



Element Specifications for Cisco Unified CVP VXML Server and Cisco Unified Call Studio Release 11.6(1)

First Published: 2017-08-24

Americas Headquarters

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
<http://www.cisco.com>
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883



Preface

- [Change History, on page iii](#)
- [About this Guide, on page iv](#)
- [Audience, on page iv](#)
- [Related Documents, on page iv](#)
- [Communications, Services, and Additional Information, on page iv](#)
- [Documentation Feedback, on page v](#)

Change History

This table lists and links to changes made to this guide and gives the dates those changes were made. Earliest changes appear in the bottom rows.

Change	See	Date
Initial Release of document for release 11.6(1)		August 2017
Updated Customer_Lookup element data information.	Element Data	
Updated Subflow Call description by adding a note.	Subflow Call	
Updated Rest_Client Element notes for XPath and JSONPath	Settings	
Updated Currency, Currency_with_Confirm, Date, Date_with_confirm, Digits, Digits_with_confirm, Form, Form_with_Confirm, Number, Number_with_Confirm, Phone, Phone_with_Confirm, Time, and Time_with_confirm for secure logging related information.	Settings	
Updated Web Service Elements for WSDL	Settings	

About this Guide

This document provides specifications for the elements that are included with VXML Server.

Audience

This guide is intended for network administrators who install or upgrade the Unified CVP software.

Related Documents

- *Documentation Guide for Cisco Unified Customer Voice Portal*
- *Compatibility Matrix for Unified CCE*
- *Configuration Guide for Cisco Unified Customer Voice Portal*

Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at [Cisco Profile Manager](#).

- To get the business impact you're looking for with the technologies that matter, visit [Cisco Services](#).
- To submit a service request, visit [Cisco Support](#).
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit [Cisco Marketplace](#).
- To obtain general networking, training, and certification titles, visit [Cisco Press](#).
- To find warranty information for a specific product or product family, access [Cisco Warranty Finder](#).

Cisco Bug Search Tool

[Cisco Bug Search Tool](#) (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.

Documentation Feedback

Provide your comments about this document to: mailto:contactcenterproducts_docfeedback@cisco.com.



CONTENTS

PREFACE

Preface **iii**

 Change History **iii**

 About this Guide **iv**

 Audience **iv**

 Related Documents **iv**

 Communications, Services, and Additional Information **iv**

 Documentation Feedback **v**

CHAPTER 1

Element Specifications **1**

 Introduction **1**

CHAPTER 2

Subflow Start **5**

 Events **5**

 Exit **5**

CHAPTER 3

Subflow Return **7**

CHAPTER 4

Subflow Call **9**

 Events **9**

 Exit **10**

CHAPTER 5

Application_Modifier **11**

 Settings **11**

 Exit States **12**

 Folder and Class Information **12**

 Events **12**

CHAPTER 6**Audio 13**

Audio Groups 13

Audio Playback 13

Custom VoiceXML Properties 13

Folder and Class Information 14

Events 14

CHAPTER 7**Counter 15**

Settings 15

Element Data 16

Exit States 16

Folder and Class Information 16

Events 16

CHAPTER 8**Callback_Add 17**

Settings 17

Element Data 18

Exit States 18

Folder and Class Information 18

Events 19

CHAPTER 9**Callback_Disconnect Caller 21**

Settings 21

Element Data 21

Exit States 22

Folder and Class Information 22

Events 22

CHAPTER 10**Callback_Enter_Queue 23**

Settings 23

Element Data 23

Exit States 23

	Folder and Class Information	24
	Events	24
<hr/>		
CHAPTER 11	Callback_Get_Status	25
	Settings	25
	Element Data	25
	Exit States	26
	Folder and Class Information	26
	Events	26
<hr/>		
CHAPTER 12	Callback_Reconnect	27
	Settings	27
	Element Data	27
	Exit States	28
	Folder and Class Information	28
	Events	28
<hr/>		
CHAPTER 13	Callback_Set_Queue_Defaults	29
	Settings	29
	Element Data	31
	Exit States	31
	Folder and Class Information	31
	Events	32
<hr/>		
CHAPTER 14	Callback_Update_Status	33
	Settings	33
	Element Data	34
	Exit States	34
	Folder and Class Information	34
	Events	34
<hr/>		
CHAPTER 15	Callback_Validate	35
	Settings	35

Element Data	35
Exit States	35
Folder and Class Information	36
Events	36

CHAPTER 16

Callback_Wait	37
Settings	37
Exit States	37
Folder and Class Information	37
Events	38

CHAPTER 17

Currency	39
Settings	40
Element Data	41
Exit States	42
Audio Groups	42
Currency Capture	42
End	42
Folder and Class Information	43
Events	43

CHAPTER 18

Currency_with_Confirm	45
Settings	46
Element Data	47
Exit States	48
Audio Groups	48
Currency Capture	48
Currency Confirm	49
End	49
Folder and Class Information	49
Events	50

CHAPTER 19

Customer_Lookup	51
Settings	51

Element Data	51
Exit States	52
Folder and Class Information	53
Events	53

CHAPTER 20

CVP Subdialog Return 55

Settings	55
Exit States	56
Folder and Class Information	56

CHAPTER 21

CVP Subdialog Start 57

Settings	57
Exit States	58
Folder and Class Information	58

CHAPTER 22

Database 59

Settings	60
Element Data	60
Session Data	61
Exit States	61
Folder and Class Information	61
Events	61
Create JNDI Database Connection in Tomcat for Use in VXML Applications	62
Summary	62
Steps	62

CHAPTER 23

Date 65

Settings	65
Element Data	66
Exit States	67
Audio Groups	67
Date Capture	67
End	68
Folder and Class Information	68

Events 68

CHAPTER 24

Date_with_Confirm 69

Settings 69

Element Data 70

Exit States 71

Audio Groups 72

 Date Capture 72

 Date Confirm 72

 End 73

Folder and Class Information 73

Events 73

CHAPTER 25

Digits 75

Settings 75

Element Data 76

Exit States 77

Audio Groups 78

 Digits Capture 78

 End 78

Folder and Class Information 78

Events 78

CHAPTER 26

Digits_with_Confirm 79

Settings 79

Element Data 81

Exit States 82

Audio Groups 82

 Digits Capture 82

 Digits Confirm 82

 End 83

Folder and Class Information 83

Events 83

CHAPTER 27**Alert 85**

- Settings 85
- Events 85
- Exit States 86

CHAPTER 28**Email 87**

- Settings 87
- Exit States 88
- Folder and Class Information 89
- Events 89
- Set Up Email Element 89

CHAPTER 29**Form 91**

- Settings 91
- Element Data 97
- Exit States 98
- Audio Groups 98
 - Form Data Capture 98
 - End 99
- Folder and Class Information 99
- Events 99

CHAPTER 30**Form_with_Confirm 101**

- Settings 101
- Element Data 108
- Exit States 109
- Audio Groups 110
 - Form Data Capture 110
 - Form Data Confirm 110
 - End 110
- Folder and Class Information 111
- Events 111

CHAPTER 31	FTP_Client	113
	Settings	113
	Element Data	115
	Exit States	116
	Other	116
	Events	116

CHAPTER 32	Math	117
	Examples	117
	Settings	117
	Operators and Functions	118
	Element Data	119
	Session Data	119
	Exit States	119
	Folder and Class Information	119
	Events	119

CHAPTER 33	Local Variables	121
	Set Value Element	121
	Change Implementation Order of Local Variables	121

CHAPTER 34	Menu Support for 2_Option_Menu Through 10_Option_Menu	123
	Settings	123
	Element Data	126
	Exit States	126
	Audio Groups	127
	Menu Option Capture	127
	End	127
	Folder and Class Information	127

CHAPTER 35	Number	129
	Settings	129

Element Data	130
Exit States	131
Audio Groups	132
Number Capture	132
End	132
Folder and Class Information	132
Events	132

CHAPTER 36

Number_with_Confirm	133
Events	133
Settings	133
Element Data	135
Exit States	136
Audio Groups	136
Number Capture	136
Number Confirm	136
End	137
Folder and Class Information	137

CHAPTER 37

Phone	139
Settings	139
Element Data	140
Exit States	141
Audio Groups	141
Phone Capture	141
End	142
Folder and Class Information	142
Events	142

CHAPTER 38

Phone_With_Confirm	143
Settings	143
Element Data	145
Exit States	145
Audio Groups	146

Phone Capture	146
Phone Confirm	146
End	147
Folder and Class Information	147
Events	147

CHAPTER 39	POD_Add	149
	Settings	149
	Element Data	150
	Session Data	150
	Exit States	151
	Folder and Class Information	151
	Events	151

CHAPTER 40	POD_Read	153
	Settings	153
	Element Data	154
	Exit States	154
	Folder and Class Information	154
	Events	155

CHAPTER 41	POD_Update	157
	Settings	157
	Element Data	158
	Exit States	158
	Folder and Class Information	158
	Events	159

CHAPTER 42	Record	161
	Settings	161
	Element Data	165
	Exit States	165
	Audio Groups	166
	Record Capture	166

Folder and Class Information 166

Events 166

CHAPTER 43

Record_With_Confirm 167

Settings 167

Element Data 171

Exit States 171

Audio Groups 172

Record Capture 172

Record Confirm 172

Folder and Class Information 173

Events 173

CHAPTER 44

Rest_Client Element 175

Rest_Client 175

Settings 175

Element Data 180

Exit States 180

Events 181

CHAPTER 45

ReqICMLabel 183

Settings 183

Element Data 184

Session Data 185

Exit States 185

Folder and Class Information 185

Events 185

CHAPTER 46

Subdialog Invoke 187

Settings 187

Exit States 188

Folder and Class Information 188

Events 188

CHAPTER 47	Subdialog Return	189
	Settings	189
	Exit States	190
	Folder and Class Information	190

CHAPTER 48	Subdialog Start	191
	Settings	191
	Exit States	192
	Folder and Class Information	192

CHAPTER 49	Time	193
	Settings	193
	Element Data	194
	Exit States	195
	Audio Groups	195
	Time Capture	195
	End	196
	Folder and Class Information	196
	Events	196

CHAPTER 50	Time_With_Confirm	197
	Settings	197
	Element Data	199
	Exit States	200
	Audio Groups	200
	Time Capture	200
	Time Confirm	200
	End	201
	Folder and Class Information	201
	Events	201

CHAPTER 51	Transfer	203
-------------------	-----------------	------------

Settings	203
Element Data	205
Exit States	205
Audio Groups	205
Transfer Audio	205
End	206
Folder and Class Information	206
Events	206

CHAPTER 52
VideoConnect 207

Settings	207
Element Data	208
Exit States	208
Events	208
Others	208

CHAPTER 53
Web Service Elements 209

Exit States	210
Element Data	210
Settings	211
Configuring Request Parameters	216
Configuring Response Parameters	217

CHAPTER 54
Yes_No_Menu 219

Settings	219
Element Data	220
Exit States	220
Audio Groups	221
Yes / No Capture	221
End	221
Folder and Class Information	221
Events	221

CHAPTER 55
Throw 223

General	223
---------	-----



CHAPTER 1

Element Specifications

- [Introduction, on page 1](#)

Introduction

Every element included with Call Studio and VXML Server must be configured before it can be used. This reference file contains a detailed specification for each of the core Cisco Unified Customer Voice Portal (Unified CVP) elements, listing all the options available in the configuration. The specifications must be followed, or the element may complain with an error message or behave erratically.

Each element specification in this reference file presents information on some or all of the following topics:

- **Overview** – Each specification starts with a brief description of the element’s behavior including what it does, how it reacts to various settings and audio groups, and other miscellaneous behavior. This information should help the developer decide whether to use these elements in an application or to rely on custom elements.
- **Settings** – Settings contain information that affects how the element behaves. Each setting has the following attributes:
 - **Type** – The type of data accepted such as a string, text, boolean, integer, or enumeration.



Note For string data type, white spaces are accepted (or truncated) in input. However, for text data type, white spaces are not allowed.

- **Required** – This defines whether the setting is required to have a value *if the setting is active* (available to be configured in Builder for Studio).



Note The definition of required in this case is that the setting must have an appropriate value for Builder for Studio to validate the voice element configuration.

- **Single setting value** – This defines whether the setting can have multiple values. If set to `true`, then the setting may have only a single configuration value. Multiple value settings are created in Builder for Studio by right clicking on the setting and choosing the *add setting name* option.

- **Substitution allowed** – This setting attribute determines if the setting value can include substitution.
- **Default** – The initial value of an element setting when a new element is dragged to the workspace.
- **Events** – Event and exceptions occurring in a Cisco Unified Call Studio application can be handled by event handlers defined in the applications. When event handlers are configured for elements, the corresponding exit states are created. Following events types are supported:
 - Custom Exception- User defined application specific exception.
 - Java Exception - Java Exception occurring on a VXML server when executing applications.
 - VXML Event - VXML events thrown by the Voice XML browser.
 - Hotlink - Local Hotlinks defined for voice elements.

Event handlers can be configured with the following attributes:

- **Name** - The Event Handler name can be changed according to the requirement and the Event Handler name will be added as an exit state.
- **Event Type** - You can select the event handler type depending on the element, the applicable event types are listed in the drop-down list.
- **Event List** - You can select from a pre-defined list of VXML and Java exceptions using the drop-down list. In addition, you can also enter a comma separated list of VXML events, Java exceptions, or user defined custom exceptions based on the Event Type selected.

Example 1, for VXML event you can enter error.badfetch to catch a VXML event named error.badfetch

Example 2, for Java Exception event you can enter "java.io.FileNotFoundException" to catch a Java exception named java.io.FileNotFoundException

Example 3, for Custom Exception event you can enter "com.cisco.CustomException" to catch a user defined exception named com.cisco.CustomException



Note You can enter *.* to handle all the events and exceptions. * is allowed only at the end of the event name followed by "."(dot).

- **DTMF** - A digit which activates the hotlink. This attribute is applicable to Hotlink event handlers.
- **Speech** - A spoken keyword or keywords which activate the hotlink. This attribute is applicable to Hotlink event handlers.
- **Throw Hotevent** - The Voice XML event to be thrown when Hotlink is activated. When choosing the option to throw an event, the full name of the VoiceXML event must be entered in the provided text box.
- **Element Data** – Some elements capture data or yield information that may be useful to other elements, or for logging purposes. The variables created by each element are listed here.
- **Exit States** – Each element may have one or more exit states that indicate the dialog status when the element execution has completed. These are pre-defined Exit states that do not appear in an element configuration and cannot be changed. However, when an Event handler is associated to the elements,

the corresponding Exit state (<event handler type>-<event handler name> is added along with the pre-defined state.

- **Audio Groups** – Voice elements define audio groups that define the different places within the element that audio can be played. Application designers configure the contents of audio groups as a list of audio items that are played one after the other. Audio items may be pre-recorded audio files, text-to-speech (TTS) phrases, and Say It Smart types (playback of formatted data such as dates, currency amounts, and so on). Each audio group can be required or optional and can also define multiple counts. Audio groups with multiple counts are used to define different audio to play each time a certain VoiceXML event occurs (often known as tapered prompts).

**Note**

You can create your custom elements or use additional Java classes in the Cisco Call Studio. If you need support in developing or troubleshooting it, you must have a developer support services contract or work with a Cisco partner/Cisco Advanced Services who has a developer support services contract.



CHAPTER 2

Subflow Start

Subflow Start element is the first element for a subflow. This element is not created from the element view however, it is created automatically when a new subflow is created. Subflow Start element cannot be deleted it can just be renamed. You can have only one Subflow Start element in a subflow. Subflow Start element provides the definition of a subflow using its configuration. This element defines the parameters subflow can receive during the subflow execution. Subflow Start Element uses a data model to save its configuration which is implemented in `SubflowStartConfig` class. The Subflow Argument Data available at the Element Configuration view. Subflows accepts inputs from the calling flows as arguments. Subflow Call element allows to send multiple arguments of different types to a subflow. The set of arguments in Subflow Start should match with the set of arguments in Subflow Call.

- [Events, on page 5](#)
- [Exit, on page 5](#)

Events

Name (Label)	Notes
Event Type	You can select Java Exception, VXML Event, or Custom Exception event handler type for this element from the drop-down list.

Exit

Name	Notes
next	The default exit state. The events that are entered for this element as added as the exit state in the call flow.



