirect** step to transfer a call to another extension, or with a **Play Prompt** step to play a voice prompt.

Figure 44 shows the customizer window for the **Get Call Contact Info** step.

Figure 44 Get Call Contact Info Customizer Window



Call Contact: --Triggering Contact--
Calling Number:
Called Number:
Arrival Type:
Last Redirected Number:
Original Called Number:

Table 8 describes the fields of the **Get Call Contact Info** customizer window.

Table 8 Get Call Contact Info Customizer Window Fields

Field	Description
Call Contact	Contact for which you want to get information. Default is Triggering Contact, unless another contact is defined. The name of the selected **Call Contact** variable appears next to the **Get Call Contact Info** step icon in the Design pane.
Calling Number	Variable that stores the number of the originator of the call.
Called Number	Variable that stores the number called by the calling party.
Arrival Type	Variable that holds the arrival type of the call. (See Table 9 for supported arrival types.)
Last Redirected Number	The number from which the last call redirect or transfer was invoked. This is the number at which the call was placed immediately before the current number.
Original Called Number	Number called from the perspective of the called party.

Table 9 describes the arrival types of the **Get Call Contact Info** step.

Table 9 Get Call Contact Info: Arrival Types

(Event) Arrival Type	Description
UNKNOWN	The system is unable to determine how the call arrived.
DIRECT	Incoming call that came directly from the originator.

Table 9 Get Call Contact Info: Arrival Types (continued)

(Event) Arrival Type	Description
REDIRECT	Incoming call that was redirected to this application.
FORWARD_ALL	Incoming call that was forwarded from its original destination.
FORWARD_BUSY	Call that was forwarded to the current application because the original extension was busy.
FORWARD_NO_ANSWER	Call that was forwarded to the current application because the original extension exceeded the maximum number of rings.
TRANSFER	Incoming call that originated locally as part of the Transfer feature.
OUTBOUND	Call that was the result of an outgoing call created by an application.
TIME_OF_DAY	Call that was the result of a time-of-day forwarding.
DO_NOT_DISTURB	Call that was the result of a do-not-disturb forwarding.
FOLLOW_ME	Call that was the result of a follow-me forwarding.
OUT_OF_SERVICE	Call that was received because the originally called party was out of service.
AWAY	Call that was received because the originally called party was away.

Place Call (IVR Only)

Use the **Place Call** step to place outbound calls.

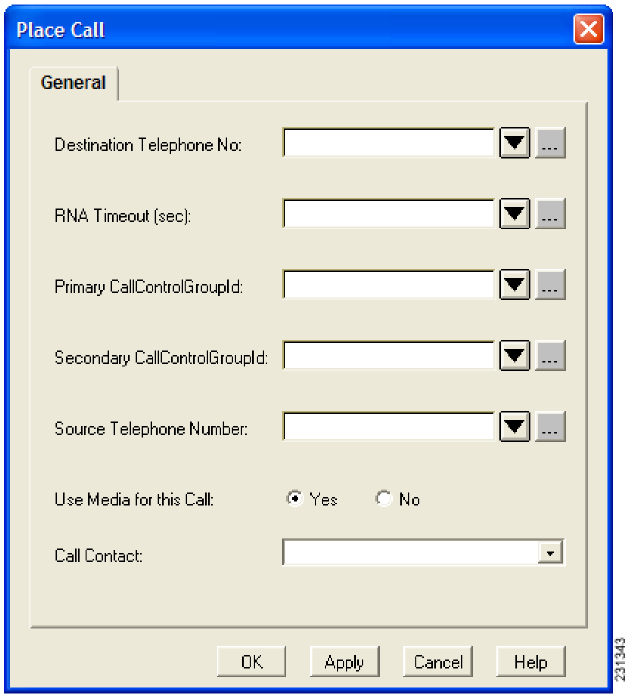
This step, for example, can be used to place calls to patients at preconfigured times to inform them of the current status of their interaction. The notification calls are considered successful if they are answered by a person, voice mail system, answering machine, or fax machine.

The **Place Call** step produces six output branches:

- **Successful**: The call is successful.
- **NoAnswer**: The call was attempted, and the Ring No Answer (RNA) time-out limit was reached.
- **Busy**: The call was attempted, and the line was busy.
- **Invalid**: The call was attempted, and the extension was not a valid extension.
- **NoResource**: The call was not attempted because a resource was not available to make the call.
- **Unsuccessful**: The call was attempted and failed because of an internal system error.

Figure 45 shows the customizer window for the **Place Call** step.

Figure 45 Place Call Customizer Window



Destination Telephone No:	
RNA Timeout (sec):	
Primary CallControlGroupId:	
Secondary CallControlGroupId:	
Source Telephone Number:	
Use Media for this Call:	[✓ Yes] No
Call Contact:	

Table 10 describes the fields of the **Place Call** customizer window.

Note If the values of primary and secondary CallControlGroupIds are not set correctly, as described in Table 10, the outbound call might not be successfully established.

Table 10 Accept Customizer Window Field

Field	Description
Destination Telephone No	Variable that stores the destination number of the outbound call.
RNA Timeout (sec)	Length of time, in seconds, before a Ring No Answer (RNA) condition stops the script from waiting for the remote side to answer, and then returning "NoAnswer" through the output branch.
Primary CallControlGroupId	The identification of the primary call control group that is to be used for the outbound call. This value must be set to 0 for the Cisco Unified Communications Manager Express (formerly known as Cisco Unified CallManager Express) integrations and to 1 for the Cisco Unified Communications Manager (formerly known as Cisco Unified CallManager) integrations.
Secondary CallControlGroupId	The identification of the secondary call control group that is to be used for this outbound call. This field is not used for the Cisco Unified Communications Manager Express integrations. The value must be set to 0 for the Cisco Unified Communications Manager integrations.

Table 10 Accept Customizer Window Field (continued)

Field	Description
Source Telephone Number	(For Future Use) Variable that stores the source number of the inbound call.
Use Media for this Call	If **Yes**, the media channels are also established as part of the call. If **No**, no prompts are played for this call.
Call Contact	Variable that stores the call contact that is created when the step succeeds.

Contact Steps (IVR Only)

The steps in the **Contact** palette provide designers with a way to control contacts. For more information about contact steps, see the *Cisco Unity Express 7.0 IVR CLI Administrator Guide*.

A *contact* represents one form of connection, such as a telephone call, with a remote user. Scripts use contacts to track connections through the system. The contact is established when the connection is made. The contact lasts until the connection is terminated, for example, when the script transfers or disconnects a telephone call.

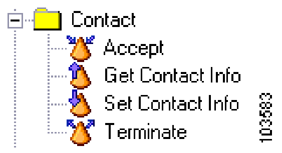
Configure each step that acts on contacts to accept the implicit contact (by choosing the "-- Triggering Contact --" default) or to use a variable that can hold the identifier for this contact. Use the **Set Contact Info** step of the **Contact** palette to mark the contact as Handled, which is important for reporting purposes.

The **Contact** palette contains the following steps:

- Accept, page 67
- Get Contact Info, page 68
- Set Contact Info, page 70
- Terminate, page 71

Figure 46 shows the steps in the **Contact** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 46 Contact Palette Steps



Accept

Use the **Accept** step to accept a particular contact.

After the **Start** step, the **Accept** step is normally the first step in a Cisco Unity Express script, triggered by an incoming contact.

The caller hears ringing until the script reaches this step.

Figure 47 shows the customizer window for the **Accept** step.

Figure 47 Accept Customizer Window

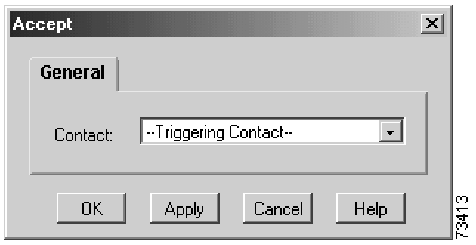


Table 11 describes the field of the **Accept** customizer window.

Table 11 Accept Customizer Window Field

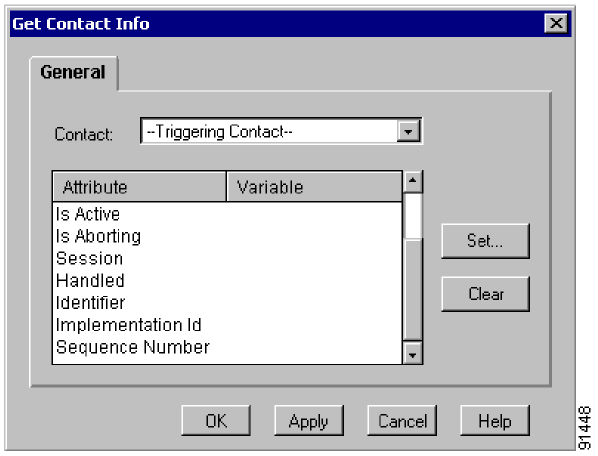
Field	Description
Contact	Contact variable. Default is Triggering Contact, unless another contact is defined. The name of the **Contact** variable appears next to the **Accept** step icon in the Design pane.

Get Contact Info

Use the **Get Contact Info** step to extract information from a contact and store it in script variables so that this contact information is available to subsequent steps in the script.

Figure 48 shows the customizer window for the **Get Contact Info** step.

Figure 48 Get Contact Info Customizer Window



Attribute	Variable
Is Active	
Is Aborting	
Session	
Handled	
Identifier	
Implementation Id	
Sequence Number	

Table 12 describes the fields of the **Get Contact Info** customizer window.

Table 12 Get Contact Info Customizer Window Fields

Field	Description
Contact	Contact variable for which you want to get information. Default is Triggering Contact, unless another contact is defined. The name of the selected Contact variable appears next to the **Get Contact Info** step icon in the Design pane.
Attribute/Variable	Attributes and variables of contact information types.
Set...	Displays Get Contact Type dialog box. The name of the selected variable appears in the **Variable** column next to the selected attribute in the **Get Contact Info** customizer window.
Clear	Clears the variable from the selected Attribute.

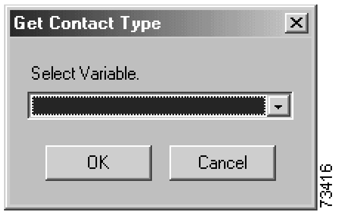
Table 13 describes the information that the **Get Contact Info** step makes available to other steps in the script.

Table 13 Get Contact Info Attributes

Attribute	Description
Type	String representing the type of contact. For Cisco Unity Express, the type is a call.
Language	This option is for future use.
ASR Supported	This option is for future use.
Is Active	Boolean value indicating whether the call is still active.
Is Aborting	Boolean value indicating whether the call is being aborted.
Session Handled	Boolean value indicating whether the contact was previously marked as handled.
Identifier	Integer value containing the contact identifier assigned by the system guaranteed to be unique among all contacts.
Implementation ID	String value containing the implementation-specific identifier for the contact. This value is unique for a given contact type. For a Cisco JTAPI call contact, this value is equivalent to the global call identifier obtained by the Cisco Unified Communications Manager software.
Sequence Number	Integer value containing the sequence number of the contact assigned by the system if the contact is associated with a session. The value is -1 if the contact is not associated with a session. For every new contact associated with a session, the system increments the value by one.
Session	Session object associated with the contact. Null if none is found.

The **Get Contact Type** dialog box is shown in Figure 49.

Figure 49 Get Contact Type Dialog Box



Set Contact Info

Use the **Set Contact Info** step to modify the context information associated with a contact.

The **Set Contact Info** step often follows a **Redirect** step in the script to mark the contact as Handled.

A contact can be marked Handled only while it is active. After a contact becomes inactive (for example, after a successful transfer), the script has a maximum of 5 seconds to mark the contact as Handled; otherwise the mark has no effect in reporting.

Note You cannot mark a contact as *unhandled*. After a contact is reported as Handled, it is always reported with that status.

Figure 50 shows the customizer window for the **Set Contact Info** step.

Figure 50 Set Contact Info Customizer Window

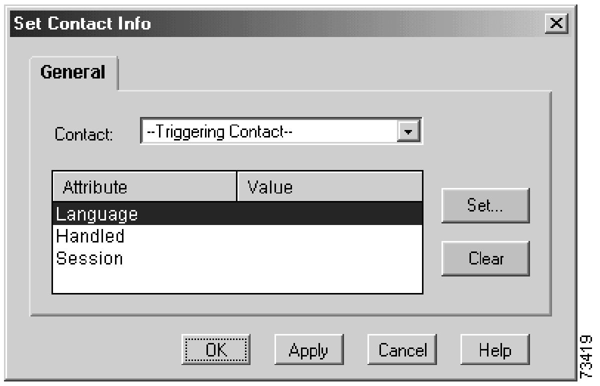


Table 14 describes the fields of the **Set Contact Info** customizer window.

Table 14 Set Contact Info Customizer Window Fields

Field	Description
Contact	Contact variable for which you want to set information. Default is Triggering Contact, unless another contact is defined. The name of the selected **Contact** variable appears next to the **Set Contact Info** step icon in the Design pane.
Attribute/Value	Attributes and values of contact information types.

Table 14 Set Contact Info Customizer Window Fields (continued)

Field	Description
Set...	Sets the Handled attribute for a contact variable. An X appears as the value.
Clear	Clears the Handled attribute for a contact variable.

Table 15 describes the attribute information provided in the customizer window of the **Set Contact Info** step.

Table 15 Set Contact Info Attributes

Attribute	Description
Language	This option is for future use.
Handled	Final result of contact; this is important for reporting purposes.
Session	This option is for future use.

Terminate

Use the **Terminate** step to disconnect the call.

Figure 51 shows the customizer window for the **Terminate** step.

Figure 51 Terminate Customizer Window

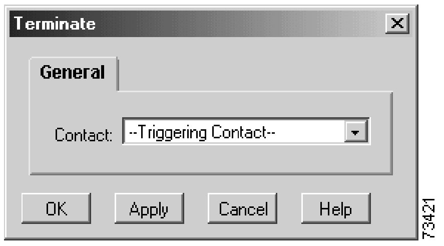


Table 16 describes the field of the **Terminate** customizer window.

Table 16 Terminate Customizer Window Field

Field	Description
Contact	Contact variable that you want terminated. Default is Triggering Contact, unless another contact is defined. The name of the selected **Contact** variable appears next to the **Terminate** step icon in the Design pane.

Database Steps (IVR Only)

The steps in the **Database** palette provide designers with the steps to read and write data from database tables and views. For more information about setting up the database, see the *Cisco Unity Express 7.0 IVR CLI Administrator Guide*.

The following prerequisites must be met before a script is created using **Database** steps:

- When the Script Editor software cannot connect to the Cisco Unity Express router, you can still create a script by manually entering the schema information in the database steps field.

  For example, a table name or field name is only refreshed if there is connectivity between the Script Editor software and the Cisco Unity Express router, and the Cisco Unity Express router can connect to the external database using a defined database profile configuration.
- The Cisco Unity Express router must be reachable over the network from the Cisco Unity Express Script Editor over Java RMI protocol.
- The IP address or Domain Name Server (DNS) hostname of the Cisco Unity Express router must be defined. The network information of the module can be specified by clicking **Editor Tools** and Options from the Cisco Unity Express Script Editor.
- Database profiles must be created using either the Cisco Unity Express graphical user interface (GUI) or command-line interface (CLI). **Database** steps are not usable for script writing without database profiles, and an error message appears if database profiles have not been created.

Database steps have a *resource name* associated with them that are used to link multiple steps. For example, if the **DB Read** step is issued with the resource name *myquery* to execute a query, *myquery* will be specified in the **DB Get** step to retrieve results from that query. The *myquery* resource name is also required before the **DB Release** step can release database resources when it is prompted by a query.

If a resource name is used for a query, and another **DB Read** step is configured to use the same resource name, the results of the previous query are overwritten by the second **DB Read** step.

A resource name is entered in text fields associated with the **DB Read** and **DB Write** steps. Each resource name uses one database connection, and each profile is allowed a finite number of connections. If this number of allowed connections is exceeded during step execution, a Structured Query Language (SQL) Exception will be issued. To ensure that the number of connections do not go over profile maximums, you must use the **DB Release** step, which releases connection resources after a **DB Read** or **DB Write** step is executed. Terminating the script execution also releases database resources.

The **DB Read** and **DB Get** steps are executed in sequence by the **Database** script. The **DB Read** step performs a database query and stores the returned data internally in the Enterprise Database Subsystem (EDBS). The **DB Get** step uses data retrieved by the **DB Read** step. If the actual database contents are changed after executing the **DB Read** step, to the **DB Get** step will not reflect those changes and you must issue another **DB Read** to get updated data from the database.

The **Database** palette contains the following steps:

- DB Read, page 73
- DB Get, page 76
- DB Write, page 79
- DB Release, page 83

Figure 52 shows the steps in the **Database** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 52 Database Palette Steps

DB Read

Use the **DB Read** step to issue a database selection query. The **DB Read** step allows you to specify which columns to return in the select query and to specify the variables to put the resulting data into through the **DB Get** step (see the "DB Get" section on page 76). The **DB Read** step allows you to specify more than one database table or view name from which to query.

The **DB Read** step has the following restrictions:

- The **DB Read** step will not support the **count(*)** or **min(*)** form of the SQL statement variant, such as **select count(*)** and **select min(*)**. The **DB Read** step supports database views, so you can create a view in a database that uses these SQL queries.
- The **DB Read** step does not support semantic checking on the *where* clause of an SQL statement.
- The **DB Read** step does not support *join* queries that retrieve columns with the same name from different tables.
- The **DB Read** step does not support the use of aliases of column names. For example, you cannot create **select a.x, b.x from a,b** and **select a.x as y from a**. No checks will be made to ensure such queries are not entered. The **DB Read** step only allows **select** SQL statements to be entered.
- **Update** or **insert** SQL statements are not supported.

When those results returned from **DB Read** step are no longer needed, use the **DB Release** step to clear the results and release database resources.

The **DB Read** step produces three output branches:

- **Successful**: Indicates that no errors occurred. Zero or more rows of data were found.
- **Connection Not Available**: Indicates that no database connection was available to execute the query.
- **SQL Error**: Indicates that errors occurred that were related to SQL, timeout, or any other exception encountered.

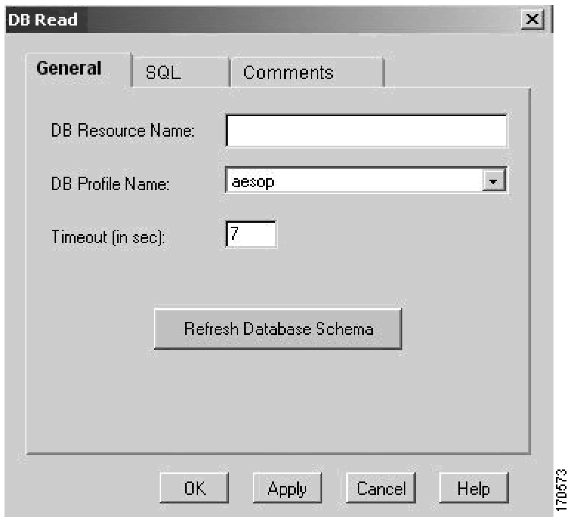
The **DB Read** customizer window contains three tabs:

- General Tab, page 73
- SQL Tab, page 74
- Comments Tab, page 76

General Tab

Use the **General** tab of the **DB Read** customizer window, as shown in Figure 53, to configure the database resource name, database profile name, and time out value (in seconds).

Figure 53 DB Read Customizer Window: General Tab



DB Resource Name:

DB Profile Name: aesop

Timeout (in sec): 7

Refresh Database Schema

Table 17 describes the fields of the **General** tab.

Table 17 DB Read Customizer Window Fields: General Tab

Property	Description
DB Resource Name	Name that identifies this database query. Each DB Read step must have a unique DB Resource Name. The DB Read step Resource Name must not be used in any DB Write step. The same DB Resource Name is used in the accompanying **DB Get** and **DB Release** steps.
DB Profile Name	The name of the database profile that you want to access. This list contains the database profiles configured on the Cisco Unity Express module.
Timeout (in sec)	Interval that prevents the script from waiting indefinitely if the database is unavailable. If the value for the timeout interval is 0, an indefinite wait occurs. Note that an indefinite wait may block the script from responding to events such as a remote disconnection.
Refresh Database Schema	Updates the Editor with the fields defined in the selected Cisco Unity Express database.

SQL Tab

Use the **SQL** tab of the **DB Read** customizer window, as shown in Figure 54, to configure SQL commands.

Figure 54 DB Read Customizer Window: SQL Tab

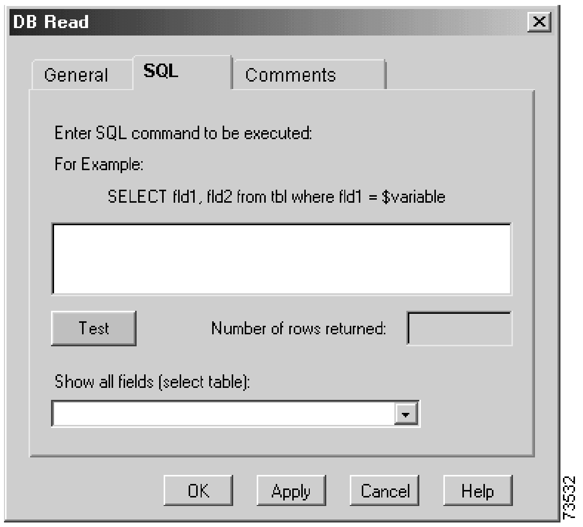

DB Read
General
SQL
Comments
Enter SQL command to be executed:
For Example:
SELECT fld1, fld2 from tbl where fld1 = $variable
Test
Number of rows returned:
Show all fields (select table):
OK
Apply
Cancel
Help
73532

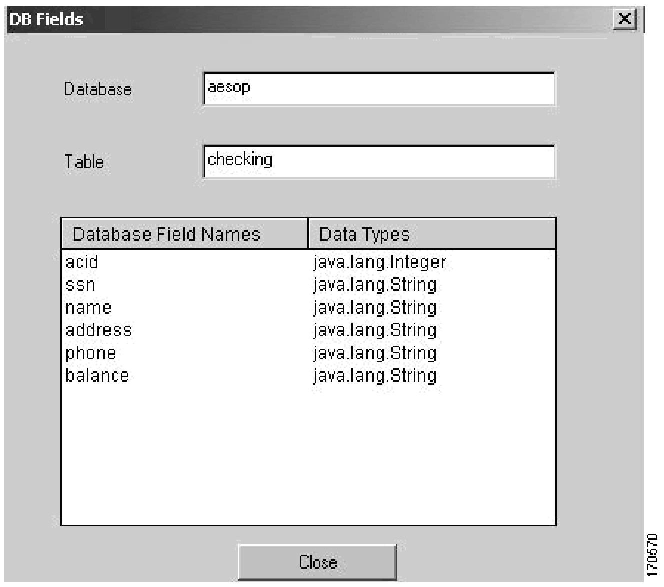
Table 18 describes the fields of the **SQL** tab.

Table 18 DB Read Customizer Window Fields: SQL Tab

Property	Description
Enter SQL Command to be Executed	SQL command that you want to be executed.
Number of Rows Returned	Rows returned when the **Test** button is clicked.
Show All Fields (Select Table)	Fields defined in a particular table in this database. This will list all the tables for a given profile. When a table name is selected, the DB Field dialog box opens, as shown in Figure 55, displaying the fields for this table and their data types.

After a particular table is selected in the **Show All Fields** drop-down list, the **DB Field** dialog box opens as shown in Figure 55.

Figure 55 DB Field Dialog Box



Database: aesop

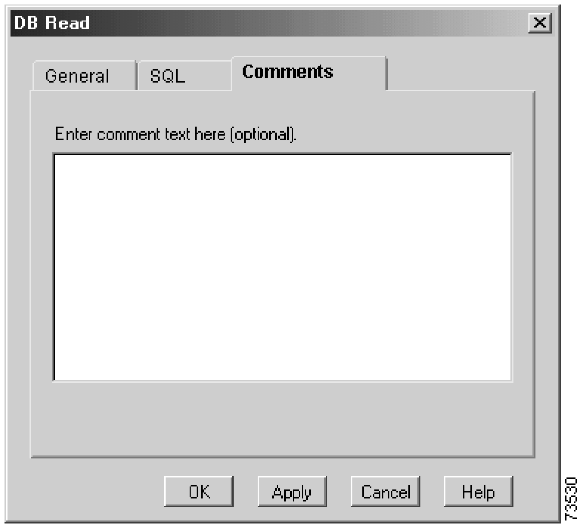
Table: checking

Database Field Names	Data Types
acid	java.lang.Integer
ssn	java.lang.String
name	java.lang.String
address	java.lang.String
phone	java.lang.String
balance	java.lang.String

Comments Tab

Use the **Comments** tab of the **DB Read** customizer window, as shown in Figure 56, to enter text comments.

Figure 56 DB Read Customizer Window: Comments Tab



DB Get

Use the **DB Get** step to retrieve the results from the **DB Read** step.

Inside (or script variables used while configuring) the **DB Get** step, variables are mapped to the table or view fields selected. Outside (or script variables that can be used anywhere in the script) the **DB Get** step, data can be accessed by using the script variable names. When finished with results, use the **DB Release** step to release query resources.

Each time a script executes the **DB Get** step, it retrieves one row of the results returned by the **DB Read** step and assigns variables to them. To move to the next row in the result set, you must execute the **DB Get** step again. When all of the rows of the result are read, **DB Get** goes into the **No Data** branch.

Note It is highly recommended that you place any DB Get steps under the Successful branch of its corresponding DB Read step. If the DB Read step is unsuccessful when its corresponding DB Get step is attempted, the script will abort.

The **DB Get** step produces three output branches:

- **Successful**: No errors occurred. Data was found.
- **No Data**: Query returned nothing in **DB Read** step. This will also be the case when all the rows of the result set have been read.
- **SQL Error**: This could include SQL, timeout, or any exceptions found.

The **DB Get** customizer window contains two tabs:

- General Tab, page 73
- Field Selection Tab, page 78

General Tab

Use the **General** tab of the **DB Get** customizer window, as shown in Figure 57, to select a database resource and database profile from **DB Read** step results.

Figure 57 DB Get Customizer Window: General Tab

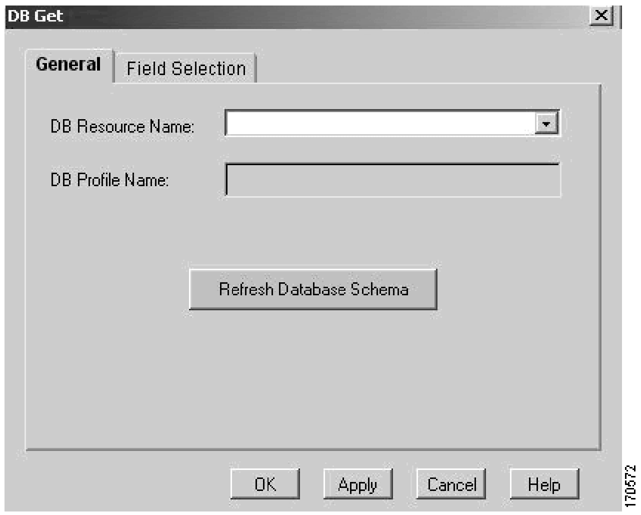


Table 17 describes the fields of the **General** tab.

Table 19 DB Get Customizer Window Fields: General Tab

Property	Description
DB Resource Name	Query defined by a **DB Read** step. You can choose a specific query when your script has multiple **DB Read** steps open at the same time.
DB Profile Name	Name of the database profile associated with this DB Resource Name. This field is populated automatically when DB Resource Name is selected.
Refresh Database Schema	Updates the Editor with the fields defined in the selected Cisco Unity Express database.

Field Selection Tab

Use the **Field Selection** tab of the **DB Get** customizer window, as shown in Figure 58, to select a table, fields, data type, and local variables from **DB Read** step results.

Figure 58 Field Selection Customizer Window: Field Selection Tab

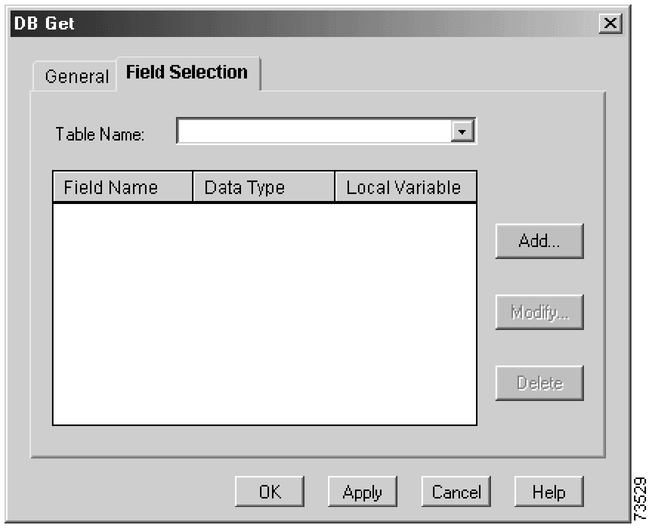


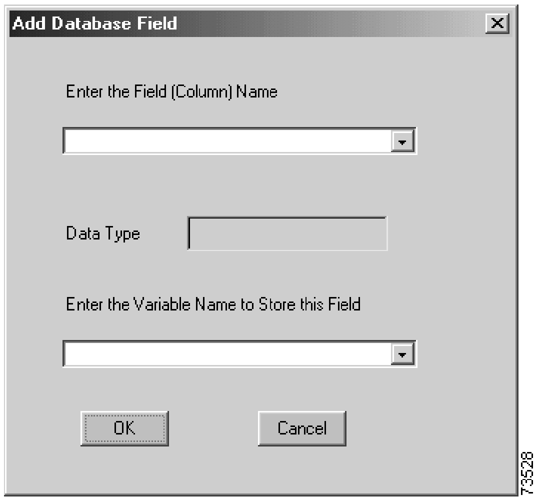
Table 20 describes the fields of the **Field Selection** tab.

Table 20 DB Get Customizer Window Fields: Field Selection Tab

Property	Description
Table Name	Name of the table from the selected database.
Field Name	Name of the field (column) in the selected table of the database.
Data Type	Data type of the variable.
Local Variable	Variables that store the values of the associated fields.

As shown in Figure 58, clicking **Add** opens the **Add Database Field** dialog box shown in Figure 59.

Figure 59 Add Database Field Dialog Box


Add Database Field
Enter the Field (Column) Name
Data Type
Enter the Variable Name to Store this Field
OK
Cancel
73528

DB Write

Use the **DB Write** step to enter SQL *update*, *insert*, and *delete* statements for refreshing the database records.

Note The **DB Write** step supports only these three SQL statements: *update*, *insert*, and *delete.*

After completing the **DB Write** step, use the **DB Release** step (see the "DB Release" section on page 83) to release database resources.