CHAPTER 3

Subflow Return

The `Subflow Return` element is the exit point for the subflow execution. The `Subflow Return` element execution returns the call flow control back to the `Call Subflow` element. `Subflow Return` element has no exit state as it is the last element in a subflow execution. The `Subflow Return` element is used to returned data configured to a calling application. `Subflow Return` Element uses a data model to save its configuration which is implemented in the `SubflowReturnConfig` class. The Element configuration view displays the configuration of `Subflow Return` element implemented in `SubflowReturnDataPage` class which extends `BaseConfigPage` class. The `Subflow Return` Data is available in the Element Configuration view. `Subflow Call` element allows to accept multiple return values of different types from a subflow.



CHAPTER 4

Subflow Call

The `Subflow call` element is used to call the subflows from any call flows inside the application. The Subflow Call element is available in elements view. The Subflow Call element can be deleted, renamed, or can be used multiple times. The Call Subflow Element has the following three configurable tabs:

- **General** - This tab provides the means to associate a subflow call with Call Subflow element. It provides a drop down list of all the available subflows in a project. Only one of the subflow can be selected from the list.
- **Data** - This tab provides the information about subflow argument data and return data.
 - **Subflow Argument Data** - Subflows accepts inputs from the calling flows as arguments. Subflow Call element allows to send multiple arguments of different types to a subflow.
 - **Subflow Return Data** - Subflows returns data as processed output. Subflow Call element allows to accept multiple return values of different types from a subflow.

Note Sub flow call parameters (Argument Data and Return Data) are auto populated from the sub flow start and return elements respectively. If changes are done to sub flow start or return after the call element is created and assigned to the sub flow, call element needs to be reloaded. This can be done by clicking out and clicking back on the sub flow call element.

Cisco Unified Call Studio allows you to modify both the Subflow Argument Data and Subflow Return Data variable value directly from the Variables View while debugging a call flow. You can modify the data value directly from the value pane or right-click on the data variable and select **Change Value** to modify the value.

- [Events, on page 9](#)
- [Exit, on page 10](#)

Events

Name (Label)	Notes
Event Type	You can select Java Exception, VXML Event, or Custom Exception event handler type for this element from the drop-down list.

Exit

Name	Notes
next	The default exit state. The events that are entered for this element as added as the exit state in the call flow.



CHAPTER 5

Application_Modifier

The `Application_Modifier` action element is used to modify context variables and remove session data values at runtime in a voice application. It allows for a developer to change the application's environment anywhere in the callflow. A typical use for the `Application_Modifier` element would be for multi-language support because it can be used to change the application level `xml:lang` and encoding values. Visiting an `Application_Modifier` element instance will update the application for the current session only.

- [Settings, on page 11](#)
- [Exit States, on page 12](#)
- [Folder and Class Information, on page 12](#)
- [Events, on page 12](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
<code>maintainer</code> (Maintainer)	string	No	true	true	None	This setting specifies the e-mail address of the voice application administrator. This value is set in a VoiceXML <code><meta></code> tag.
<code>language</code> (Language)	string	No	true	true	None	This setting specifies the language identifier to specify in each VoiceXML document's <code>xml:lang</code> attribute. This value is set in the <code><vxml></code> tag.
<code>encoding</code> (Encoding)	string	No	true	true	None	This setting specifies the encoding to use when creating VoiceXML documents. This value is set in the <code><xml></code> tag.
<code>default_audio_path</code> (Default Audio Path)	string	No	true	true	None	This setting specifies a partial URI to a path containing the audio content for this voice application.

remove_session_data (Session Data to Remove)	string	No	false	true	None	This setting specifies the names of session data values to remove from this voice application.
Voice Name	String	No	true	true	None	This can take voice names provided by Google. For more information see https://cloud.google.com/text-to-speech/docs/voices

Exit States

Name	Notes
done	The application's context variables were modified and session data values were removed.

Folder and Class Information

Studio Element Folder Name	Class Name
Context	com.audium.server.action.context.ApplicationModifier

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 6

Audio

The `Audio` voice element simply outputs a VoiceXML page with the contents of a single audio group. The `Audio` element is used for greetings, error messages and any other time audio is to be played in a situation not associated with an input state.

- [Audio Groups, on page 13](#)
- [Custom VoiceXML Properties, on page 13](#)
- [Folder and Class Information, on page 14](#)
- [Events, on page 14](#)

Audio Groups

Audio Playback

Name (Label)	Max1	Req'd	Notes
<code>initial_audio_group</code> (Initial)	Yes	Yes	The audio group containing the audio to play.

Custom VoiceXML Properties

Name (Label)	Type	Description
<code>http.streaming</code>	boolean	Indicates whether media streaming is enabled. Set the value of this parameter to <code>true</code> to enable media streaming.

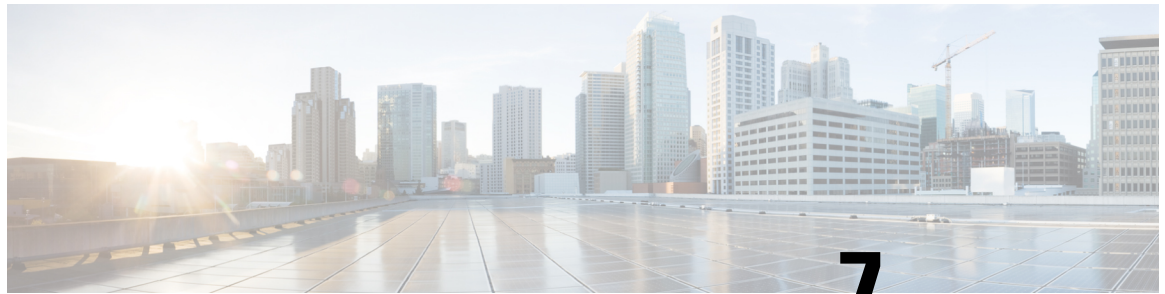
Name (Label)	Type	Description
		<p>Note</p> <ul style="list-style-type: none"> • Streaming is supported only for static URLs using u-law and A-law audio codec. • Streaming supports a maximum of 150 simultaneous callers for a single conference or a maximum of five simultaneous conferences each having a maximum of 30 simultaneous callers. • Each caller can hear live streaming for a maximum duration of 30 minutes. • DTMF recognition and buffering are not supported for streaming prompts. • Caller can barge-in the live stream using DTMF if barge-in is enabled.

Folder and Class Information

Studio Element Folder Name	Class Name
Top Level	com.audium.server.voiceElement.audio.MAudio

Events

Name (Label)	Notes
Event Handler	You can select either VXML Event or Java Exception as event handler type from the drop-down list.



CHAPTER 7

Counter

The `Counter` action element is used to keep track of a count stored as element data. The initial value of the count is defined as a configuration setting. In addition, the element may be configured to increment or decrement with a user defined step size. A typical use for the Counter element would be in a loop in the call flow that increments the count until a decision element decides that the loop must end. Revisiting a Counter element instance will automatically update the count.

- [Settings, on page 15](#)
- [Element Data, on page 16](#)
- [Exit States, on page 16](#)
- [Folder and Class Information, on page 16](#)
- [Events, on page 16](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
initial (Initial Count)	int	Yes	true	true	<i>None</i>	This setting specifies at which integer value this counter should start.
type (Type)	string enum	Yes	true	true	<i>None</i>	This setting specifies whether the counter should be incremented or decremented. Possible values are: <code>decrement</code> <code>increment</code> .
step (Step Size)	int	Yes	true	true	1	This setting specifies by how much this counter should be incremented or decremented.

Element Data

Name	Type	Notes
count	string	The current count

Exit States

Name	Notes
done	The counter was updated.

Folder and Class Information

Studio Element Folder Name	Class Name
Calculation	com.audium.server.action.counter.CounterAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 8

Callback_Add

The `Callback_Add` element is used to add a callback object to the database after all the callback information has been collected from the caller. In addition, it can be optionally configured to automatically delete old recorded files at specified intervals. These recorded files are the files produced by the `Record` element when the user records his/her name if they want a call back in the `CallbackEntry` application.

- [Settings, on page 17](#)
- [Element Data, on page 18](#)
- [Exit States, on page 18](#)
- [Folder and Class Information, on page 18](#)
- [Events, on page 19](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Callback Number	string	Yes	true	true	<i>None</i>	The phone number the callers specifies to call back.
Recorded Name File	string	Yes	true	true	<i>None</i>	The URL to the recorded file for playback when the caller is called back.
Recorded Name Path	string	No	true	true	<i>None</i>	<p>Path to the recorded file. If specified, files starting with <i>audio</i> in this folder are deleted automatically based on the file retention time and interval specified in <code>Recorded File Retention</code> and <code>Recorded File Deletion Interval</code> settings.</p> <p>Note All files created by the <code>Record</code> element start with <i>audio</i>.</p> <p>If this setting is left blank, recorded files are not deleted automatically.</p>

						The value of this setting may be either the path to a folder or a path to a file. If a path to a file is specified, then the folder in which the file resides is the folder to be managed. The path to the folder must be accessible to the VXMLServer.
Recorded File Retention	Int	No	true	true	240	Number of minutes to retain recorded files before they are eligible for automatic deletion. This setting only takes effect if <code>Recorded name Path</code> is specified.
Recorded File Deletion Interval	Int	No	true	true	30	Number of interval minutes for checking when recorded files can be deleted. This setting only takes effect if <code>Recorded name Path</code> is specified

Element Data

Name	Type	Notes
Result	string	<p>Result of request to add callback object to the database. Valid string values are <i>valid</i>, <i>no_validation</i> and <i>invalid_time</i>.</p> <ul style="list-style-type: none"> • valid – signifies that the request was successful. • no_validation – occurs when a callback object cannot be created because <code>Callback_Validate</code> element was not executed in the script. • invalid_time – means that the time selected for the scheduled callback is invalid.

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.AddCallback

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 9

Callback_Disconnect_Caller

The `Callback_Disconnect_Caller` element is responsible for disconnecting the caller's leg of the call. The IP leg of the call for Unified CVP is preserved to hold the caller's *place in line* until the callback is made back to the caller.

- [Settings, on page 21](#)
- [Element Data, on page 21](#)
- [Exit States, on page 22](#)
- [Folder and Class Information, on page 22](#)
- [Events, on page 22](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Probe Type	string enum	Yes	Yes	No	Disconnect Caller	The probe type can be one of: Disconnect Caller Intercept Caller Hangup No Intercept Caller Hangup

Element Data

Name	Type	Notes
Result	string	The call outcome from the attempt to disconnect the caller's leg.

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.DisconnectCaller

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 10

Callback_Enter_Queue

The `Callback_Enter_Queue` element is responsible for adding a new caller to queue. This element must be executed for all callers even if the caller may not be offered a callback.

- [Settings, on page 23](#)
- [Element Data, on page 23](#)
- [Exit States, on page 23](#)
- [Folder and Class Information, on page 24](#)
- [Events, on page 24](#)

Settings

None.

Element Data

Name	Type	Notes
ewt	int	The calculated estimated wait time for caller in queue.

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.EnterQueue

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JSON Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 11

Callback_Get_Status

The `Callback_Get_Status` element is responsible for retrieving all information about the callback related to the current call (if a callback exists).

- [Settings, on page 25](#)
- [Element Data, on page 25](#)
- [Exit States, on page 26](#)
- [Folder and Class Information, on page 26](#)
- [Events, on page 26](#)

Settings

None.

Element Data

Name	Type	Notes
startCallback	boolean	Specifies whether the application should call the caller, given current caller position in queue and rate of de-queue.
ewt	int	Current estimated remaining wait time in seconds for this caller before the callback should be initiated.
qpos	int	Current position in queue.
rec	string	Recording URL that was stored in the callback table. This only needs to be returned if startCallback is true.
DORateA	int	Average number of seconds that it takes for each caller in this queue to leave the queue. This includes both callers leaving queue by going to agents and callers in queue abandoning.
DORateB	int	Average number of seconds that it takes for the #1 caller in this queue to leave the queue.

RORate	int	Average number of seconds that it takes to get the caller back after starting the callback. The rate is the same for all queues. This includes dial time, ring time, and IVR time spent asking the caller if they are ready to take the callback.
cli	string	The Calling Line ID to be used for this callback
rna	int	Ring No Answer timeout for this call
dn	string	Destination number for this outbound call

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

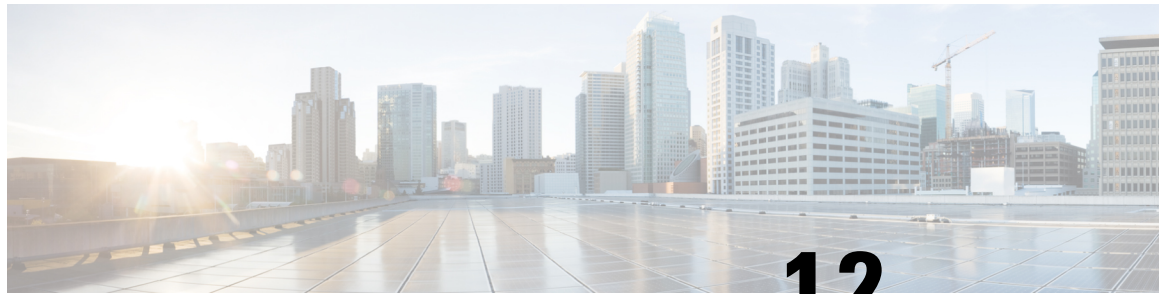
Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.GetStatus

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 12

Callback_Reconnect

The `Callback_Reconnect` element is responsible for reconnecting the caller's leg of the call.

- [Settings, on page 27](#)
- [Element Data, on page 27](#)
- [Exit States, on page 28](#)
- [Folder and Class Information, on page 28](#)
- [Events, on page 28](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Dialed Number	string	Yes	true	true	<i>None</i>	Destination for the outbound call.
Calling Line ID	string	Yes	true	true	<i>None</i>	The calling line ID to be used for the callback.
Ring No Answer Timeout	string	Yes	true	true	30	Ring No Answer timeout in seconds, The default is 30, minimum is 0 and maximum is 300 seconds.
User-to-User Information	string	No	true	true	<i>None</i>	The user-to-user information (UII) to include in the callback.

Element Data

Name	Type	Notes
------	------	-------

result	string	Contains the reconnect exit state.
--------	--------	------------------------------------

Exit States

Name	Notes
noanswer	The callback was attempted and not answered.
busy	The callback was attempted and the calling line was busy.
invalid_number	The callback number was not a valid number.
connected	The callback was attempted and connected.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.Reconnect

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 13

Callback_Set_Queue_Defaults

The `Callback_Set_Queue_Defaults` element is responsible for updating the DBServlet with the values that should be used for each queue. There is always a *default* queue type. The values are used whenever a queue type is encountered for which there are no explicitly defined values. For example, if an administrator has defined values for a *billing* and *default* queues, but the caller is queued for *mortgages*. In that case, the application uses the values from `Callback_Set_Queue_Defaults`.

Note When the DBServlet is not reachable to check the callback status for the duration of keepalive interval, the callback entry in the Reporting Server gets marked as a stale cached entry and subsequently gets cleared. As a result, a callback is not initiated.

- [Settings, on page 29](#)
- [Element Data, on page 31](#)
- [Exit States, on page 31](#)
- [Folder and Class Information, on page 31](#)
- [Events, on page 32](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Queue Name	string	Yes	true	false	<i>None</i>	The name of the queue.
Maximum Percentage	integer	No	true	false	50	Maximum percentage of callbacks that can exist in the queue. Maximum is 100, minimum is 0.
Maximum Count	integer	No	true	false	9999999	Absolute number of callbacks that can exist in a queue.
Refresh Interval	integer	No	true	false	30	Number of minutes between DBServlet refreshes of this reference data. Maximum is 1440 minutes, minimum is 1 minute.

Maximum Estimated Wait Time	integer	No	true	false	900	Callbacks are only offered for this queue when the estimated wait time (ewt) is greater than or equal this number of seconds. If 0, then callbacks are offered regardless of ewt. Maximum is 86400 seconds, minimum is 0.
Timezone	string enum	No	true	false	<i>None</i>	The timezone to apply to this queue. Valid options available from pull-down menu.
Keepalive Interval	integer	No	true	false	180	Maximum keepalive interval in seconds. Maximum is 300, minimum is 1. 'Ring No Answer Timeout' setting must be less than this value.
Dialed Number	string	No	true	false	<i>None</i>	Dialed Number to which a callback is directed for this queue.
Reconnect Time	integer	No	true	false	30	Approximate average time in seconds to reconnect caller. Take into account both ringtime and IVR time when determining this value. Maximum is 300, minimum is 1.
Service Level Agreement (SLA)	integer	No	true	false	60	Average number of seconds to wait before connecting to an agent after a caller is called back.
Calling Line ID	string	Yes	true	false	<i>None</i>	The CLI to be used on the callback.
Sample	string	No	true	false	0	Number of minutes in the interval used to calculate average time to leave queue. Maximum is 1440, minimum is 15.
Burst	string	No	true	false	10:1	X:Y, where X requests to method LeaveQueue in Y seconds. This is used to detect abnormal system failures so that the requests do not get included in the average time to leave queue calculation.
Ring No Answer Timeout	integer	No	true	false	30	The RNA timeout for the callback. Maximum is 300, minimum is 0. Must be less than the Keepalive Interval.

Sunday Time Range	string	No	true	false	00:00:00 – 23:59:59	Time range per day when callbacks can occur. Value “none” means no callbacks are allowed on that day. The default is all day if no value is specified. 00:00:00 – 23:59:59 means all day.
Monday Time Range						
Tuesday Time Range						
Wednesday Time Range						
Thursday Time Range						
Friday Time Range						
Saturday Time Range						
Max No Response Count	string	No	true	false	3,300	Max attempts to try the callback when this error occurs and the next the interval (in seconds) in which to retry the call.
Max Busy Count	string	No	true	false	4,300	
Max No Answer Count	string	No	true	false	4,300	
Max Trunks Busy Count	string	No	true	false	4,300	
Max Error Count	string	No	true	false	4,300	

Element Data

Name	Type	Notes
result	string	Contains the reconnect exit state.

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.SetQueueDefaults

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JSON Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 14

Callback_Update_Status

The `Callback_Update_Status` element is responsible for updating the database after a callback disconnect or reconnect.

- [Settings, on page 33](#)
- [Element Data, on page 34](#)
- [Exit States, on page 34](#)
- [Folder and Class Information, on page 34](#)
- [Events, on page 34](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
status	enum string	Yes	true	true	<i>None</i>	Callback status can be one of the following: <ul style="list-style-type: none">• PENDING• INPROGRESS• COMPLETED• ADD TO QUEUE• DROP FROM QUEUE
reason	enum string	*	true	true	None	Required if status is COMPLETED, one of the following: <ul style="list-style-type: none">• error• busy• noanswer• noresponse

						<ul style="list-style-type: none"> • invalid_number • connected • trunkbusy • caller_cancelled
--	--	--	--	--	--	--

Element Data

Name	Type	Notes
result	string	<p>Tells the application whether to cancel the existing callback or to retry, can be one of the following:</p> <ul style="list-style-type: none"> • cancel • retry • done

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.UpdateStatus

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 15

Callback_Validate

The `Callback_Validate` element is responsible for verifying whether or not a callback can be offered to the caller during this call. Depending on the outcome of the validation, the `Validate` element exits with one of four states.

- [Settings, on page 35](#)
- [Element Data, on page 35](#)
- [Exit States, on page 35](#)
- [Folder and Class Information, on page 36](#)
- [Events, on page 36](#)

Settings

None.

Element Data

Name	Type	Notes
result	string	Contains the exit state result.
ewt	int	EWT value passed from Unified ICM.
gw	string	Gateway identifier.
loc	string	Gateway location information.
capacity	int	Gateway capacity.

Exit States

Name	Notes
------	-------

preemptive	This callback is valid.
none	The callback is not allowed.
refresh	The validation could not be performed because the DBServlet needs a reference data refresh. The application must call SetQueueDefaults before validation can occur.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.Validate

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 16

Callback_Wait

The `Callback_Wait` element is responsible for *sleeping* the application for X seconds. The application hands control back to `cvp_ccb_vxml.tcl` with the parameter `wait=X`.

- [Settings, on page 37](#)
- [Exit States, on page 37](#)
- [Folder and Class Information, on page 37](#)
- [Events, on page 38](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Wait Time	integer	Yes	true	false	None	Amount of time in seconds to wait. Maximum is 60, minimum is 0.

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Callback	com.cisco.cvp.vxml.custelem.callback.Wait

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 17

Currency

The `Currency` voice element captures from the caller a currency amount in dollars and cents. The currency amount can be entered using the keypad or spoken. The captured value will be stored in element data as a decimal value (without the \$ character).

There are several different formats for speaking a currency amount or entering it through the keypad. Voice browsers may use different grammars and therefore accept different utterances. However, the spoken formats listed below should result in the same behavior for all supported browsers. The tables below list each input and the value that is stored in the element variable as a result. If some data is left out, the system assumes a default value for the missing information.

Note You cannot use the * character to represent a decimal point in the `Currency` voice element, if you have defined it as a **termchar** in the **Root Doc Settings**.

Utterance	Stored Value	Example	Description
[dollar] "dollar(s)" ("and") [cent] "cent(s)"	D.CC	"thirteen dollars and fifty cents" = 13.50	Dollars are whole numbers ≥ 0 . Cents are from 00 to 99. The word <i>and</i> is optional.
[dollar] "dollar(s)" "[cent]"	D.CC	"thirteen dollars five" = 13.05	Dollars are whole numbers ≥ 0 . Cents are from 00 to 99.
[dollar] "dollar(s)"	D.00	"three hundred fifty" = 350.00	A plain whole number is interpreted as dollars with no cents.
[cent] "cent(s)"	0.CC	"three cents" = 0.03	To specify cents only, the word <i>cents</i> to be uttered. Cents are from 00 to 99.

DTMF Entry	Stored Value	Example	Description
[D]*[CC]	D.CC	3*99 = 3.99	The decimal is represented by the * button.

There are other formats that are possible, particularly when entering via DTMF and inputting incomplete amounts. These inputs may yield differing results on various voice browsers. The returned variable will always be a decimal value with the appropriate number of padded zeros, if applicable.

- [Settings, on page 40](#)
- [Element Data, on page 41](#)
- [Exit States, on page 42](#)
- [Audio Groups, on page 42](#)
- [Folder and Class Information, on page 43](#)
- [Events, on page 43](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: <code>voice</code> <code>dtmf</code> <code>both</code> .
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time length allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s
max_noinput_count (Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during currency input capture. 0 = infinite noinputs allowed.
max_nomatch_count (Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during currency input capture. 0 = infinite nomatches allowed.
currency_confidence_level (Currency Confidence Level)	decimal (0.0 to 1.0)	Yes	true	true	0.40	The confidence level threshold to use during currency capture.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	Whether or not to temporarily disable all hotlink grammars (global or local) and universal grammars. If set to true, only the currency grammars will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes <code>utterance</code> , <code>interpretation</code> , <code>value</code> , <code>nbestUtteranceX</code> and <code>nbestInterpretationX</code> are masked in VXML server logs. The format used to render secure element attributes is to add a <code>_secureLogging</code>

						suffix. For example nbestUtterance1_secureLogging,****.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Note Refer to the Element Data table below for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
Value	string	The currency amount captured. This will always be a decimal number with the appropriate number of padded zeros (up to 2).
value_confidence	float	This is the confidence value of the captured utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2	string	This set of element data stores the input modes of captured n-best utterances.

...		
nbestInputmodeX		

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
done	The currency capture was completed.

Audio Groups

Currency Capture

Name (Label)	Req'd	Max 1	Notes
initial_audio_group (Initial)	Yes	Yes	Played when the voice element first begins.
nomatch_audio_group (NoMatch)	No	No	Played when a nomatch event occurs.
noinput_audio_group (NoInput)	No	No	Played when a noinput event occurs.
help_audio_group (Help)	No	No	Played when the caller asked for help. If not specified, by default help is treated as a nomatch.

End

Name (Label)	Req'd	Max 1	Notes
done_audio_group (Done)	No	Yes	Played when the currency capture is completed and the voice element exits with the <i>done</i> exit state.

Folder and Class Information

Studio Element Folder Name	Class Name
Commerce	com.audium.server.voiceElement.currency.MBasicCurrency

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 18

Currency_with_Confirm

The `Currency_With_Confirm` voice element captures from the caller a currency amount in dollars and cents, and presents a confirmation menu allowing the caller to either accept their entry or re-enter the currency value. The currency amount can be entered using the keypad or spoken. The captured value will be stored in element data as a decimal value (without the \$ character).

There are several different formats for speaking a currency amount or entering it through the keypad. Voice browsers may use different grammars and therefore accept different utterances. However, the spoken formats listed below should result in the same behavior for all supported browsers. The tables below list each input and the value that is stored in element data as a result. If some data is left out, the system assumes a default value for the missing information.

Utterance	Stored Value	Example	Description
[dollar] "dollar(s)" ("and") [cent] "cent(s)"	D.CC	"thirteen dollars and fifty cents" = 13.50	Dollars are whole numbers ≥ 0 . Cents are from 00 to 99. The word <i>and</i> is optional.
[dollar] "dollar(s)" "[cent]"	D.CC	"thirteen dollars five" = 13.05	Dollars are whole numbers ≥ 0 . Cents are from 00 to 99.
[dollar] "dollar(s)"	D.00	"three hundred fifty" = 350.00	A plain whole number is interpreted as dollars with no cents.
[cent] "cent(s)"	0.CC	"three cents" = 0.03	To specify cents only, the word <i>cents</i> to be uttered. Cents are from 00 to 99.

DTMF Entry	Stored Value	Example	Description
[D]*[CC]	D.CC	3*99 = 3.99	The decimal is represented by the * button.

There are other formats that are possible, particularly when entering via DTMF and inputting incomplete amounts. These inputs may yield differing results on various voice browsers. The returned variable will always be a decimal value with the appropriate number of padded zeros if applicable.

- [Settings, on page 46](#)

- [Element Data, on page 47](#)
- [Exit States, on page 48](#)
- [Audio Groups, on page 48](#)
- [Folder and Class Information, on page 49](#)
- [Events, on page 50](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: <code>voice</code> <code>dtmf</code> <code>both</code> .
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time length allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
currency_max_noinput_count (Currency Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during currency input capture. 0 = infinite noinputs allowed.
currency_max_nomatch_count (Currency Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during currency input capture. 0 = infinite nomatches allowed.
confirm_max_noinput_count (Confirm Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during currency input confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during currency input confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int ≥ 0	Yes	true	true	3	The maximum number of times a caller is allowed to disconfirm a captured input. 0 = infinite disconfirmations allowed.
currency_confidence_level (Currency Confidence Level)	decimal (0.0 to 1.0)	Yes	true	true	0.40	The confidence level threshold to use during currency capture.

confirm_confidence_level (Confirm Confidence Level)	Decimal (0.0 to 1.0)	Yes	true	true	0.50	The confidence level threshold to use during confirmation.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Currency_With_Confirm element (the currency and boolean grammars) will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example <code>nbestUtterance1_secureLogging,****.</code>
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Element Data

Name	Type	Notes
Value	string	The currency amount captured. This will always be a decimal number with the appropriate number of padded zeros (up to 2).
value_confidence	float	This is the confidence value of the captured currency utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
confirm_confidence	float	This is the confidence value of the captured confirm utterance.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.
nbestInterpretation1	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX

nbestInterpretation2 ... nbestInterpretationX		values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
max_disconfirmed	The maximum number of disconfirmations has occurred. If the max disconfirmed count is set to 0, this exit state will never occur.
done	The currency capture was confirmed.

Audio Groups

Currency Capture

Name (Label)	Req'd	Max 1	Notes
currency_initial_audio_group (Currency Initial)	Yes	Yes	Played when the voice element first begins.
currency_nomatch_audio_group (Currency NoMatch)	No	No	Played when a nomatch event occurs during a currency capture.

currency_noinput_audio_group (Currency NoInput)	No	No	Played when a noinput event occurs during a currency capture.
currency_help_audio_group (Currency Help)	No	No	Played when the caller asked for help during a currency capture. If not specified, by default help is treated as a nomatch.

Currency Confirm

Name (Label)	Req'd	Max 1	Notes
confirm_initial_audio_group (Confirm Initial)	Yes	Yes	Played when confirmation first begins.
confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during confirmation. The nomatch event count corresponds to the audio group count.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during confirmation. The noinput event count corresponds to the audio group count.
confirm_help_audio_group (Confirm Help)	No	No	Played when a help event occurs during confirmation. The help event count corresponds to the audio group count. If not specified, by default help throws a nomatch.
disconfirmed_audio_group (Disconfirmed)	No	No	Played after the caller disconfirms a captured currency entry. Upon reaching the <code>max_disconfirmed_count</code> , the prompt should be about exiting with the <code>max_disconfirmed</code> exit state.

End

Name (Label)	Req'd	Max 1	Notes
yes_audio_group (Yes)	No	Yes	Played after the caller chooses the <code>yes</code> option. If not specified, no audio will be played when this option is chosen.

Folder and Class Information

Studio Element Folder Name	Class Name
Commerce	com.audium.server.voiceElement.currency.MBasicCurrencyWithConfirm

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 19

Customer_Lookup

Use the `Customer_Lookup` element to search for the existing customers present in Context Service based on the query parameters. You can combine multiple query parameters by using a logical AND or OR operator.

- [Settings, on page 51](#)
- [Element Data, on page 51](#)
- [Exit States, on page 52](#)
- [Folder and Class Information, on page 53](#)
- [Events, on page 53](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Query Parameters	string	Yes	true	true	None	The actual search parameters based on which a search is performed. For example, if the customer listings in Context Service contain data elements such as email address and phone number, you can use the name-value pair to search for data in Context Service. Note You can create up to ten query parameters.
Operator	string	No	true	false	AND	The logical AND or OR operator used to combine multiple query parameters.

Element Data

Name	Type	Notes
------	------	-------

context_city	string	Contains the Context_City data element associated with the Customer.
context_country	string	Contains the Context_Country data element associated with the Customer.
context_customer_external_id	string	Contains the Context_Customer_External_ID data element associated with the Customer.
context_first_name	string	Contains the Context_First_Name data element associated with the Customer.
context_home_email	string	Contains the Context_Home_Email data element associated with the Customer.
context_home_phone	string	Contains the Context_Home_Phone data element associated with the Customer.
context_last_name	string	Contains the Context_Last_Name data element associated with the Customer.
context_mobile_phone	string	Contains the Context_Mobile_Phone data element associated with the Customer.
context_state	string	Contains the Context_State data element associated with the Customer.
context_street_address_1	string	Contains the Context_Street_Address_1 data element associated with the Customer.
context_street_address_2	string	Contains the Context_Street_Address_2 data element associated with the Customer.
context_work_email	string	Contains the Context_Work_Email data element associated with the Customer.
context_zip	string	Contains the Context_Zip data element associated with the Customer.
customer_id	string	Contains the Customer ID of the last updated Customer which matches the search criteria.
customer_count	string	Contains the number of customers that match your search criteria.
search_result_as_json	string	Contains data of all customers in the JSON format.

Exit States

Name	Notes
done	The element execution is complete and the search operation is successful.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Context Service	com.audium.server.action.contextservice.CustomerLookUpAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 20

CVP Subdialog Return

For a Cisco Unified CVP Voice application invoked as a subdialog, the `CVP Subdialog Return` element must be used to return data back to the calling application. The element should be used in place of Hang Up elements throughout the call flow. Like a Hang Up element, the element has no exit states.

Note There is one exception to the above description. If the voice application will only ever be called by a Subdialog Invoke element (that is, never by Unified ICM), then the Subdialog Start and Subdialog Return elements may be used instead. Refer to

The settings for this element are used to define what data to pass back to the calling application. The `Caller Input` setting must be assigned a value in order for the application to validate, since it is required to have a value. Each element setting corresponds to an ICM ECC external variable name, and therefore the configuration values must conform to requirements associated with ICM ECC variables. Refer to the Unified CVP documentation for further details.

The CVP Subdialog Return element can be used to enable multiple types of transfer in call failure conditions. In case of a Hook Flash (HF) or Two B-Channel Transfer (TBCT) transfer, for example, `Caller Input` should be set to the transfer destination number prefixed with *HF* or *TBCT* (as in HF800xxxxxxx or TBCT800xxxxxxx). An HF or TBCT transfer will be invoked after the `Caller Input` was passed back from the CVP Subdialog Return element.