The **DB Write** step produces three output branches:

- **Successful**: No errors occurred. The step entered the specified SQL statements successfully.
- **Connection Not Available**: No database connection was available to execute this query.
- **SQL Error**: An error occurred. This could include SQL, timeout, or any exceptions found.

The **DB Write** customizer window contains four tabs:

- General Tab, page 79
- SQL Tab, page 80
- Test Tab, page 82
- Comments Tab, page 82

General Tab

Use the **General** tab of the **DB Write** customizer window, as shown in Figure 60, to select a database resource and database profile to which you can choose to apply the SQL **update**, **insert**, and **delete** statements.

Figure 60 DB Write Customizer Window: General Tab

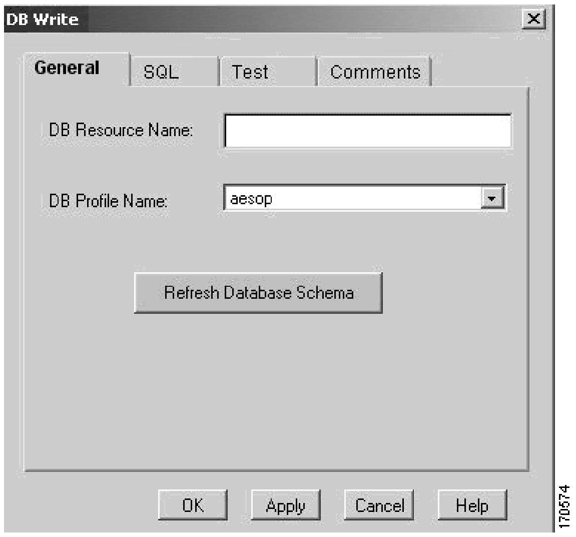


Table 21 describes the fields of the **General** tab.

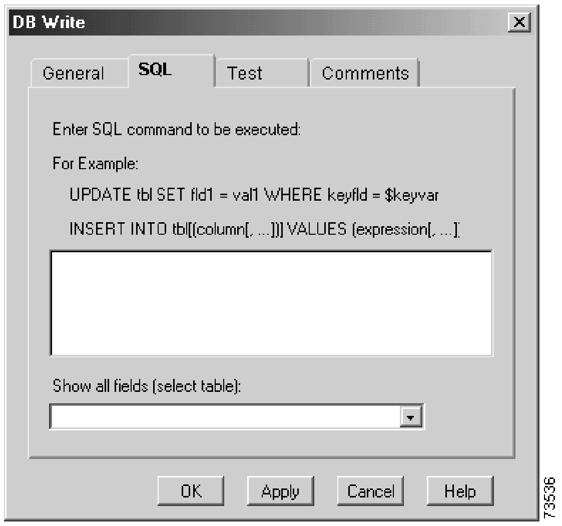
Table 21 DB Write Customizer Window Fields: General Tab

Property	Description
DB Resource Name	Name assigned to identify this database query.
DB Profile Name	Name of the database profile on which you want to run your update query.
Refresh Database Schema	Updates the Editor with the fields defined in the selected Cisco Unity Express database.

SQL Tab

Use the **SQL** tab of the **DB Write** customizer window, as shown in Figure 61, to enter SQL commands.

Figure 61 DB Write Customizer Window: SQL Tab



Enter SQL command to be executed:

For Example:

UPDATE tbl SET fld1 = val1 WHERE keyfld = $keyvar

INSERT INTO tbl[(column[, ...])] VALUES (expression[, ...]

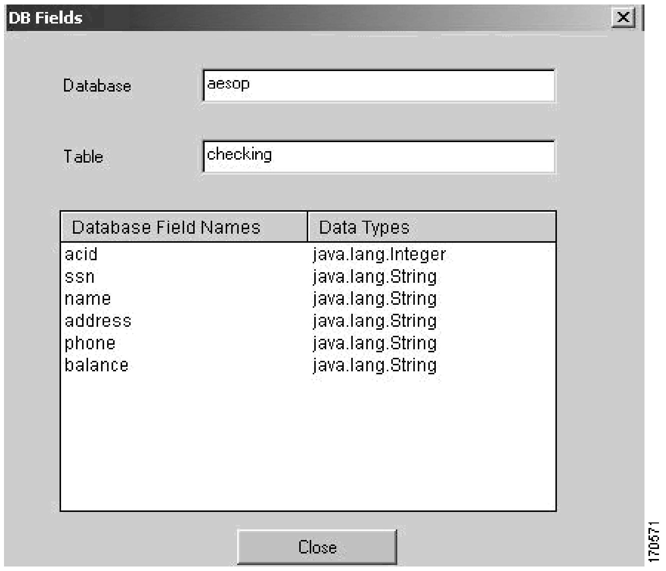
Show all fields (select table):

Table 22 describes the fields of the **SQL** tab.

Table 22 DB Write Customizer Window Fields: SQL Tab

Property	Description
Enter SQL Command to be Executed	SQL command that you want to be executed.
Show All Fields (Select Table)	Fields defined in a particular table in this database. Select a table from the drop-down list to display the DB Fields dialog box, as shown in Figure 62.

Figure 62 DB Fields Dialog Box



Database aesop

Table checking

Database Field Names	Data Types
acid	java.lang.Integer
ssn	java.lang.String
name	java.lang.String
address	java.lang.String
phone	java.lang.String
balance	java.lang.String

Test Tab

Use the **Test** tab of the **DB Write** customizer window, shown in Figure 63, to execute the SQL *update*, *insert*, or *delete* statement and view the results without changing a database.

Figure 63 DB Write Customizer Window: Test Tab

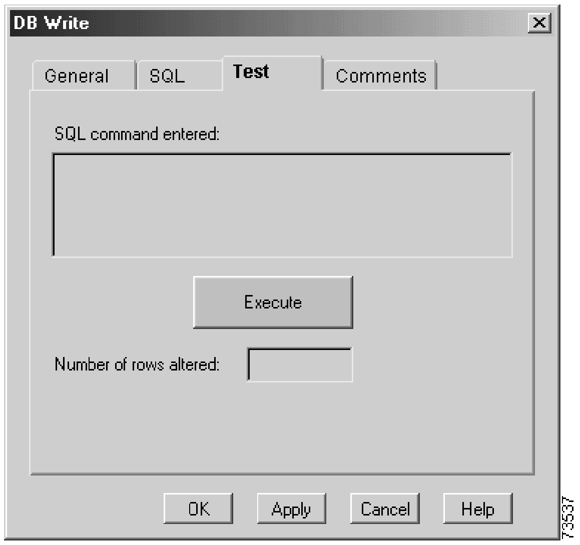


Table 23 describes the fields of the **Test** tab.

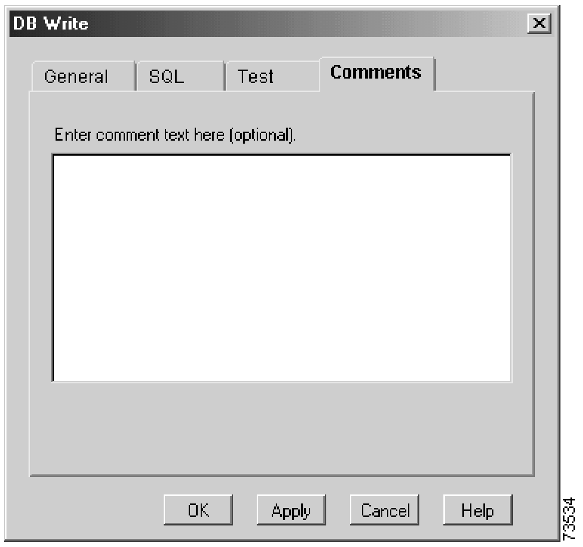
Table 23 DB Write Customizer Window Fields: Test Tab

Property	Description
SQL Command Entered	Enter the SQL *update*, *insert*, or *delete* statement to run a trial statement execution.
Number of Rows Altered	Displays the number of rows altered by the test.

Comments Tab

Use the **Comments** tab of the **DB Write** customizer window, as shown in Figure 64, to enter text comments regarding test results (see the "Test Tab" section on page 82).

Figure 64 DB Write Customizer Window: Comments Tab



Enter comment text here (optional).

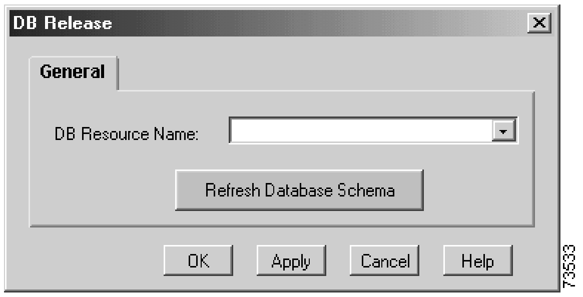
DB Release

Use the **DB Release** step to release database resources associated with a resource name.

The **DB Read** and **DB Write** steps (see the "DB Read" section on page 73 and "DB Write" section on page 79) must use the **DB Release** step to release resources when results returned by them are no longer needed.

Figure 65 shows the customizer window for the **DB Release** step.

Figure 65 DB Release Customizer Window



DB Resource Name:

Table 18 describes the field of the **DB Release** customizer window.

Table 24 DB Release Customizer Window Field

Property	Description
DB Resource Name	Name of the database resource from which you want to release resources.

Document Steps (IVR Only)

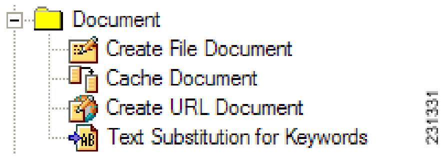
The steps in the **Document** palette provide designers with the steps to handle a variety of documents. For more information about document steps, see the *Cisco Unity Express 7.0 IVR CLI Administrator Guide*.

The **Document** palette contains the following steps:

- Cache Document, page 84
- Create File Document (IVR Only), page 85
- Create URL Document, page 86
- Text Substitution for Keywords, page 88

Figure 66 shows the steps in the **Document** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 66 Document Palette Steps



Cache Document

Use the **Cache Document** step is to perform an input/output (I/O) operation, such as making a Hypertext Transfer Protocol (HTTP) request, and to cache the resulting document in the memory buffer.

Note The **Cache Document** step can use up to 512 KB of memory. If the size of the cache document exceeds 512 KB of memory, any data that exceeds 512 KB is truncated.

The I/O operation is specified by the document defined in the step that precedes this step, such as the **Create File Document** step (see the "Create File Document (IVR Only)" section on page 85) or the **Create URL Document** step (see the "Create URL Document" section on page 86) or by a document expression that contains a hard-coded document. When the **Create File Document** or **Create URL Document** step executes, it creates the document variable and does not send a URL request or access the file system.

I/O operation occurs when another step, such as the Send Response step (see the "Send Response" section on page 123), references the document and permits the I/O operation to be completed before any subsequent steps are executed. In the following example, the **Cache Document** step makes an HTTP request to mybank.money.com. Without the **Cache Document** step, the I/O would not occur until the Send eMail step (see the "Send eMail" section on page 93) step executes.

```
doc=Create URL Document("http://mybank.money.com/debit?amount=500")

doc=Cache Document(doc)

. . .

SendEmail(From, To, doc)
```

Figure 67 shows the customizer window for the **Cache Document** step.

Figure 67 Cache Document Customizer Window

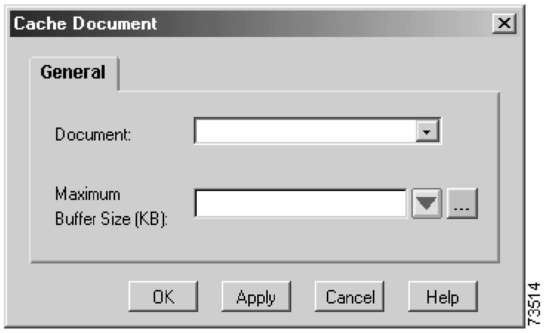


Table 25 describes the fields of the **Cache Document** customizer window.

Table 25 Cache Document Customizer Window Fields

Property	Description
Document	Permits the selection of a document to be cached from the drop-down list.
Maximum Buffer Size (KB)	Maximum buffer size for documents: • A large buffer can affect system performance. • Documents that are larger than the configured maximum buffer size are truncated. • Valid range is 1–512 KB.

Create File Document (IVR Only)

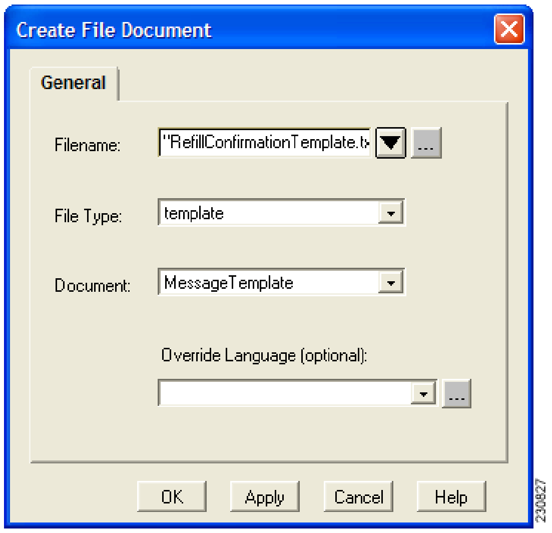
Use the **Create File Document** step to incorporate templates, Tagged Image File Format (TIFF) attachments, and other generic documents into a script.

Documents created by this step can be included in the body of an e-mail or as an attachment, or be used in the same way for sending faxes.

When using the **Create File Document** step, you must select the name of a file that has been uploaded onto a Cisco Unity Express module. Files can be loaded onto the module using the CLI or GUI. You must also select a file type, which will be assigned to the file when it is uploaded onto the module.

Figure 68 shows the customizer window for the **Create File Document** step.

Figure 68 Create File Document Customizer Window


Create File Document

General

Filename: "RefillConfirmationTemplate.tx

File Type: template

Document: MessageTemplate

Override Language (optional):

Table 26 describes the fields of the **Create File Document** customizer window.

Table 26 Create File Document Customizer Window Fields

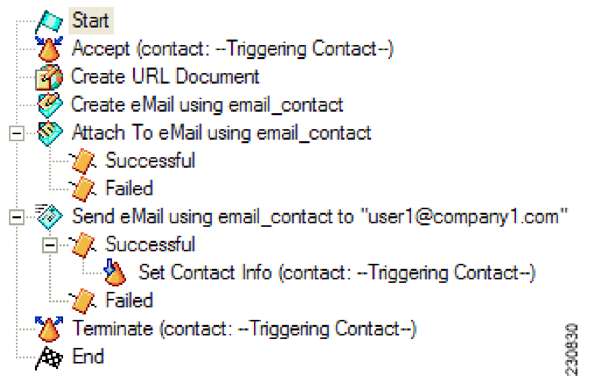
Property	Description
Filetype	File type is selected from a drop-down list. The list includes Template, TIFF, and document.
Filename	Filename of the document to be associated with the document variable. The filename may contain the name of the file in quotations or reference a string variable.
Document	Variable that represents the specified document. Select from the drop-down list.

Create URL Document

Use the **Create URL Document** step to define a document variable by entering an HTTP URL. For the **Create URL Document**, you can use FTP or HTTP.

The **Create URL Document** step does not issue the HTTP request. The request occurs when the document is used by another step, such as the Send Response, Send eMail, Send Fax, or Cache Document step. In the example shown in Figure 71, when the **Send Email** step is started, the HTTP request is made and the resulting HTML document is sent as an e-mail to user1@company1.com.

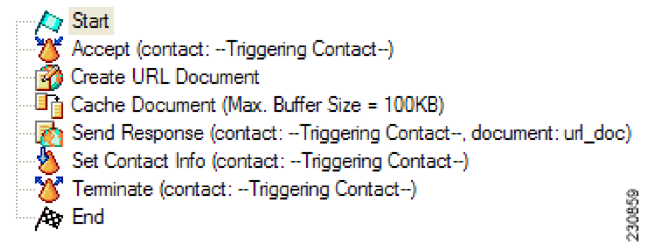
Figure 69 *Send Attached E-mail*



```
Start
Accept (contact: --Triggering Contact--)
Create URL Document
Create eMail using email_contact
Attach To eMail using email_contact
    Successful
    Failed
Send eMail using email_contact to "user1@company1.com"
    Successful
        Set Contact Info (contact: --Triggering Contact--)
    Failed
Terminate (contact: --Triggering Contact--)
End
```

In the example shown in Figure 70, if you want to use the **Create URL Document** step to make an HTTP request only and do not want to send an e-mail, fax, or HTTP response, use the **Cache Document** step to make the request.

Figure 70 *Send Cache URL Response*



```
Start
Accept (contact: --Triggering Contact--)
Create URL Document
Cache Document (Max. Buffer Size = 100KB)
Send Response (contact: --Triggering Contact--, document: url_doc)
Set Contact Info (contact: --Triggering Contact--)
Terminate (contact: --Triggering Contact--)
End
```

As shown in Table 27, there are two HTTP request methods: GET and POST. Table 27 describes the fields of the **Create URL Document** customizer window.

Table 27 *Create URL Document: Customizer Window Fields*

Property	Description
URL	URL for the document. The URL can be a quoted string or a string variable. It must be a well-formed HTTP URL. The URL string is not validated to ensure that it is a well-formed HTTP URL.
Method	Method to use if the URL represents an HTTP request. • **GET**: Appends parameters to the URL. This step is equivalent to using the document expression form URL[url?name=value,name=value]. • **POST**: Includes the parameters as if they were entered in an HTML form.
Parameters	Name-value pairs that form a parameter string to send to the web server. The name-value pairs are converted into a URL-encoded parameter string.
Document	Variable name in which the resulting document object is stored.

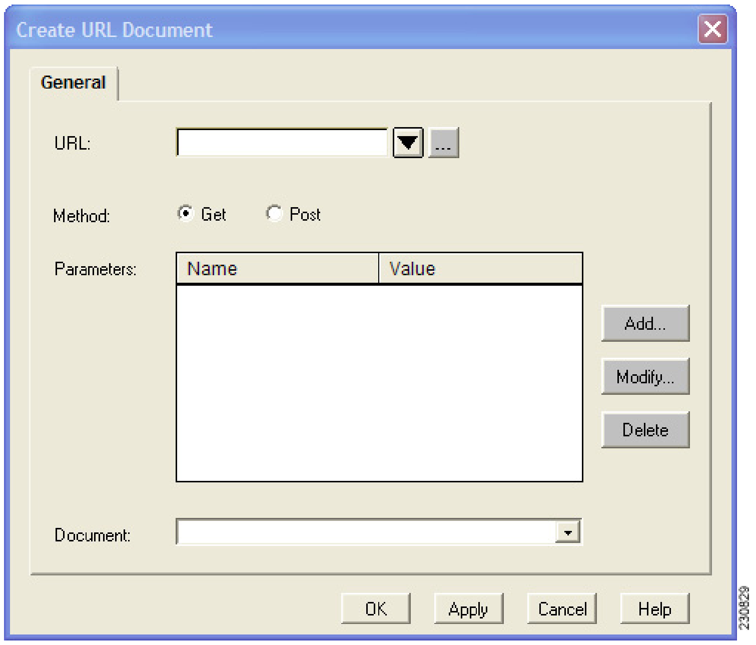
For GET HTTP, parameters are usually appended to a URL in an HTTP request to provide data required to execute the request. For example, the following HTTP request is made when the value of the total amount is set to 500:

```
http://mybank.money.com/debit?amount=500
```

For POST, the parameters are passed in the body of the HTTP request as if entered in an HTML form.

Figure 71 shows the customizer window for the **Create URL Document** step.

Figure 71 Create URL Document Customizer Window



Text Substitution for Keywords

Use the **Text Substitution for Keywords** step to create dynamic content for HTTP, e-mail, and fax by replacing keywords in a document with values contained in local variables.

Keywords are mapped to variables. The **Text Substitution for Keywords** step replaces keywords with the current values of local variables. Keywords are indicated by percent (%) characters in the document, such as %NAME%, which can be dynamically replaced with a string variable with a person's name.

Figure 72 shows the customizer window for the **Text Substitution for Keywords** step.

Figure 72 Text Substitution for Keywords Customizer Window

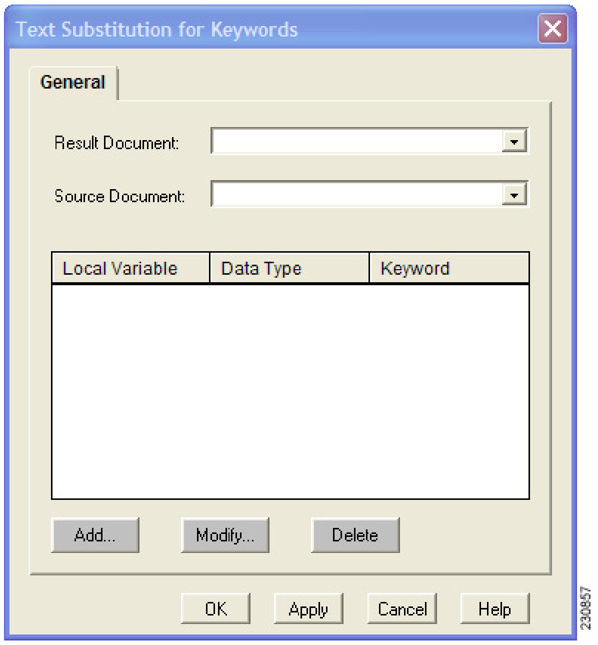


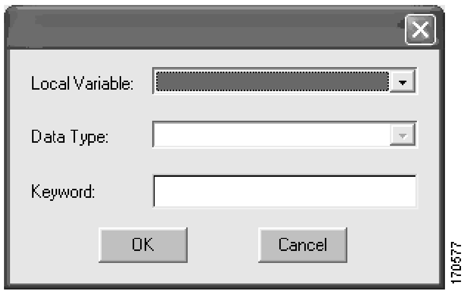
Table 28 describes the fields of the **Text Substitution for Keywords** customizer window.

Table 28 Text Substitution for Keywords Customizer Window Fields

Property	Description
Result Document	Variable that stores the document containing the text substitutions.
Source Document	Variable that identifies the source template document. The lower half of the window can be modified to configure the local variables that will replace the specific keywords in the document template. These can be added, modified, or deleted as desired.
Local Variable	Variable used to replace the keyword.
Data Type	Data type of the local variable, such as string and integer. This field is populated with the data type of the local variable selected automatically.
Keyword	Keyword in the source document indicated by a percent (%) character that is to be replaced. When entering a keyword, in this field do not use the % character. Keywords should contain only alphanumeric characters. Dash (–) and underscore (_) characters are permitted.
Add	Adds a new replacement entry. When **Add** is clicked, the **Keyword Mapping** dialog box appears, as shown in Figure 73.
Modify	Modifies an existing entry. When **Modify** is clicked, the **Keyword Mapping** dialog box appears, as shown in Figure 73.
Delete	Deletes an entry.

The **Keyword Mapping** dialog box, as seen in Figure 73 appears when **Add** or **Modify** is clicked in the **Text Substitution for Keywords** customizer window, as seen in Figure 72.

Figure 73 Keyword Mapping Dialog Box



eMail Contact Steps (IVR Only)

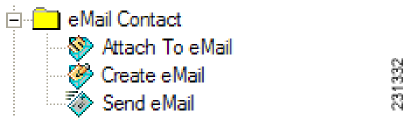
The steps in the eMail Contact palette provide the facility for creating, attaching, and sending e-mail. The e-mail steps are available in the e-mail palette in the Cisco Unity Express Script Editor. For more information about the e-mail contact steps, see the *Cisco Unity Express 7.0 IVR CLI Administrator Guide*.

The eMail Contact palette contains the following steps:

- Attach To eMail, page 90
- Create eMail, page 92
- Send eMail, page 93

Figure 74 shows the steps in the eMail Contact Palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 74 eMail Contact Palette Steps



Attach To eMail

Use the **Attach to eMail** step to attach an existing document to an e-mail to be sent at a later time.

Note that the e-mail must have been previously created with the **Create eMail** step (see the "Create eMail" section on page 92) and matched to the corresponding e-mail contact. The contents of the attachment will be specified by a document variable.

The **Attach To eMail** step produces two output branches:

- **Successful**: The document was attached to the e-mail object.
- **Failed**: The document could not be attached to the e-mail object.

To attach an e-mail, click **Add** in the **Attach to eMail** customizer window, as shown in Figure 75, and complete the **Add Attachment** customizer window as seen in Figure 76.

Figure 75 Attach to eMail Customizer Window

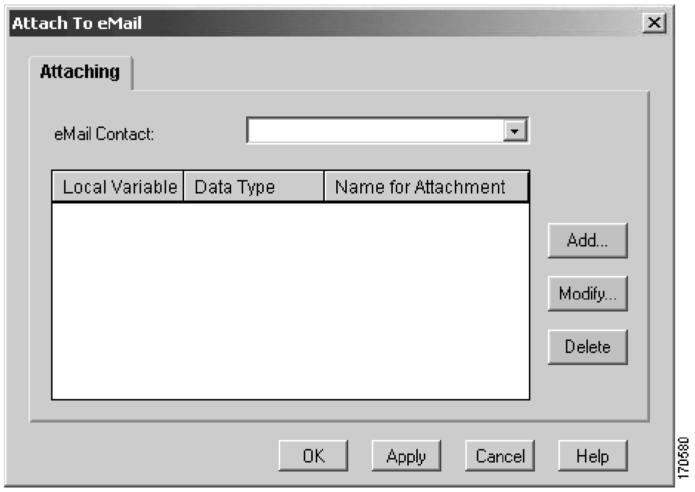


Figure 76 Add Attachment Customizer Window

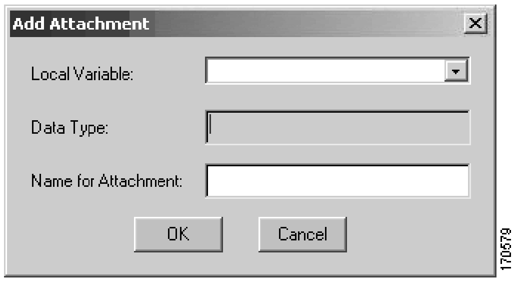


Table 29 describes the fields of the **Attach to eMail** and **Add Attachment** customizer windows.

Table 29 Attach to eMail and Add Attachment Customizer Window Fields

Field	Description
eMail Contact	Contact associated with the e-mail message that was configured in the **Create eMail** step (see the "Create eMail" section on page 92). The name can be entered directly as text in quotations or with a string variable.
Local Variable	Specifies the document to be sent as an e-mail attachment.
Data Type	Data type of the local variable. The default data type is Document. After a local variable is selected, Document appears in the Data Type field automatically.
Name for attachment	Text name of the e-mail attachment.
Add	Adds an attachment. A maximum of five attachments can be added.
Modify	Modifies an attachment.
Delete	Deletes an attachment.

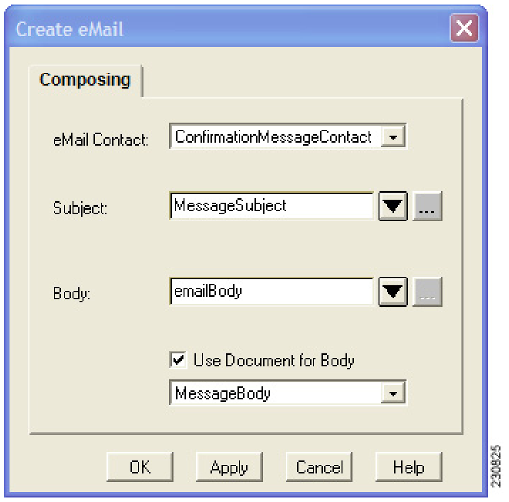
Create eMail

Use the **Create eMail** step to compose an e-mail. Include the subject and the text body of the e-mail to be associated with the e-mail contact. To specify the body of the e-mail:

1. Choose the variable you want to use in the Body field for the body of the e-mail message.
2. Click "..." to the right of the Body field, and then enter a string expression into the popup Expression Editor window.
3. If you want to use a separate document for the body of the message, check the Use Document for Body check box, and then choose a document variable from the drop-down list to the right of the message field.

Figure 77 shows the customizer window for the **Create eMail** step.

Figure 77 Create eMail Customizer Window



eMail Contact: ConfirmationMessageContact
Subject: MessageSubject
Body: emailBody
[✓] Use Document for Body
MessageBody

Table 30 describes the fields of the **Create eMail** customizer windows.

Table 30 Create eMail Customizer Window Fields

Field	Description
eMail Contact	Variable that describes the e-mail.
Subject	Description of the e-mail subject. This can be entered with a reference to a string variable or directly with text in quotations. The subject string must not exceed 256 characters. If a string larger that 256 characters is entered, it will be truncated following the 256th character.
Body	Text body of the e-mail message. This can reference a string variable, a document variable, or a direct entry of text in quotations. If a document variable is selected, the reference document should contain only text. If the referenced document contains binary data, such as .JPEG and .GIF file, the system attempts to determine the content type of the document, and the e-mail might not appear properly when it is received. Choose the Use Document for Body to use a separate document for the e-mail body. When the box is checked, you must select a document variable listed in the drop-down list.

Send eMail

Use the **Send eMail** step to send an e-mail.

Prior to sending an e-mail, the e-mail must have been created with the Create eMail step (see the "Create eMail" section on page 92).

The **Send eMail** step sets the To and From addresses and sends the e-mail to the configured Simple Mail Transfer Protocol (SMTP) server for delivery. The SMTP server can be configured using the Cisco Unity Express GUI or CLI. The From address can be set to any desirable address to which a reply can be sent. This address can also be advised of undeliverable messages received from the SMTP server.

E-mail can be sent immediately or queued to be sent later as a background process.

- **Immediate**: If the immediate option is selected, the e-mail will be sent to the SMTP server, and the client will be kept waiting until the message is accepted by the SMTP server. If the server is unavailable because of server or network problems, the client must wait until the transaction times out (30 seconds). It may be necessary to issue a prompt to alert the client that the e-mail is being sent, and to hold the line until the transmission of the e-mail message is complete.
- **Queued**: If the queued option is selected, the step will return immediately, but the script does not receive notification of whether e-mail was sent successfully or not.

The **Send eMail** step generates the following three branches:

- **Successful**: The e-mail was sent and its transmission completed.
- **Failed**: The e-mail could not be sent. The failure reasons can be that the SMTP server was not properly configured, SMTP authentication failed, or the SMTP server was not reachable and caused the transaction to time out.

Figure 78 shows the customizer window for the **Send eMail** step.

Figure 78 Send eMail Customizer Window

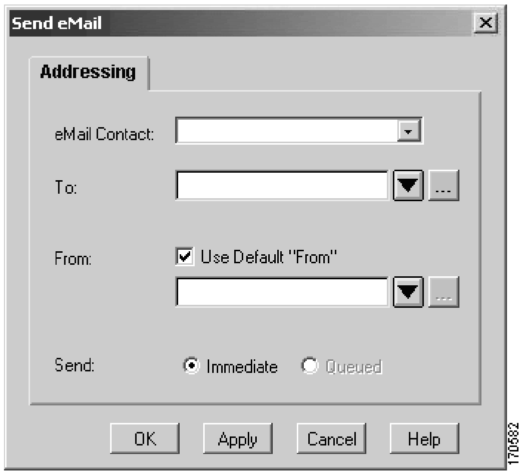


Table 31 describes the fields of the **Send eMail** customizer windows.

Table 31 Send eMail Customizer Window Fields

Field	Description
eMail Contact	Contact associated with the e-mail message that was configured in the **Create eMail** step (see the "Create eMail" section on page 92). May be entered directly as text in quotations or with a string variable.
To	Person to whom you are sending the e-mail. A variable or string expression can be used.
Use Default From Check Box	• **Checked box**: Causes Cisco Unity Express to use the default account for the e-mail subsystem that was configured through the Cisco Unity Express GUI or CLI. • **Unchecked box**: Requires the entry or selection of a From e-mail address.
From	Account from which you are sending the e-mail. This may be entered directly as text in quotations or with a string variable.
Send	Method of fax transport. • **Immediate**: Sends the fax immediately to the SMTP server and waits until the message is accepted by the SMTP server. • **Queued**: Sends fax in a separate background process.

Fax Contact Steps (IVR Only)

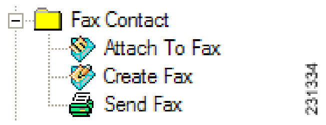
The steps in the **Fax Contact** palette of the Cisco Unity Express Script Editor provide fax programming functionality for scripting. For more information about the fax contact steps, see the *Cisco Unity Express 7.0 IVR CLI Administrator Guide*.

The **Fax Contact** palette contains the following steps:

- Create Fax, page 94
- Attach to Fax, page 95
- Send Fax, page 96

Figure 79 shows the steps in the **Fax** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 79 Fax Contact Palette Steps



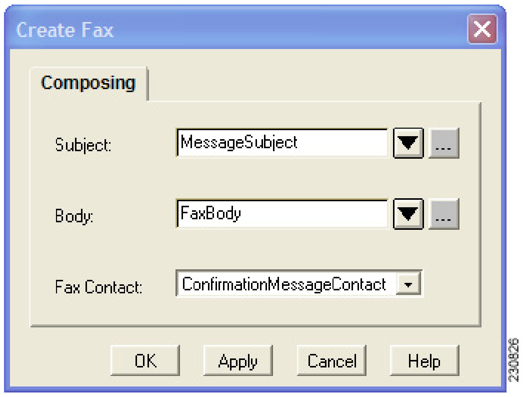
Create Fax

Use the **Create Fax** step to construct a fax message.

Creating a fax involves entering the subject and the body of the fax and associating an e-mail contact with the fax.

Figure 80 shows the customizer window for the **Create Fax** step.

Figure 80 Create Fax Customizer Window



Subject: MessageSubject
Body: FaxBody
Fax Contact: ConfirmationMessageContact

Table 32 describes the fields of the **Create Fax** customizer window.

Table 32 Create Fax Customizer Window Fields

Property	Description
Subject	Subject of the fax. A string variable or text in quotations can be entered.
Body	Body of the fax message. A string variable or text in quotations can be entered. The body of the fax is limited to 1 KB. If a document variable is selected, the referenced document should contain text only; otherwise, the fax transmission will fail. The **Create Fax** step does not check the bodies of faxes for binary data.
Fax Contact	Variable that identifies the fax.

Attach to Fax

Use the **Attach to Fax** step to send either a text document or a .TIFF image as an attachment to a fax.

Prior to attaching a fax, you must first create a fax using the **Create Fax** step (see the "Create Fax" section on page 94) and use the same fax contact. The contents of the attachment are identified with a document variable entered in the Local Variable field. The file referenced in the Local Variable field can contain either text or a .TIFF image. The .TIFF image must conform to Profile-F as defined in RFC2306. If the attachment is not a text or a .TIFF image, the fax is not sent. If the .TIFF image does not conform to the Profile-F requirements, the fax will be sent but the image might not be transmitted correctly.

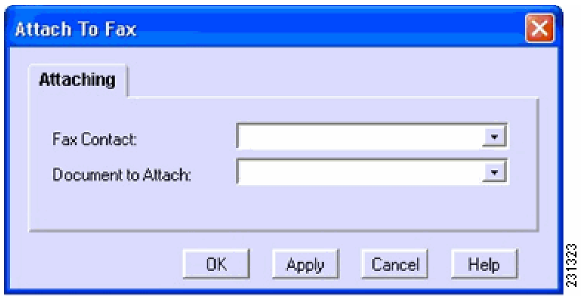
Note The **Send Fax** step does not validate the formats of attached documents.

The **Attach to Fax** step generates the following two branches:

- **Successful**: The document was attached to the fax object.
- **Failed**: The document was not attached to the fax object.

Figure 81 shows the customizer window for the **Attach to Fax** step.

Figure 81 Attach to Fax Customizer Window



Fax Contact:

Document to Attach:

Table 33 describes the fields of the **Attach to Fax** customizer window.

Table 33 Attach to Fax Customizer Window Fields

Property	Description
Fax Contact	Contact associated with the fax message created in the **Create Fax** step (see the "Create Fax" section on page 94).
Document to Attach	Local variable referencing the document to be sent as a fax attachment. This is restricted to a document data type containing either ASCII text or a .TIFF image.

Send Fax

Use the **Send Fax** step to send a fax.

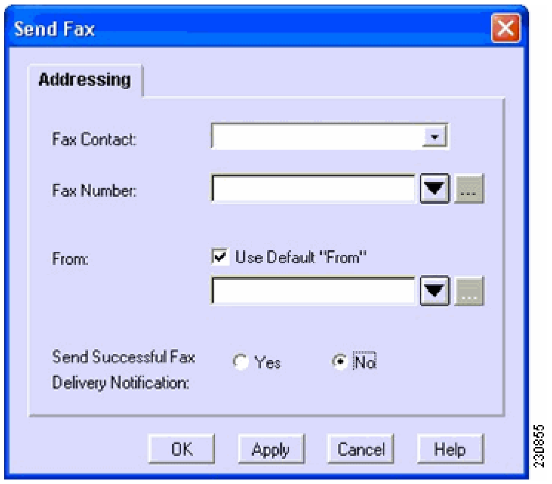
The Send Fax step returns to the steps immediately after queuing the fax request. Prior to attaching a fax, you must first create a fax using the Create Fax step (see the "Create Fax" section on page 94) and use the same fax contact. The fax may also have an attachment. See the "Attach to Fax" section on page 95.

The **Send Fax** step produces two output branches:

- **Successful**: The fax was successfully queued to be sent to the Fax SMTP server.
- **Failed**: The fax could not be queued for sending.

Figure 82 shows the customizer window for the **Send Fax** step.

Figure 82 Send Fax Customizer Window



Fax Contact:
Fax Number:
From: ☑ Use Default "From"
Send Successful Fax Delivery Notification: Yes [✓ No]

Table 33 describes the addressing fields of the **Send Fax** customizer window.

Table 34 Send Fax Customizer Window Fields

Property	Description
Fax Contact	Contact associated with the fax message that was created in the **Create Fax** step (see the "Create Fax" section on page 94).
Fax Number	Variable or string expression to use for the phone number of the intended fax machine. This may be entered directly as text in quotations or with a string variable. This number must include any necessary digits used to make an outbound call.
***Default From* Check Box**	Enables or disables the use of the account configured for the fax subsystem through the Cisco Unity Express GUI or CLI. • **Checked Box**: Enables the use of the account configured for the Fax subsystem through the Cisco Unity Express GUI or CLI. If a *Default From* value has not been configured, the *From* field in the sent Fax e-mail will be set to localhost@<*localhostname*>. If the fax is sent successfully, then the localhost@<*localhostname*> address will be used as the sender e-mail address in the sent fax. If the fax is not sent successfully, then a failure notification will be sent to the default "From" address. • **Unchecked Box**: Disables the use of the account configured for the Fax subsystem through the Cisco Unity Express GUI or CLI.
From	Account from which the e-mail will be sent. Text in quotations, or a string variable can be used. The address entered into this field must be one that is capable of receiving e-mails.
Send Fax Successful Notification Radio Buttons	Does not send notification e-mails on successful fax transmission. • **Yes**: The notification e-mail is sent. • **No**: The notification e-mail is not sent.

General Steps

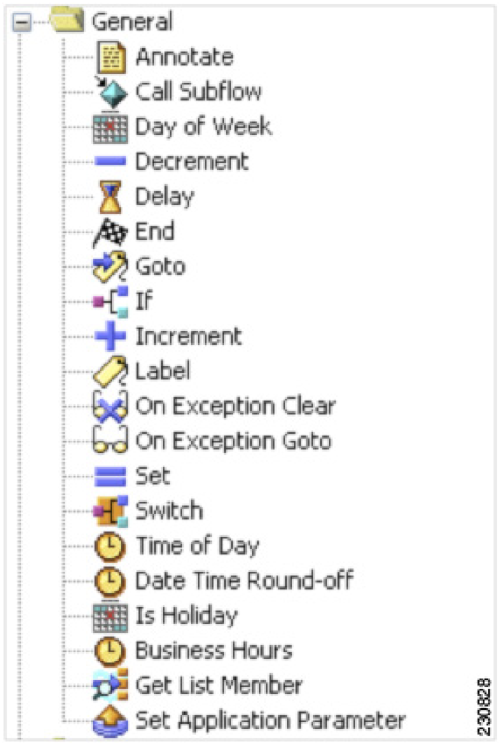
The steps in the **General** palette of the Cisco Unity Express Script Editor provide basic programming functionality for scripting.

The **General** palette contains the following steps:

Figure 83 shows the steps in the **General** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Note The Set Application Parameter step is only available in Cisco Unity Express 8.0 and later versions.

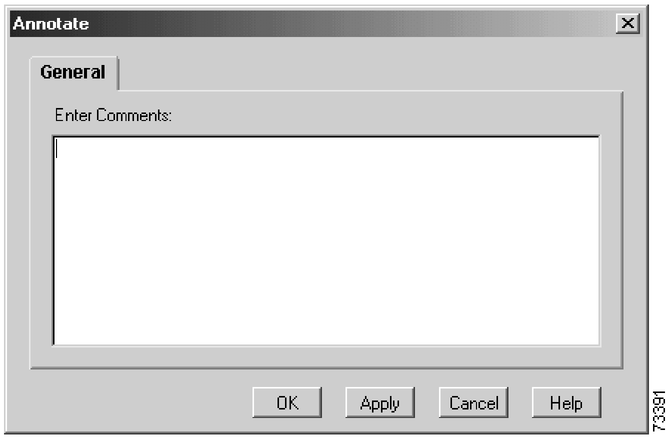
Figure 83 General Palette Steps



Annotate

Use the **Annotate** step to enter comments at any point in a script. This step has no impact on script logic. Figure 84 shows the customizer window for the **Annotate** step.

Figure 84 Annotate Customizer Window



To annotate a script, enter your comments in the **Enter Comments** field and click **OK**.

The **Annotate** customizer window closes and the first words of your annotation appear next to the **Annotate** icon in the Design pane of the Cisco Unity Express Script Editor.

Business Hours

Use the **Business Hours** step to determine if the business is open when the auto-attendant receives a call. This step can play a "Business is closed" prompt if the system receives a call during business- closed hours. When configured, the name of the **Schedule** variable appears next to the **Business Hours** step in the Design pane.

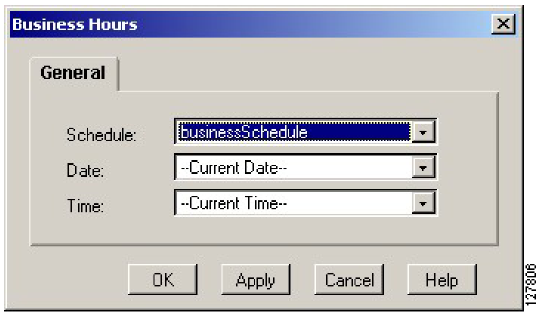
The **Business Hours** step automatically adds two output branches:

- **Open**: Steps following this branch execute if the business is open at the specified date and time.
- **Closed**: Steps following this branch execute if the business is closed at the specified date and time.

For more information about configuring business schedules, see the *Cisco Unity Express 7.0 GUI Administrator Guide* or the *Cisco Unity Express Voice-Mail and Auto-Attendant CLI Administrator Guide for 3.0 and Later Versions*..

Figure 85 shows the customizer window for the **Business Hours** step.

Figure 85 Business Hours Customizer Window



Schedule: businessSchedule
Date: --Current Date--
Time: --Current Time--

Table 35 describes the fields of the **Business Hours** customizer window.

Table 35 Business Hours Customizer Window Fields

Field	Description
Schedule	Name of the Business Hours Schedule. Select one of the schedules you created or customized using the Cisco Unity Express GUI options or CLI commands.
Date	Current date requires no configuration. You can also select a date variable for which you want to check the business hours.
Time	Current time requires no configuration. You can also select a time variable for which you want to check the business hours.

Call Subflow

Use the **Call Subflow** step to execute a subflow, also called a subroutine or module in structured programming.

Use the Cisco Unity Express Script Editor to create the subflow as an independent script that you can reuse in other scripts. Subflows can be nested; that is, you can call subflows from within scripts that are themselves used as subflows.

During run time, if an exception occurs within a subflow and you do not handle the exception within the subflow, the exception is available to the parent script for processing. For more information about exceptions, see the "On Exception Goto" section on page 110.

The **Call Subflow** customizer window contains two tabs:

- General Tab, page 101
- Parameter Mapping Tab, page 101

General Tab

Use the **General** tab of the **Call Subflow** customizer window, shown in Figure 86, to specify the filename of the subflow you want to call.

Figure 86 Call Subflow Customizer Window: General Tab

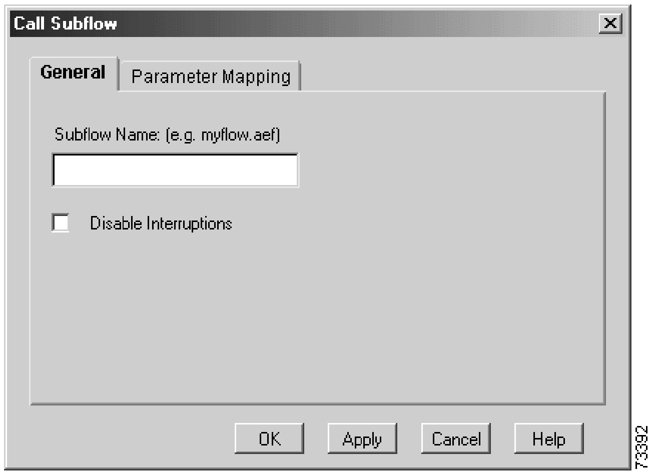


Table 36 describes the fields of the **General** tab.

Table 36 Call Subflow Customizer Window Fields: General Tab

Field	Description
Subflow Name	Filename of the subflow you want to call. This name appears next to the **Call Subflow** step icon in the Design pane.
Disable Interruptions	If the check box is checked, execution of the step cannot be interrupted by external events.

Parameter Mapping Tab

Use the **Parameter Mapping** tab of the **Call Subflow** customizer window, shown in Figure 87, to map variables or expressions from the main script to variables in the subflow you specified in the **General** tab of the **Call Subflow** customizer window.

Note You must define variables in the map script before you can map them.

You can map variables only to variables of the same type. For example, you can map a string variable in the main script only to a string variable in the subflow.

You can pass in any valid expression; for example, “4” or an expression such as “counter + 3”.

When the script calls a subflow, the subflow has access to the variables from the main script that you specify on the **Parameter Mapping** tab. If the subflow changes the value of a mapped variable, that change carries over to the main script when the subflow returns control to the main script.

Figure 87 Call Subflow Customizer Window: Parameter Mapping Tab

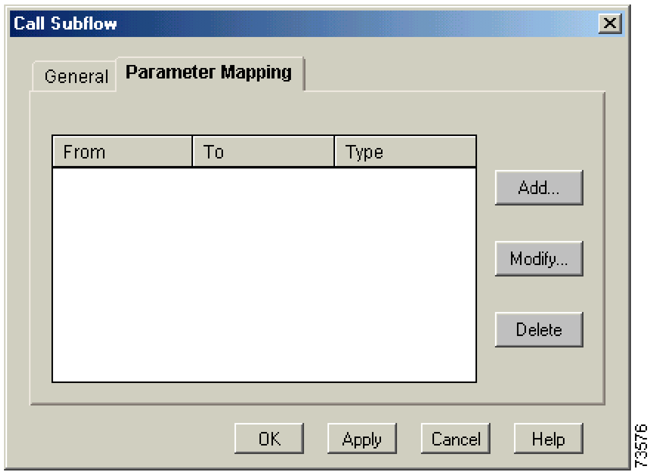


Table 37 describes the fields of the **Parameter Mapping** tab.

Table 37 Call Subflow Customizer Window Fields: Parameter Mapping Tab

Field	Description
From	Displays the name of the variable from the main script that receives the value of the variable from the subflow script.
To	Displays the name of the variable from the subflow script that is assigned to the variable in the main script.
Type	Displays the variable type.
Add	Click to add a variable from the main script and map it to map to a subflow. A separate dialog box appears.
Modify	Click to modify the selected variable.
Delete	Click to delete the selected variable.

The **Parameter Mapping** dialog box is shown in Figure 88.

Figure 88 Parameter Mapping Dialog Box

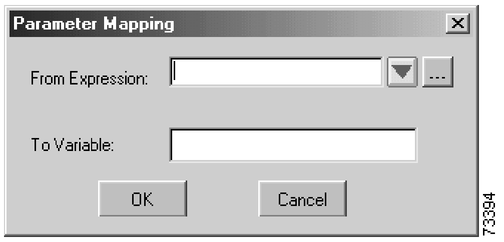


Table 38 Parameter Mapping Dialog Box

Field	Description
From Expression	Enter the variable name or expression from the main script.
To Variable	Enter the variable name from the subflow. This name appears in the list box of the **Call Subflow** window.
...	Click to display the Expression editor.

Date Time Round-off

Use the **Date Time Round-off** step utility to round up or round down the current date and time. When the Cisco Unity Express system clock matches the approximate time of day with a rounding value set, the time associated with a connection including the rounding value, the script executes any steps configured for that output branch.

In the following examples:

- Time 6:43 is rounded up to 6:45 if the rounding setting on this step is set to 15 minutes.
- Time 6:43 is rounded up to 7:25 if the rounding setting on this step is set to 25 minutes.
- Time 6:43 is rounded up to 8:00 if the rounding setting on this step is set to 2 hours.

Day of Week

Use the **Day of Week** step to direct the script to different connection output branches depending on the current day of the week.

When the Cisco Unity Express system clock matches one of the days associated with a connection, the script executes any steps that you configured for that day's connection branch.

Configure all days with output branches and assign each day its own connection(s). If a day is not assigned to at least one output branch, the Cisco Unity Express Script Editor displays a warning dialog box when you close the **Day of Week** customizer window.

Figure 89 shows the customizer window for the **Day of Week** step.

Figure 89 Day of Week Customizer Window

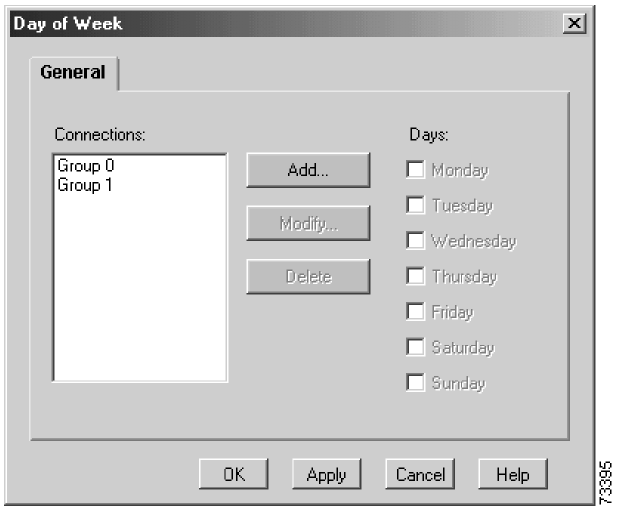


Table 39 describes the fields of the **Day of Week** customizer window.

Table 39 Day of Week Customizer Window Fields

Field	Description
Connections	Output branches that execute depending on a specified day of week. Select the connection in the **Connections** list box and check the check boxes for the days you want to associate with that branch.
Days	Days of the week for each connection branch.
Add	Click to add a connection name.
Modify	Click to modify the selected connection name. To modify the name of an already existing connection output branch to make it easier to understand your script, for example, select the connection in the **Connections** list box, and then click **Modify**. The **Modify Connection Name** dialog box contains the same field as the **Add Connection Name** dialog box and is configured in the same way.

The **Add Connection Name** dialog box is shown in Figure 90, and the field is described in Table 39.

Figure 90 Add Connection Name Dialog Box

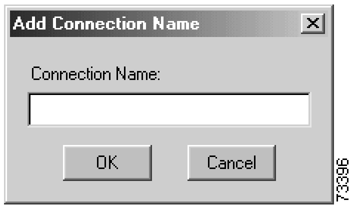


Table 40 Add Connection Name Dialog Box

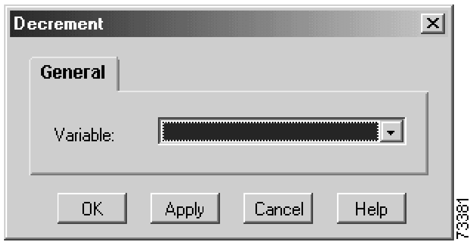
Field	Description
Connections Name	Enter a name for the connection branch. This name appears in the **Connections** list box of the **Day of Week** customizer window.

Decrement

Use the **Decrement** step to decrease the value of a chosen Integer variable by one. This step is a specialized version of the **Set** step of the **General** palette, which you use to assign any value to a variable.

Figure 91 shows the customizer window for the **Decrement** step.

Figure 91 Decrement Customizer Window



Select the desired variable from the **Variable** drop-down menu. The variable appears next to the **Decrement** step icon in the Design pane.

Delay

Use the **Delay** step to pause the processing of a script for a specified number of seconds.

Figure 92 shows the customizer window for the **Delay** step.

Figure 92 Delay Customizer Window

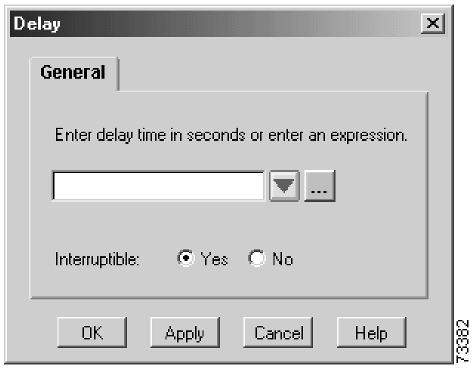


Table 41 describes the fields of the **Delay** customizer window.

Table 41 Delay Customizer Window Fields

Field	Description
Enter delay time in seconds or enter an expression	Enter either the length of time, in seconds, for the delay, or an expression that specifies the length of the delay.
...	Displays the expression editor. Enter an expression that specifies the length of the delay.
Interruptible	Click **Yes** to interrupt the delay by external events.

End

Use the **End** step at the end of a script to complete processing and free all allocated resources.

You can also use the **End** step at the end of a branch of logic in a script. Any call still active by the time this step is executed will automatically be processed by the system default logic.

This step has no properties and does not require a customizer.

Get List Member (IVR Only)

The **Get List Member** step retrieves a member (token) from a specific position (index) out of a delimiter separated string list.

You can use this step to parse string data retrieved from the database and to extract a portion of it. The extracted member (token) can then be used by other steps. For example, a database query might retrieve all the active prescription numbers for a patient as a comma-separated list. This step can be used to extract individual prescription numbers out of that list, and then each of these prescription numbers can be played to the caller using the Play Prompt step, prompting them to select one of them for refill.

The **Get List Member** step produces two output branches:

- **Success**: The data was retrieved successfully and placed in the appropriate output string.
- **Invalid Position**: The specified position (index) from which to extract the data was invalid. If the members are being retrieved in a loop, this may signify the end of the loop.

Figure 93 shows the customizer window for the **Get List Member** step.

Figure 93 Get List Member Window

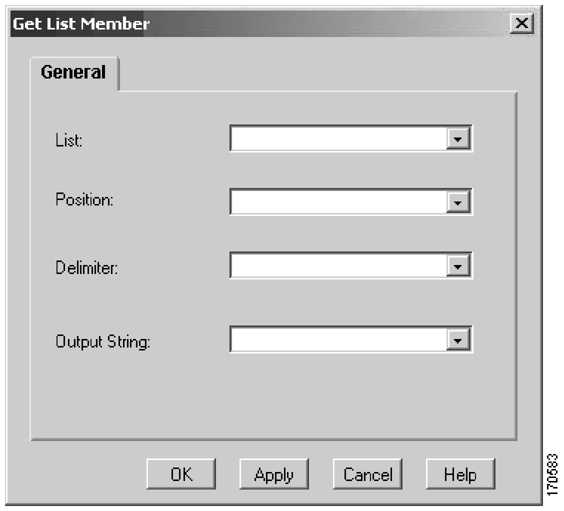


Table 42 describes the fields of the **Get List Member** customizer window.

Table 42 Get List Member Customizer Window Fields

Field	Description
List	Variable containing the string list.
Position	Variable containing the position (index) from which the string member is to be extracted.
Delimiter	Variable specifying the delimiter separating the members in the string list.
Output String	Variable that stores the extracted output string from the position (index) in the string list, if the step is successful.

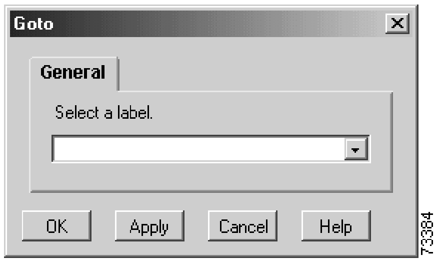
Goto

Use the **Goto** step to cause the script logic to branch to a specified **Label** step within the script.

Note Use a **Label** step to indicate where the **Goto** step should branch to.

Figure 94 shows the customizer window for the **Goto** step.

Figure 94 Goto Customizer Window



Select the Label step from the **Select a Label** drop-down menu. This name appears next to the **Goto** step icon in the Design pane.

If

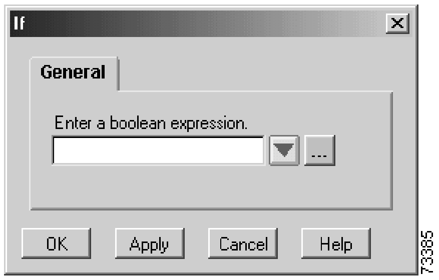
Use the **If** step to cause the script to choose one of two branches based on the evaluation of a specified Boolean expression.

The **If** step automatically adds two output branches, **True** and **False**:

- **True**: Steps following this output branch execute if the expression is true.
- **False**: Steps following this output branch execute if the expression is false.

Figure 95 shows the customizer window for the **If** step.

Figure 95 If Customizer Window



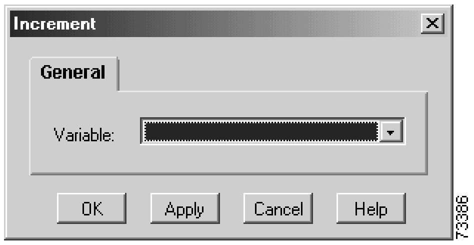
Enter an expression in the text field or click the **Expression Editor** (**...**) button to enter an expression. The expression appears next to the **If** step icon in the Design pane.

Increment

Use the **Increment** step to increase the value of a chosen Integer variable by one. This step is a specialized version of the **Set** step of the **General** palette, which you use to assign any value to a variable.

Figure 96 shows the customizer window for the **Increment** step.

Figure 96 Increment Customizer Window



Select the Integer type variable from the **Variable** drop-down menu. The variable appears next to the Increment step icon in the Design pane.

Is Holiday

The **Is Holiday** step is a conditional step which branches the flow according to the predefined holidays. This step can play a “Business closed for the holiday” prompt.

The **Is Holiday** step automatically adds two output branches:

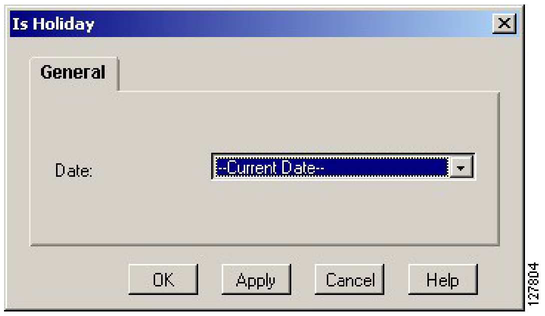
- **True**: Steps following this branch execute if the specified date is a holiday.
- **False**: Steps following this branch execute if the specified date is not a holiday.

When the step executes, the system compares the specified date with the list of holidays. If the specified date is a holiday, the **True** branch executes. If the specified date is not a holiday, the **False** branch executes.

For more information about configuring holiday schedules, see the *Cisco Unity Express 7.0 GUI Administrator Guide* or the *Cisco Unity Express Voice-Mail and Auto-Attendant CLI Administrator Guide for 3.0 and Later Versions*.

Figure 97 shows the customizer window for the Is Holiday step.

Figure 97 Is Holiday Customizer Window



Date: --Current Date--

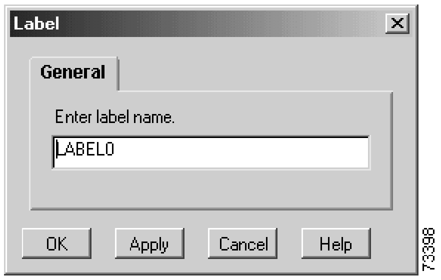
The **Date** field contains the date variable that the system uses to check for holidays. You can leave the **Current Date** variable or click the **Date** drop-down menu to select a date variable. The name of the **Date** variable appears next to the **Is Holiday** step icon in the Design pane.

Label

Use the **Label** step to serve as a target for a **Goto** step. The Label step indicates a section of the script that can be executed by more than one Goto step. The Label and Goto steps must be in the same script.

Figure 98 shows the customizer window for the **Label** step.

Figure 98 Label Customizer Window



Enter label name.
LABEL0

Enter a name in the **Enter Label Name** text field. The Label name appears next to the **Label** step icon in the Design pane.

On Exception Clear

Use the **On Exception Clear** step to remove an exception set by a previous **On Exception Goto** step.