- [Settings, on page 55](#)
- [Exit States, on page 56](#)
- [Folder and Class Information, on page 56](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
caller_input (Caller Input)	string	Yes	true	true	None	Required return argument that holds a value to be returned to the calling application.
FromExtVXML0 (External VXML 0)	string	No	true	true	None	Optional return argument that is returned to the calling application.

FromExtVXML1 (External VXML 1)	string	No	true	true	<i>None</i>	Optional return argument that is returned to the calling application.
FromExtVXML2 (External VXML 2)	string	No	true	true	<i>None</i>	Optional return argument that is returned to the calling application.
FromExtVXML3 (External VXML 3)	string	No	true	true	<i>None</i>	Optional return argument that is returned to the calling application.

**Note**

The following characters are not allowed in the return arguments:

< > " ' &

Exit States

Name	Notes
done	The element execution is complete.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco	com.audium.server.voiceElement.internal.CiscoSubdialogReturnElement



CHAPTER 21

CVP Subdialog Start

For a Cisco CVP voice application invoked as a subdialog, the `CVP Subdialog Start` element must be used, which receives data from a calling application and creates corresponding element data or session data. The element should be placed at the entrance point of the application, immediately after the Start of Call element.

Data can be passed to the VoiceXML application either as HTTP parameters or VoiceXML parameters (using the `<param>` tag). In the first case (that is, as HTTP parameters), Cisco Unified CVP VoiceXML Server will automatically create session data using the name of the data received. In the second case (that is, as VoiceXML parameters), the `CVP Subdialog_Start` element must be configured appropriately in order for the data to be available as element or session data for the duration of the call session. For each data passed as a VoiceXML parameter, the *Parameter* setting must be configured with the same exact name as the data. The *Store As* setting can be configured to store the passed data either as session or element data. The *Enable Digits Bypass* setting is used to activate a VoiceXML workaround to ensure expected functionality for a particular TDM or analog phone. When this setting is set to *true*, a new setting named *Audio Filler URI* will be enabled in VoiceXML Studio and can be configured to set a reference to a silence wave file to be played in the digits field. For IP phones the *Enable Digits Bypass* setting should be set to *false*.

- [Settings, on page 57](#)
- [Exit States, on page 58](#)
- [Folder and Class Information, on page 58](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Parameter (Parameter)	string	No	false	true	<i>None</i>	Holds the name of a parameter passed as input to the subdialog. It must match the exact value specified in the VoiceXML page that calls the subdialog. This is a repeatable setting, so multiple values can be specified.
Where (Store As)	string	No	true	false	Session Data	Determines whether the parameter passed to the subdialog will be stored as element data or session data. By

						making it element data, the information will “belong” only to this element, and so there is no chance that these variables will overwrite any other variables.
enable_digits_bypass (Enable Digits Bypass)	boolean	Yes	true	true	false	Determines whether the digits field is used at the beginning of an application. By default this is disabled.
audio_filler_uri (Audio Filler URI)	string	No	true	true	<i>None</i>	Configures a URI for a silence wave file to be played in the above digits field.

Exit States

Name	Notes
done	The element execution is complete.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco	com.audium.server.voiceElement.internal.CiscoSubdialogStartElement



CHAPTER 22

Database

The `database` element provides the ability to execute an SQL command on external databases within a voice application call flow. The element requires JNDI to be configured in the Java application server to handle database connections. Only a single SQL statement can be executed per element. There are four types of commands that can be made:

- **Single** – This is used to run a SQL query that returns only a single row. Element data will be created with the variable names being the names of the columns returned and the value of that column as the element data value (as a string). If no row is returned, no element data will be set.
- **Multiple** – This is used to run a SQL query that returns multiple rows. A Unified CVP-defined Java data structure, the Java class `ResultSetList`, stores the full result and is placed in session data. If no rows are returned, the `ResultSetList` object in session data will be empty. For detail about the `ResultSetList` data structure, refer to the javadocs for this class.
- **Inserts** – This is used to run a SQL INSERT command that inserts information into the database.
- **Updates** – This is used to run a SQL UPDATE command that updates information in the database.

The developer can utilize substitution to create dynamic queries. The Database element is ideal for performing simple queries and updates. It may not be sufficient for performing complex database interactions such as multiple dependent queries or stored procedure calls. One would use a custom configurable or generic action element for these tasks. Also note that in order to avoid performance issues creating database connections, you must implement database pooling on the application server.

- [Settings, on page 60](#)
- [Element Data, on page 60](#)
- [Session Data, on page 61](#)
- [Exit States, on page 61](#)
- [Folder and Class Information, on page 61](#)
- [Events, on page 61](#)
- [Create JNDI Database Connection in Tomcat for Use in VXML Applications, on page 62](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
type (Type)	string enum	Yes	true	true	single	The type of query: single, multiple, insert or update. Note The "xml_resultset" element data is not created when insert or update is selected.
jndiName (JNDI Name)	string	Yes	true	true	None	This JNDI name for the SQL datasource of the database.
key (Session Data Key)	string	Yes	true	true	None	For queries of type multiple, the name of the session variable which the results of the query will be stored.
query (SQL Query)	string	Yes	true	true	None	The SQL query to be executed.
enableXmlResultSet (Result-Set XML)	Boolean	Yes	true	false	true	If the Result-Set XML option is set to False, the "xml_resultset" element data is not created when the XML Data conversion functionality is disabled.

Element Data

In the substitution tag, the two element data `num_rows_processed` and `xml_resultset` are available by default when a database element is selected. The `{Data.Element.DBElement1.num_rows_processed}` and `{Data.Element.Database_01.xml_resultset}` are the two tags that can be added for these element data respectively. The Database element `num_rows_processed` carries the number of rows fetched when query is selected from the database and the number of rows updated when any update, delete or insert operation is made in the database. The `xml_resultset` carries the database result in the XML form for a single query or multiple select query. The `num_rows_processed` can be used for any data *type* settings. The `xml_resultset` can only be used for Insert and Update *type* settings. However, when the *type* setting is set to *single* for an Element data, the names of the return columns are created containing the respective return values.

For example, if a query returned the following information:

```
foo bar
123 456
```


The following element data will be created: *foo* with the value *123* and *bar* with the value *456*.

Session Data

Session data is created *only* when the `type` setting is set to *multiple*. In all other cases, no session data is created.

Name	Type	Notes
[value of setting "key"]	ResultSetList	The Java data structure that stores the returned values from a multiple type query. The name of the session data variable is specified by the developer in the <code>key</code> setting.

Exit States

Name	Notes
done	The database query was successfully completed.

Folder and Class Information

Studio Element Folder Name	Class Name
Integration	com.audium.server.action.database.DatabaseAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.

Create JNDI Database Connection in Tomcat for Use in VXML Applications

Summary

Steps

This section explains how to create a new JNDI database connection in Tomcat. These instructions are useful when you would like to use the built-in Studio Database element, or create some custom code that accesses database functionality through JNDI.

1. In order enable database access on your application server, a compatible JDBC driver must be installed. These drivers, typically packaged as JAR files, should be placed in a directory accessible to the application server classpath (on Tomcat, for example, place in `%CVP_HOME%\VXMLServer\Tomcat\lib`).

Note The database must exist for this connection to work. CVP VXML Server will not create the database for you.

2. Add a Tomcat Context for the database connection so that the CVP VXML Server knows how to communicate with your database. For more information, see <https://tomcat.apache.org/tomcat-8.0-doc/jndi-datasource-examples-howto.html>.
3. In Audium Builder for CVP Studio, edit the configuration of the Database element in question. Enter the string you entered *below* in `<LABEL_YOU_CHOOSE>` from the Tomcat Context into the JNDI Name property of the Settings tab of your Database element.

Note You should not include the `jdbc/` portion here.

Here is an example that uses MySQL (edit `context.xml` from `AUDIUM_HOME\Tomcat\conf` folder):

```
<Context>
<Resource name="jdbc/<LABEL_YOU_CHOOSE>"
auth="Container"
type="javax.sql.DataSource"
username="USER_NAME"
password="USER_PW"
driverClassName="com.mysql.jdbc.Driver"
url="jdbc:mysql://HOSTNAME_OR_IP:PORT/DB_NAME" />
</Context>
```

The default port number for MySQL is 3306. An example url for the above Context would be `jdbc:mysql://localhost:3306/DB_name`

Note Alternately, the `<Resource>` can be configured in the `server.xml` file under `<GlobalNamingResources>`, and a `<ResourceLink>` created in `context.xml` under `<Context>`

4. Under heavy load conditions, enable Database Connection Pooling.

A database connection pool creates and manages a pool of connections to a database. Recycling and reusing already existing connections to a database is more efficient than opening a new connection. For further information on Tomcat Database Pooling, see <https://tomcat.apache.org/tomcat-8.0-doc/jndi-datasource-examples-howto.html>.

Note Tomcat 8.0 has two connection pool libraries: commons-dbcp and tomcat-jdbc-pool. Due to a known issue with tomcat-jdbc-pool connection pool library, if the connection between the CVP VXML server and the remote SQL server goes down, the connections are not re-established automatically. The connections can be re-established only after the VXMLServer tomcat service is restarted.

The commons-dbcp connection pool library does not have this problem. The commons-dbcp library is used by default, and the tomcat-jdbc-pool is only used if the tomcat context.xml file contains the following line:

```
factory="org.apache.tomcat.jdbc.pool.DataSourceFactory"
```

Due to this issue, Cisco does not recommended using the tomcat-jdbc-pool library.



CHAPTER 23

Date

The `Date` voice element captures a date input from the caller. The date can be entered using DTMF input (in the `YYYYMMDD` format). It can also be spoken in natural language including a month, day and year. The captured value will be stored in element data as a fixed-length date string in the `YYYYMMDD` format. If the year is not specified in the input, `YYYY` is stored as “????”. And if the month or the day is not specified, `MM` and `DD` will be stored as “??”.

- [Settings, on page 65](#)
- [Element Data, on page 66](#)
- [Exit States, on page 67](#)
- [Audio Groups, on page 67](#)
- [Folder and Class Information, on page 68](#)
- [Events, on page 68](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: <code>voice</code> <code>dtmf</code> <code>both</code> .
Noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time length allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
collect_max_noinput_count (Date Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events. 0 = infinite noinputs allowed.

collect_max_nomatch_count (Date Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed. 0 = infinite nomatches allowed.
collect_confidence_level (Date Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use during date capture.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Date element will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example <code>nbestUtterance1_secureLogging,*****.</code>
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the following Element Data table for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
value	string	The date stored in the YYYYMMDD format.
value_confidence	float	This is the confidence value of the captured date utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.

nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the max nomatch count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the max noinput count is 0, this exit state will never occur.
done	The data capture was completed.

Audio Groups

Date Capture

Name (Label)	Req'd	Max 1	Notes
collect_initial_audio_group (Date Initial)	Yes	Yes	Played when the voice element first begins.
collect_noinput_audio_group (Date NoInput)	No	No	Played when a noinput event occurs during date input. The noinput event count corresponds to the audio group count.

collect_nomatch_audio_group (Date NoMatch)	No	No	Played when a nomatch event occurs during date input. The nomatch event count corresponds to the audio group count.
collect_help_audio_group (Date Help)	No	No	Played when a help event occurs during date input. The help event count corresponds to the audio group count. If not specified, a help event is treated as nomatch.

End

Name (Label)	Req'd	Max1	Notes
done_audio_group (Done)	No	Yes	Played after the date capture is completed. If not specified, no audio will be played.

Folder and Class Information

Studio Element Folder Name	Class Name
Date & Time	com.audium.server.voiceElement.date.MBasicDate

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 24

Date_with_Confirm

The `Date_With_Confirm` voice element captures a date input from the caller, and presents a confirmation menu allowing the caller to either accept their entry or re-enter the date. The date can be entered using DTMF input (in the YYYYMMDD format). It can also be spoken in natural language including a month, day and year. The captured value will be stored in element data as a fixed-length date string in the YYYYMMDD format. If the year is not specified in the input, YYYY is stored as “????”. If the month or the day is not specified, MM and DD will be stored as “??”.

- [Settings, on page 69](#)
- [Element Data, on page 70](#)
- [Exit States, on page 71](#)
- [Audio Groups, on page 72](#)
- [Folder and Class Information, on page 73](#)
- [Events, on page 73](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time length allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
collect_max_noinput_count (Date Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during date input capture. 0 = infinite noinputs allowed.

collect_max_nomatch_count (Date Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during date input capture. 0 = infinite nomatches allowed.
confirm_max_noinput_count (Confirm Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during date input confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during date input confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int ≥ 0	Yes	true	false	3	The maximum number of times a caller is allowed to disconfirm a captured input. 0 = infinite disconfirmations allowed.
collect_confidence_level (Date Confidence Level)	decimal (0.0 – 1.0)	Yes	true	false	0.40	The confidence level threshold to use during date capture.
confirm_confidence_level (Confirm Confidence Level)	decimal (0.0 – 1.0)	Yes	true	false	0.50	The confidence level threshold to use during confirmation.
modal (Disable Hotlinks)	boolean	Yes	true	false	false	If set to true, only the grammars of the current Date_With_Confirm element (the built-in date and boolean grammars) will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	false	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example <code>nbestUtterance1_secureLogging,*****.</code>
maxnbest (Maxnbest)	int ≥ 1	Yes	true	false	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the following Element Data table for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
------	------	-------

value	string	The date stored in the YYYYMMDD format.
value_confidence	float	This is the confidence value of the captured date utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
confirm_confidence	float	This is the confidence value of the captured confirm utterance.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the max nomatch count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the max noinput count is 0, this exit state will never occur.

max_disconfirmed	The maximum number of disconfirmations occurred. If the <code>max_disconfirmed_count</code> is set to 0, this exit state will never occur.
done	The date captured was confirmed.

Audio Groups

Date Capture

Name (Label)	Req'd	Max 1	Notes
collect_initial_audio_group (Date Initial)	Yes	Yes	Played when the voice element first begins.
collect_noinput_audio_group (Date NoInput)	No	No	Played when a noinput event occurs during date input. The noinput event count corresponds to the audio group count.
collect_nomatch_audio_group (Date NoMatch)	No	No	Played when a nomatch event occurs during date input. The nomatch event count corresponds to the audio group count.
collect_help_audio_group (Date Help)	No	No	Played when a help event occurs during date input. The help event count corresponds to the audio group count. If not specified, a help event is treated as nomatch.

Date Confirm

Name (Label)	Req'd	Max 1	Notes
confirm_initial_audio_group (Confirm Initial)	Yes	Yes	Played when the captured date is confirmed.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during date confirmation. The noinput event count corresponds to the audio group count.
confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during date confirmation. The nomatch event count corresponds to the audio group count.
confirm_help_audio_group (Confirm Help)	No	No	Played when a help event occurs during date confirmation. The help event count corresponds to the audio group count. If not specified, by default help is treated as nomatch.

disconfirmed_audio_group (Disconfirmed)	No	No	Played after the caller disconfirms a date entry.
--	----	----	---

End

Name (Label)	Req'd	Max 1	Notes
yes_audio_group (Yes)	No	Yes	Played after the caller chooses the <i>yes</i> option. If not specified, no audio will be played when this option is chosen.

Folder and Class Information

Studio Element Folder Name	Class Name
Date & Time	com.audium.server.voiceElement.date.MBasicDateWithConfirm

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 25

Digits

The `Digits` voice element captures a string of numerical digits. It may be used to collect small or large strings of digits. The digit string can be spoken or entered using the keypad. The captured value will be stored in element data as a string. The string cannot contain any non-numerical characters. Using speech input, the number is spoken one digit at a time (that is, 49678 is spoken *four nine six seven eight*). DTMF input can be terminated by a # keypress if desired (if not used, the entry is considered terminated when the input timeout has been reached).

With the `Digits` voice element, the application designer has the ability to set length restrictions on the digit string. A minimum and maximum length can be given to narrow the criteria. If a string of a specific length is required, the minimum and maximum lengths should be set to the same value. If fewer digits are entered, a nomatch event will be thrown. A string of digits with length greater than the maximum length cannot be entered.