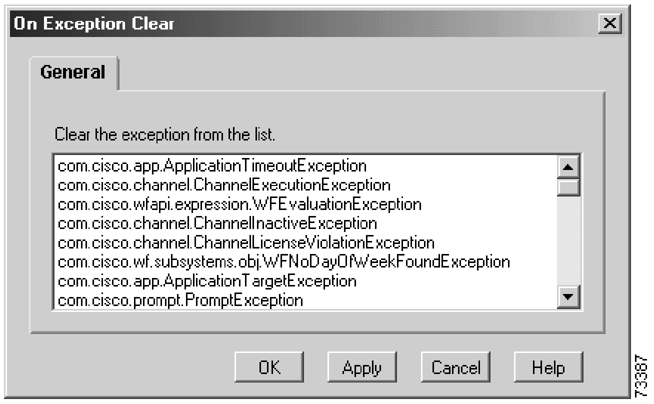
Typically, this step is used in the following sequence:

1. An **On Exception Goto** step directs the script to a **Label** step.
2. The **Label** step is configured with a script to handle the exception.
3. An **On Exception Clear** step clears the exception.

Also, use this step when you no longer need to handle the selected exception within the script.

Figure 99 shows the customizer window for the **On Exception Clear** step.

Figure 99 On Exception Clear Customizer Window



Clear the exception from the list.

com.cisco.app.ApplicationTimeoutException
com.cisco.channel.ChannelExecutionException
com.cisco.wfapi.expression.WFEvaluationException
com.cisco.channel.ChannelInactiveException
com.cisco.channel.ChannelLicenseViolationException
com.cisco.wf.subsystems.obj.WFNoDayOfWeekFoundException
com.cisco.app.ApplicationTargetException
com.cisco.prompt.PromptException

Select the specific exception from the list box. The exception being cleared appears next to the **On Exception Clear** step icon in the Design pane.

On Exception Goto

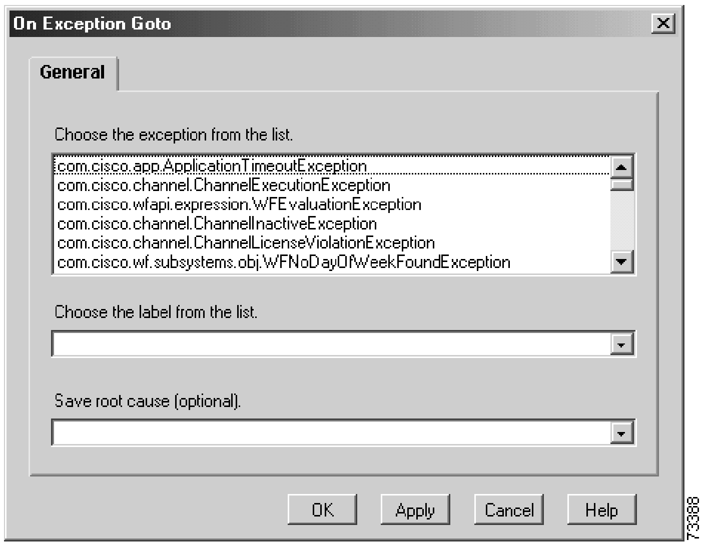
Use the **On Exception Goto** step to catch problems occurring during script execution and allow a graceful exit from the situation.

You can include any script steps in the Exception Flow branch that you want to use to respond to the exception.

If you are using subflows and the subflow does not handle an exception, the exception is returned to the script and the script can respond to it.

Figure 100 shows the customizer window for the **On Exception Goto** step.

Figure 100 On Exception Goto Customizer Window



Choose the exception from the list.

com.cisco.app.ApplicationTimeoutException
com.cisco.channel.ChannelExecutionException
com.cisco.wfapi.expression.WFEvaluationException
com.cisco.channel.ChannelInactiveException
com.cisco.channel.ChannelLicenseViolationException
com.cisco.wf.subsystems.obj.WFNoDayOfWeekFoundException

Choose the label from the list.

Save root cause (optional).

Table 43 describes the fields of the **On Exception Goto** customizer window.

Table 43 On Exception Goto Fields

Field	Description
Choose the exception from the list	Exception that triggers the execution of the step. This appears next to the **On Exception Goto** step icon in the Design pane.
Choose the label from the list	Label to which the script branches.
Save root cause (optional)	Cause of the exception, saved in an exception object using the "Save root cause" field. The object type must correspond to the type of exception being caught or to a base class of that exception. If it does not, no warning will be generated at design time, and an error will result at run time.

Set

Use the **Set** step to change the value of a variable.

The **Set** step supports type casting (with possible loss of precision) from any Number data type (Integer, Float, Long, Double, BigInteger, BigDecimal) to any other Number data type.

You can also use the **Set** step to convert a String variable to any Number data type. For String conversions, the system replaces all "*" characters with a decimal point (".") before performing the conversion.

Figure 101 shows the customizer window for the **Set** step.

Figure 101 Set Customizer Window

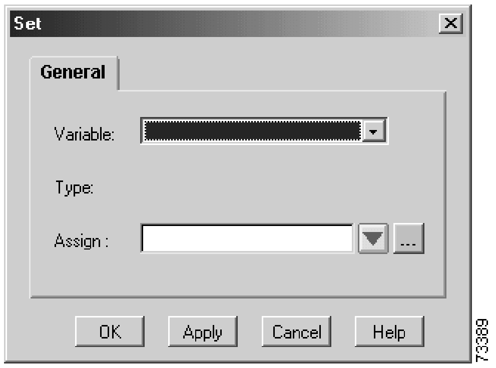


Table 44 describes the fields of the **Set** customizer window.

Table 44 Set Customizer Window Fields

Field	Description
Variable	Variable for which the value is set. This appears next to the **Set** step icon in the Design pane.
Type	Variable type. The application software assigns this value.
Assign	Value for the specified variable. Choose the value from the **Assign** drop-down menu, or click the **Expression Editor** (**...**) button to enter any valid expression.

Set Application Parameter

Prerequisite: Cisco Unity Express 8.0 or Later

Use the Set Application Parameter step to configure application parameters using the TUI. This step is equivalent to setting the script parameter using the **ccn application** commands. Any script parameter can be set using the Set Application Parameter step.

Note The sample script “ sampleSetAppParams.aef” included in the script editor package provides information on how to use the Set Application Parameter step.

Figure 102 shows the customizer window for the Set Application Parameter step.

Figure 102 Set Application Parameter Window

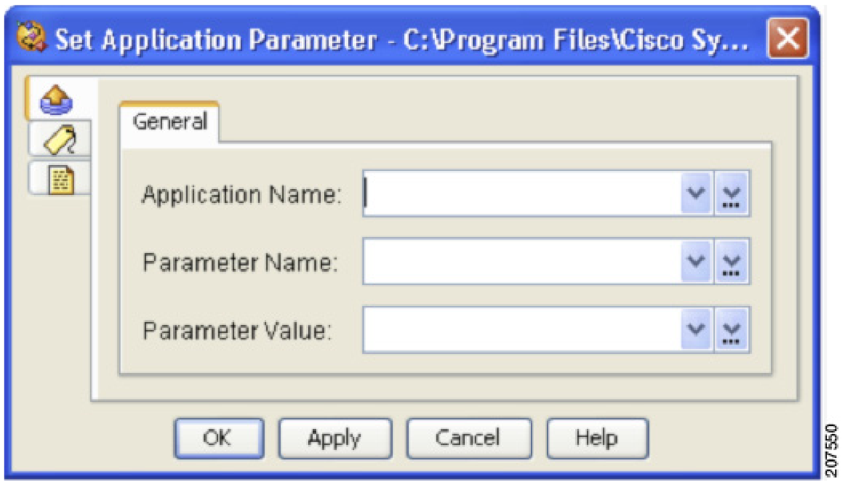


Table 45 describes the fields of the **Set Application Parameter** customizer window.

Table 45 Set Application Parameter Window Fields

Field	Description
Application Name	Value for the specified script name. Corresponds to the *application-name* argument for the **ccn application** command.
Parameter Name	Value for the specified parameter name. Corresponds to the *parameter-name* argument for the **parameter (ccn application)** command.
Parameter Value	Value for the specified parameter. Corresponds to the *"value"* argument for the **parameter (ccn application)** command.

Start

The Cisco Unity Express Script Editor automatically adds the **Start** step when you create a new script. This step has no properties and does not require a customizer. It is not shown in any palette.

Switch

Use the **Switch** step to cause the program logic to branch to one of a number of cases based on the evaluation of a specified expression.

A *case* provides script logic based on the value of a variable. A variable can have one of several values. You can assign one case for each value. The **Switch** step lets you define any number of case output branches. You can then create a separate script logic for each (case) branch.

The **Switch** step supports switching based on the following variables:

- Integer: Comparison of integers.
- String: Comparison of string variables (case insensitive).

The type of switching is automatically determined by the type of the specified expression.

If the integer or string expression you specify for a case is equal to the global expression defined in the **Switch Expression** field, the script executes the steps configured for that case output branch.

The **Default** branch of the step allows you to handle cases where none of the branches matches the expression.

Figure 103 shows the customizer window for the **Switch** step.

Figure 103 Switch Customizer Window

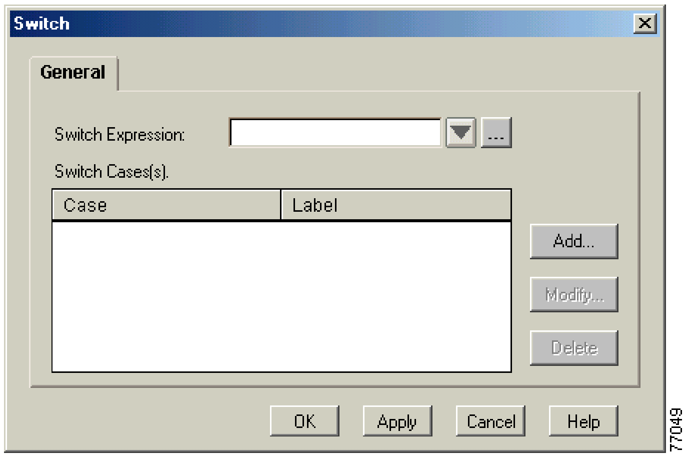


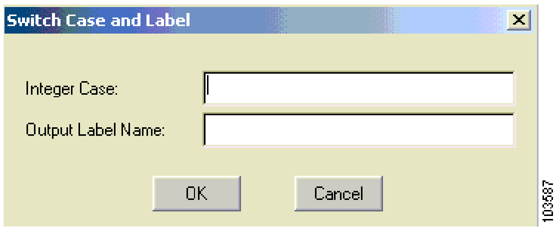
Table 46 describes the fields of the **Switch** customizer window.

Table 46 Switch Customizer Window Fields

Field	Description
Switch Expression	Expression to be executed. Select a variable or expression, or click the **Expression Editor** (**...**) button to enter any valid expression.
Switch Case(s)	• Case: Output branch containing script logic specific to one possible variable value. • Label: Target to which the script branches when the variable equals a specific value.
Add	Adds a case.
Modify	Modify a case.
Case	Integer or Case. Determined automatically by switch expression type.
Output Label Name	Enter an output label name. The script branches to this Label when the variable equals the value specified in the **Case** field.

The **Switch Case and Label** dialog box is shown in Figure 104.

Figure 104 Switch Case and Label Dialog Box



Time of Day

Use the **Time of Day** step to cause the script to branch to different connection branches depending on the current time of day.

When the Cisco Unity Express system clock indicates that the time of day matches the time associated with a connection, the script executes any steps configured for that output branch.

Associate each output branch with a specified range of time.

During run time, if the current time falls out of the configured time range, the script follows the **Rest** output branch of the **Time of Day** step.

Figure 105 shows the customizer window for the **Time of Day** step.

Figure 105 Time of Day Customizer Window

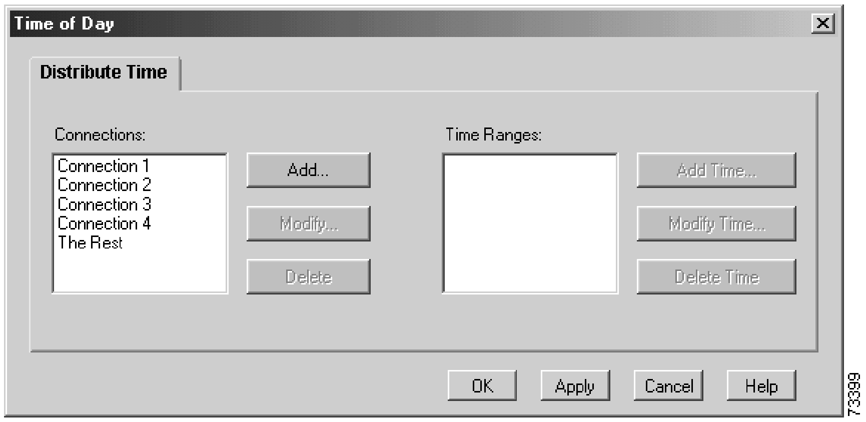


Table 47 describes the fields of the **Time of Day** customizer window.

Table 47 Time of Day Customer Window Fields

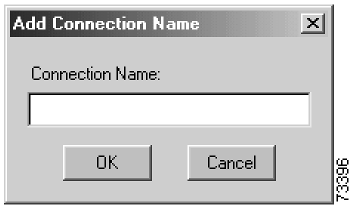
Field	Description
Connections	Output branches that execute depending on specified time of day.
Time Ranges	Time ranges for each connection branch.
Add	Click to add a connection. Enter a connection name in the dialog box.
Modify	Click to modify a connection.
Delete	Click to delete the selected connection.

Table 47 Time of Day Customer Window Fields (continued)

Field	Description
Add Time	Click to add time to a connection. To specify a range of hours for the connection, select the **Start Time** and **End Time**.
Modify Time	Click to modify time for a connection.
Delete Time	Click to delete the selected time.

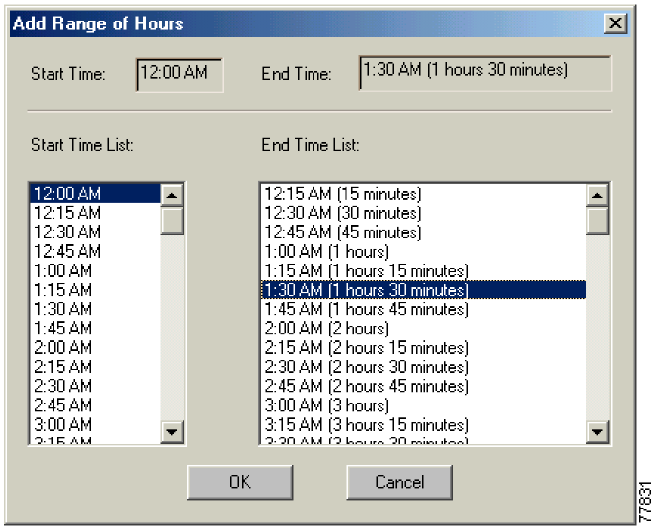
The **Add Connection Name** dialog box is shown in Figure 106.

Figure 106 Add Connection Name Dialog Box



The **Add Range of Hours** dialog box is shown in Figure 107.

Figure 107 Add Range of Hours Dialog Box


Start Time: 12:00 AM
End Time: 1:30 AM (1 hours 30 minutes)

Start Time List:

```
12:00 AM
12:15 AM
12:30 AM
12:45 AM
1:00 AM
1:15 AM
1:30 AM
1:45 AM
2:00 AM
2:15 AM
2:30 AM
2:45 AM
3:00 AM
```

End Time List:

```
12:15 AM (15 minutes)
12:30 AM (30 minutes)
12:45 AM (45 minutes)
1:00 AM (1 hours)
1:15 AM (1 hours 15 minutes)
1:30 AM (1 hours 30 minutes)
1:45 AM (1 hours 45 minutes)
2:00 AM (2 hours)
2:15 AM (2 hours 15 minutes)
2:30 AM (2 hours 30 minutes)
2:45 AM (2 hours 45 minutes)
3:00 AM (3 hours)
3:15 AM (3 hours 15 minutes)
```

HTTP Contact Steps (IVR Only)

The steps in the **HTTP Contact** palette of the Cisco Unity Express Script Editor allow you to receive HTTP requests and send HTTP responses in web-enabled server applications. For more information about the HTTP contact steps, see the *Cisco Unity Express 7.0 IVR CLI Administrator Guide*.

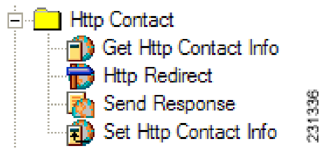
The **HTTP Contact** palette contains the following steps:

- Get Http Contact Info, page 117

- Http Redirect, page 123
- Send Response, page 123
- Set Http Contact Info, page 124

Figure 108 shows the steps in the **HTTP Contact** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 108 HTTP Contact Palette Steps



Get Http Contact Info

Use the **Get Http Contact Info** step to map parameters from an HTTP request to locally defined variables.

The **Get Http Contact Info** step obtains URL parameters, HTTP headers, cookies, or Common Gateway Interface (CGI) environment variables. URL parameters, header information, cookies, and CGI variables are mapped to locally defined variables.

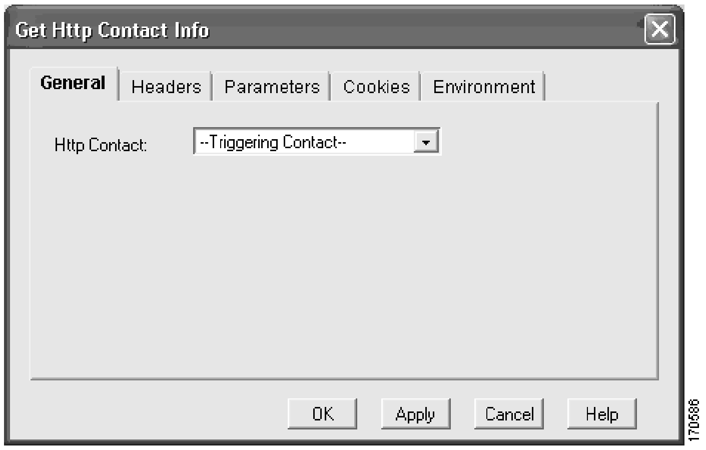
The **GET Http Contact Info** customizer window contains five tabs:

- General Tab, page 117
- Headers Tab, page 118
- Parameters Tab, page 119
- Cookies Tab, page 120
- Environment Tab, page 121

General Tab

Use the **General** tab of the **Get Http Contact Info** customizer window, shown in Figure 109, to select the type of variable to trigger the execution of the **Get Http Contact Info** step.

Figure 109 Get Http Contact Customizer Window: General Tab



Http Contact: --Triggering Contact--

Table 48 describes the field of the **General** tab.

Table 48 Get Http Contact Info Customizer Window Field: General Tab

Field	Description
Http Contact	Contact variable that triggers the execution of the step. The default is Triggering Contact, unless another contact is specified.

Headers Tab

Use the **Headers** tab of the **Get Http Contact Info** customizer window to display the HTTP headers that are mapped to local variables.

HTTP headers contain general information such as the type of browser or HTTP version. Each header provides one value, which is identified by the header name. For example, information from HTTP headers can be used in more complex scripts to customize the behavior of the script for different HTTP versions or for different browser types.

HTTP provides four types of headers:

- **General**: Used by both servers and clients (browsers)
- **Server**: Used only by servers
- **Request**: Used only by clients (browsers)
- **Entity**: Used by servers and by clients using POST or PUT methods

The following are common HTTP Request headers:

- **Accept**: Preferred media type
- **Authorization**: Client username and password
- **From**: E-mail address of the client
- **Host**: Hostname and port number of the server receiving the original request
- **Referrer**: URL of the source document
- **User-Agent**: Browser type

Note For detailed information about these or other headers, see any HTTP reference guide.

Figure 110 shows the customizer window for the **Headers** tab.

Figure 110 Get HTTP Contact Info Customizer Window: Headers Tab

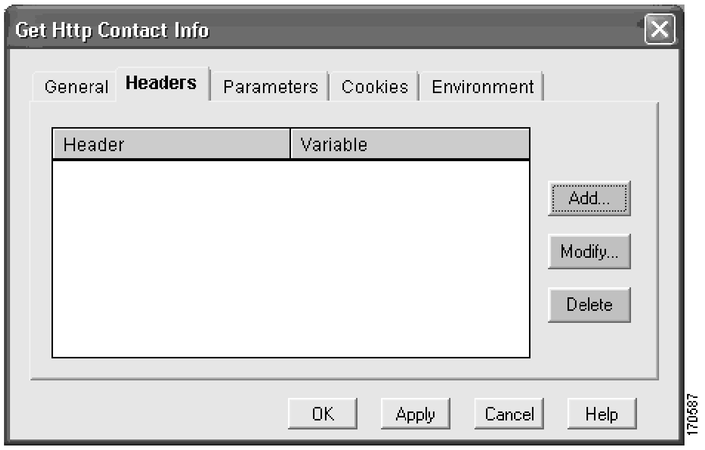


Table 49 describes the fields of the **Headers** tab.

Table 49 Get Http Contact Info Customizer Window Fields: Headers Tab

Field	Description
Header	Name of the header.
Variable	Variable that maps to the header.

Parameters Tab

Use the **Parameters** tab of the **Get Http Contact Info** customizer window, shown in Figure 111, to map scripts parameters to variables in the Cisco Unity Express Editor.

After an HTML form has been completed, the values are typically passed as parameters to the web server. The **Get Http Contact Info** step reads the values of these parameters from the HTTP request using both the GET and POST methods (see Table 27 on page 87) and updates the local variables in the script.

Figure 111 Get Http Contact Info Customizer Window: Parameters Tab

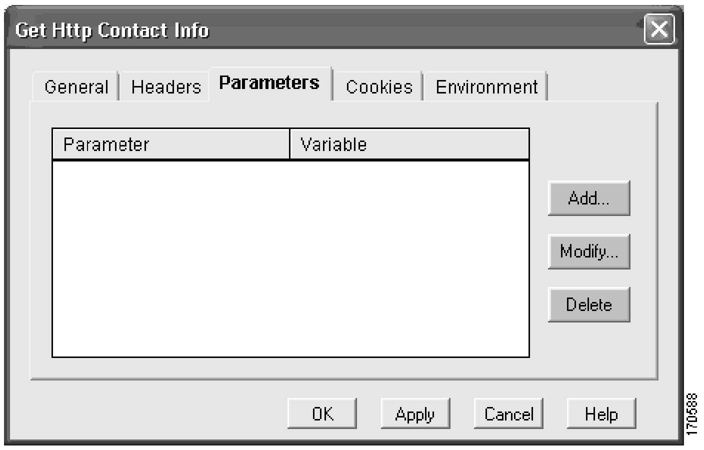


Table 50 describes the fields of the **Parameters** tab.

Table 50 Get Http Contact Info Customizer Window Fields: Parameters Tab

Field	Description
Parameter	Name of the parameter.
Variable	Name of the variable that maps to the parameter.

Cookies Tab

Use the **Cookies** tab of the **Get Http Contact Info** customizer window to map information from a local variable to a cookie.

A cookie is information maintained by the browser that is typically sent by an HTTP server. The information in cookies can improve access to web pages. Most cookies store authentication or identifying information. Once the server authenticates a browser, it can send authentication credentials or other user identifiers to the browser cookie. The web page can then be accessed without further authentication or identification.

Cookies can also be used to store a mapping identifier to a session object so that on subsequent requests, the original Session object associated with the previous HTTP request can be retrieved and reassociated with the new HTTP contact.

Note The use of cookies for authentication may present a security risk if an authenticated browser is left unattended.

Figure 112 shows the customizer window for the **Cookies** tab and Table 51 describes the field. Figure 113 shows the Add Cookies window, in which you add the cookies name and variable name, and Table 52 describes the fields. Use the variable name to refer to the data that the variable contains.

Figure 112 Get Http Contact Info Customizer Window: Cookies Tab

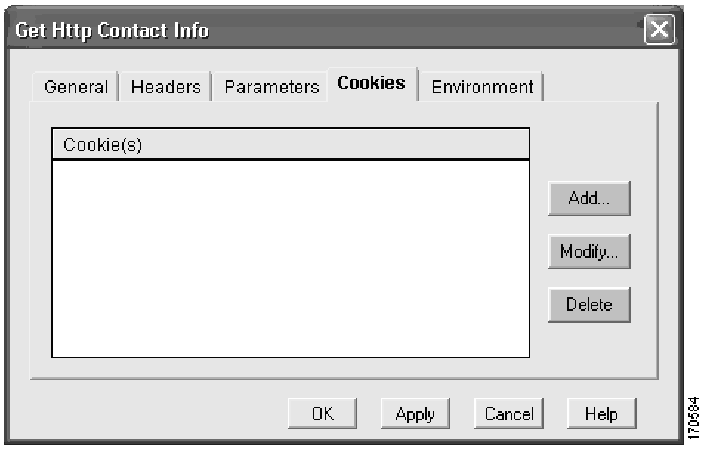


Table 51 Get Http Contact Info Customizer Window Fields: Cookies Tab

Field	Description
Cookie	Names of the cookies

Figure 113 Add Cookies Window

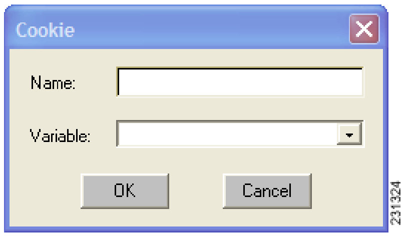


Table 52 Add Cookies Window Fields

Field	Description
Name	Names of the cookies
Variable	Name of variable

Environment Tab

Use the **Environment** tab of the **Get Http Contact Info** customizer window to map information from CGI environment variables to local variables.

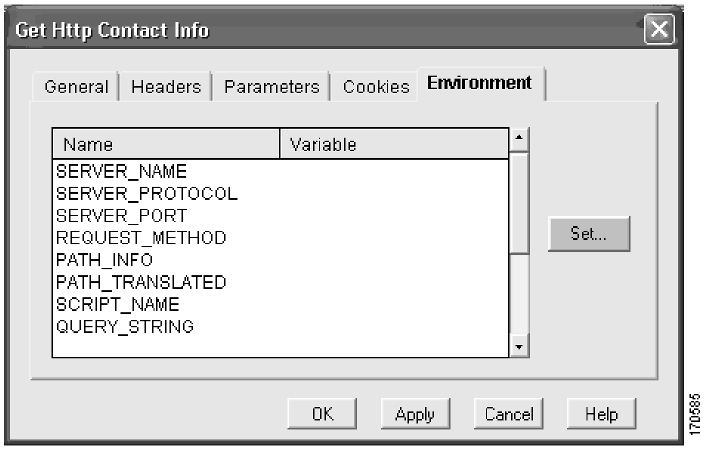
The following is an alphabetical list of the environment variables:

- **AUTH_TYPE**: Protocol-specific authentication method used to validate the user when the server supports user authentication and the script requires authentication.
- **CONTENT_LENGTH**: Content length of the data as specified by the client.
- **CONTENT_TYPE**: Content type of the data for queries such as HTTP GET and HTTP POST that have attached information.

- **PATH_INFO**: Extra path information as given by the client. Scripts can be accessed by a virtual pathname, followed by extra information at the end of the path. The extra information is sent as PATH_INFO. The server decodes this information if it is from a URL before it is passed to the script.
- **PATH_TRANSLATED**: Translated version of PATH_INFO, with any associated virtual-to-physical mapping.
- **QUERY_STRING**: Information that follows the question mark (?) in the URL that references this script. This information is the query information. Do not decode it. Always set this variable when there is query information, regardless of command line decoding.
- **REMOTE_ADDR**: IP address of the remote host making the request.
- **REMOTE_HOST**: Hostname making the request. If the server does not have this information, it sets REMOTE_ADDR.
- **REMOTE_USER**: Authenticated username when the server supports user authentication and the script requires authentication.
- **REQUEST_METHOD**: Method of the request that was made, such as GET or POST (see Table 27 on page 87).
- **SCRIPT_NAME**: Virtual path to the script being executed, used for self-referencing URLs.
- **SERVER_NAME**: Server's hostname, DNS alias, or IP address as it would appear in a self-referencing URL.
- **SERVER_PORT**: TCP port number of the request.
- **SERVER_PROTOCOL**: Name and revision of the information protocol of the request, uses the protocol/revision format.

Figure 114 shows the customizer window for the **Environment** tab.

Figure 114 Get HTTP Contact Info Customizer Window: Environment Tab



Name	Variable
SERVER_NAME	
SERVER_PROTOCOL	
SERVER_PORT	
REQUEST_METHOD	
PATH_INFO	
PATH_TRANSLATED	
SCRIPT_NAME	
QUERY_STRING	

Table 53 describes the fields of the **Environment** tab.

Table 53 Call Subflow Customizer Window Fields: General Tab

Field	Description
Name	Name of the environmental variable.
Variable	Name of the variable that maps to the environment variable.

Http Redirect

Use the **Http Redirect** step to redirect the browser to a specified URL instead of respond to an HTTP request.

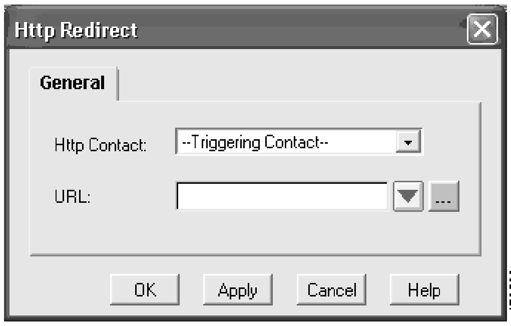
Either the **Http Direct** step or the **Send Response** step (see the "Send Response" section on page 123) can be used to respond to an HTTP request. Do not use both steps. If both steps are used, the HTTP Contact script will shift to a final state after a response is returned back to the browser and be unable send another response because one was already sent.

If a script uses conditional logic, the **Http Direct** step can be used with one condition and **Send Response** step can be used with the other.

The **Terminate** step from the **Contact** palette (see the "Terminate" section on page 71) can also be used as responses to an HTTP request.

Figure 115 shows the customizer window for the **Http Redirect** step.

Figure 115 HTTP Redirect Customizer Window



Http Contact: --Triggering Contact--

URL:

Table 54 describes the fields of the **Http Redirect** customizer window.

Table 54 Http Redirect Customizer Window Fields

Field	Description
Http Contact	Triggers the execution of the **Redirect** step. The default is Triggering Contact, unless another contact is specified.
URL	URL to which the browser is redirected.

Send Response

Use the **Send Response** step to send a document to a browser.

A document must be stored in a document variable before it is sent (see the "Document Steps (IVR Only)" section on page 84). To update a document with dynamic information before it is sent to a browser, place the **Text Substitution for Keywords** step (see the "Text Substitution for Keywords" section on page 88) before the **Send Response** step in the script.