- [Settings, on page 75](#)
- [Element Data, on page 76](#)
- [Exit States, on page 77](#)
- [Audio Groups, on page 78](#)
- [Folder and Class Information, on page 78](#)
- [Events, on page 78](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
inputmode (Input Mode)	String enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: <code>voice</code> <code>dtmf</code> <code>both</code> .
noinput_timeout (Noinput Timeout)	String	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s

						(for seconds) or 3000ms (for milliseconds). Default = 5s.
max_noinput_count (Digits Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during digits input capture. 0 = infinite noinputs allowed.
max_nomatch_count (Digits Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during digits input capture. 0 = infinite nomatches allowed.
digits_confidence_level (Digits Confidence Level)	Decimal (0.0 to 1.0)	Yes	true	true	0.40	The confidence level threshold to use during digits capture.
min_digit (Min Digits)	int > 0	Yes	true	true	None	Minimum number of digits allowed.
max_digit (Max Digits)	int ≥ 0	Yes	true	true	None	Maximum number of digits allowed.
modal (Disable Hotlinks)	Boolean	Yes	true	true	false	If set to true, only the grammars of the current Digits element will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	Boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example nbestUtterance1_secureLogging,*****.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the following Element Data table for information about nbestUtteranceX and nbestInterpretationX

Element Data

Name	Type	Notes
Value	string	The digit string value captured.

value_confidence	float	This is the confidence value of the captured utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
done	The digit string capture was completed.

Audio Groups

Digits Capture

Name (Label)	Req'd	Max1	Notes
digits_initial_audio_group (Digits Initial)	Yes	Yes	Played when the voice element first begins.
digits_nomatch_audio_group (Digits NoMatch)	No	No	Played when a nomatch event occurs.
digits_noinput_audio_group (Digits NoInput)	No	No	Played when a noinput event occurs.
digits_help_audio_group (Digits Help)	No	No	Played when the caller asked for help. If not specified, help is treated as a nomatch by default.

End

Name (Label)	Req'd	Max1	Notes
done_audio_group (Done)	No	Yes	Played when the digits capture is completed and the voice element exits with the done exit state.

Folder and Class Information

Studio Element Folder Name	Class Name
Number Capture	com.audium.server.voiceElement.digit.MBasicDigit

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 26

Digits_with_Confirm

The `Digits_With_Confirm` voice element captures a string of numerical digits, and presents a confirmation menu allowing the caller to either accept their entry or re-enter the digits. It may be used to collect small or large strings of digits. The digit string can be spoken or entered using the keypad. The captured value will be stored in element data as a string. The string cannot contain non-numerical characters. Using speech input, the number is spoken one digit at a time (i.e. 49678 is spoken "four nine six seven eight"). DTMF input can be terminated by a # keypress if desired (otherwise, the entry is considered terminated when the input timeout is reached).

With the `Digits_With_Confirm` voice element, the application designer has the ability to set length restrictions on the digit string. A minimum and maximum length can be given to narrow the criteria. If a string of a specific length is required, the minimum and maximum lengths should be set to the same value. If fewer digits are entered, a nomatch event will be thrown. A string of digits with length greater than the maximum length cannot be entered.

- [Settings, on page 79](#)
- [Element Data, on page 81](#)
- [Exit States, on page 82](#)
- [Audio Groups, on page 82](#)
- [Folder and Class Information, on page 83](#)
- [Events, on page 83](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input (during digits capture and confirmation). Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time length allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both

						a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
digits_max_noinput_count (Digits Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during digits input capture. 0 = infinite noinputs allowed.
digits_max_nomatch_count (Digits Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during digits input capture. 0 = infinite nomatches allowed.
confirm_max_noinput_count (Confirm Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during digits input confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during digits input confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int ≥ 0	Yes	true	true	3	The maximum number of times a caller is allowed to disconfirm a captured digits input. 0 = infinite disconfirmations allowed.
digits_confidence_level (Digits Confidence Level)	decimal (0.0 to 1.0)	Yes	true	true	0.40	The confidence level threshold to use during digits capture.
confirm_confidence_level (Confirm Confidence Level)	decimal (0.0 to 1.0)	Yes	true	true	0.50	The confidence level threshold to use during confirmation.
min_digit (Min Digits)	int > 0	Yes	true	true	None	Minimum number of digits allowed.
max_digit (Max Digits)	int > 0	Yes	true	true	None	Maximum number of digits allowed.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Digits_With_Confirm element (the builtin digits and boolean grammars) will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element

						attributes is to add a <i>_secureLogging</i> suffix. For example <code>nbestUtterance1_secureLogging, ****</code> .
maxnbest (Maxnbest)	$\text{int} \geq 1$	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the following Element Data table for information about `nbestUtteranceX` and `nbestInterpretationX`

Element Data

Name	Type	Notes
Value	string	The digit string captured.
value_confidence	float	This is the confidence value of the captured digit string utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
confirm_confidence	float	This is the confidence value of the captured confirm utterance.
nbestLength	$\text{int} \geq 1$	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of <code>nbestUtteranceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestUtterance1</code> holds the utterance of the top hypothesis in the n-best list and <code>nbestUtteranceX</code> holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of <code>nbestInterpretationX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestInterpretation1</code> holds the interpretation of the top hypothesis in the n-best list and <code>nbestInterpretationX</code> holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of <code>nbestConfidenceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestConfidence1</code> holds the confidence score of the top hypothesis in the n-best list and <code>nbestConfidenceX</code> holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2	string	This set of element data stores the input modes of captured n-best utterances.

...		
nbestInputmodeX		

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
max_disconfirmed	The maximum number of disconfirmations has occurred. If the max disconfirmed count is set to 0, this exit state will never occur.
done	The digit string captured was confirmed.

Audio Groups

Digits Capture

Name (Label)	Req'd	Max1	Notes
digits_initial_audio_group (Digits Initial)	Yes	Yes	Played when the voice element first begins.
digits_nomatch_audio_group (Digits NoMatch)	No	No	Played when a nomatch event occurs during digits capture.
digits_noinput_audio_group (Digits NoInput)	No	No	Played when a noinput event occurs during digits capture.
digits_help_audio_group (Digits Help)	No	No	Played when the caller asks for help during digits capture. If not specified, by default help is treated as a nomatch.

Digits Confirm

Name (Label)	Req'd	Max1	Notes
confirm_initial_audio_group (Confirm Initial)	Yes	Yes	Played when confirmation first begins.

confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during confirmation. The nomatch event count corresponds to the audio group count.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during confirmation. The noinput event count corresponds to the audio group count.
confirm_help_audio_group (Confirm Help)	No	No	Played when a help event occurs during confirmation. The help event count corresponds to the audio group count. If not specified, by default help throws a nomatch.
disconfirmed_audio_group (Disconfirmed)	No	No	Played after the caller disconfirms a captured digits entry. Upon reaching the max_disconfirmed_count, the prompt should be about exiting with the max_disconfirmed exit state.

End

Name (Label)	Req'd	Max 1	Notes
yes_audio_group (Yes)	No	Yes	Played after the caller chooses the <code>yes</code> option. If not specified, no audio will be played when this option is chosen.

Folder and Class Information

Studio Element Folder Name	Class Name
Number Capture	com.audium.server.voiceElement.digit. MBasicDigitWithConfirm

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



Alert

The `Alert` element is used to generate syslog alerts and SNMP alerts based on the values set in the Element Configuration view.

- [Settings, on page 85](#)
- [Events, on page 85](#)
- [Exit States, on page 86](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
SNMP	Boolean	Yes	true	false	true	This settings specifies whether SNMP alert to be generated.
Syslog	Boolean	Yes	false	false	false	This settings specifies whether Syslog alert to be generated.
Message	Boolean	Yes	Not Applicable	true	Blank	The alert message to be logged in SNMP ans Syslog.

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the `Customer_Lookup` element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.

Exit States

Name	Notes
done	The element execution is complete.



CHAPTER 28

Email

The `Email` action element sends messages to the provided email address. Additionally the message can include attachments. The application server must be configured to set a JNDI datasource for mail sessions. The `to` and `toList` fields are not individually required; however, at least one must be defined. Email addresses are not verified for syntax or validity. Attachments that do not exist will be skipped but the message will still be sent. Repeated email addresses are sent the message multiple times. The `toList`, `ccList` and `bccList` settings must refer to session data variables that holds a `ResultSetList` Java class holding a list of email addresses (retrieved from a Database element).

- [Settings, on page 87](#)
- [Exit States, on page 88](#)
- [Folder and Class Information, on page 89](#)
- [Events, on page 89](#)
- [Set Up Email Element, on page 89](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
jndiName (JNDI Name)	string	Yes	true	true	<i>None</i>	The configured JNDI datasource for mail sessions under the java application server.
to (To)	string	No	false	true	<i>None</i>	The email address this message will be sent to. This setting is repeatable so that each setting value contains a separate email address.
toList (To List)	string	No	true	true	<i>None</i>	The name of a session data variable containing a <code>ResultSetList</code> object holding a list of email addresses as retrieved from a Database element. The email will be sent to every address in this list.

from (From)	string	Yes	true	true	None	The email address this message will be sent from.
cc (Cc)	string	No	false	true	None	The email address this message will be carbon copied to. This setting is repeatable so that each setting value contains a separate email address.
ccList (Cc List)	string	No	true	true	None	The name of a session data variable containing a ResultSetList object holding a list of email addresses as retrieved from a Database element. The email will be carbon copied to each address in this list.
bcc (Bcc)	string	No	false	true	None	The email address this message will be blind carbon copied to. This setting is repeatable so that each setting value contains a separate email address.
bccList (Bcc List)	string	No	true	true	None	The name of a session data variable containing a ResultSetList object holding a list of email addresses as retrieved from a Database element. The email will be blind carbon copied to each address in this list.
subject (Subject)	string	No	true	true	None	Subject field of the email.
attachment (Attachment)	string	No	false	true	None	Full local path of the file to be attached. This setting is repeatable so that each setting value contains a reference to separate attachments.
messageBody (Message Body)	string	Yes	true	true	None	The message body of the email.

Exit States

Name	Notes
done	The database query successfully completed.

Folder and Class Information

Studio Element Folder Name	Class Name
Notification	com.audium.server.action.email.EmailAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.

Set Up Email Element

For the Email element to work, add a mail session under Tomcat manually.

Procedure

Step 1 Edit the \Tomcat\conf\context.xml file.

Step 2 Within the <Context> </Context> tags, add the following:

```
<Resource name="mail/ChrisMail"
type="javax.mail.Session"
mail.smtp.host="xmb-sjc-22d.amer.cisco.com"/>
```

Here, the name must be mail/*ANY_NAME_YOU_CHOOSE*, type must be javax.mail.Session, and mail.smtp.host must be a working SMTP server.

Note In Studio, edit the configuration of the Email element in question. Set the JNDI name to the *ANY_NAME_YOU_CHOOSE* portion of what you entered in the Tomcat settings. In the preceding example, you can enter *ChrisMail* but ensure that you do not include the mail/ portion here.



CHAPTER 29

Form

The `Form` voice element is used to capture any input from the caller, based on application designer-specified grammars. The valid caller inputs can be specified either directly in the voice element settings (which will create an inline grammar) or with external grammar files. Information returned by the grammar are saved in element data that then can be analyzed by developer-defined components. A `Form` voice element can be configured to listen for voice input only, DTMF input only, or both voice and DTMF input. In short, the `Form` element is the most flexible of included Unified CVP elements as it allows almost any custom information to be captured without requiring a separate voice element. If a Unified CVP or third-party voice element does not capture the information desired, one can always use a `Form` element before embarking on constructing a custom voice element.

The `Form` element provides support for custom control over the VoiceXML code generation. For example, the developer can decide what name to use for the VoiceXML field, whether or not to include a field-level slot attribute and how to name the slot attribute. The element also supports separate options for activating help prompts and the ability to set modality for `Form`.

Multiple DTMF and speech external grammars can be referenced within a single `Form` element, and the application designer has the ability to specify grammar weights for speech grammars and set MIME types for both speech and DTMF grammars. Additionally, the `Form` element can be used to capture multiple slots, and the developer can specify for which slot(s) they want the recognition values stored as element data. N-best processing can be enabled, and standard n-best results are stored in element data and the activity log.

- [Settings, on page 91](#)
- [Element Data, on page 97](#)
- [Exit States, on page 98](#)
- [Audio Groups, on page 98](#)
- [Folder and Class Information, on page 99](#)
- [Events, on page 99](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allow	Default	Notes
inputmode	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: <code>voice</code> <code>dtmf</code> <code>both</code> .

(Input Mode)						The adapter type Cisco DTMF is not compatible with input modes <code>voice</code> and <code>both</code> .
<code>noinput_timeout</code> (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
<code>form_max_noinput_count</code> (Form Max NoInput)	<code>int ≥ 0</code>	Yes	true	true	3	0 = infinite noinputs allowed.
<code>form_max_nomatch_count</code> (Form Max NoMatch)	<code>int ≥ 0</code>	Yes	true	true	3	0 = infinite nomatches allowed.
<code>confidence_level</code> (Form Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use for data capture.
<code>voice_grammar</code> (Voice Grammar)	string	*No	false	true	None	<p>Defines an external voice grammar for Form, in a string format delimited with semi-colons specifying these values in the following order:</p> <ol style="list-style-type: none"> 1. The language context in which the current grammar should be used (optional). If omitted the language will be the same as the page-scoped language. 2. The language code to assign to the <code>xml:lang</code> attribute of the parent <code><grammar></code> tag (optional). If omitted the attribute will not have an <code>xml:lang</code> attribute and the standard scoping rules apply. 3. The grammar weight (optional) 4. The grammar type (optional) 5. URL of the grammar file (required) 6. builtin: speech/transcribe <p>The type can be left blank to use the adapter default or set to <code>null</code> to not include a type at all. If one of the optional parameters is defined, four semi-colons must be used, even if the other parameters are not used. For example:</p> <ul style="list-style-type: none"> • <code>en-US;en-US;0.6;application/srgs+xml;http://IP:PORT/ mygrammar.grxml</code> • <code>fr-FR;en-US;;application/srgs+xml;http://IP:PORT/ mygrammar.grxml</code>

						<ul style="list-style-type: none"> • ;;0.6;;http://IP:PORT/mygrammar.grxml • ;fr-FR;0.6;null;http://IP:PORT/mygrammar.grxml • http://IP:PORT/mygrammar.grxml <p>This setting is repeatable so multiple external grammar sources may be specified. None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without a grammar.</p>
dtmf_grammar (DTMF Grammar)	URI	*No	false	true	None	<p>Defines an external DTMF grammar for Form, in a string format delimited with a semi-colon specifying four values in the following order:</p> <ol style="list-style-type: none"> 1. The language context in which the current grammar should be used (optional). If omitted the language will be the same as the page-scoped language. 2. The language code to assign to the <code>xml:lang</code> attribute of the parent <code><grammar></code> tag (optional) . If omitted the attribute will not have an <code>xml:lang</code> attribute and the standard scoping rules apply. 3. The grammar type (optional) 4. URL of the grammar file (required) <p>The type can be left blank to use the adapter default or set to <i>null</i> to not include a type at all. If one of the optional parameters is defined, three semi-colons must be used, even if the other parameters are not used. For example:</p> <ul style="list-style-type: none"> • en-US;en-US;application/srgs+xml;http://IP:PORT/ mygrammar.grxml • ;fr-FR;null;http://IP:PORT/mygrammar.grxml • en-US;;;http://IP:PORT/mygrammar.grxml • http://IP:PORT/mygrammar.grxml <p>This setting is repeatable so multiple external grammar sources may be specified. None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without a grammar.</p>
voice_keyword	string	*No	false	true	None	<p>Defines the inline voice grammar for Form, with each configuration of this repeatable setting</p>

(Voice Keyword)						<p>specifying one option for the grammar. The valid format is a string separated with a semi-colon specifying four values in the following order:</p> <ol style="list-style-type: none"> 1. The language context in which the current input should be included in the inline grammar (optional). If omitted the language will be the same as the page-scoped language. 2. The language code to assign to the <code>xml:lang</code> attribute of the <code><item></code> tag inside the inline grammar (optional) . If omitted the attribute will not have an <code>xml:lang</code> attribute and the standard scoping rules apply. 3. The weight of the grammar item (optional) 4. The grammar item (required) <p>Note The grammar item may either contain the input itself followed by an optional return value, or just the input. If one of the optional parameters is defined, three semi-colons must be used, even if the other parameters are not used.</p> <p>Sample configurations values are:</p> <ul style="list-style-type: none"> • en-US;en-US;0.6;news report [news] • ;fr-FR;0.6;news report • news report [news] • news report <p>None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without at least one grammar.</p>
dtmf_keypress (DTMF Keypress)	character (0-9, #,)	*No	false	true	None	<p>Defines the inline DTMF grammar for Form, with each configuration of this repeatable setting specifying one option for the grammar. The valid format is a string separated with a semi-colon specifying three values in the following order:</p> <ol style="list-style-type: none"> 1. The language context in which the current input should be included in the inline grammar (optional). If omitted the language will be the same as the page-scoped language. 2. The language code to assign to the <code>xml:lang</code> attribute of the <code><item></code> tag inside the inline grammar (optional) . If omitted the attribute

						<p>will not have an <code>xml:lang</code> attribute and the standard scoping rules apply.</p> <p>3. A character (0-9, #, *) representing the keypress, followed by an optional return value.</p> <p>Note The grammar item may either contain the input itself followed by an optional return value, or just the input. If one of the optional parameters is defined, two semi-colons must be used, even if the other parameters are not used.</p> <p>Sample configurations values are:</p> <ul style="list-style-type: none"> • en-US;en-US;1 [news] • ;fr-FR;1 • 1 [news] • 1 <p>None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without at least one grammar.</p>
help_voice_keyword (Help Voice Keyword)	string	No	false	true	None	<p>Specifies a custom inline voice grammar to activate the help audio group. Each value of this repeatable setting adds another valid utterance. The format is a string specifying just the utterance (for example, <i>news report</i>).</p> <p>If this setting is configured, a custom inline voice grammar will be generated, replacing the default help grammar used by a browser, and the custom grammar will be active only within the current Form element.</p>
help_dtmf_keypress (Help DTMF Keypress)	character (0-9, #, *)	No	false	true	None	<p>Specifies a custom inline DTMF grammar to activate the help audio group. Each value of this repeatable setting adds another valid DTMF keypress. The format is a character (0-9, #, *) representing just the keypress.</p> <p>If this setting is configured, a custom inline DTMF grammar will be generated, and it will be active only within the current Form element.</p>
modal (Disable Hotlinks)	boolean	Yes	true	true	false	<p>Whether or not to temporarily disable all hotlink grammars (global or local) and universal grammars. If set to true, only the current Form</p>

						element grammars will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
field_name (Field Name)	string	Yes	true	true	foundation_fld	<i>foundation_fld</i> - The value to assign to the VXML field name attribute.
slot_name (Field Slot)	string	No	true	true	None	The name to assign to the VXML field slot attribute. If left unspecified, the field will not include a slot attribute.
slot_element_data (Slot Element Data)	string	No	false	true	None	Specifies for which grammar slot the return value should be stored as element data. This is a repeatable setting so multiple slot names can be specified. See notes below for further details.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example nbestUtterance1_secureLogging,*****.

- VXML 2.0-compliant browsers typically require top-level slot names in the grammar (inline or external) to match the field-level slot attribute (if it exists) or the field name attribute, in order for the field name variable (and hence the *value* element data) to be defined. For inline grammars, the Form element automatically generates the grammar slot name to match the slot attribute (if available) or the field name. For custom grammars that are referenced from an external source, the application designer needs to set *Field Name* and *Field Slot* properly based on the slot name returned by the grammar.
- If a grammar returns different slots for different inputs or multiple slots per utterance, there are two ways to configure the Form element to store this data:
 - Leave the *slot_element_data* setting empty. The Form element will create element data named “nbestInterpretationX” (where X is from 1 to the length of the n-best list) that contains a string that uses delimiters “+” and “.” to separate the multiple slot names from their values. For example: “+Slot1:value1+Slot2:value2...”. A developer would then need to parse this string in a subsequent element to obtain the different slot name and value pairs.
 - Configure the *slot_element_data* setting with the names for all the slots that can be returned. The Form element will create a new set of n-best element data to store the recognition results for each slot listed in that setting. The element data will be named as *<SLOT_ELEMENT_DATA_X>* (where *SLOT_ELEMENT_DATA* is a string identical to the setting value and X is from 1 to the length of the n-best list). For example, if *slot_element_data* had two values *city* and *state* and there are three n-best results triggered, then six element data in the names of *city1*, *city2*, *city3*, *state1*, *state2*, and *state3* will be created to store each of the n-best values for the *city* and *state* slots. Note that if n-best processing is disabled by setting the *maxnbest* setting to 1, then only one interpretation

result will be returned per recognition and thereby only one element data per slot (*city1* and *state1*) will be created.

Element Data

Name	Type	Notes
value	string	This stores the value of the VXML field name variable.
value_confidence	float	This stores the confidence score of the captured Form utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
<SLOT_ELEMENT_DATA1> <SLOT_ELEMENT_DATA2> ... <SLOT_ELEMENT_DATAX*>	string	A separate set of element data stores the interpretation values for each filled slot of captured n-best utterances. While the maximum number of <SLOT_ELEMENT_DATAX> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is dependent on speech recognition at runtime, where <SLOT_ELEMENT_DATA1> holds the slot value of the top hypothesis in the n-best list and <SLOT_ELEMENT_DATAX> holds the slot value of the last hypothesis. Note If the <code>slot_element_data</code> setting is blank, these sets of element data will not be created.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of <code>nbestUtteranceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestUtterance1</code> holds the utterance of the top hypothesis in the n-best list and <code>nbestUtteranceX</code> holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of <code>nbestInterpretationX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestInterpretation1</code> holds the interpretation of the top hypothesis in the n-best list and <code>nbestInterpretationX</code> holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of <code>nbestConfidenceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestConfidence1</code> holds the confidence score of the top hypothesis in the n-best list and <code>nbestConfidenceX</code> holds the confidence score of the last hypothesis.

nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances. This stores the number of no input events that the browser returned during the collection phase of the VXML field name variable.
collect_noinput_count	int ≥ 0	This stores the number of no input events that the browser returned during the collection phase of the VXML field name variable.
collect_nomatch_count	int ≥ 0	This stores the number of no match events that the browser returned during the collection phase of the VXML field name variable.

* SLOT_ELEMENT_DATA is a string identical to the configuration value of the slot_element_data setting, and X is from 1 to the length of the n-best list. If more than one such value is configured, then multiple sets of element data using the same naming convention will be created.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
done	The caller input matched the grammar correctly.

Audio Groups

Form Data Capture

Name (Label)	Req'd	Max1	Notes
initial_audio_group (Initial)	Yes	Yes	Played when the voice element first begins.
nomatch_audio_group (NoMatch)	No	No	Played when a nomatch event occurs.
noinput_audio_group (NoInput)	No	No	Played when a noinput event occurs.

help_audio_group (Help)	No	No	Played when the caller asks for help. If not specified, help is treated as a nomatch event by default.
----------------------------	----	----	--

End

Name (Label)	Req'd	Max 1	Notes
done_audio_group (Done)	No	Yes	Played when the form data capture is completed, and the voice element exits with the done exit state.

Folder and Class Information

Studio Element Folder Name	Class Name
Form	com.audium.server.voiceElement.form. MFoundationForm

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 30

Form_with_Confirm

The `Form_With_Confirm` voice element is used to capture and confirm input from the caller, based on application designer-specified grammars. The valid caller inputs can be specified either directly in the voice element settings (which will create an inline grammar) or with external grammar files. Information returned by the grammar are saved in element data that then can be analyzed by developer-defined components. A `Form_With_Confirm` voice element can be configured to listen for voice input only, DTMF input only, or both voice and DTMF input. In short, the `Form_With_Confirm` element is the most flexible of included elements that have confirmation menus as it allows almost any custom information to be captured and confirmed without requiring a separate voice element. If a Unified CVP or third-party voice element does not capture and confirm the information desired, one can always use a `Form_With_Confirm` element before embarking on constructing a custom voice element.

The `Form_With_Confirm` element provides support for custom control over the VoiceXML code generation. For example, the developer can decide what name to use for the VoiceXML field, whether or not to include a field-level slot attribute and how to name the slot attribute. The element also supports separate options for activating help prompts and the ability to set modality for Form.

Multiple DTMF and speech external grammars can be referenced within a single `Form_With_Confirm` element, and the application designer has the ability to specify grammar weights for speech grammars and set MIME types for both speech and DTMF grammars. Additionally, the `Form_With_Confirm` element can be used to capture multiple slots, and the developer can specify for which slot(s) they want the recognition values stored as element data. N-best processing can be enabled, and standard n-best results are stored in element data and the activity log.

- [Settings, on page 101](#)
- [Element Data, on page 108](#)
- [Exit States, on page 109](#)
- [Audio Groups, on page 110](#)
- [Folder and Class Information, on page 111](#)
- [Events, on page 111](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allow	Default	Notes

inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: <code>voice</code> <code>dtmf</code> <code>both</code> . The adapter type Cisco DTMF is not compatible with input modes <code>voice</code> and <code>both</code> .
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
form_max_noinput_count (Form Max NoInput)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during form input capture. 0 = infinite noinputs allowed.
form_max_nomatch_count (Form Max NoMatch)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during form input capture. 0 = infinite nomatches allowed.
confirm_max_noinput_count (Confirm Max NoInput)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during form input confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during form input confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int ≥ 0	Yes	true	true	3	The maximum number of times a caller is allowed to disconfirm a captured input. 0 = infinite disconfirmations allowed.
form_confidence_level (Form Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use for capture of the form data.
confirm_confidence_level (Confirm Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.50	The confidence level threshold to use for confirmation of the form data.
voice_grammar (Voice Grammar)	string	*No	false	true	None	Defines an external voice grammar for Form_With_Confirm, in a string format delimited with semi-colons specifying five values in the following order: 1. The language context in which the current grammar should be used (optional). If omitted the language will be the same as the page-scoped language.

						<ol style="list-style-type: none"> The language code to assign to the <code>xml:lang</code> attribute of the parent <code><grammar></code> tag (optional). If omitted the attribute will not have an <code>xml:lang</code> attribute and the standard scoping rules apply. The grammar weight (optional) The grammar type (optional) URL of the grammar file (required) <p>The type can be left blank to use the adapter default or set to 'null' to not include a type at all. If one of the optional parameters is defined, four semi-colons must be used, even if the other parameters are not used. For example:</p> <ul style="list-style-type: none"> en-US;en-US;0.6;application/srgs+xml;http://IP:PORT/mygrammar.grxml fr-FR;en-US;;application/srgs+xml;http://IP:PORT/mygrammar.grxml ;;0.6;;http://IP:PORT/mygrammar.grxml ;fr-FR;0.6>null;http://IP:PORT/mygrammar.grxml http://IP:PORT/mygrammar.grxml <p>This setting is repeatable so multiple external grammar sources may be specified. None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without a grammar.</p>
dtmf_grammar (DTMF Grammar)	URI	*No	false	true	None	<p>Defines an external DTMF grammar for Form_With_Confirm, in a string format delimited with a semi-colon specifying four values in the following order:</p> <ol style="list-style-type: none"> The language context in which the current grammar should be used (optional). If omitted the language will be the same as the page-scoped language. The language code to assign to the <code>xml:lang</code> attribute of the parent

						<p><grammar> tag (optional) . If omitted the attribute will not have an <code>xml:lang</code> attribute and the standard scoping rules apply.</p> <p>3. The grammar type (optional)</p> <p>4. URL of the grammar file (required)</p> <p>The type can be left blank to use the adapter default or set to 'null' to not include a type at all. If one of the optional parameters is defined, three semi-colons must be used, even if the other parameters are not used. For example:</p> <ul style="list-style-type: none"> • en-US;en-US;application/srgs+xml;http://IP:PORT/mygrammar.grxml • ;fr-FR;null;http://IP:PORT/mygrammar.grxml • en-US;;;http://IP:PORT/mygrammar.grxml • http://IP:PORT/mygrammar.grxml <p>This setting is repeatable so multiple external grammar sources may be specified. None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without a grammar.</p>
voice_keyword (Voice Keyword)	string	*No	false	true	None	<p>Defines the inline voice grammar for Form_With_Confirm, with each configuration of this repeatable setting specifying one option for the grammar. The valid format is a string separated with a semi-colon specifying four values in the following order:</p> <ol style="list-style-type: none"> 1. The language context in which the current input should be included in the inline grammar (optional). If omitted the language will be the same as the page-scoped language. 2. The language code to assign to the <code>xml:lang</code> attribute of the <item> tag inside the inline grammar (optional) . If omitted the attribute will not have an <code>xml:lang</code> attribute and the standard scoping rules apply.

						<p>3. The weight of the grammar item (optional)</p> <p>4. The grammar item (required)</p> <p>Note The grammar item may either contain the input itself followed by an optional return value, or just the input. If one of the optional parameters is defined, three semi-colons must be used, even if the other parameters are not used.</p> <p>Sample configurations values are:</p> <ul style="list-style-type: none"> • en-US;en-US;0.6;news report [news] • ;fr-FR;0.6;news report • news report [news] • news report <p>None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without a grammar.</p>
dtmf_keypress (DTMF Keypress)	character (0-9, #,)	*No	false	true	None	<p>Defines the inline DTMF grammar for Form_With_Confirm, with each configuration of this repeatable setting specifying one option for the grammar. The valid format is a string separated with a semi-colon specifying three values in the following order:</p> <ol style="list-style-type: none"> 1. The language context in which the current input should be included in the inline grammar (optional). If omitted the language will be the same as the page-scoped language. 2. The language code to assign to the <code>xml:lang</code> attribute of the <code><item></code> tag inside the inline grammar. If omitted the attribute will not have an <code>xml:lang</code> attribute and the standard scoping rules apply. 3. A character (0-9, #, *) representing the keypress, followed by an optional return value.

						<p>Note The grammar item may either contain the input itself followed by an optional return value, or just the input. If one of the optional parameters is defined, two semi-colons must be used, even if the other parameters are not used.</p> <p>Sample configurations values are:</p> <ul style="list-style-type: none"> • en-US;en-US;1 [news] • ;fr-FR;1 • 1 [news] • 1 <p>None of the four settings - <code>voice_grammar</code>, <code>dtmf_grammar</code>, <code>voice_keyword</code> and <code>dtmf_keypress</code> - is required, but at least one must be specified since a form cannot be completed without a grammar.</p>
<code>help_voice_keyword</code> (Help Voice Keyword)	string	No	false	true	None	<p>Specifies a custom inline voice grammar to activate the help audio group. Each value of this repeatable setting adds another valid utterance. The format is a string specifying just the utterance (for example, <i>news report</i>).</p> <p>If this setting is configured, a custom inline voice grammar will be generated, replacing the default help grammar used by a browser, and the custom grammar will be active only within the current <code>Form_With_Confirm</code> element.</p>
<code>help_dtmf_keypress</code> (Help DTMF Keypress)	character (0-9, #, *)	No	false	true	None	<p>Specifies a custom inline DTMF grammar to activate the help audio group. Each value of this repeatable setting adds another valid DTMF keypress. The format is a character (0-9, #, *) representing just the keypress.</p> <p>If this setting is configured, a custom inline DTMF grammar will be generated, and it will be active only within the current <code>Form_With_Confirm</code> element.</p>
<code>modal</code> (Disable Hotlinks)	boolean	Yes	true	true	false	<p>Whether or not to temporarily disable all hotlink grammars (global or local) and universal grammars. If set to true, only the current <code>Form_With_Confirm</code> element</p>

						grammars (including the builtin boolean grammar for confirmation) will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
field_name (Field Name)	string	Yes	true	true	foundation_fld	<i>foundation_fld</i> - The value to assign to the VXML field-level name attribute.
slot_name (Field Slot)	string	No	true	true	None	The name to assign to the VXML field-level slot attribute. If left unspecified (i.e. the default value), the field will not have a slot attribute.
slot_element_data (Slot Element Data)	string	No	false	true	None	Specifies for which grammar slot the return value should be stored as element data. This is a repeatable setting so multiple slot names can be specified. See notes below for further details.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example <i>nbestUtterance1_secureLogging, *****</i> .
recordutterance	boolean	No	true	true	false	If inputmode is set to <i>both</i> or <i>voice</i> , this element is displayed. When <i>recordutterance</i> is set to <i>true</i> , the record utterance file name is returned for <i>done</i> and <i>nomatch</i> states.