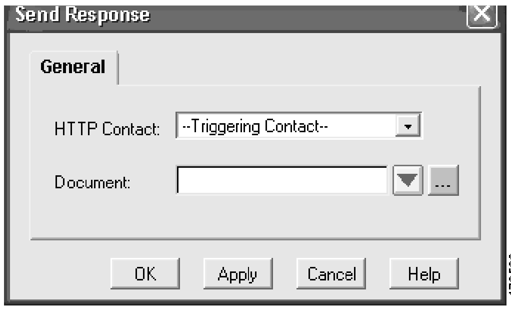
Either the **Send Response** step or the **Http Direct** step (see the "Http Redirect" section on page 123) can be used to respond to an HTTP request. Do not use both steps. If both steps are used, the HTTP Contact script shifts to a final state after a response is returned to the browser and is unable to send another response because one was already sent.

If a script uses conditional logic, the **Http Direct** step can be used with one condition and **Send Response** step can be used with the other.

The **Terminate** step from the **Contact** palette (see the "Terminate" section on page 71) can also be used to respond to an HTTP request.

Figure 116 shows the customizer window for the **Send Response** step.

Figure 116 Send Response Customizer Window



HTTP Contact: --Triggering Contact--
Document:

Table 55 describes the fields of the **Send Response** customizer window.

Table 55 Send Response Window Fields

Field	Description
Http Contact	Contact variable that triggers the execution of the step. The default is Triggering Contact, unless another contact is specified.
Document	Variable that stores the document to be sent.

Set Http Contact Info

Use the **Set Http Contact Info** step to set the values of HTTP headers and cookies in an HTTP response.

Note To use his step for advanced scripts, specialized knowledge of HTTP is required.

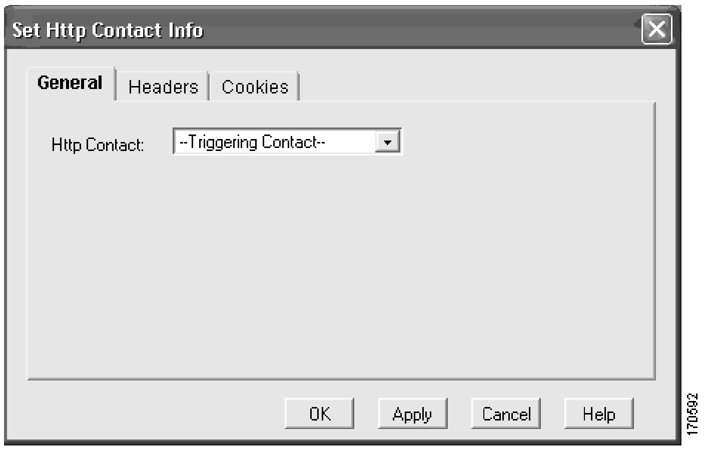
The **Set Http Contact Info** customizer window contains three tabs:

- General Tab, page 124
- Headers Tab, page 125
- Cookies Tab, page 126

General Tab

Use the **General** tab of the **Set Http Contact Info** customizer window, shown in Figure 117, to specify the HTTP contact variable that triggers the execution of the step.

Figure 117 Set Http Contact Info Customizer Window: General Tab



Http Contact: --Triggering Contact--

Table 56 describes the field of the **General** tab.

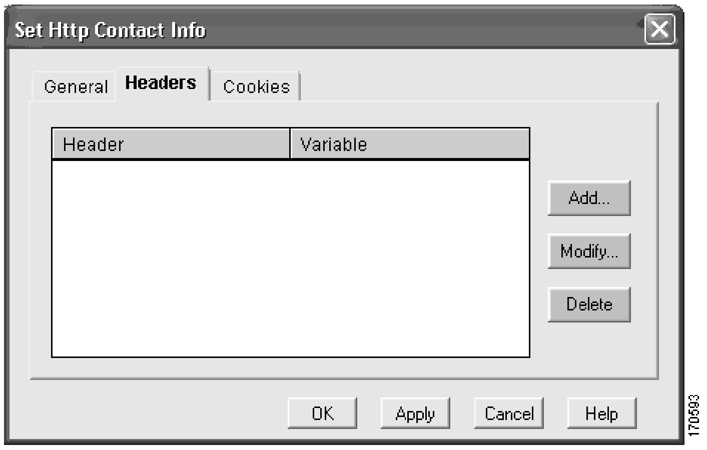
Table 56 Set Http Contact Info Customizer Window Field: General Tab

Field	Description
Http Contact	Contact variable that triggers the execution of the step. Unless another contact is specified, the default is Triggering Contact.

Headers Tab

Use the **Headers** tab of the **Set Http Contact Info** customizer window, shown in Figure 118, to map each HTTP header to a local variable or to a valid expression from which the header value will be obtained when the step executes.

Figure 118 Set Http Contact Info Customizer Window: Headers Tab



Header	Variable

Table 57 describes the fields of the **Headers** tab.

Table 57 Set Http Contact Info Customizer Window Fields: Headers Tab

Field	Description
Header	Name of the header.
Variable	Variable that maps to the header.

Cookies Tab

Use the **Cookies** tab of the **Set Http Contact Info** customizer window, shown in Figure 119, to map each cookie to a local variable from which to obtain the cookie value when the step executes.

A cookie is information maintained by the browser that is sent by the HTTP server in response to an HTTP request. For more information about cookies, see the "Cookies Tab" section on page 120.

Figure 119 Set Http Contact Info Customizer Window: Cookies Tab

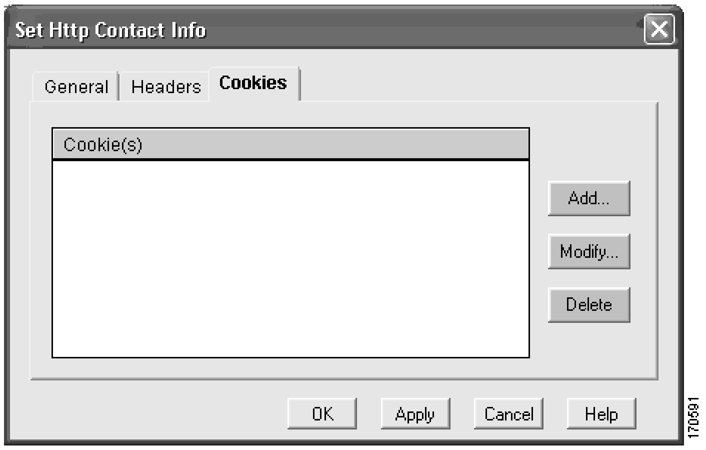


Table 58 describes the field of the **Cookies** tab.

Table 58 Set Http Contact Info Customizer Window Fields: Cookies Tab

Field	Description
Cookie	Name or names of the cookies mapped to variables.

Media Steps

The steps in the **Media** palette of the Cisco Unity Express Script Editor provide script designers with a way to process media interactions with callers.

Media interactions can include playing prompts and acquiring Dual Tone Multi-frequency (DTMF) input.

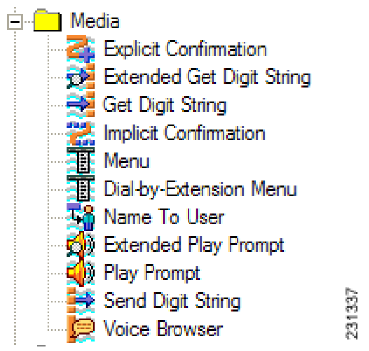
The **Media** palette contains the following steps:

- Explicit Confirmation, page 127
- Implicit Confirmation, page 139
- Get Digit String, page 134

- Menu, page 140
- Name To User, page 144
- Play Prompt, page 150
- Extended Play Prompt, page 153
- Send Digit String (IVR Only), page 155
- Dial-by-Extension Menu, page 155
- Voice Browser (IVR Only), page 160

Figure 46 shows the steps in the **Media** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 120 Media Palette Steps



Note If you apply any of these **Media** steps to a contact that is not associated with a **Media** channel (also called a *dialog* channel), you receive a ChannelUnsupportedException error.

Explicit Confirmation

Use the **Explicit Confirmation** step to confirm an explicit response to a prompt. The **Explicit Confirmation** step accepts 1 for yes and 2 for no.

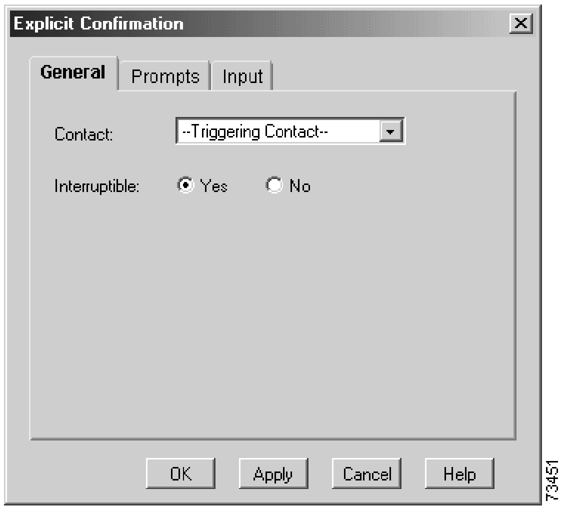
The customizer window of the **Explicit Confirmation** step contains three tabs:

- General Tab, page 127
- Prompts Tab, page 128
- Input Tab, page 129

General Tab

Use the **General** tab of the **Explicit Confirmation** customizer window, as shown in Figure 121, to select the contact on which to perform the confirmation and to set the Interruptible option.

Figure 121 Explicit Confirmation General Tab



Contact: --Triggering Contact--
Interruptible: [✓ Yes] No

Table 59 describes the fields of the **General** tab.

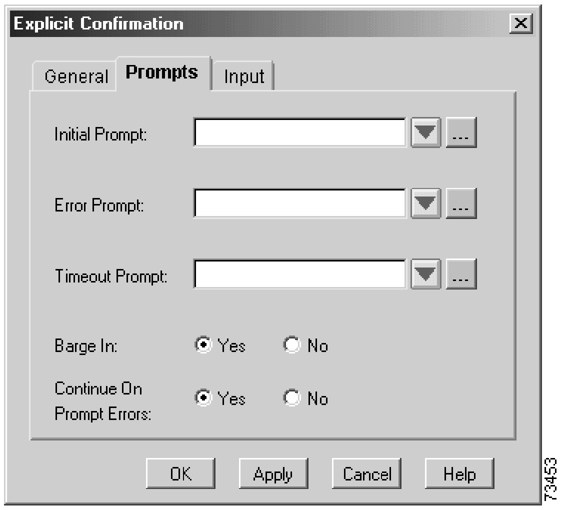
Table 59 Explicit Confirmation Customizer Window Fields: General Tab

Property	Description
Contact	Contact that triggers the execution of the step. Unless another contact is specified, the default is Triggering Contact.
Interruptible	If **Yes,** an external event (such as a caller hanging up) can interrupt the step. If **No**, the step must complete before any other process can execute.

Prompts Tab

Use the **Prompts** tab of the **Explicit Confirmation** customizer window, as shown in Figure 122, to specify initial, error, and timeout prompts and to set Barge In and Continue On Prompt Errors options.

Figure 122 Explicit Confirmation Customizer Window: Prompts Tab



Initial Prompt:
Error Prompt:
Timeout Prompt:
Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 60 describes the fields of the **Prompts** tab.

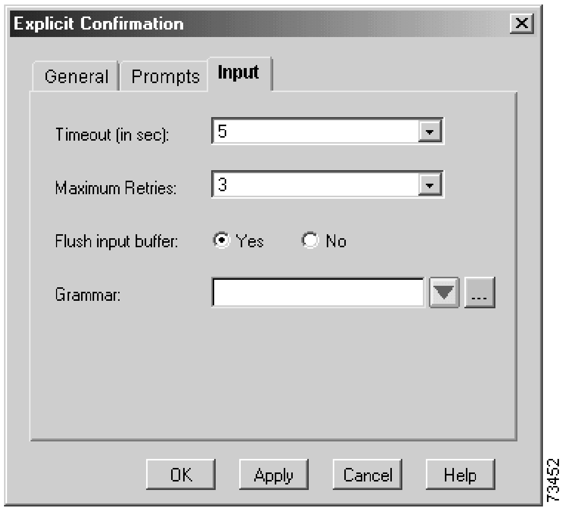
Table 60 Explicit Confirmation Customizer Window Fields: Prompts Tab

Field	Description
Initial Prompt	First prompt to be played back. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Error Prompt	Prompt to be played if an input error occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Timeout Prompt	Prompt to be played if a timeout occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Barge In	If **Yes**, the caller can interrupt the prompt. If **No**, the prompt must complete playback before the caller can respond.
Continue on Prompt Errors	If **Yes**, the step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. If **No**, an exception results, which can then be handled in the script.

Input Tab

Use the **Input** tab of the **Explicit Confirmation** customizer window, as shown in Figure 123, to set timeout duration, maximum number of retries, and Flush Input Buffer options.

Figure 123 Explicit Confirmation Customizer Window: Input Tab



Timeout (in sec): 5
Maximum Retries: 3
Flush input buffer: [✓ Yes] No
Grammar:

Table 61 describes the fields of the **Input** tab.

Table 61 Explicit Confirmation Customizer Window Fields: Input Tab

Field	Description
Timeout (in sec)	Number of seconds that the step waits for a response before timing out.
Maximum Retries	Number of times a new entry can be entered after a timeout or invalid key.
Flush input Buffer	If **Yes**, the system erases previously entered input before capturing caller input. If **No**, the system does not erase previously entered input before capturing caller input.
Grammar	Optional grammar expression to be used for recognizing **Yes** or **No**. If supplied, the grammar overrides the system default grammar. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a grammar expression.

Extended Get Digit String Step

Use the **Extended Get Digit String** step to capture DTMF entries in a script. This step waits for input until the timeout is reached or the caller does one of the following:

- Presses the terminating key
- Exhausts the maximum number of retries
- Enters the maximum number of keys

The **Extended Get Digit String** step acts exactly like the existing **Get Digit String** step except:

- The Get Digit String step provides a Boolean expression for the "Interruptible" and "Clear DTMF Buffer on Retry" fields.
- Although the same limits apply to both steps, you can define most of the Extended Get Digit String step properties using variables that you can change while the script is running.

The terminating key and the cancel key fields in the step customizer support Character objects and String objects. When strings are passed as the terminating or cancel keys, only the first character of the string is used. The special string *none* can specify that either no terminating key or no cancel key is required.

The **Extended Get Digit String** step has the following output branches:

- **Successful** - Input was valid.
- **Timeout** - After the retry limit was reached, the last try timed out.
- **Unsuccessful** - After the retry limit was reached, an invalid key was entered.

When the step returns an error (Timeout or Unsuccessful), all collected digits are returned and stored in the specified input variable.

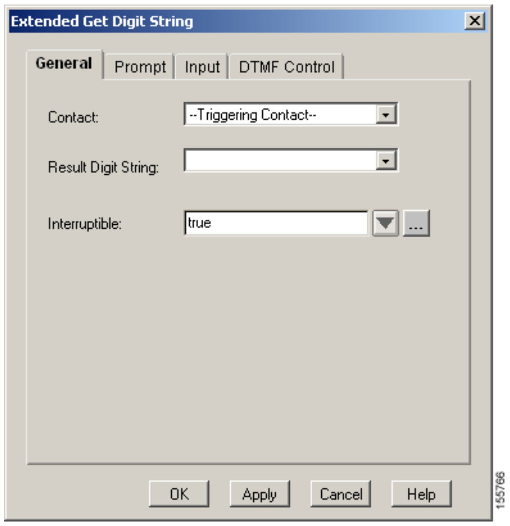
The customizer window of the **Extended Get Digit String** step contains four tabs:

- General Tab, page 131
- Prompt Tab, page 131
- Input Tab, page 132
- DTMF Control Tab, page 133

General Tab

Use the **General** tab of the **Extendedly Digit String** step, as shown in Figure 128, to choose the contact, specify the variable that stores the digit string, and specify whether the step is interruptible by external events.

Figure 124 Extended Get Digit String Customizer Window: General Tab



Contact: --Triggering Contact--
Result Digit String:
Interruptible: true

Table 62 describes the fields of the **General** tab.

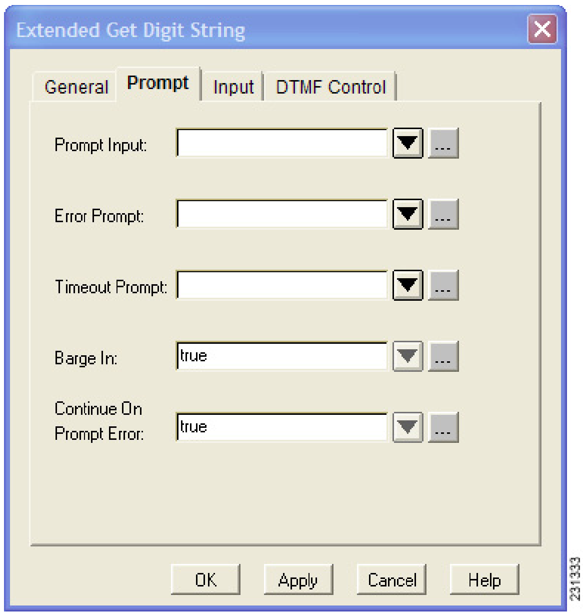
Table 62 Extended Get Digit String Customizer Window: General Tab

Property	Description
Contact	Unless you specify another contact, the default is Triggering Contact.
Result Digit String	Variable that holds the resulting digit string.
Interruptible	If true, an external event, such as a call being remotely disconnected by the caller, can interrupt the step. If false, the step completes before any other process can execute.

Prompt Tab

Use the **Prompt** tab of the **Get Digit String** customizer window, as shown in Figure 129, to specify a prompt, and to set **Barge In** and **Continue on Prompt Errors** options.

Figure 125 Extended Get Digit String Customizer Window: Prompt Tab



Prompt Input:
Error Prompt:
Timeout Prompt:
Barge In: true
Continue On Prompt Error: true

Table 63 describes the field of the **Prompt** tab.

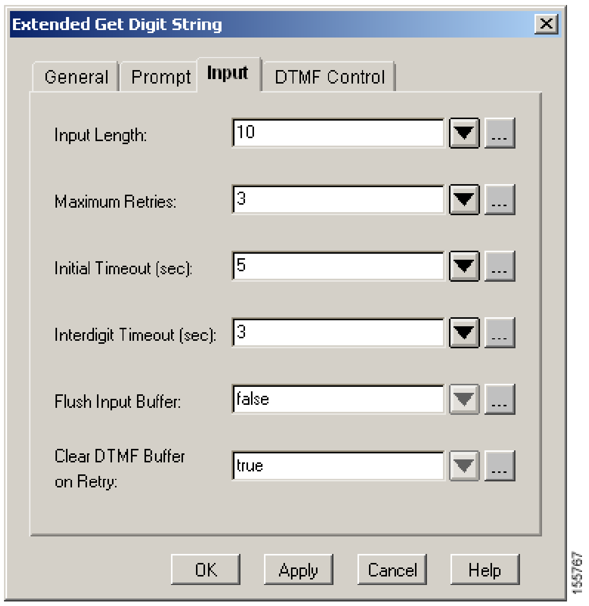
Table 63 Extended Get Digit String: Prompt Tab

Property	Description
Prompt Input	Prompt that is played to callers asking them to input a digit string.
Error Prompt	Prompt to be played if an input error occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Timeout Prompt	Prompt to be played if a timeout occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Barge In	If true, the caller can interrupt the prompt; if false, the prompt must complete playing before the caller's input is accepted.
Continue On Prompt Error	If Yes, the step continues with the next prompt in the list, or if this was the last prompt, waits for input from the caller. If No, an exception is thrown, which can then be handled in the script.

Input Tab

Use the **Input** tab of the **Extended Get Digit String** customizer window, as shown in Figure 130, to set conditions for receiving caller input.

Figure 126 Extended Get Digit String Customizer Window: Input Tab



Input Length: 10
Maximum Retries: 3
Initial Timeout (sec): 5
Interdigit Timeout (sec): 3
Flush Input Buffer: false
Clear DTMF Buffer on Retry: true

Table 64 describes the field of the **Filter** tab.

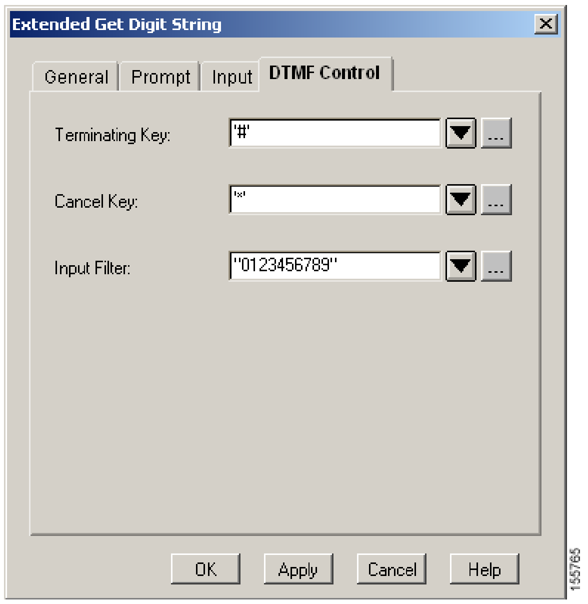
Table 64 Extended Get Digit String: Input Tab

Property	Description
Input Length	Maximum number of digits this step accepts. When this limit is reached, the step returns Successful.
Maximum Retries	Number of times the entry can be started over after a timeout or an invalid key. A 0 value means no retries and that the script must handle the retry scenario.
Initial Timeout	If Yes, the step continues with the next prompt in the list, or if this was the last prompt, awaits the input from the caller. If No, an exception is thrown, which can then be handled in the script.
Interdigit Timeout	Time (in seconds) that the system waits for the caller to enter the next digit after receiving the first digit from the caller.
Flush Input Buffer	If the expression evaluates to true, the system discards any previously entered digits before executing this step.
Clear DTMF Buffer on Retry	If the expression evaluates to true, the script clears the DTMF buffer before each retry.

DTMF Control Tab

Use the DTMF Control tab of the **Extended Get Digit String** customizer window, as shown in Figure 127, to set the Terminating Key, Cancel Key, and Input Filter.

Figure 127 Extended Get Digit String Customizer Window: DTMF Control Tab



Terminating Key: '#'

Cancel Key: '*'

Input Filter: "0123456789"

Table 65 describes the field of the **DTMF Control** tab.

Table 65 Extended Get Digit String Customizer Window: DTMF Control Tab

Property	Description
Terminating Key	Key used to indicate the end of caller input. The terminating key overrides the Maximum Input Length to terminate input. Leaving this field empty or setting it to *none* or *n* means that no terminating key exists.
Cancel Key	Key the caller can press to restart. Leaving this field empty or setting it to *none* or *n* means that no cancel key exists.
Input Filter	Expression that defines the valid DTMF keys that can be entered (excluding the terminating and cancel keys).

Get Digit String

Use the **Get Digit String** step to capture a DTMF digit string from the caller in response to a prompt.

The **Get Digit String** step waits for input until the caller does one of the following:

- Presses the terminating key (DTMF only).
- Exhausts the maximum number of retries.
- Enters the maximum number of keys (DTMF only).
- Does not respond before the timeout length is reached.

Note When any previous escalating prompt in the script enters the **Get Digit String** step, the previous escalating prompt is reset to the first prompt in its list.

The **Get Digit String** step provides three output branches:

- **Successful**: Input was valid.
- **Timeout**: After the retry limit was reached, the last try timed out.
- **Unsuccessful**: After the retry limit was reached, an invalid key was pressed.

Note If an error occurs, the accumulated digits are returned and saved in the specified variable before the script exits through the unsuccessful or timeout output branches.

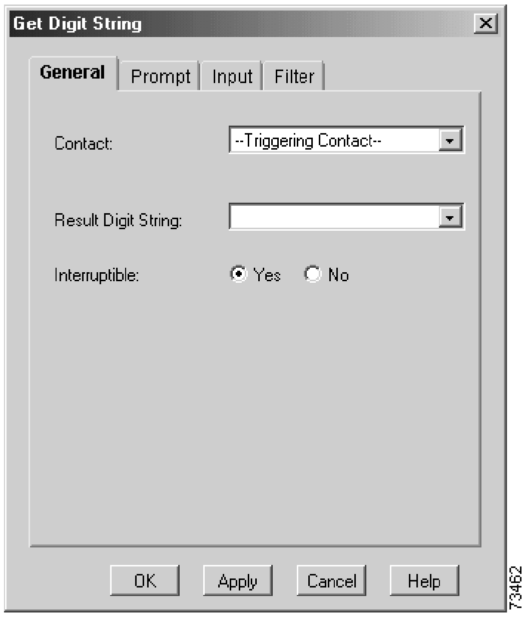
The customizer window of the **Get Digit String** step contains four tabs:

- General Tab, page 135
- Prompt Tab, page 136
- Input Tab, page 137
- Filter Tab, page 138

General Tab

Use the **General** tab of the **Get Digit String** step, as shown in Figure 128, to choose the contact, specify the variable that will store the digit string, and specify whether or not the step is interruptible by external events.

Figure 128 Get Digit String Customizer Window: General Tab



Contact: --Triggering Contact--

Result Digit String:

Interruptible: [✓ Yes] No

Table 66 describes the fields of the **General** tab.

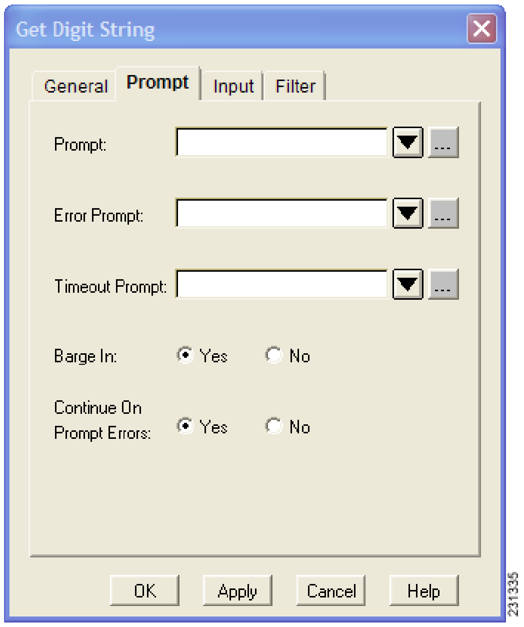
Table 66 Get Digit String Customizer Window Fields: General Tab

Field	Description
Contact	Contact that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
Result Digit String	Name of the variable that stores the digits that the caller enters.
Interruptible	If **Yes**, an external event (such as a caller hanging up) can interrupt the step. If **No**, the step must complete before any other process can execute.

Prompt Tab

Use the **Prompt** tab of the **Get Digit String** customizer window, as shown in Figure 129, to specify a prompt, and to set **Barge In** and **Continue on Prompt Errors** options.

Figure 129 Get Digit String Customizer Window: Prompt Tab



Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 67 describes the fields of the **Prompt** tab.

Table 67 Get Digit String Customizer Window Fields: Prompt Tab

Field	Description
Prompt	Prompt to be played back.
Error Prompt	Prompt to be played if an input error occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.

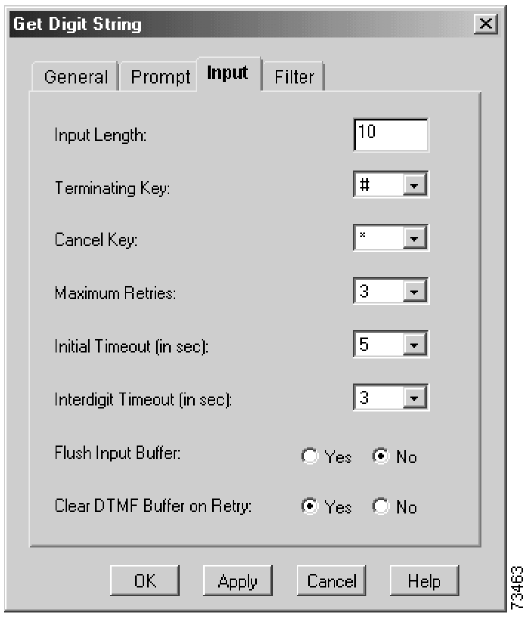
Table 67 Get Digit String Customizer Window Fields: Prompt Tab (continued)

Field	Description
Timeout Prompt	Prompt to be played if a timeout occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Barge In	If **Yes**, the caller can interrupt the prompt. If **No**, the prompt must complete playback before the caller can respond.
Continue on Prompt Errors	If **Yes**, the step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. If **No**, an exception results, which can then be handled in the script.

Input Tab

Use the **Input** tab of the **Get Digit String** customizer window, as shown in Figure 130, to set conditions for receiving caller input.

Figure 130 Get Digit String Customizer Window: Input Tab



Input Length: 10
Terminating Key: #
Cancel Key: *
Maximum Retries: 3
Initial Timeout (in sec): 5
Interdigit Timeout (in sec): 3
Flush Input Buffer: Yes [✓ No]
Clear DTMF Buffer on Retry: [✓ Yes] No

Table 68 describes the fields of the **Input** tab.

Table 68 Get Digit String Customizer Window Fields: Input Tab

Field	Description
Input Length	Maximum number of digits or characters. When this limit is reached, the step stops accumulating digits and returns.
Terminating Key	Key used to indicate the end of caller input (DTMF only). The terminating key overrides the **Input Length** to terminate input.
Cancel Key	Key the caller presses to start over.
Maximum Retries	Number of times a new entry can be entered after a timeout or invalid key. After the maximum number of retries is reached, the step continues on the **Timeout** or **Unsuccessful** output branch, depending on whether the last try timed out or an invalid key was entered. On a retry because of an invalid key, a system prompt plays. A "0" value means that no retry is allowed; in this case, the script must handle the retry scenario.
Initial timeout (in sec)	Number of seconds the system waits for initial input from the caller.
Interdigit timeout (in sec)	Number of seconds the system waits for the caller to enter the next digit after receiving initial input from the caller (DTMF).
Flush Input Buffer	If **Yes**, the system erases previously entered input before capturing caller input. If **No**, the system does not erase previously entered input before capturing caller input.
Clear DTMF Buffer on Retry	If **Yes**, the step clears the DTMF buffer before each retry. If **No**, the step does not clear the DTMF buffer before each retry.

Filter Tab

Use the **Filter** tab of the **Get Digit String** customizer window, as shown in Figure 131, to specify digits that can be accepted from the caller.

To specify the digits you want to accept from the caller, check the desired digit check boxes and click **OK**.

The **Get Digit String** customizer window closes. The name of the triggering contact and the result digit string variable appear next to the **Get Digit String** step icon in the Design pane.