- VXML 2.0-compliant browsers typically require top-level slot names in the grammar (inline or external) to match the field-level slot attribute (if it exists) or the field name attribute, in order for the field name variable (and hence the *value* element data) to be defined. For inline grammars, the *Form_With_Confirm* element automatically generates the grammar slot name to match the slot attribute (if available) or the field name. For custom grammars that are referenced from an external source, the application designer needs to set *Field Name* and *Field Slot* properly based on the slot name returned by the grammar.
- If a grammar returns different slots for different inputs or multiple slots per utterance, there are two ways to configure the *Form_With_Confirm* element to store this data:
 - Leave the *slot_element_data* setting empty. The *Form_With_Confirm* element will create element data named *nbestInterpretationX* (where X is from 1 to the length of the n-best list) that contains a string that uses delimiters “+” and “.” to separate the multiple slot names from their values. For

example: “+Slot1:value1+Slot2:value2...”. A developer would then need to parse this string in a subsequent element to obtain the different slot name and value pairs.

- Configure the `slot_element_data` setting with the names for all the slots that can be returned. The `Form_With_Confirm` element will create a new set of n-best element data to store the recognition results for each slot listed in that setting. The element data will be named as `<SLOT_ELEMENT_DATA<X>` (where `SLOT_ELEMENT_DATA` is a string identical to the setting value and `X` is from 1 to the length of the n-best list). For example, if `slot_element_data` had two values `city` and `state` and there are three n-best results triggered, then six element data in the names of `city1`, `city2`, `city3`, `state1`, `state2`, and `state3` will be created to store each of the n-best values for the `city` and `state` slots.

Note If n-best processing is disabled by setting the `maxnbest` setting to 1, then only one interpretation result will be returned per recognition and thereby only one element data per slot (`city1` and `state1`) will be created.

Element Data

Name	Type	Notes
<code>value</code>	string	This stores the value of the VXML field name variable.
<code>value_confidence</code>	float	This stores the confidence score of the captured <code>Form_With_Confirm</code> utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
<code><SLOT_ELEMENT_DATA1></code> <code><SLOT_ELEMENT_DATA2></code> ... <code><SLOT_ELEMENT_DATA<X></code>	string	<p>A separate set of element data stores the interpretation values for each filled slot of captured n-best utterances. While the maximum number of <code><SLOT_ELEMENT_DATA<X></code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is dependent on speech recognition at runtime, where <code><SLOT_ELEMENT_DATA1></code> holds the slot value of the top hypothesis in the n-best list and <code><SLOT_ELEMENT_DATA<X></code> holds the slot value of the last hypothesis.</p> <p>Note If the <code>slot_element_data</code> setting is blank, these sets of element data will not be created.</p>
<code>nbestLength</code>	<code>int ≥ 1</code>	This stores the number of n-best hypotheses generated by the speech engine.
<code>nbestUtterance1</code> <code>nbestUtterance2</code> ... <code>nbestUtteranceX</code>	string	This set of element data stores the captured n-best utterances. While the maximum number of <code>nbestUtterance<X></code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestUtterance1</code> holds the utterance of the top hypothesis in the n-best list and <code>nbestUtteranceX</code> holds the utterance of the last hypothesis.
<code>nbestInterpretation1</code> <code>nbestInterpretation2</code> ...	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of <code>nbestInterpretation<X></code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime,

nbestInterpretationX		where <code>nbestInterpretation1</code> holds the interpretation of the top hypothesis in the n-best list and <code>nbestInterpretationX</code> holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of <code>nbestConfidenceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestConfidence1</code> holds the confidence score of the top hypothesis in the n-best list and <code>nbestConfidenceX</code> holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.
collect_noinput_count	int ≥ 0	This stores the number of no input events that the browser returned during the collection phase of the VXML field name variable.
collect_nomatch_count	int ≥ 0	This stores the number of no match events that the browser returned during the collection phase of the VXML field name variable.
confirm_noinput_count	int ≥ 0	This stores the number of no input events that the browser returned during the confirmation phase of the VXML field name variable.
confirm_nomatch_count	int ≥ 0	This stores the number of no match events that the browser returned during the confirmation phase of the VXML field name variable.

* “SLOT_ELEMENT_DATA” is a string identical to the configuration value of the “slot_element_data” setting, and X is from 1 to the length of the n-best list. If more than one such value is configured, then multiple sets of element data using the same naming convention will be created.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the <code>nomatch max count</code> is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the <code>noinput max count</code> is 0, this exit state will never occur.
max_disconfirmed	The maximum number of disconfirm events has occurred. If the <code>disconfirm max count</code> is 0, this exit state will never occur.
done	The caller input matched the grammar correctly.

Audio Groups

Form Data Capture

Name (Label)	Req'd	Max1	Notes
form_initial_audio_group (Form Initial)	Yes	Yes	Played when the voice element first begins.
form_nomatch_audio_group (Form NoMatch)	No	No	Played when a nomatch event occurs during form data capture.
form_noinput_audio_group (Form NoInput)	No	No	Played when a noinput event occurs during form data capture.
form_help_audio_group (Form Help)	No	No	Played when the caller asks for help during form data capture. If not specified, help is treated as a nomatch event by default.

Form Data Confirm

Name (Label)	Req'd	Max1	Notes
confirm_initial_audio_group (Confirm Initial)	Yes	Yes	Played after the caller enters a value, requesting the caller's confirmation of that value.
confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during confirmation.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during confirmation.
confirm_help_audio_group (Confirm Help)	No	No	Played when the caller asks for help during confirmation.
disconfirmed_audio_group (Disconfirmed)	No	No	Played when the caller disconfirms the value.

End

Name (Label)	Req'd	Max 1	Notes
--------------	-------	-------	-------

yes_audio_group (Yes)	No	Yes	Played after the caller chooses the <i>yes</i> option. If not specified, no audio will be played when this option is chosen.
--------------------------	----	-----	--

Folder and Class Information

Studio Element Folder Name	Class Name
Form	com.audium.server.voiceElement.form. MFoundationFormWithConfirm

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 31

FTP_Client

The `FTP_Client` element is used to upload a local file to one or more FTP servers. If there are multiple FTP servers specified, the file is uploaded concurrently to the FTP servers.

- [Settings, on page 113](#)
- [Element Data, on page 115](#)
- [Exit States, on page 116](#)
- [Other, on page 116](#)
- [Events, on page 116](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Validation Enforced by Call Studio	Notes
filename (Name of file to be transferred)	string	Yes	true	true	None	Must be a valid Windows filename.	This setting specifies the full pathname of the file to transfer. Alternatively, a path relative to the application directory can be used.
remote_filename (Remote Filename)	string	No	true	true	None	If specified, must be a valid Windows filename.	This is the FTP server target filename. If a remote filename is not specified, the remote filename will be the same as the input filename.
ftp_hosts (FTP Server or FTP Servers)	string	Yes	true	true	None	Must conform to the format listed in "Notes".	This is the list of FTP server host names or IP addresses to transfer the file to. Each FTP server entry may optionally specify a port number (default port:21), username and password in the format

						Validation will fail if the password is set, but the username is not.	host port username password. Server entries are delimited by a space character. You can enter multiple hosts on one line or separate lines or both. If any field requires spaces, vertical bars () or equals symbols (=), they may be escaped with \s, \p or \e, respectively.
ftp_user (Default Username)	string	Yes	true	true	None	Validation will fail if the password field is set while this field is not set.	User name to use when transferring the file. This value may be overridden on a per-server basis. If left blank, "anonymous" will be assumed.
ftp_password (Default Password)	string	No	true	true	None	n/a	This is the password to use when transferring the file. This value can be overridden on a per-server basis.
ftp_path (FTP Path)	string	No	true	true	None	Must be a valid Windows pathname.	This is the directory on the FTP server where to transfer the file. Use the forward slash as the directory delimiter dir/subdir. The directory will be created if it does not already exist.
delete_file_on_success (Delete file if file transferred successfully)	boolean	No	true	true	true	n/a	This setting deletes the file after it has been successfully transferred to all FTP Server(s).



Note Default ftp_user/ftp_password will be used if ftp_hosts setting does not include a username/password in its definition.



Note It is important to ensure that the FTP Server(s) are open for write access.



Note The file to be uploaded is assumed to be a binary file.



Note If a large file is to be transferred and the network connection to the FTP servers is slow and there are multiple FTP servers, consider implementing VXML 'fetchaudio' functionality in the element before the FTP element so that the caller does not hear silence while the FTP operation is in progress.



Note The http client response timeout setting on the gateway must be set to accommodate the time it takes to complete the largest anticipated FTP file transfer. If an FTP file transfer takes longer than the configured duration in seconds for http client response timeout, the FTP transfer will complete correctly, but the **call will drop as soon as the configured timeout duration is met**.

Element Data

Element data is created *only* when the `exit state` setting is not *done*. If the exit state is *done*, no element data is created.

Name	Type	Notes
failed_servers	string	One or more space delimited host names or IP addresses of Server(s) where the input file was not successfully transferred. This data is created only if the exit state is not <i>done</i> .
failed_server_reasons	string	<p>One or more space delimited reason codes indicating why a file was not successfully transferred:</p> <ul style="list-style-type: none"> • <code>connection_error</code>: There was an error connecting to the FTP server. This may be caused by an invalid or blocked port. • <code>extraneous_data</code>: There were extra fields for a given server in the <code>ftp_hosts</code> setting. • <code>invalid_filename</code>: The name of the file to transfer is invalid or the file doesn't exist. • <code>invalid_port</code>: The port for an FTP server is invalid. • <code>missing_username</code>: The password for an FTP server was specified, but the username was left blank. They must either both be specified or both left blank. • <code>unknown</code>: An unknown error has occurred. • <code>unknown_host</code>: An FTP server could not be reached. Possible reasons include an incorrect hostname or network connectivity problems. A three-digit number: An FTP server sent back an unexpected reply code. Additional information will appear in the error log. • <code>A three-digit number</code>: An FTP server sent back an unexpected reply code. Additional information will appear in the error log.

		<ul style="list-style-type: none"> A <code>Java exception</code>: An unexpected exception was handled. Additional information will appear in the error log.
failed_servers_count	string	Number of failed FTP transfers. This data is created only if the exit state is not <i>done</i> .

Exit States

Name	Notes
error	This exit state is used if an error occurred and the file was not transferred to any FTP Server(s).
partial_success	This exit state is used when not all FTP transfers were successful.
done	This exit state means the file was successfully transferred to all FTP Server(s).

Other

Studio Element Folder: Integration

Class Name: com.cisco.cvp.vxml.custelem.FTP

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 32

Math

The `Math` action element is used to evaluate basic mathematical expressions. The mathematical expression is composed of operators and functions in the form of a string which is passed as a setting to the element, parsed and evaluated at runtime. The result is a double value stored as a string in either element data or session data. All common arithmetic operators are supported. Boolean operators are also fully supported. Boolean expressions are evaluated to be either 1.0 or 0.0 (*true* or *false* respectively).

- [Examples, on page 117](#)
- [Settings, on page 117](#)
- [Operators and Functions, on page 118](#)
- [Element Data, on page 119](#)
- [Session Data, on page 119](#)
- [Exit States, on page 119](#)
- [Folder and Class Information, on page 119](#)
- [Events, on page 119](#)

Examples

Expression: <code>2 * 4</code> Result: 8.0	Expression: <code>sqrt(16)</code> Result: 4.0	Expression: <code>{Data.Session.myNumber} == 4</code> Result: 1.0
---	--	--

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Type (Type)	string enum	Yes	true	false	Element	This setting specifies the type of data that will store the result of the mathematical expression. Possible values are: <code>Element</code> <code>Session</code> . Default = <code>Element</code> .

Name (Name)	string	Yes	true	true	None	This setting specifies the name to assign to the data that will store the result of the mathematical expression.
Expression (Expression)	string	Yes	true	true	None	This setting specifies the mathematical expression to parse and evaluate. For supported operators and functions see tables below.

Operators and Functions

Operator Name	Operator	Function Name	Syntax
Power	^	Sine	sin(x)
Boolean Not	!	Cosine	cos(x)
Unary Plus, Unary Minus	+x, -x	Tangent	tan(x)
Modulus	%	Arc Sine	asin(x)
Division	/	Arc Cosine	acos(x)
Multiplication	*	Arc Tangent	atan(x)
Addition, Subtraction	+, -	Arc Tangent (with 2 parameters)	atan2(y, x)
Less or Equal, More or Equal	<=, >=	Hyperbolic Sine	sinh(x)
Less Than, Greater Than	<, >	Hyperbolic Cosine	cosh(x)
Not Equal, Equal	!=, ==	Hyperbolic Tangent	tanh(x)
Boolean And	&&	Inverse Hyperbolic Sine	asinh(x)
Boolean Or		Inverse Hyperbolic Cosine	acosh(x)
		Inverse Hyperbolic Tangent	atanh(x)
		Natural Logarithm	ln(x)
		Logarithm base 10	log(x)
		Exponential	exp(x)
		Absolute Value / Magnitude	abs()
		Modulus	mod()
		Square Root	sqrt()
		Sum	sum()
		If	if()

Element Data

Element data is created *only* when the `type` setting is set to *Element*. In all other cases, no element data is created.

Name	Type	Notes
[value of setting “name”]	string	The result of the mathematical expression.

Session Data

Session data is created *only* when the `type` setting is set to *Session*. In all other cases, no session data is created.

Name	Type	Notes
[value of setting “name”]	string	The result of the mathematical expression.

Exit States

Name	Notes
done	The mathematical expression was evaluated and the result was stored as either element data or session data.

Folder and Class Information

Studio Element Folder Name	Class Name
Math	com.audium.server.action.math.MathAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 33

Local Variables

- [Set Value Element, on page 121](#)
- [Change Implementation Order of Local Variables, on page 121](#)

Set Value Element

The Set Value element allows you to define and assign values to local variables. It supports basic mathematical operation, string operation, and Java script. The Set Value element allows you to specify a Java script which does the required programming in the application. The Java script allows substitution of other element data. The evaluation result of Java script is stored in the variable specified in the **Settings** tab. The scope of the local variable is restricted to a particular subflow or main flow in which it is defined and is not available in another subflow or main flow.

You can perform the following operations on local variables:

- Add Variable
- Delete Variable
- Update Variable
- Move Variable



Note

The **Settings** tab does not display the **Delete Variable** and **Update Variable** options when you add a variable for the first time.

Change Implementation Order of Local Variables

Cisco Unified Call Studio allows you to select and move the local variables up and down on the **Settings** tab to change the order in which they are implemented. The implementation order of local variables will be same as the order as defined in the **Settings** tab.

Follow these steps to change the implementation order of local variables in the Settings tab.

Procedure

- Step 1** On the **Settings** tab, right-click the local variable you want to move up or down and choose **Mark Variable**.
- Step 2** Choose the location where you want to move the marked local variable, right-click and choose **Move Variable**.
-



CHAPTER 34

Menu Support for 2_Option_Menu Through 10_Option_Menu

These voice elements define menus that support from 2 to 10 options. The Menu voice elements are similar to the Form voice element, however the number of choices is fixed and all grammars are defined in the voice element itself. Additionally, there is an exit state for each option, therefore the captured value does not have to be analyzed afterwards to determine the next dialog in the call flow. Use Menu elements when the situation defines a fixed number of choices where each choice does something different in the call flow.

Because the number of exit states is fixed for a voice element, there are separate voice elements for Menu voice elements with 2 to 10 options. For each additional option, three additional settings are added to handle the spoken keyword, DTMF entry, and interpretation value for each option. The audio groups and element data saved are the same for all Menu voice elements.

Each option must be assigned an interpretation value that the element will return as element data named `value` when any of the keywords or DTMF key presses assigned to that option are captured. The element variable (`value`) will contain the same value regardless of the input mode (speech or DTMF).