Figure 131 Get Digit String Customizer Window: Filter Tab

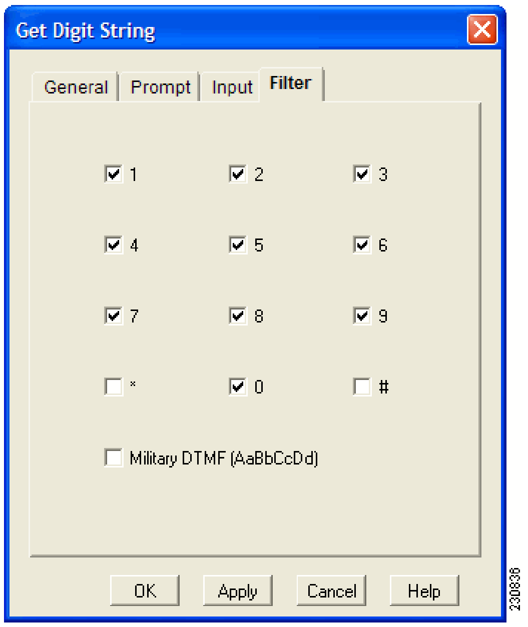


Table 69 describes the field of the **Filter** tab.

Table 69 Get Digit String Customizer Window Fields: Filter Tab

Field	Description
Digit selection box	Use the **Filter** tab to specify the digits that you want to accept from the caller (excluding the terminating and cancel keys). If the caller enters digits that you do not choose, the system plays an error prompt for the caller and retries the **Input** step until the maximum numbers of retries is reached. At that time, the **Unsuccessful** output branch executes.

Implicit Confirmation

Use the **Implicit Confirmation** step to confirm an action without asking a question.

A prompt explaining the action to be taken is played back and the system waits a configured number of seconds for input from the caller. If the caller presses any DTMF digits before the configured timeout, the confirmation is considered to have failed, and an **Explicit Confirmation** step must be used.

Note When any previous escalating prompt in the script enters the **Implicit Confirmation** step, the previous escalating prompt is reset to the first prompt in its list.

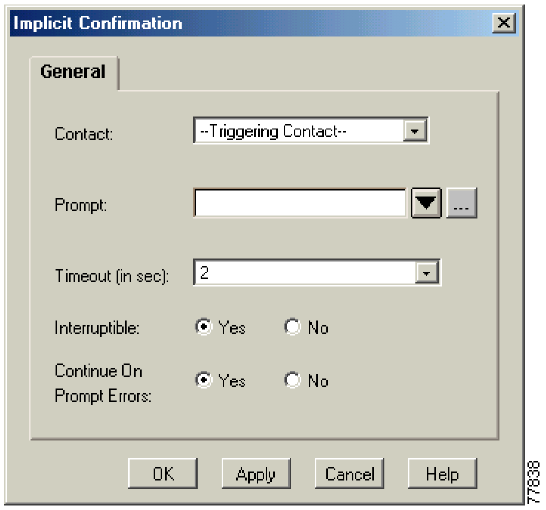
For example, when a valid string of digits is received, a prompt plays the extension that will be dialed, based on the caller's input. The **Implicit Confirmation** step is configured in this example to give the caller two seconds after hearing the prompt to decline confirmation before timeout.

Under the **No** output branch of the **Implicit Confirmation** step, an **If** step tracks the number of times the confirmation is attempted before the script moves to a subsequent step.

If the extension played back to the caller is accurate and the caller makes no effort to stop the operation, the **Yes** output branch executes and a **Call Redirect** step attempts to connect the caller to the desired extension.

Figure 132 shows the customizer window for the **Implicit Confirmation** step.

Figure 132 Implicit Confirmation Customizer Window



Implicit Confirmation

General

Contact: --Triggering Contact--

Prompt:

Timeout (in sec): 2

Interruptible: [✓ Yes] No

Continue On Prompt Errors: [✓ Yes] No

Table 70 describes the fields of the **Implicit Confirmation** customizer window.

Table 70 Implicit Confirmation Fields

Field	Description
Contact	Contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified. The name of the triggering contact appears next to the **Implicit Confirmation** step icon in the Design pane.
Prompt	Prompt played to the caller.
Timeout (in secs)	Number of seconds without a caller response before confirmation is considered successful. (Usual value is 2 seconds.)
Interruptible	If **Yes**, an external event (such as a caller hanging up) can interrupt the step. If **No**, the step must complete before any other process can execute.
Continue on Prompt Errors	If **Yes**, the step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. If **No**, an exception results, which can then be handled in the script.

Menu

Use the **Menu** step to provide a menu from which callers can choose a series of options. The **Menu** step receives a single digit entered by a caller and maps this entry to a series of option output branches. The system executes the steps that you add after each of these option output branches.

Note When any previous escalating prompt in the script enters the **Menu** step, the previous escalating prompt is reset to the first prompt in its list.

Although the **Menu** step combines the functionality of a **Get Digit String** step and a **Switch** step, it allows the caller to enter only one digit.

By default, the **Menu** step has the following output branches:

- **Output 1**
- **Output 2**
- **Output 3**
- **Timeout**
- **Unsuccessful**

You can add more output branches in the **General** tab of the **Menu** customizer window.

The **Menu** step retries for either a timeout or an invalid digit entry (a digit that is not associated with any connections). If the maximum number of retries is reached, the **Menu** step follows either the **Timeout** or **Unsuccessful** connection, depending on the reason for the latest failure.

The customizer window of the **Menu** step contains three tabs:

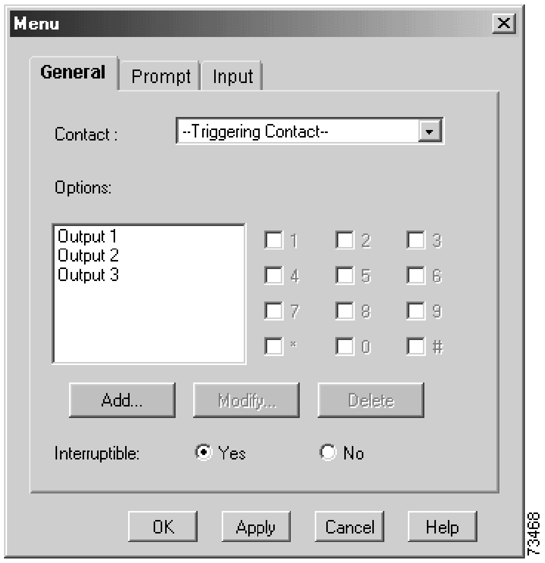
- General Tab, page 141
- Prompt Tab, page 143
- Input Tab, page 143

General Tab

Use the **General** tab of the **Menu** customizer window, shown in Figure 133, to associate digits (typically entered by the caller from a telephone keypad) with an output branch label.

You can associate multiple inputs with a single output branch label, and you can associate only one output branch label with a given input.

Figure 133 Menu Customizer Window: General Tab



Contact : --Triggering Contact--

Options:

Output 1
Output 2
Output 3

☐ 1 ☐ 2 ☐ 3
☐ 4 ☐ 5 ☐ 6
☐ 7 ☐ 8 ☐ 9
☐ * ☐ 0 ☐ #

Interruptible: [✓ Yes] No

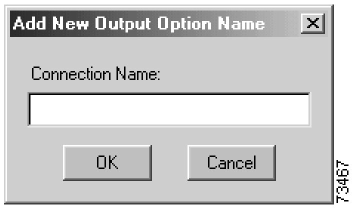
Table 71 describes the fields of the **General** tab.

Table 71 Menu Customizer Window Fields Customizer Window: General Tab

Field	Description
Contact	Contact that triggers the execution of the step. Unless another contact is specified, default is Triggering Contact.
Options	One label for each possible output value.
Add	Add a new option. The new option appears in the **Options** list box.
Modify	Modifies the selected option using the **Rename Output Option** dialog box, which contains the same field as the **Add New Output Option Name** dialog box and is configured in the same way.
Interruptible	If **Yes**, an external event (such as a caller hanging up) can interrupt the step. If **No**, the step must complete before any other process can execute.

The **Add New Output Option Name** dialog box is shown in Figure 134.

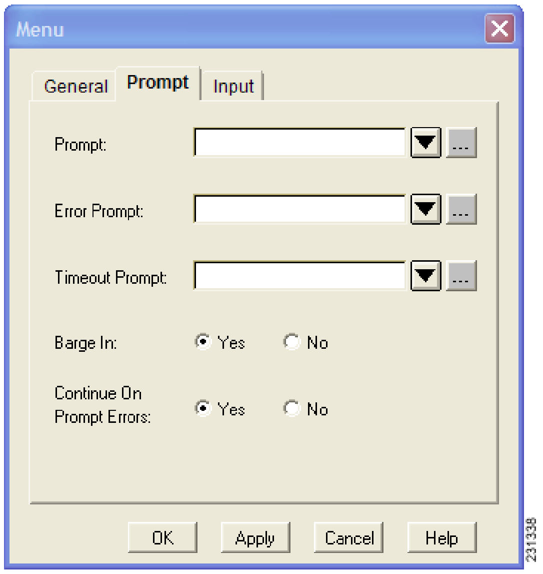
Figure 134 Add New Output Option Name Dialog Box



Prompt Tab

Use the **Prompt** tab of the **Menu** customizer window, as shown in Figure 135, to choose the prompt to be played back and to set the **Barge In** and **Continue on Prompt Errors** options.

Figure 135 Menu Customizer Window: Prompt Tab



Prompt:

Error Prompt:

Timeout Prompt:

Barge In: [✓ Yes] No

Continue On Prompt Errors: [✓ Yes] No

Table 72 describes the fields of the **Prompt** tab.

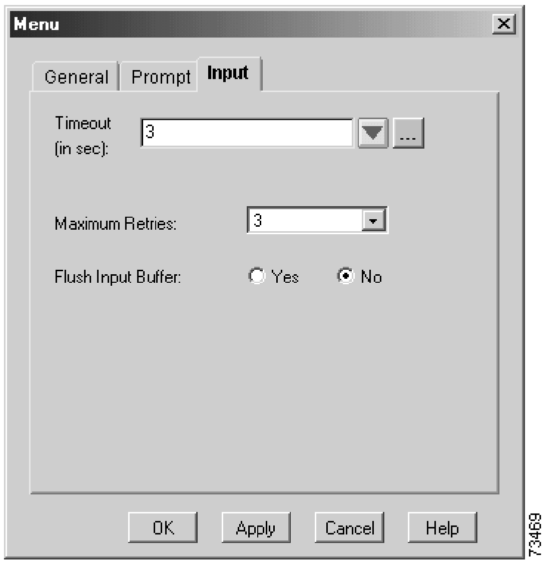
Table 72 Menu Customizer Window Fields: Prompt Tab

Field	Description
Prompt	Specifies prompt to be played back to caller.
Error Prompt	Prompt to be played if an input error occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Timeout Prompt	Prompt to be played if a timeout occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Barge In	If **Yes**, the caller can interrupt the prompt. If **No**, the prompt must complete playback before the caller can respond.
Continue on Prompt Errors	If **Yes,** the step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. If **No**, an exception results, which can then be handled in the script.

Input Tab

Use the **Input** tab of the **Menu** customizer window, as shown in Figure 136, to set the timeout setting, maximum number of retries, and **Flush Input Buffer** options.

Figure 136 Menu Customizer Window: Input Tab



Timeout (in sec): 3

Maximum Retries: 3

Flush Input Buffer: Yes [✓ No]

Table 73 describes the fields of the **Input** tab.

Table 73 Menu Customizer Window Fields: Input Tab

Property	Description
Timeout	Amount of time the system waits for input from the caller. When this timer expires, the system either replays the prompt or plays the system prompt that asks if the caller is still there. Enter a value, choose the variable that stores the timeout value from the **Timeout** drop-down menu, or click the **Expression Editor** (**...**) button and enter a number.
Maximum Retries	Number of times the entry can be restarted after a timeout or invalid input response. After the maximum number of retries is reached, the **Menu** step follows the **Timeout** or **Unsuccessful** output branches depending on whether the last try timed out or an invalid input response was entered. A “0” value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	If **Yes,** the system erases previously entered input before capturing caller input. If **No**, the system does not erase previously entered input before capturing caller input.

Name To User

The **Name To User** step is typically used to prompt a caller for the name of the person being called (using DTMF), and then to compare the name entered by the caller with names stored in a directory. The **Name To User** step is often used in a script to automatically transfer a caller to the extension of the person being called.

Another useful function of the **Name To User** step is to assign a value to a variable that can later be queried using the **Get User Info** step to retrieve information such as the extension, e-mail address, and spoken name of the user selected by the caller.

Note When any previous escalating prompt in the script enters the **Name To User** step, the previous escalating prompt is reset to the first prompt in its list.

The **Name To User** step receives DTMF input from a caller, using the following numeric keypad mapping:

- 2 = ABC
- 3 = DEF
- 4 = GHI
- 5 = JKL
- 6 = MNO
- 7 = PQRS
- 8 = TUV
- 9 = WXYZ

Using the information from this step, the script creates a subsequent prompt that plays the prerecorded name of the user selected by the caller if it exists. If no recording exists, the script will spell the user's name.

Note The **Name To User** step is limited to spelling back names with ASCII-only characters, which may be a limitation under some international conditions.

The **Name To User** step produces the following output branches:

Note If the **Name To User** step matches the caller input with a single user in the Lightweight Access Directory Protocol (LDAP) directory, that result will be returned immediately without requiring the caller to confirm the selection.

- **Successful**: A successful match is made between the input from the caller and a name in the directory.
- **Timeout**: The step has reached the maximum number of retries (as configured in the customizer window) without receiving input from the caller.
- **Unsuccessful**: The input from the caller does not match a name in the directory.
- **Operator**: The operator's extension was entered.

  Note The **Operator** output branch appears under the **Name To User** step in the script only if **Yes** is selected for the **Operator** option in the **General** tab of the **Name To User** customizer window. (See Figure 137.)

The customizer window of the **Name To User** step contains three tabs:

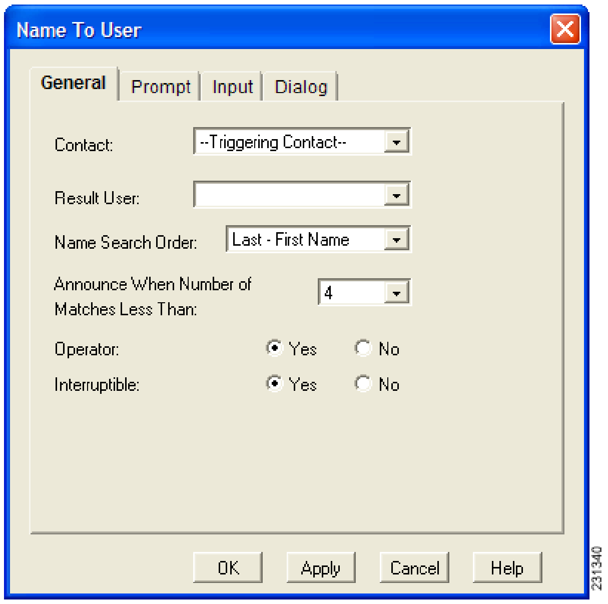
- General Tab, page 146

- Prompt Tab, page 147
- Input Tab, page 148

General Tab

Use the **General** tab, as shown in Figure 137, to specify the **Result User** variable and to set other properties for the **Name To User** step.

Figure 137 Name To User Customizer Window: General Tab



Contact: --Triggering Contact--
Result User:
Name Search Order: Last - First Name
Announce When Number of Matches Less Than: 4
Operator: [✓ Yes] No
Interruptible: [✓ Yes] No

Table 74 describes the fields of the **General** tab.

Follow these steps to configure the **General** fields of the **Name To User** step:

Table 74 Name To User Customizer Window Fields: General Tab

Field	Description
Contact	Contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified. The name of the triggering contact and the name of the result user variable appear in the Design pane.
Result User	Variable that stores a user object representing the user selected by the caller.
Announce When Number of Matches Less Than	If the number of matches is less than this value, the step prompts the caller to choose the correct entry from the list of matches. If the number of matches is greater than or equal to this value, the step prompts the caller to enter additional letters to reduce the number of matches.
Name Search Order	Selects the search to use either the caller's first or last name.

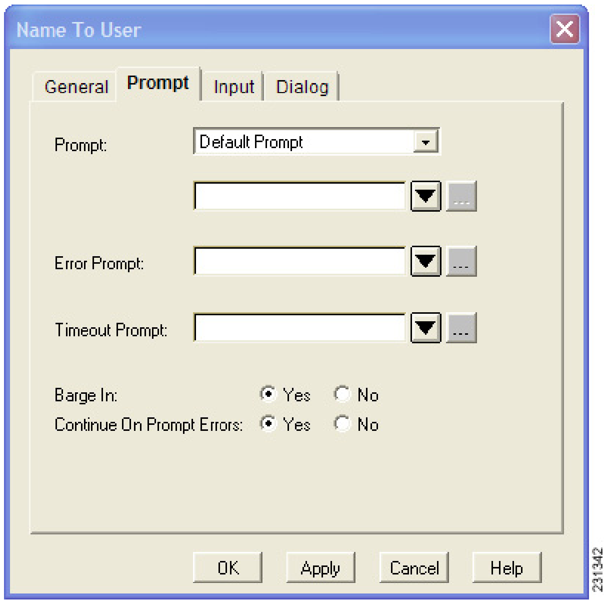
Table 74 Name To User Customizer Window Fields: General Tab (continued)

Field	Description
Operator	If **Yes**, the caller has the option to connect to an operator by pressing “0”. If **No**, the caller is not offered the option to connect to an operator.
Interruptible	If **Yes**, an external event (such as a caller hanging up) can interrupt the step. If **No**, the step must complete before any other process can execute.

Prompt Tab

Use the **Prompt** tab, as shown in Figure 138, to specify prompts to be played back by the **Name To User** step and to set the **Barge In** and **Continue on Prompt Errors** options.

Figure 138 Name To User Customizer Window: Prompt Tab



Prompt: Default Prompt

Error Prompt:

Timeout Prompt:

Barge In: [✓ Yes] No

Continue On Prompt Errors: [✓ Yes] No

Table 75 describes the fields of the **Prompt** tab.

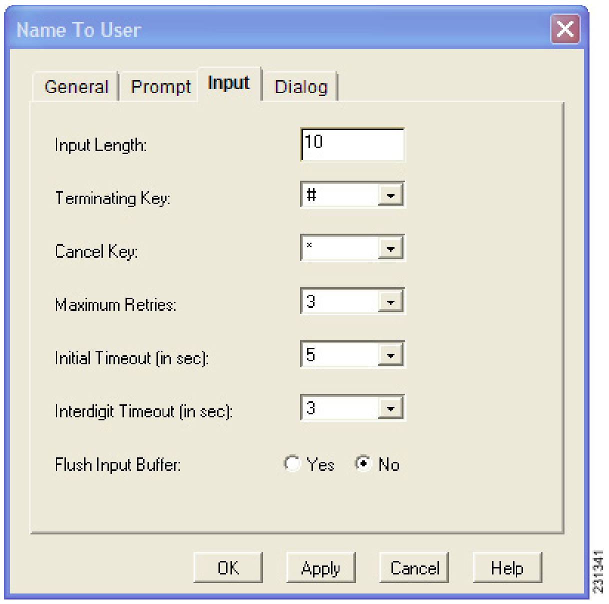
Table 75 Name To User Customizer Window Fields: Prompt Tab

Field	Description
Prompt	Specifies the prompt to be played back to the caller. Options from the drop-down menu: • **Default prompt**: System prompt bundled with the Cisco Unity Express software: "Spell the last name followed by the first name." • **Customized prompt**: Prompt created by the script designer. • **No prompt**: No prompt is played. Enter a value, choose a prompt to play from the **List of Prompts** drop-down menu, click the **Expression Editor** (**...**) button and enter an expression that specifies the prompt to play.
Error Prompt	Prompt to be played if an input error occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Timeout Prompt	Prompt to be played if a timeout occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Barge In	If **Yes**, the caller can interrupt the prompt. If **No**, the prompt must complete playback before the caller can respond.
Continue on Prompt Errors	If **Yes**, the step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. If **No**, an exception results, which can then be handled in the script.

Input Tab

Use the **Input** tab, as shown in Figure 139, to configure various input properties for the **Name To User** step.

Figure 139 Name To User Customizer Window: Input Tab



Input Length: 10
Terminating Key: #
Cancel Key: *
Maximum Retries: 3
Initial Timeout (in sec): 5
Interdigit Timeout (in sec): 3
Flush Input Buffer: Yes [✓ No]

Table 76 describes the fields of the **Input** tab.

Table 76 Name To User Customizer Window Fields: Input Tab

Property	Description
Input Length	Minimum number of digits required before automatically checking for a caller match.
Terminating Key	Key used to indicate the end of caller input.
Cancel Key	Key the caller presses to restart. The **Cancel** key works only until the number of maximum retries is reached.
Maximum Retries	Number of times the step attempts to receive valid input. A “0” value means that no retry is allowed; in this case, the script must handle the retry scenario.
Initial Timeout (in sec)	Number of seconds that the system waits for initial input from the caller.
Interdigit Timeout (in sec)	Number of seconds that the system waits for the caller to enter the next digit, after receiving initial input from the caller.
Flush Input Buffer	If **Yes**, the system erases previously entered input before capturing caller input. If **No**, the system does not erase previously entered input before capturing caller input.

Dialog Tab

Use the **Dialog** tab, as shown in Figure 140, to configure various input properties for the **Name To User** step.

Figure 140 Name To User Customizer Window: Dialog Tab

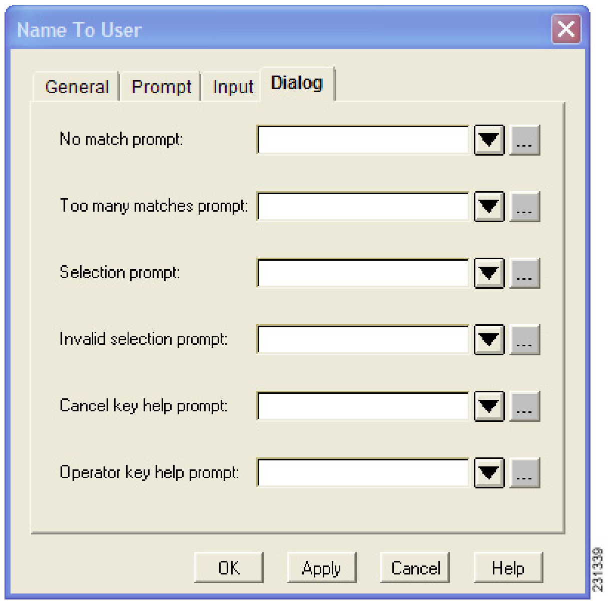


Table 76 describes the fields of the **Dialog** tab.

Table 77 Name To User Customizer Window Fields: Input Tab

Property	Description
No match prompt	Specifies the prompt to be played when no match is found after checking for a caller match.
Too many matches prompt	Specifies the prompt to be played when too many matches are found after checking for a caller match.
Selection prompt	Specifies the prompt to be played when a caller must make another selection.
Invalid selection prompt	Specifies the prompt to be played when an unrecognized selection is made by the caller.
Cancel key help prompt	Specifies the prompt to be played when the cancel help key is selected by the caller.
Operator key help prompt	Specifies the prompt to be played when the operator help key is selected by the caller.

Play Prompt

Use the **Play Prompt** step to play back specified prompts to the caller.

Note When any previous escalating prompt in the script enters the **Play Prompt** step, the previous escalating prompt is reset to the first prompt in its list.

The customizer window of the **Play Prompt** step contains three tabs:

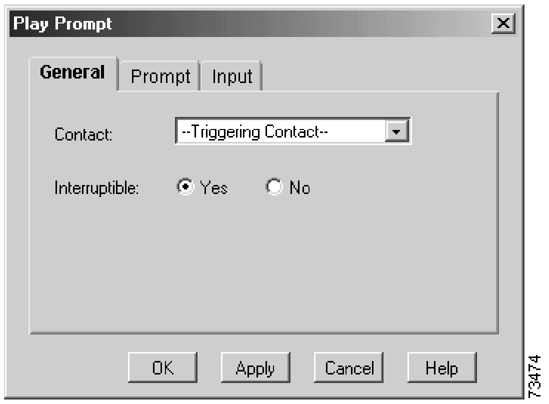
- General Tab, page 151

- Prompt Tab, page 151
- Input Tab, page 152

General Tab

Use the **General** tab, as shown in Figure 141, to identify the contact and to set the **Interruptible** option.

Figure 141 Play Prompt Customizer Window: General Tab



Contact: --Triggering Contact--
Interruptible: [✓ Yes] No

Table 78 describes the fields of the **General** tab.

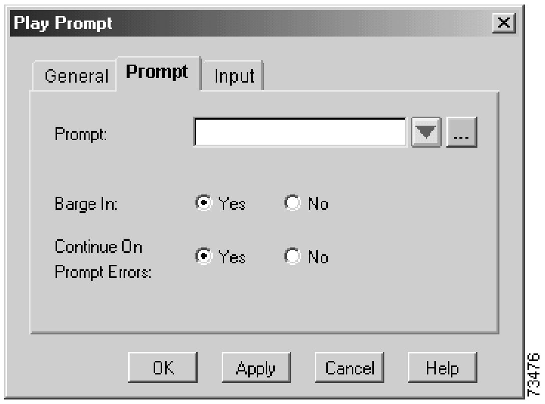
Table 78 Play Prompt Customizer Window Fields: General Tab

Field	Description
Contact	Contact that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified. The name of the triggering contact and the name of the prompt variable appears in the Design pane.
Interruptible	If **Yes**, an external event (such as a caller hanging up) can interrupt the step. If **No**, the step must complete before any other process can execute.

Prompt Tab

Use the **Prompt** tab of the **Play Prompt** customizer window, as shown in Figure 142, to specify the prompt to be played back, and to set the **Barge In** and **Continue on Prompt Errors** options.

Figure 142 Play Prompt Customizer Window: Prompt Tab



Prompt:

Barge In: [✓ Yes] No

Continue On Prompt Errors: [✓ Yes] No

Table 79 describes the fields of the **Prompt** tab.

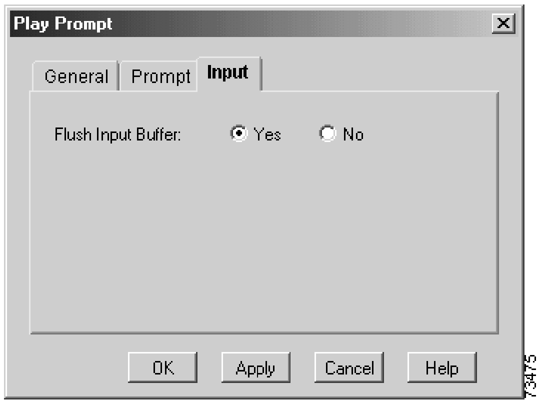
Table 79 Play Prompt Customizer Window Fields: Prompt Tab

Field	Description
Prompt	Specifies prompt to be played.
Barge In	If **Yes**, the caller can interrupt the prompt. If **No**, the prompt must complete playback before the caller can respond.
Continue on Prompt Errors	If **Yes**, the step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. If **No**, an exception results, which can then be handled in the script.

Input Tab

Use the **Input** tab of the **Play Prompt** step, as shown in Figure 143, to specify whether or not to erase previously entered input before capturing caller input.

Figure 143 Play Prompt Customizer Window: Input Tab



Flush Input Buffer: [✓ Yes] No

Table 80 describes the field of the **Input** tab

Table 80 *Play Prompt Customer Window Field: Input Tab*

Field	Description
Flush Input Buffer	If **Yes**, the system erases previously entered input before capturing caller input. If **No**, the system does not erase previously entered input before capturing caller input.

Extended Play Prompt

Use the **Extended Play Prompt** step to play prompts back to the caller. Use this step instead of the Play Prompt step, to include conditional testing using the Expression Editor.

The Extended Play Prompt Step does not have any output branches.

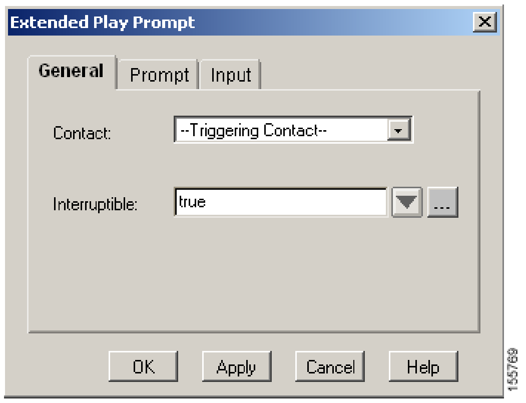
The customizer window of the **Extended Play Prompt** step contains three tabs:

- General Tab, page 153
- Prompt Tab, page 154
- Input Tab, page 154

General Tab

Use the **General** tab of the **Extended Play Prompt** customizer window, as shown in Figure 144, to select the contact on which to perform the confirmation and to set the Interruptible option.

Figure 144 *Extended Play Prompt Customizer Window: General Tab*



Contact: --Triggering Contact--
Interruptible: true

Table 81 describes the field of the **General** tab.

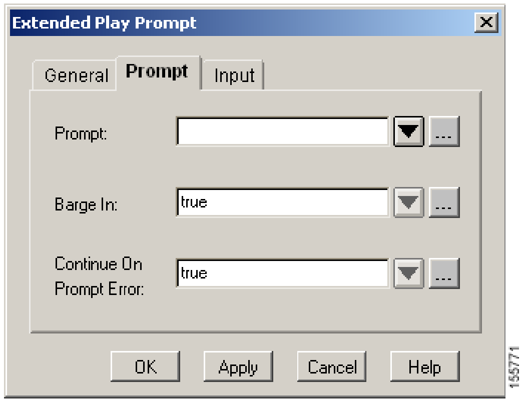
Table 81 *Extended Play Prompt Customizer Window: General Tab*

Property	Description
Contact	Unless another contact is specified, default is Triggering Contact.
Interruptible	If true, an external event, such as a call being remotely disconnected by the caller, can interrupt the step. If false, the step completes before any other process can execute.

Prompt Tab

Use the Prompt tab of the Extended Play Prompt customizer window, as shown in Figure 145, to set the Prompt Input, and to set the Barge in and Continue On Prompt Error conditions.

Figure 145 Extended Play Prompt Customizer Window: Prompt Tab



Prompt:
Barge In: true
Continue On Prompt Error: true

Table 82 describes the field of the **Prompt** tab.

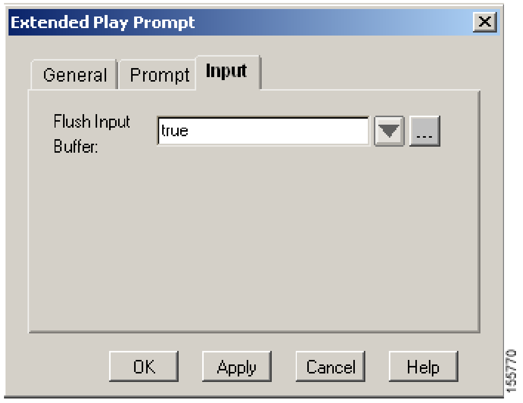
Table 82 Extended Play Prompt Customizer Window: Prompt Tab

Property	Description
Prompt	Specifies which prompt is to be played.
Barge In	If true, the caller can interrupt the prompt. If false, the prompt must complete playing before the caller's input is accepted.
Continue On Prompt Error	If **Yes**, the step continues with the next prompt in the list. If **No**, an exception is thrown, which can then be handled in the script.

Input Tab

Use the Input tab of the **Extended Play Prompt** customizer window, as shown in Figure 146, to set the Flush Input Buffer condition.

Figure 146 Extended Play Prompt Customizer Window: Input Tab



Flush Input Buffer: true

Table 83 describes the field of the **Prompt** tab.

Table 83 Extended Get Digit String Customizer Window: Prompt Tab

Property	Description
Flush Input Buffer	If the expression evaluates to true, the system discards any previously entered digits before running this step.

Send Digit String (IVR Only)

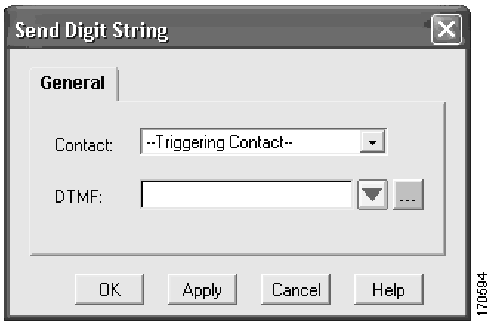
Use the **Send Digit String** step to out pulse or send back a specified set of DTMF digits to the caller.

The DTMF digits are sent out-of-band and out pulsed inband by the gateways.

Note You can insert a comma (,) in the sequence of DTMF digits to instruct the script to insert a 1-second pause before continuing to out pulse the remaining DTMF digits. You can use multiple commas to increase the length of the pause.

Figure 147 shows the **Send Digit String** customizer window.

Figure 147 Send Digit String Customizer Window



The Table 84 describes the fields of the **Send Digit String** customizer window.

Table 84 Send Digit String Customizer Window Fields

Field	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
DMTF	Variable or expression indicating the sequence of the DTMF digits to be sent.

Dial-by-Extension Menu

Use the **Dial-by-Extension Menu** step to associate the digits entered by the caller with an output branch label.

Use the **General** tab of the **Dial-by-Extension Menu** customizer window, as shown in Figure 148, to associate digits (typically entered by the caller from a telephone keypad) with an output branch label.

You can associate multiple inputs with a single output branch label, and you can associate only one output branch label with a given input.

The **Dial-by-Extension Menu** step receives a single digit entered by a caller and maps this entry to a series of option output branches. The system executes the steps that you add after each of these option output branches.

The customizer window of the **Dial-by-Extension Menu** step contains four tabs:

- General Tab, page 156
- Prompt Tab, page 158
- Input Tab, page 158
- Dial-by-Extension Tab, page 160

General Tab

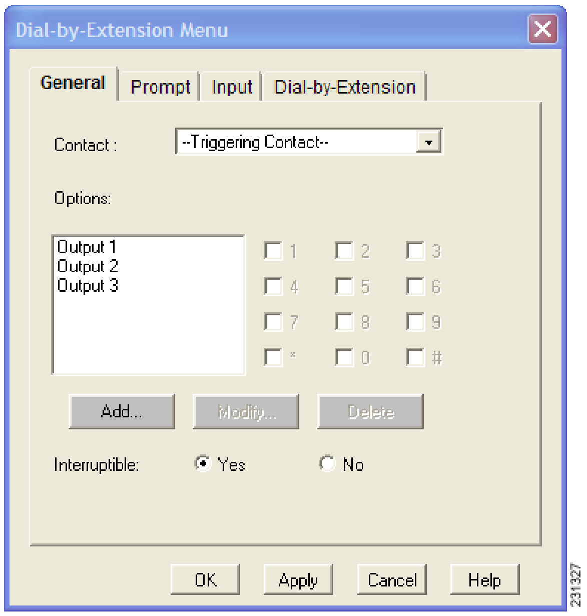
By default, the **Dial-by-Extension Menu** step has the following output branches:

- **Output 1**
- **Output 2**
- **Output 3**
- **Timeout**
- **Unsuccessful**
- **Digits Collected**

You can add more output branches in the **General** tab of the **Dial-by-Extension Menu** customizer window.

The **Dial-by-Extension Menu** step retries for either a timeout or an invalid digit entry (a digit that is not associated with any connections). If the maximum number of retries is reached, the **Dial-by-Extension Menu** step follows either the **Timeout, Unsuccessful**, or **Digits Collected** connection, depending on the reason for the latest failure.

Figure 148 Dial by Extension Menu Customizer Window: General Tab



Contact : --Triggering Contact--

Options:

Output 1
Output 2
Output 3

☐ 1 ☐ 2 ☐ 3
☐ 4 ☐ 5 ☐ 6
☐ 7 ☐ 8 ☐ 9
☐ * ☐ 0 ☐ #

Interruptible: [✓ Yes] No

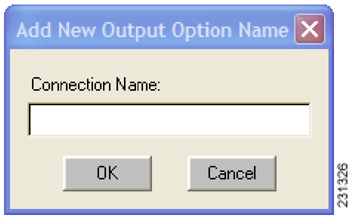
Table 85 describes the field of the **General** tab.

Table 85 Dial-by-Extension Menu Fields Customizer Window: General Tab

Field	Description
Contact	Contact that triggers execution of the step. Unless another contact is specified, default is Triggering Contact.
Options	One label for each possible output value.
Add	Add a new option. The new option appears in the **Options** list box.
Modify	Modifies the selected option using the **Rename Output Option** dialog box, which contains the same field as the **Add New Output Option Name** dialog box and is configured in the same way.
Interruptible	If **Yes**, an external event (such as a caller hanging up) can interrupt the step. If **No**, the step must complete before any other process can execute.

The **Add New Output Option Name** window is shown in Figure 149.

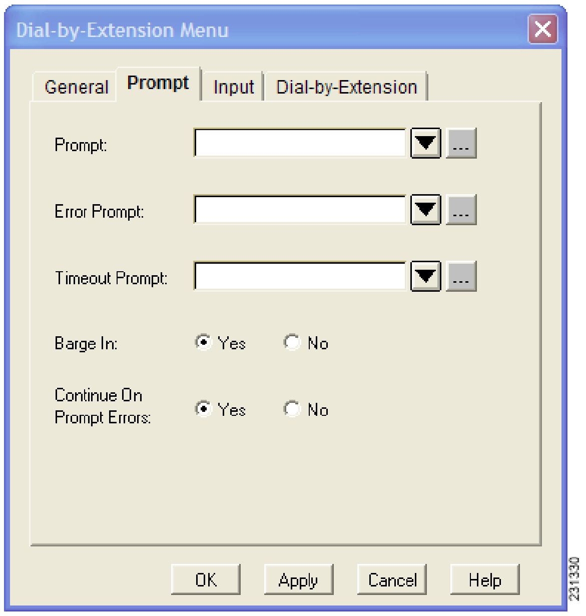
Figure 149 Add New Output Option Name Window



Prompt Tab

Use the **Prompt** tab of the **Dial-by-Extension Menu** customizer window, as shown in Figure 150, to set the Prompt Input, and to set the Barge in and Continue On Prompt Error conditions.

Figure 150 Dial-by-Extension Menu Customizer Window: Prompt Tab



Prompt:
Error Prompt:
Timeout Prompt:
Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 86 describes the field of the **Prompt** tab.

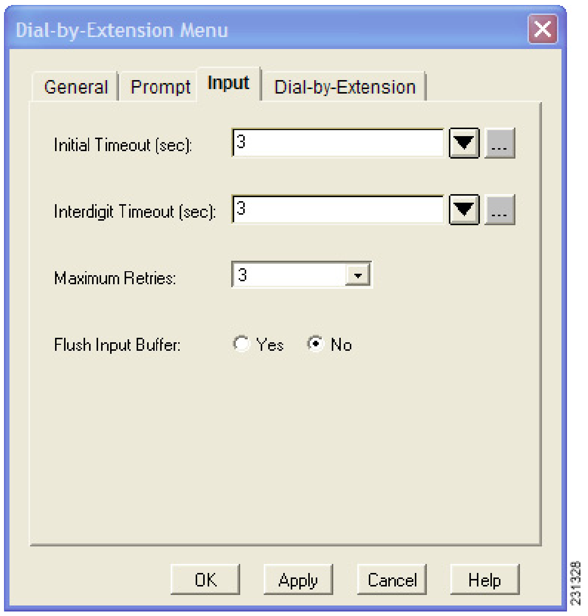
Table 86 Dial-by-Extension Menu Customizer Window: Prompt Tab

Property	Description
Prompt	Specifies which prompt is to be played.
Error Prompt	Prompt to be played if an input error occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Timeout Prompt	Prompt to be played if a timeout occurs. Enter a value, select a variable from the drop-down menu, or use the expression editor to enter a prompt expression.
Barge In	If true, the caller can interrupt the prompt. If false, the prompt must complete playing before the caller's input is accepted.
Continue On Prompt Errors	If Yes, the step continues with the next prompt in the list. If No, an exception is thrown, which can then be handled in the script.

Input Tab

Use the **Input** tab of the **Dial-by-Extension Menu** customizer window, as shown in Figure 151, to set the timeout setting, maximum number of retries, and **Flush Input Buffer** options.

Figure 151 Dial-by-Extension Menu Customizer Window: Input Tab



Initial Timeout (sec): 3
Interdigit Timeout (sec): 3
Maximum Retries: 3
Flush Input Buffer: Yes [✓ No]

Table 87 describes the field of the **Prompt** tab.

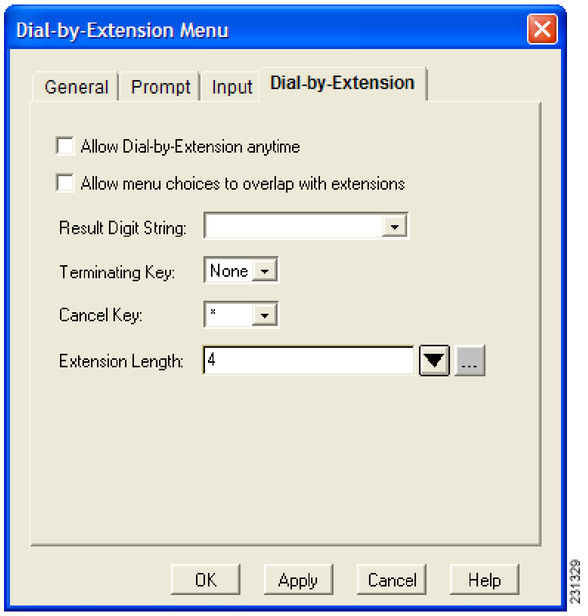
Table 87 Dial-by-Extension Menu Window Fields: Input Tab

Property	Description
Initial Timeout (sec)	Amount of time the system waits for input from the caller. When this timer expires, the system either replays the prompt or plays the system prompt that asks if the caller is still there. Enter a value, choose the variable that stores the timeout value from the **Timeout** drop-down menu, or click the **Expression Editor** (**...**) button and enter a number.
Interdigit Timeout (sec)	Amount of time the system waits for input between digits that are pressed by the caller. When this timer expires, the system either replays the prompt or plays the system prompt that asks if the caller is still there. Enter a value, choose the variable that stores the timeout value from the **Timeout** drop-down menu, or click the **Expression Editor** (**...**) button and enter a number.
Maximum Retries	Number of times the entry can be restarted after a timeout or invalid input response. After the maximum number of retries is reached, the **Menu** step follows the **Timeout** or **Unsuccessful** output branches depending on whether the last try timed out or an invalid input response was entered. A "0" value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	If **Yes,** the system erases previously entered input before capturing caller input. If **No**, the system does not erase previously entered input before capturing caller input.

Dial-by-Extension Tab

Use the **Prompt** tab of the **Dial-by-Extension Menu** customizer window, as shown in Figure 152, to set the dial-by-extension options.

Figure 152 Dial-by-Extension Menu Customizer Window: Dial by Extension Input Tab



Allow Dial-by-Extension anytime
Allow menu choices to overlap with extensions
Result Digit String:
Terminating Key: None
Cancel Key: *
Extension Length: 4

Table 88 lists and describes the field of the **Dial-by-Extension** tab.

Table 88 Dial-by-Extension Menu Window Fields: Dial-by-Extension Tab

Property	Description
Allow Dial-by-Extension anytime	• **Checked box**: Causes Cisco Unity Express to allow the dial-by-extension function all the time. • **Unchecked box**: Causes Cisco Unity Express to not allow the dial-by-extension function all the time.
Allow menu choices to overlap with extensions	• **Checked box**: Causes Cisco Unity Express to allow the dial-by-extension function to overlap with extensions. • **Unchecked box**: Causes Cisco Unity Express to not allow the dial-by-extension function to overlap with extensions.
Result Digit String	Variable that holds the resulting digit string.
Terminating Key	Key the caller presses to terminate entering the extension number.
Cancel Key	Key the caller presses to cancel entering the extension number.
Extension Length	Number of digits in the extension.

Voice Browser (IVR Only)

Use the **Voice Browser** step to invoke a VoiceXML application.

The **Voice Browser** step supports DTMF signaling only. A sample script (webapp.aef) is available on Cisco.com at the "Download Software" site for Cisco Unity Express.

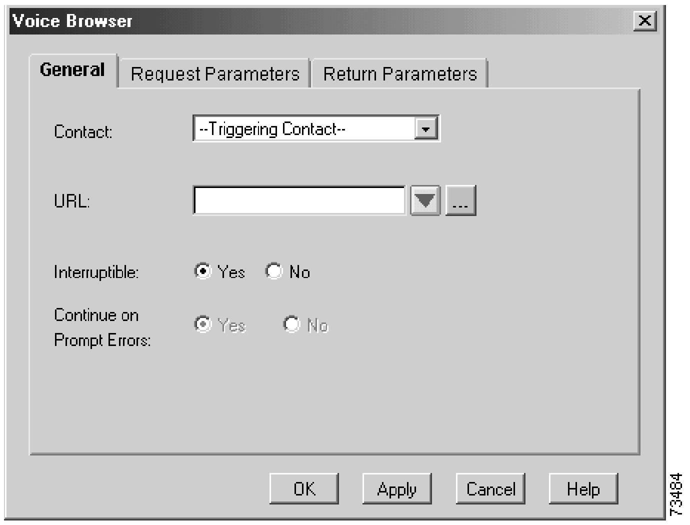
The customizer window of the **Voice Browser** step contains three tabs:

- General Tab, page 161
- Request Parameters Tab, page 162
- Return Parameters Tab, page 162

General Tab

Use the **General** tab, as shown in Figure 153, to identify the contact and to choose a variable that points the browser toward a specific URL.

Figure 153 Voice Browser Customizer Window: General Tab



Contact: --Triggering Contact--
URL:
Interruptible: [✓ Yes] No
Continue on Prompt Errors: [✓ Yes] No

Table 89 describes the fields of the **General** tab.

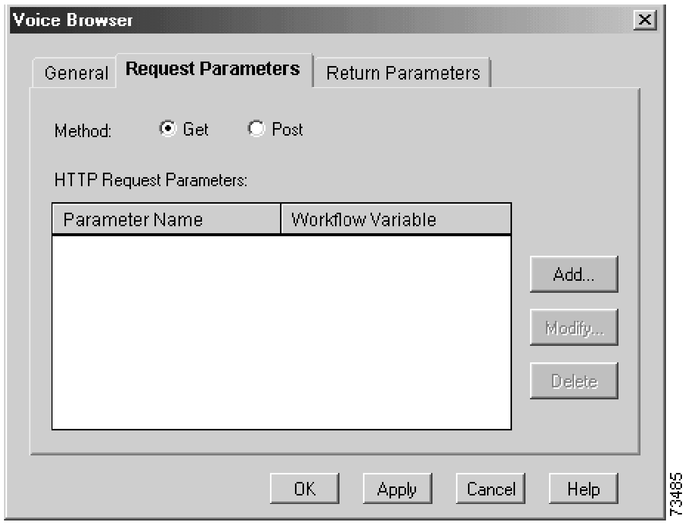
Table 89 Voice Browser Customizer Window Fields: General Tab

Field	Description
Contact	The default is Triggering Contact, unless another contact is specified.
URL	A URL can be entered manually or a variable containing the URL can be selected from the drop-down list.
Interruptible	Enable or disable interruptions caused by external events. • Yes: The step can be interrupted by external events, • No: The step will terminate before any external events are considered.
Continue on Prompt Errors	Disabled for the **Voice Browser** step.

Request Parameters Tab

Use the **Request Parameters** tab, as shown in Figure 154, to define parameters that are to be passed with an HTTP request to retrieve the desired VoiceXML content.

Figure 154 Voice Browser Customizer Window: Request Parameters



Method: [✓ Get] Post

HTTP Request Parameters:

Parameter Name	Workflow Variable

Table 90 describes the fields of the **Request Parameters** tab.

Table 90 Voice Browser Customizer Window: Request Parameter Tab

Field	Description
Method	Method to use if the URL represents an HTTP request. • **GET**: Appends parameters to the URL. This step is equivalent to using the document expression form URL[url?name=value,name=value]. • **POST**: Includes the parameters as if they were entered in an HTML form.
HTTP Request Parameters list box	HTTP Request parameter name and its corresponding workflow variable name.

Return Parameters Tab

Use the **Return Parameter** tab, as shown in Figure 155, to return parameter information back to the .aef script file from the VoiceXML application.

Figure 155 Voice Browser Customizer Window: Return Parameters Tab

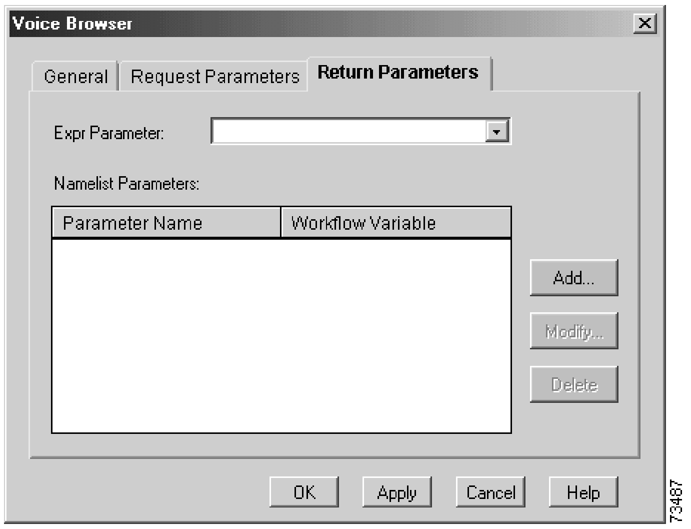


Table 91 describes the fields of the **Return Parameter** tab.

Table 91 Voice Browser Customizer Window: Return Parameters Tab

Field	Description
Expr Parameter	Variable that will store the return value if the VoiceXML application uses the following exit statement form: `<exit expr = "`*`ECMAScript_Expression`*`"/>`
Namelist Parameters	Name of the parameter returned by the VoiceXML application and the corresponding script variable in which its value is stored if the VoiceXML application uses the following exit statement form: `<exit namelist = "`*`string`*`"/>`

Prompt Steps

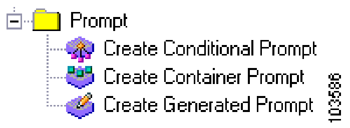
The steps in the **Prompt** palette of the Cisco Unity Express Script Editor provide script designers with a way to create intelligent prompts.

The **Prompt** palette contains the following steps:

- Create Conditional Prompt, page 164
- Create Container Prompt, page 165
- Create Generated Prompt, page 168

Figure 156 shows the steps in the **Prompt** palette as they appear in the Palette pane of the Cisco Unity Express Script Editor.

Figure 156 Prompt Palette Steps



Create Conditional Prompt

Use the **Create Conditional Prompt** step to create a prompt based on the result of evaluating a specified Boolean expression.

The prompts passed are evaluated immediately as prompt objects, and they are not resolved until the time of playback. This means that if the values of any variables entered as part of the expression change between the time this prompt was created and the time the prompt is played back, then the new value of the variable is used to evaluate the conditional expression.

Figure 157 shows the customizer window for the **Create Conditional Prompt** step.

Figure 157 Create Conditional Prompt Customizer Window

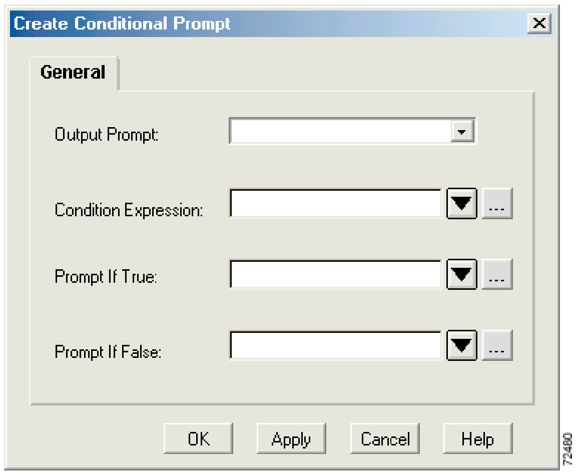


Table 92 describes the fields of the **Create Conditional Prompt** customizer window.

Table 92 Create Conditional Prompt Fields

Field	Description
Output Prompt	Variable that stores the prompt that results from the **Create Conditional Prompt** step. The name of the conditional prompt appears next to the **Create Conditional Prompt** step icon in the Design pane.
Condition Expression	Boolean expression the script uses to decide which one of the two prompts to play back. Enter a value, choose the variable that stores the expression to be used to evaluate the condition from the **Condition Expression** drop-down menu, or click the **Expression Editor (...)** button, and enter the expression for evaluating the condition.

Table 92 Create Conditional Prompt Fields (continued)

Field	Description
Prompt If True	Prompt to use if the expression is True. Enter a value, choose the variable that stores the prompt to use if the expression evaluates to True from the **Prompt If True** drop-down menu, or click the **Expression Editor** (**...**) button, and enter an expression that specifies the prompt to use if the expression evaluates to True.
Prompt If False	Prompt to use if the expression is False. Enter a value, choose the variable that stores the prompt to be used if the expression evaluates to False from the **Prompt If False** drop-down menu, click the **Expression Editor** (**...**) button, and enter an expression that specifies the prompt to use if the expression evaluates to False.

Create Container Prompt

Use the **Create Container Prompt** step to combine multiple prompts into one larger prompt. You can create three types of container prompts:

- **Concatenated Prompt**: Contains a list of prompt phrases that are played back in a specific sequence to create a single prompt.

  For example, for a prompt of “Your checking account balance is one hundred and sixty-eight dollars,” you can create a concatenated prompt that (1) begins with a user prompt “Your”; (2) continues with a conditional prompt that specifies a condition such as <accountType == “check”>, and plays “checking account” if the condition is True or “savings account” if the condition is False; and (3) ends with the balance amount.

- **Escalating Prompt**: Provides an initial question prompt with a minimal amount of information, and then adds additional prompt phrases if no response is given.

  For example, for a prompt that provides the caller with more information as needed, you can create an escalating prompt that, when passed to a media step such as the **Get Digit String** step, begins by playing the first concise prompt inside the escalating prompt, such as “What is your account number”?

  If the step fails to collect the account number because of the caller’s failure to provide it, a second prompt plays, such as “Please provide your account number by entering the account number using your touch tone phone followed by the pound key.”

- **Random Prompt**: Creates a prompt that plays back one phrase from the supplied list in a random order; for example, the system could play back a series of promotional or informational messages in a random order while a caller is waiting for an available agent.

Figure 158 shows the customizer window for the **Create Container Prompt** step.

Figure 158 Create Container Prompt Customizer Window

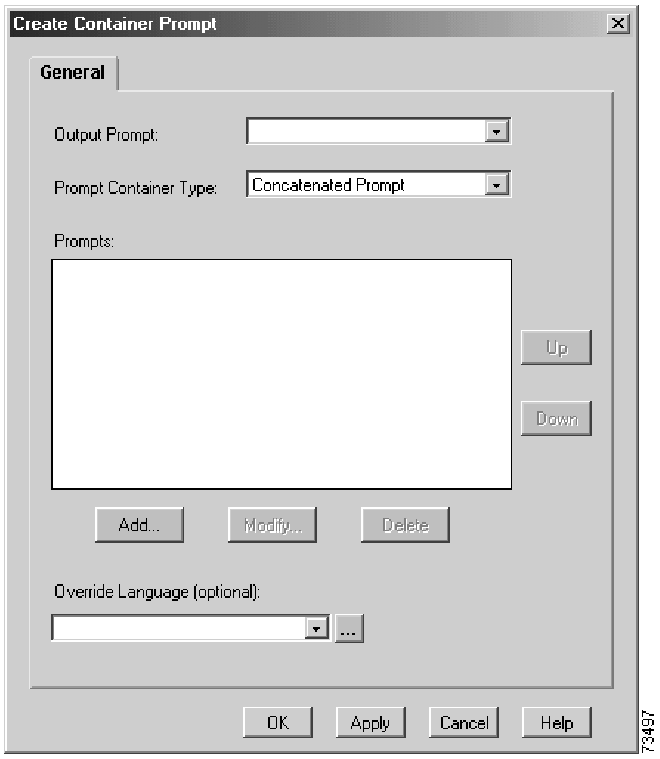


Table 93 describes the properties of the **Create Container Prompt** customizer window.

Table 93 Create Container Prompt Fields

Property	Description
Output Prompt	Script variable that holds the combined prompt generated by the Create Container Prompt step. The name of the concatenated prompt appears next to the **Create Container Prompt** step icon in the Design pane.
Prompt Container Type	Concatenated, escalating, or random prompt.
Prompts	List of prompts to be combined into the container prompt.
Add	Add a prompt. Enter a value, choose the variable that stores the prompt you want to add from the **Prompt** drop-down menu, or click the **Expression Editor** (**...**) button and enter an expression that specifies the prompt you want to add. For a random prompt, choose a number to represent the priority of the prompt in the sequence from the **Weight** drop-down menu. The name of the prompt variable appears in the **Prompts** list box in the **Create Container Prompt** customizer window.
Modify	Modify selected prompt using the **Modify Prompt** dialog box, which contains the same field as the **Add Prompt** dialog box and is configured in the same way.

Table 93 Create Container Prompt Fields (continued)

Property	Description
UP, **Down**	Determines the order of playback of the prompts. Select an individual prompt and click **Up** to move it up a level or click **Down** to move it down a level.
Override Language (optional)	This option is for future use.

The **Add Prompt** dialog box is shown in Figure 159 and the Random Prompt window is shown in Figure 160.

Figure 159 Add Prompt Dialog Box

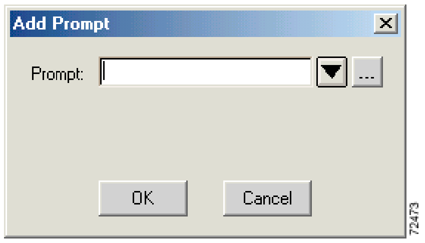
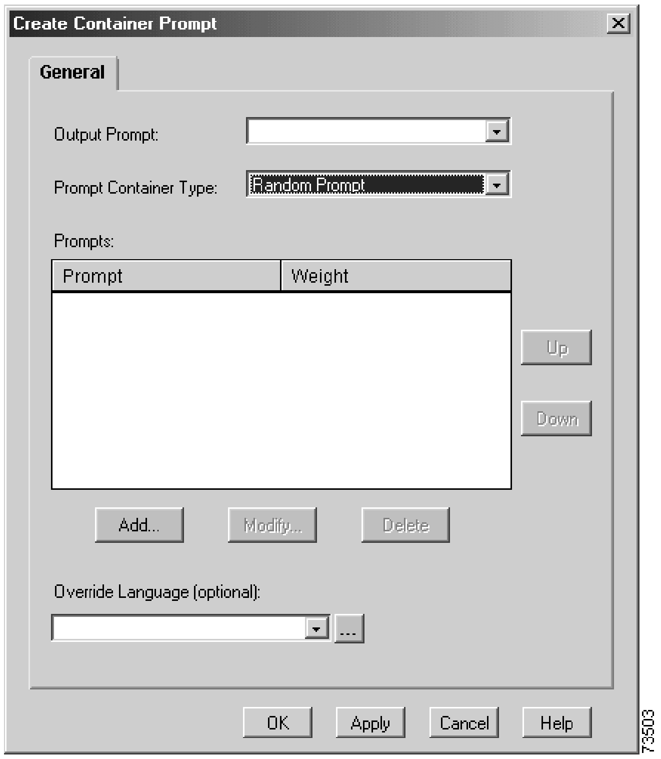
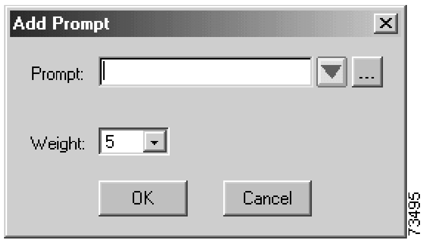


Figure 160 Create Container Prompt Customizer Window: Random Prompt



The **Add Prompt** dialog box is shown in Figure 161.

Figure 161 Add Prompt Dialog Box



Create Generated Prompt

Use the **Create Generated Prompt** step to create prompt phrases from intermediate variables whose values are dynamically determined based on run-time script information.

For example, you can create the prompt phrase of "account balance is one hundred and sixty-eight dollars" by querying the database of account balances at a particular point in the script and using a currency generator to generate the number.

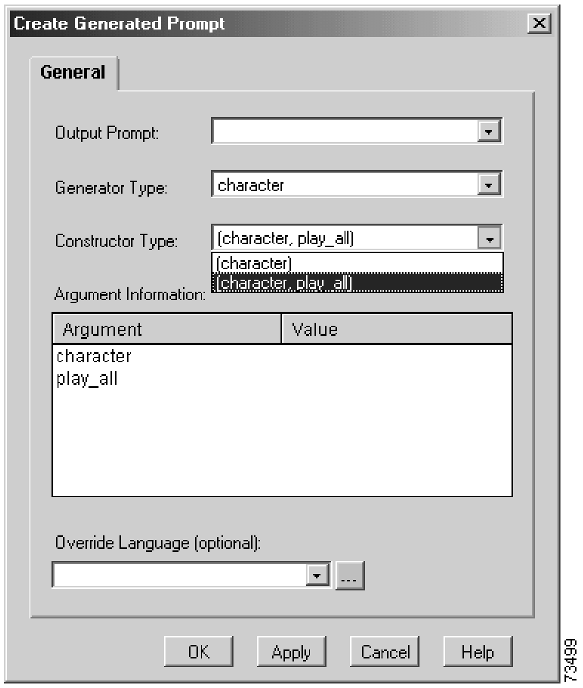
Note You cannot use the Script Editor to query a database.

Note The **Create Generated Prompt** step accepts only the 4-digit year format. A 3-digit date format is not accepted.

Note If the **Create Generated Prompt** step encounters an invalid time, it outputs 4:00 P.M. Specify a valid time between 0000 and 2400.

Figure 162 shows the customizer window for the **Create Generated Prompt** step.

Figure 162 Create Generated Prompt Customizer Window



Output Prompt:

Generator Type: character

Constructor Type: (character, play_all)

(character)
(character, play_all)

Argument Information:

Argument	Value
character	
play_all	

Override Language (optional):

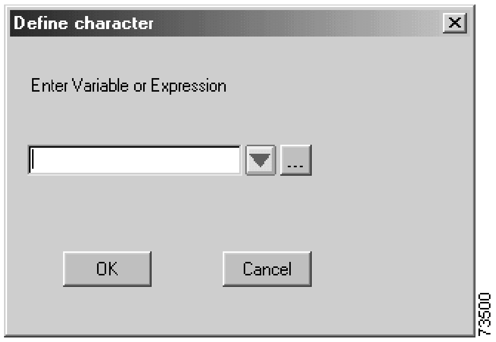
Table 94 describes the fields of the **Create Generated Prompt** customizer window.

Table 94 Create Generated Prompt Fields

Field	Description
Output Prompt	Prompt script variable in which the prompt object resulting from this step is stored. The name of the generated prompt appears next to the **Create Generated Prompt** step icon in the Design pane.
Generator Type	Type of information generated. See the following sections for descriptions of the 12 supported generator types.
Constructor Type	Constructor type that corresponds to the generator type.
Argument Information	Arguments and their values. Double-click to define a value for an item.
Enter Variable or Expression	Define a value for an argument. Enter a value, choose the variable that holds the value for the argument from the **Enter Variable Expression** drop-down menu, or click the **Expression Editor** (**...**) button and enter any valid expression. The name of the argument and its value appear in the **Argument Information** list box of the **Create Generated Prompt** customizer window.
Override Language (optional)	This option is for future use.

The **Define Character** dialog box is shown in Figure 163.

Figure 163 Define Character Dialog Box



Generator Types

The **Create Generated Prompt** step supports the following generator types:

- Number Generator, page 170
- Character Generator, page 171
- Spelling Generator, page 171
- Date Generator, page 171
- Time Generator, page 172
- Ordinal Generator, page 172
- Currency Generator, page 173
- Country Generator, page 173
- Language Generator, page 174
- Telephone Number Generator, page 174
- Credit Card Number Generator, page 174
- Credit Card Expiration Date Generator, page 174
- Fullname Generator, page 174

Number Generator

The **Number** generator type supports the following constructors:

- (Number number)
- (String number)
- (Number number, Number gender)
- (String number, Number gender)
- (Number number, Boolean play.full)
- (String number, Boolean play.full)
- (Number number, Boolean play.full, Number gender)
- (Number number, Boolean play.full, Number gender)

The three parameters are:

- Number: Any Number object (for example; Integer, Long, Float, Double, BigInteger, BigDecimal) or String object defining the number to be played back.
- Gender: When the number must be played back in a specific gender context, this parameter specifies the context. Valid values are 0 for neutral, 1 for male, and 2 for female.
- Play.full: Plays the number in full format if this optional Boolean argument is true or omitted. For example, "709" is played as "Seven Hundred and Nine." Otherwise, the number plays in brief format. For example, "709" is played as "Seven Oh Nine."

Note If the number is played in full format, the maximum number supported is ± 999,999,999,999.

Character Generator

The **Character** generator type supports the following constructors:

- (Character character)
- Character character, Boolean play_all)

The two parameters are:

- Character: The character object to be played back.
- Play_all: Optional Boolean flag indicating whether to play spaces, punctuation, and other special characters normally instead of playing them as silence (ranging from 250 to 500 ms).

Spelling Generator

The **Spelling** generator type supports the following constructors:

- (String string)
- (String string, Boolean punctuation)
- (Object object)
- (Object object, Boolean punctuation)

The three parameters are:

- String: String object to be played back.
- Object: Object for which the string representation returned by the `String.valueOf()` method should be spelled out.
- Punctuation: An optional Boolean flag indicating whether to play spaces, punctuations, and special characters normally or as silences.

Note Punctuation default behavior in the **Spelling** generator is different from the Play_all default behavior in the **Character** generator.

Date Generator

The **Date** generator type supports the following constructors:

- (Date date)
- (Date date, Boolean skip.current.year)

- (Number year)
- (Number year, Number month)
- (Number year, Number month, Boolean skip.current.year)
- (Number year, Number month, Number day)
- (Number year, Number month, Number day, Boolean skip.current.year)

The five parameters are:

- Date: Any Date object from which to extract the date to be played back.
- Skip.current.year: If set to true, the year does not play back if it is the same as the current year.
- Year: Year of the date to be played back. This year must be specified in full (for example, 2007).

Note The system plays any number given, so the caller is responsible for ensuring that the specified year is valid.

- Month: The month of the date to be played back. Valid values range from 1 to 12, where 1 represents January and 12 represents December.
- Day: The day of the date to be played back. Valid values range from 1 to 31 and are validated at run time based on the specified month and year.

Time Generator

The **Time** generator type supports the following constructors:

- (Time)
- (Hours, Minutes)

The three parameters are:

- Time: Any Date or Time object representing the time to be played back. Time can also be defined as a Number object (Integer, Float, Long, and so forth) that specifies the time to be played, from 0 to 2359. (For example, a number such as 12:34 is played as "12 34 PM.") If the value specified is greater than 2359, then Time is considered to be the number of milliseconds starting from the standard base time known as "the epoch," which is January 1, 1970, 00:00:00 GMT.
- Hours: Number object that specifies the hour to be played.
- Minutes: Number object that specifies the minutes to be played.

Ordinal Generator

The **Ordinal** generator type supports the following constructors:

- (Number number)
- (String number)
- (Number number, Number gender)
- (String number, Number gender)

The two parameters are:

- Number: Any Number or String object defining the ordinal number to be played back. The supported range is from 1 to 999999.

- Gender: When the ordinal number must be played back in a specific gender context, this parameter specifies this context. Valid values are 0 for neutral, 1 for male, and 2 for female.

Note If the language associated with the call does not behave differently based on gender, then this parameter is ignored.

Currency Generator

The **Currency** generator type supports the following constructors:

- (Currency designator)
- (Number amount)
- (Number amount, Currency currency)
- (Number dollar, Number cent)
- (Number dollar, Number cent, Currency currency)
- (Number amount, Boolean colloquial)
- (Number amount, Boolean colloquial, Currency currency)
- (Number dollar, Number cent, Boolean colloquial, Currency currency)

The six parameters are:

- Designator: The designator of a currency to play back. (For example, "USD" is played back as "U.S. Dollar.")
- Amount: The currency amount to be played back in the system configured default currency or in the specified currency.
- Dollar: Number object representing the amount of currency unit to be played. Only the integer part of the number is played. The fractional part, if any, is ignored.
- Cent: Number object representing the currency subdivision to be played. Only the integer part of the number is played. The fractional part, if any, is ignored.

Note If the number specified exceeds the maximum value allowed for the subdivisions, the excess is added properly to the number of the currency unit. For example, specifying "5 dollars and 233" cents results in "7 dollars and 33 cents."

- Colloquial: An optional Boolean flag, which specifies whether to use colloquial currencies' representations (for example, "Dollars" instead of "US Dollars"). If omitted, the currency amount is played in colloquial format.
- Currency: The currency in which the amount should be played back. If not specified, the system default configured currency is played back.

Country Generator

The **Country** generator type supports only one constructor: (Language language).

Language Generator

The **Language** generator type supports the language constructor. The parameter "language" is a Language object from which to get the language to be played back. (For example, en_US is played back as "United States English.")

Telephone Number Generator

The **Telephone Number** generator type supports only one constructor: (String number). The parameter "number" is a String object specifying the telephone number to be played out as a sequence of digits.

The character is replaced with 250 ms of silence if the string contains any of the following characters: " - ( ) `.` Otherwise, the string is automatically formatted.

Automatic formatting of the string inserts 250 ms of silence between sections of digits. These sections follow rule: "XXX-XXX-XXX-XXXX" unless there are exactly five digits in the string, in which case the string is considered to be a single section of five digits.

Dual-Tone Multifrequency (DMTF) digits include the following digits on a standard touchtone telephone keypad: `ABCD0123456789#*`**.** In DMTF, when you touch a button on the touchtone keypad, it creates a combination of two tones (one high-frequency and one low-frequency).

An "x" character is played back as "Extension." DTMF digits ("`ABCD0123456789#*`") are played back normally.

A string of the form "`*xx`" where `x` is a DTMF digit ("`0123456789`") is played back as "star `xx`" (for example,"`*53`" is played back as "star fifty-three").

Credit Card Number Generator

The **Credit Card Number** generator type supports only one constructor: (String number). The parameter "number" is a String object specifying the credit card number to be played out as a sequence of digits.

If the specified credit card number includes "-", then it is played as is, replacing the "-" character with 250 ms of silence; otherwise the number is automatically separated into sections of four digits and played back with 250 ms of silence inserted between sections.

Credit Card Expiration Date Generator

The **Credit Card Expiration Date** generator type supports the following constructors:

- (Number year, Number month, Number day)
- (Number year, Number month)

The parameters are identical to the following **Generated Date** constructors:

- If day is 0 or omitted: GeneratedDate (year, month, true)
- All other cases: GeneratedDate (year, month, day, true)

Fullname Generator

The **Fullname** generator type supports the following constructors:

- firstname: String object specifying the first name to be played out.
- lastname: String object specifying the last name to be played out.

Session Steps

The steps in the **Session** palette provide designers with a way to store custom variables in workflow scripts.

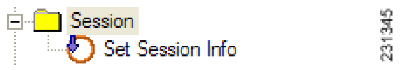
The custom variables can be assigned at run time during script execution and are stored in the historical reporting database on call completion.

The Session palette contains one step:

- Set Session Info, page 175

Figure 164 shows the step in the **Session** palette.

Figure 164 Set Session Steps



Set Session Info

The **Set Session Info** step supports the storage of custom data for use in the Call Contact Detail Record (CCDR) report.

The **Set Session Info** step contains the following two tabs:

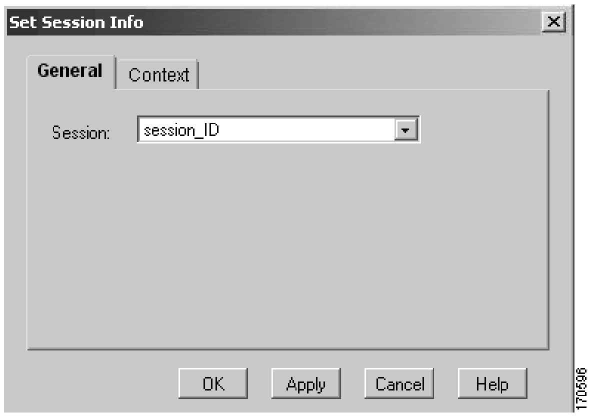
- General Tab, page 175
- Context Tab, page 176

General Tab

Use the **General** tab of the **Set Session Info** step, as shown in Figure 165, to specify the session variable for which you want to add or modify context information.

To configure session information, use the **General** tab to choose the variable from the Session drop-down menu and click Apply and complete the **Context** tab as shown in Figure 166.

Figure 165 Set Session Info Customizer Window General Tab



Session: session_ID

Table 95 describes the field of the **General** tab.

Table 95 Set Session Info Customizer Windows Field: General Tab

Field	Description
Session	The session for which you want to add or modify context information

Context Tab

Use the **Context** tab of the **Set Session Info** step to add or modify the value of attribute variables.

Some session context properties are included as part of the Call Contact Detail Record stored in the historical database for the call associated with this session. The attribute names for these properties are _ccdrVar1 to _ccdrVar10.

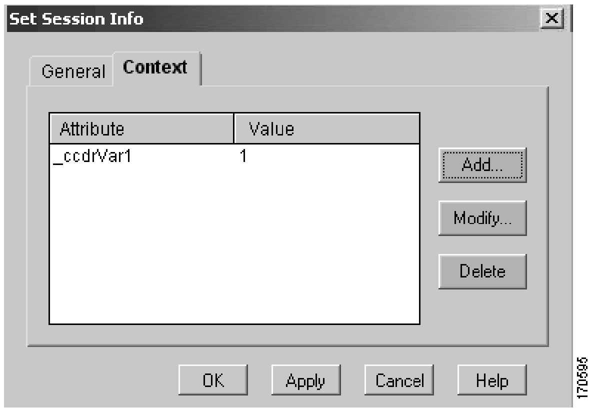
For example, the **Set Session Info** step could be used to assign the desired value(s) to one or more of the context variables named _ccdrVar1, _ccdrVar2, _ccdrVar3, _ccdrVar4, _ccdrVar5, _ccdrVar6, _ccdrVar7, _ccdrVar8, _ccdrVar9, and _ccdrVar10.

With the exception of the _ccdrVar10 field, each of the _ccdrVar fields can contain an alphanumeric value of up to 40 characters in length. The _ccdrVar10 field can contain an alphanumeric value of up to 256 characters in length.

Note The _ccdrVar names must begin with a single underscore (_) to successfully write data to the ccdr database.

Figure 166 shows the customizer window for the **Context** tab.

Figure 166 Set Session Info Customizer Window: Context Tab



Attribute	Value
_ccdrVar1	1

Table 96 describes the field of the **Context** tab.

Table 96 Set Session Info Customizer Window Fields: Context Tab

Field	Description
Attribute/Value	Attributes and associated values for the session identified in the **General** tab. You can assign values to ten Call Contact Detail record (CCDR) attributes to be stored in the historical database: _ccdrVar1 to _ccdrVar10. The attributes _ccdrVar1 to _ccdrVar9 allow a maximum of 40 characters. The _ccdrVar10 attribute allows a maximum of 255 characters.

User Steps

The steps in the **User** palette provide designers with a way to retrieve user attributes.

The User palette contains the following steps:

- Get User Info, page 177
- Extension To User, page 179
- User Authenticate, page 180
- Audit Step, page 181
- Authorize User, page 183
- Logout User, page 184

Figure 167 shows the steps in the **User Steps** palette for Cisco Unity Express 7.1 and earlier. Figure 168 shows the steps in the User Steps palette for Cisco Unity Express 8.0 and later.

Figure 167 User Steps Palette for Cisco Unity Express 7.1 and Earlier

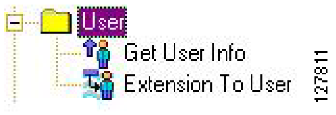
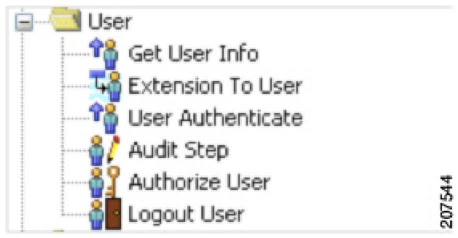


Figure 168 User Steps Palette for Cisco Unity Express 8.0 and Later



Get User Info

Use the **Get User Info** step, as shown in Figure 169, to make user attributes available to the script.

Figure 169 Get User Info Customizer Window

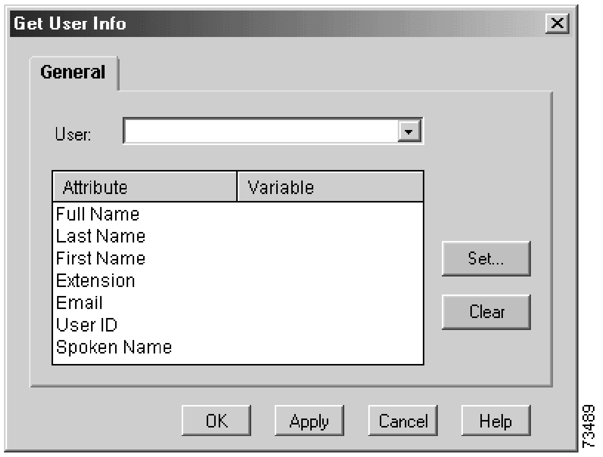


Table 97 describes the fields of the **Get User Info** customizer window.

Table 97 Get User Info Fields

Field	Description
User	User object stored in a script variable previously acquired (for example, by using a step such as the **Name To User** step). The name of the user variable appears next to the **Get User Info** step icon in the Design pane.
Attribute/Variable	Attributes and associated variables of user.
Set...	Displays the Get User Full Name dialog box. Select a variable for the user's full name.
Clear	Clears the variable set in the Get User Full Name dialog box.

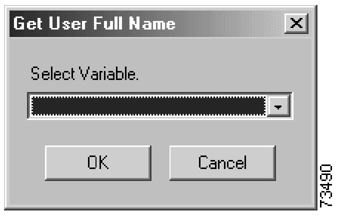
Table 98 describes the attributes that can be retrieved by using the **Get User Info** step.

Table 98 Get User Info Attributes

Attribute	Description
Full Name	String for the full name of the user as configured in the Cisco Unity Express GUI administration interface.
Last Name	String for the last name of the user.
First Name	String for the first name of the user.
Extension	String representing the primary extension selected in the Cisco Unity Express GUI administration web interface.
E-mail	String representing the e-mail ID for this user. The user ID field is currently returned.
User ID	String for the user ID configured for this user.
Spoken Name	Document object representing the recorded name of the user.

The **Get User** dialog box is shown in Figure 170.

Figure 170 Get User Dialog Box



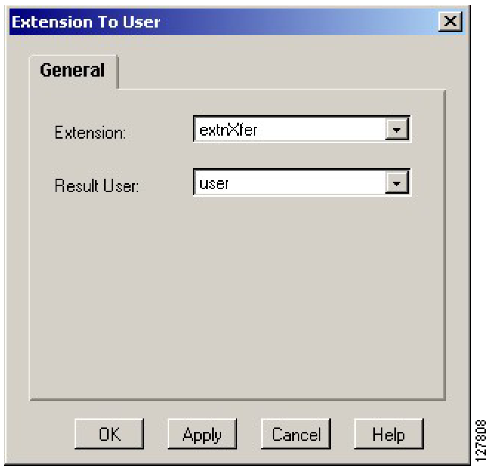
Extension To User

Use the **Extension To User** step as shown in Figure 171 to find a user based on the extension entered by the caller. The **Extension To User** step compares the extension entered by the caller with the extensions stored in a directory. If the system finds a match, it returns the user with the matched extension. This information can be used in the **Get User Info** step to get more information about the user. The **Extension To User** step can be used in a script to prevent transfers to external numbers.

The **Extension To User** step automatically adds two output branches:

- **Successful**: Steps following this branch execute if the system finds a user with an extension that matches the specified extension.
- **Unsuccessful**: Steps following this branch execute if the system does not find a user with an extension that matches the specified extension.

Figure 171 Extension To User Customizer Window



Extension: extnXfer

Result User: user

Table 99 describes the fields of the **Extension To User** window.

Table 99 Extension To User Customizer Window Fields

Field	Description
Extension	String representing the primary extension selected in the User pages of the Cisco Unity Express Administration pages. The name of the **Extension** variable appears next to the **Extension To User** step icon in the Design pane.
Result User	Variable that stores the User object that represents the user with the given extension.

User Authenticate

Prerequisite: Cisco Unity Express 8.0 or later

Use the User Authenticate step, as shown in Figure 172, to authenticate a user against the local system. Authentication can be done based on any of the following four combinations:

- User ID and password
- User ID and PIN
- User extension and password
- User extension and PIN

If authentication succeeds, the steps returns the authenticated User object. The authenticated user can be logged out of the system by executing the Logout User step. if the script execution ends and an authenticated user hasn't been logged out, the system automatically logs the user out.

Executing this step will not result an any accounting events to be generated. The Audit Step must be used with the appropriate event type (LOGIN) to generate an accounting record.

Figure 172 User Authenticate Window

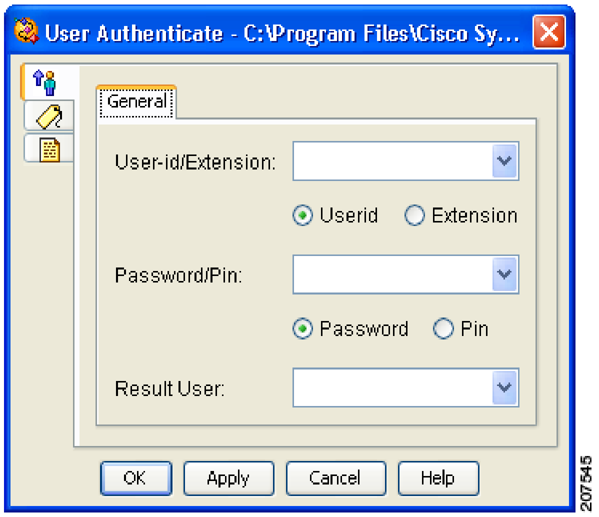


Table 100 describes the properties of the User Authenticate step.

Table 100 User Authenticate Properties

Property	Description
User-id/Extension	Input string variable containing the user-id or extension.
User-id/Extension Radio Button	Specifies whether the variable identifies a User-id or an Extension.
Password/Pin	Input string variable containing the password or PIN.
Password/Pin Radio Button	Specifies whether the variable identifies a password or a PIN.
Result User	If authentication succeeds, the resulting user object is returned in this field.

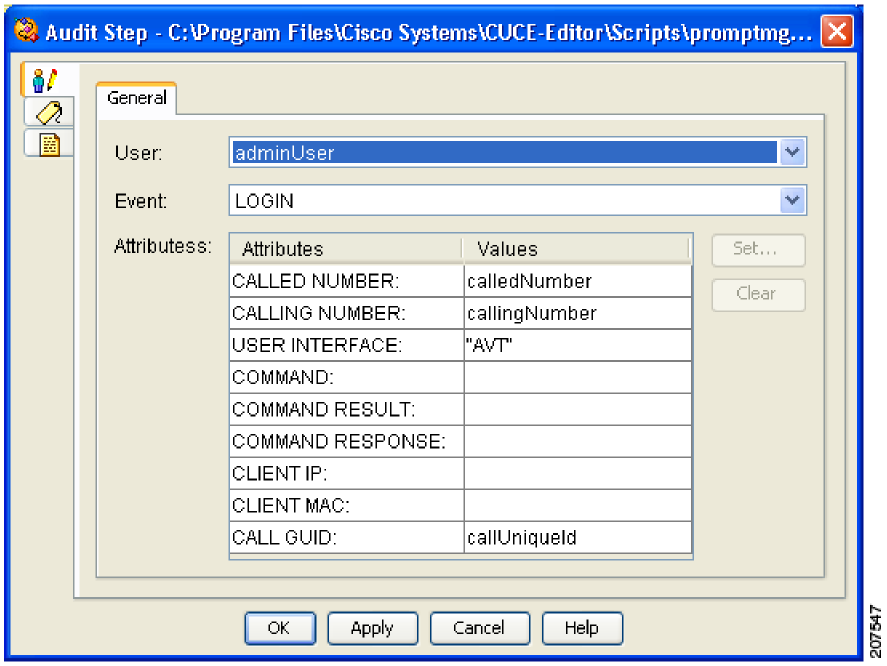
Audit Step

Prerequisite: Cisco Unity Express 8.0 or later

Use the Audit Step, as shown in Figure 173 to track the operations performed by a user within the context of an AEF script and to send the accounting records to the AAA server. The Audit Step allows for sending the following different kinds of accounting records:

1. Login: A user was successfully authenticated
2. Login_Fail: User authentication failed
3. Logout: User logged out of the system
4. Exec: User executed a command
5. Config: User made configuration changes

Figure 173 Audit Step



Audit Step - C:\Program Files\Cisco Systems\CUCE-Editor\Scripts\promptmg...

General

User: adminUser

Event: LOGIN

Attributess:

Attributes	Values
CALLED NUMBER:	calledNumber
CALLING NUMBER:	callingNumber
USER INTERFACE:	"AVT"
COMMAND:	
COMMAND RESULT:	
COMMAND RESPONSE:	
CLIENT IP:	
CLIENT MAC:	
CALL GUID:	callUniqueId

Using the Audit Step, the following attributes can be specified for the accounting record:

- CALLED NUMBER: Accounting attribute to identify the service requested by the phone number the user dialed. This information can be extracted from the Get Call Contact Info step.
- CALLING NUMBER: Accounting attribute to identify the caller by the phone number. This information can be extracted from the Get Call Contact Info step.
- USER INTERFACE: Accounting attribute to identify the user interface that accepted the user input. This can be set to the name of the script/ccn application.
- COMMAND: Accounting attribute defining the command the user requested access to.
- COMMAND RESULT: Accounting attribute indicating the result of the command execution.
- COMMAND RESPONSE: Accounting attribute indicating a more detailed explanation of the result.
- CLIENT IP: Accounting attribute identifying the IP address of the client.
- CLIENT MAC: Accounting attribute identifying the MAC address of the client.
- CALL GUID: Accounting attribute identifying the call global unique ID. This information can be extracted from the Get Call Contact Info step.

Note Cisco recommends that the Audit Step be inserted in each branch of the User Authenticate and Authorize User steps, and before the Logout User step.

Table 101 describes the properties of the Audit Step.

Table 101 Audit Step Properties

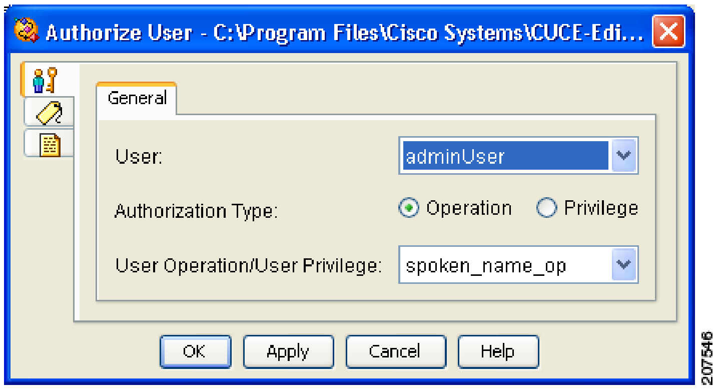
Property	Description
User	Input user variable that identifies the user being authorized. This variable can be a user object or a user name (string).
Event	Specifies the event type for the accounting record.
Attributes	Table containing the various attributes for the accounting record.

Authorize User

Prerequisite: Cisco Unity Express 8.0 or later

Use the User Authorize step, as shown in Figure 174, to authorize a previously authenticated user for a given system operation or a user-defined privilege. The step stakes as input a user object, and an operation or privilege name. If authorizing against an operation, the step returns success if the user is authorized to perform the requested operation. If authorizing against a privilege, the step returns success if the user has the requested privilege.

Figure 174 Authorize User Step


Authorize User - C:\Program Files\Cisco Systems\CUCE-Edi...

General

User: adminUser

Authorization Type: [✓ Operation] Privilege

User Operation/User Privilege: spoken_name_op

Table 102 describes the properties of the Authorize User step.

Table 102 Authorize User Properties

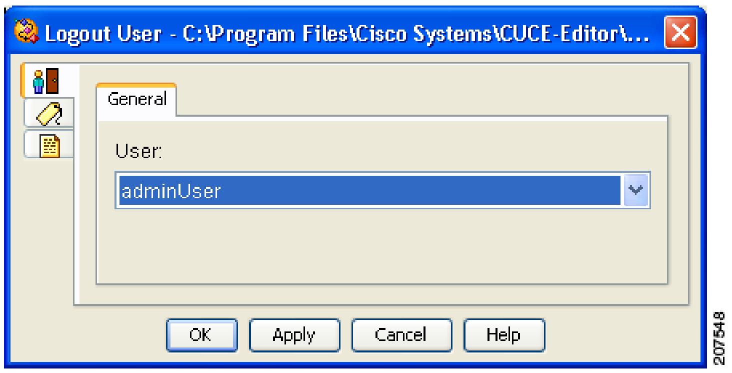
Property	Description
User	Input User variable that identifies the user being authorized.
Authorization Type	Specifies whether the authorization is against a system operation or a privilege.
User Operation/ User Privilege	If the Authorization Type is a system operation: Input String variable that identifies the system operation that the user wants to perform. If the Authorization Type is for privilege: Input String variable that identifies the privilege.

Logout User

Prerequisite: Cisco Unity Express 8.0 or later

Use the Logout User step, as shown in Figure 175, to log out a user that was previously authenticated using the User Authenticate step. If the script execution ends, and an authenticated user has not been logged out, then the system will automatically logout the user.

Figure 175 Logout User Step


Logout User - C:\Program Files\Cisco Systems\CUCE-Editor\...
General
OK
Apply
Cancel
Help
207548

User: adminUser

Table 103 describes the properties of the Logout User step.

Table 103 Logout User Step Properties

Property	Description
User	Input user variable that is to be logged out.

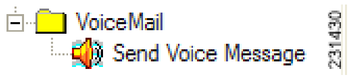
VoiceMail Step

The step in the **VoiceMail** palette provides designers with a way to configure voice-mail attributes.

The VoiceMail palette contains the Send Voice Message step.

Figure 176 shows the steps in the **VoiceMail Steps** palette.

Figure 176 VoiceMail Palette Steps

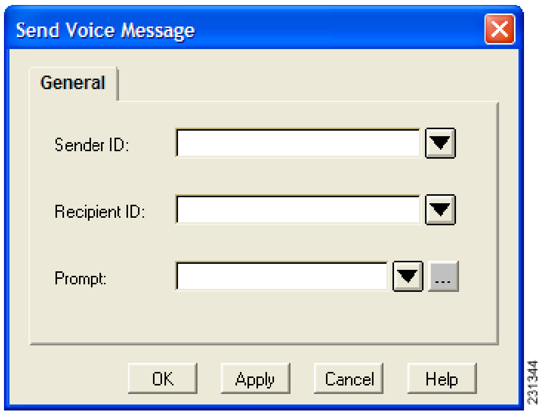


General Tab

Use the **General** tab of the **VoiceMail** step, as shown in Figure 177, to specify the Sender ID, Recipient ID, and Prompt variables for which you want to add or modify context information.

To configure the Send Voice Message information, use the **General** tab to choose the variable from the drop-down menu and click Apply.

Figure 177 Send Voice Message Customizer Window


Send Voice Message
General
Sender ID:
Recipient ID:
Prompt:
...
OK
Apply
Cancel
Help
231344