The audio groups are identical to those of the Form voice element. The `done_audio_group` group may be used for a message that is to be played regardless of what option is chosen. If you require an option specific message, use an Audio voice element after the particular choice is made and do not configure a

`done_audio_group`

- [Settings, on page 123](#)
- [Element Data, on page 126](#)
- [Exit States, on page 126](#)
- [Audio Groups, on page 127](#)
- [Folder and Class Information, on page 127](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
<code>noinput_timeout</code> (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is

						thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
max_noinput_count (Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during input capture. 0 = infinite noinputs allowed.
max_nomatch_count (Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during input capture. 0 = infinite nomatches allowed.
confidence_level (Confidence Level)	decimal (0.0 to 1.0)	Yes	true	true	0.40	The confidence level threshold to use.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	Whether or not to temporarily disable all hotlink grammars (global or local) and universal grammars. If set to true, only the grammars of the current X_Option_Menu element will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
optionX_dtmf (Option X DTMF)	Character (0-9, #, *)	No	true	true	None	<p>This setting defines the DTMF grammar that can be used to select the menu <code>optionX</code>. The valid format is a string separated with a semi-colon specifying two values in this order:</p> <ol style="list-style-type: none"> 1. The language context in which the current input should be included in the menu grammar (optional). If omitted the language used will be the same as the page-scoped language. 2. The dtmf keypress or keypresses that is included in the menu DTMF grammar (required) <p>Sample configurations values are:</p> <ul style="list-style-type: none"> • en-US;1 • 1 <p>Additional <code>optionX_dtmf</code> settings may be used to define multiple dtmf keypresses corresponding to the same return value.</p>

						<p>Note At minimum, one of the two settings: <code>optionX_dtmf</code> or <code>optionX_voice</code> must be specified.</p> <p>Note Keypresses are currently limited to single digits.</p>
optionX_voice (Option X Voice)	string	No	true	true	None	<p>This setting defines the voice grammar that can be used to select the menu <code>optionX</code>. Each configuration of this setting specifies an option for the grammar. The valid format is a string separated with semi-colons specifying three values in this order:</p> <ol style="list-style-type: none"> 1. The language context in which the current input should be included in the menu grammar (optional). If omitted the language used will be the same as the page-scoped language. 2. <i>exact</i> or <i>approximate</i> (optional) for the accept attribute value, where if <i>exact</i>, the spoken utterance must match the expected value exactly; and where if <i>approximate</i>, the spoken utterance may match one of several words 3. The voice keyword or keywords (required) that is included in the menu voice grammar. <p>If one of the optional parameters is defined, two semi-colons must be used, even if the other parameter is not used. Sample configuration values are:</p> <ul style="list-style-type: none"> • en-US;exact;news report • ;approximate;news report • fr-FR;;news report • news report <p>Additional <code>optionX_voice</code> settings may be used to define multiple matching voice keywords corresponding to the same return value.</p>

						Note At the minimum, one of the two settings: <code>optionX_dtmf</code> or <code>optionX_voice</code> <i>must</i> be specified.
optionX_value (Option X Value)	string	Yes	false	true	None	The value to be stored in the element data value for this voice element when the caller selects <i>optionX</i> . Note Only a single value is allowed for each option.

Where X is 2 – 10 as applicable.

Some voice browsers may not support menu options using * or #.

Element Data

Name	Type	Notes
value	string	The value associated with the keyword or DTMF keypress inputted by the caller is stored in this variable.
value_confidence	float	This is the confidence value of the matched utterance.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the <code>max_nomatch_count</code> is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the <code>max_noinput_count</code> is 0, this exit state will never occur.
optionX	The utterance or DTMF entry matched optionX.

Where X is 2 – 10 as applicable.

Note	Each option can react on just a spoken keyword, just DTMF keypresses, or both, but at least one method must be specified or an error will be reported.
Note	All options in the menu must have a consistent input mode. For example, a menu cannot be configured so that option 1 is chosen through both voice and DTMF but option 2 is chosen only through voice.
Note	There are no menus with more than 10 options. In cases where more are needed, use a Form voice element.

Audio Groups

Menu Option Capture

Name (Label)	Req'd	Max1	Notes
initial_audio_group (Initial)	Yes	Yes	Played when the voice element first begins.
nomatch_audio_group (NoMatch)	No	No	Played when a nomatch event occurs.
noinput_audio_group (NoInput)	No	No	Played when a noinput event occurs.
help_audio_group (Help)	No	No	Played when the caller asked for help. If not specified, by default help is treated as a nomatch.

End

Name (Label)	Req'd	Max 1	Notes
done_audio_group (Done)	No	Yes	Played when the voice element completes any of the option exit states.

Folder and Class Information

Studio Element Folder Name	Class Name
Menu	com.audium.server.voiceElement.menu.MFoundationXOptionsMenu



CHAPTER 35

Number

The `Number` voice element captures a number input from the caller. The number can be spoken or entered using the keypad. The resulting value will be stored in element data as a decimal value. The number can be negative or positive and can contain a decimal point. Using DTMF entry the number is restricted to being positive and the decimal point is entered by pressing the * key. Using speech input, the number may be spoken naturally.

Note You cannot use the * character to represent a decimal point in the `Number` voice element, if you have defined it as a **termchar** in the **Root Doc Settings**.

- [Settings, on page 129](#)
- [Element Data, on page 130](#)
- [Exit States, on page 131](#)
- [Audio Groups, on page 132](#)
- [Folder and Class Information, on page 132](#)
- [Events, on page 132](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5e	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.

max_noinput_count (Number Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during number input capture. 0 = infinite noinputs allowed.
max_nomatch_count (Number Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during number input capture. 0 = infinite nomatches allowed.
number_confidence_level (Number Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use during number capture.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	Whether or not to temporarily disable all hotlink grammars (global or local) and universal grammars. If set to true, only the grammars of the current Number element will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example nbestUtterance1_secureLogging,****.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the Element Data table for information about nbestUtteranceX and nbestInterpretationX

Element Data

Name	Type	Notes
Value	string	The number captured and stored as a whole or decimal number with an optional minus sign.
value_confidence	float	This is the confidence value of the captured utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.

nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
done	The number capture was completed.



Note

If the number to be captured is a positive whole number and the input is via DTMF, the number can be entered using this voice element or the `Digits` voice element.

Audio Groups

Number Capture

Name (Label)	Req'd	Max1	Notes
number_initial_audio_group (Number Initial)	Yes	Yes	Played when the voice element first begins.
number_nomatch_audio_group (Number NoMatch)	No	No	Played when a nomatch event occurs.
number_noinput_audio_group (Number NoInput)	No	No	Played when a noinput event occurs.
number_help_audio_group (Number Help)	No	No	Played when the caller asked for help. If not specified, by default help is treated as a nomatch.

End

Name (Label)	Req'd	Max 1	Notes
done_audio_group (Done)	No	Yes	Played when the number capture is completed and the voice element exits with the done exit state.

Folder and Class Information

Studio Element Folder Name	Class Name
Number Capture	com.audium.server.voiceElement. number.MBasicNumber

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 36

Number_with_Confirm

The `Number_With_Confirm` voice element captures a standard number, and presents a confirmation menu allowing the caller to either accept their entry or re-enter the number. The number can be spoken or entered using the keypad. The resulting value will be stored in element data as a decimal value. The number can be negative or positive and can contain a decimal point. Using DTMF entry, however, the number is restricted to being positive and the decimal point is entered by pressing the * key. Using speech input, the number may be spoken naturally.

- [Events, on page 133](#)
- [Settings, on page 133](#)
- [Element Data, on page 135](#)
- [Exit States, on page 136](#)
- [Audio Groups, on page 136](#)
- [Folder and Class Information, on page 137](#)

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is

						thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
number_max_noinput_count (Number Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during number input capture. 0 = infinite noinputs allowed.
number_max_nomatch_count (Number Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during number input capture. 0 = infinite nomatches allowed.
confirm_max_noinput_count (Confirm Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during number input confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch Count)	int ≥ 0	Yes	true	true	3	The maximum number of nomatch events allowed during number input confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int ≥ 0	Yes	true	true	3	The maximum number of times a caller is allowed to disconfirm a captured input. 0 = infinite disconfirmations allowed.
number_confidence_level (Number Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use during number capture.
confirm_confidence_level (Confirm Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.50	The confidence level threshold to use during confirmation.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Number_With_Confirm element (the builtin number and boolean grammars) will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <code>_secureLogging</code> suffix. For example <code>nbestUtterance1_secureLogging, ****</code> .

maxnbest (Maxnbest)	int \geq 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.
------------------------	--------------	-----	------	------	---	---

Refer to the Element Data table for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
Value	string	The number captured and stored as a whole or decimal number with an optional minus sign.
value_confidence	float	This is the confidence value of the captured number utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
confirm_confidence	float	This is the confidence value of the captured confirm utterance.
nbestLength	int \geq 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
max_disconfirmed	The maximum number of disconfirmations has occurred. If the max disconfirmed count is set to 0, this exit state will never occur.
done	The number captured was confirmed.

Note If the number to be captured is a positive whole number and the input is via DTMF, the number can be entered using this voice element or the `Digits_With_Confirm` voice element.

Audio Groups

Number Capture

Name (Label)	Req'd	Max1	Notes
number_initial_audio_group (Number Initial)	Yes	Yes	Played when the voice element first begins.
number_nomatch_audio_group (Number NoMatch)	No	No	Played when a nomatch event occurs during number capture.
number_noinput_audio_group (Number NoInput)	No	No	Played when a noinput event occurs during number capture.
number_help_audio_group (Number Help)	No	No	Played when the caller asks for help during number capture. If not specified, by default help is treated as a nomatch.

Number Confirm

Name (Label)	Req'd	Max1	Notes
confirm_initial_audio_group (Confirm Initial)	Yes	Yes	Played when confirmation first begins.

confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during confirmation. The nomatch event count corresponds to the audio group count.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during confirmation. The noinput event count corresponds to the audio group count.
confirm_help_audio_group (Confirm Help)	No	No	Played when a help event occurs during confirmation. The help event count corresponds to the audio group count. If not specified, help throws a nomatch by default.
disconfirmed_audio_group (Disconfirmed)	No	No	Played after the caller disconfirms a captured number entry.

End

Name (Label)	Req'd	Max 1	Notes
yes_audio_group (Yes)	No	Yes	Played after the caller chooses the <i>yes</i> option. If not specified, no audio will be played when this option is chosen.

Folder and Class Information

Studio Element Folder Name	Class Name
Number Capture	com.audium.server.voiceElement.number. MBasicNumberWithConfirm



CHAPTER 37

Phone

The `Phone` voice element captures a phone number input from the caller. The phone number can be spoken or entered using the keypad. The captured value will be stored in element data as a string. The string may contain a number of digits and an optional character “x” to indicate a phone number with an extension. Using speech input, the entire phone number (including the extension) may be spoken in natural language. Using DTMF entry, the caller can enter an extension by pressing the * keypress followed by the extension.

- [Settings, on page 139](#)
- [Element Data, on page 140](#)
- [Exit States, on page 141](#)
- [Audio Groups, on page 141](#)
- [Folder and Class Information, on page 142](#)
- [Events, on page 142](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
collect_max_noinput_count (Phone Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during phone input capture. 0 = infinite noinputs allowed.

collect_max_nomatch_count (Phone Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during phone input capture. 0 = infinite nomatches allowed.
collect_confidence_level (Phone Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use during phone capture.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Phone element will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example nbestUtterance1_secureLogging,*****.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the following Element Data table for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
Value	string	The number captured and stored as a whole or decimal number with an optional minus sign.
value_confidence	float	This is the confidence value of the captured utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the

		n-best list and <code>nbestUtteranceX</code> holds the utterance of the last hypothesis.
<code>nbestInterpretation1</code> <code>nbestInterpretation2</code> ... <code>nbestInterpretationX</code>	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of <code>nbestInterpretationX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestInterpretation1</code> holds the interpretation of the top hypothesis in the n-best list and <code>nbestInterpretationX</code> holds the interpretation of the last hypothesis.
<code>nbestConfidence1</code> <code>nbestConfidence2</code> ... <code>nbestConfidenceX</code>	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of <code>nbestConfidenceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestConfidence1</code> holds the confidence score of the top hypothesis in the n-best list and <code>nbestConfidenceX</code> holds the confidence score of the last hypothesis.
<code>nbestInputmode1</code> <code>nbestInputmode2</code> ... <code>nbestInputmodeX</code>	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
<code>max_nomatch</code>	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
<code>max_noinput</code>	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
<code>done</code>	The phone number capture was completed.

Audio Groups

Phone Capture

Name (Label)	Req'd	Max1	Notes
<code>collect_initial_audio_group</code> (Phone Initial)	Yes	Yes	Played when the voice element first begins.

collect_noinput_audio_group (Phone NoInput)	No	No	Played when a noinput event occurs.
collect_nomatch_audio_group (Phone NoMatch)	No	No	Played when a nomatch event occurs.
collect_help_audio_group (Phone Help)	No	No	Played when the caller asked for help. If not specified, help is treated as a nomatch by default.

End

Name (Label)	Req'd	Max 1	Notes
done_audio_group (Done)	No	Yes	Played after phone capture is completed.

Folder and Class Information

Studio Element Folder Name	Class Name
Number Capture	com.audium.server.voiceElement.phone.MBasicPhone

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 38

Phone_With_Confirm

The `Phone_With_Confirm` voice element captures a phone number input from the caller, and presents a confirmation menu allowing the caller to either accept their entry or re-enter the phone number. The phone number can be spoken or entered using the keypad. The captured value will be stored in element data as a string. The string may contain a number of digits and an optional character “x” to indicate a phone number with an extension. Using speech input, the entire phone number (including the extension) may be spoken in natural language. Using DTMF entry, the caller can enter an extension by pressing the * keypress followed by the extension.

- [Settings, on page 143](#)
- [Element Data, on page 145](#)
- [Exit States, on page 145](#)
- [Audio Groups, on page 146](#)
- [Folder and Class Information, on page 147](#)
- [Events, on page 147](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
collect_max_noinput_count (Phone Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during phone input capture. 0 = infinite noinputs allowed.

collect_max_nomatch_count (Phone Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during phone input capture. 0 = infinite nomatches allowed.
confirm_max_noinput_count (Confirm Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during phone input confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during phone input confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int ≥ 0	Yes	true	false	3	The maximum number of times a caller is allowed to disconfirm a captured input. 0 = infinite disconfirmations allowed.
collect_confidence_level (Phone Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use during phone capture.
confirm_confidence_level (Confirm Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.50	The confidence level threshold to use during confirmation.
Modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Phone_With_Confirm element (the builtin phone and boolean grammars) will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example nbestUtterance1_secureLogging, ****.
Maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the Element Data table that follows for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
Value	string	The number captured and stored as a whole or decimal number with an optional minus sign.
value_confidence	float	This is the confidence value of the captured utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of <code>nbestUtteranceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestUtterance1</code> holds the utterance of the top hypothesis in the n-best list and <code>nbestUtteranceX</code> holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of <code>nbestInterpretationX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestInterpretation1</code> holds the interpretation of the top hypothesis in the n-best list and <code>nbestInterpretationX</code> holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of <code>nbestConfidenceX</code> values is equal to the <code>maxnbest</code> setting value, the actual number of these values available is determined by speech recognition at runtime, where <code>nbestConfidence1</code> holds the confidence score of the top hypothesis in the n-best list and <code>nbestConfidenceX</code> holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
------	-------

max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
max_disconfirmed	The maximum number of disconfirmations has occurred. If the max disconfirmed count is set to 0, this exit state will never occur.
done	The phone number captured was confirmed.

Audio Groups

Phone Capture

Name (Label)	Req'd	Max1	Notes
collect_initial_audio_group (Phone Initial)	Yes	Yes	Played when the voice element first begins.
collect_noinput_audio_group (Phone NoInput)	No	No	Played when a noinput event occurs.
collect_nomatch_audio_group (Phone NoMatch)	No	No	Played when a nomatch event occurs.
collect_help_audio_group (Phone Help)	No	No	Played when the caller asked for help. If not specified, help is treated as a nomatch by default.

Phone Confirm

Name (Label)	Req'd	Max1	Notes
confirm_initial_audio_group (Confirm Initial)	Yes	Yes	Played when confirmation first begins.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during confirmation. The noinput event count corresponds to the audio group count.
confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during confirmation. The nomatch event count corresponds to the audio group count.

confirm_help_audio_group (Confirm Help)	No	No	Played when a help event occurs during confirmation. The help event count corresponds to the audio group count.
disconfirmed_audio_group (Disconfirmed)	No	No	Played after the caller disconfirms a captured phone entry. Upon reaching the <code>max_disconfirmed_count</code> , the prompt content should be about exiting with the <code>max_disconfirmed</code> exit state.

End

Name (Label)	Req'd	Max 1	Notes
yes_audio_group (Yes)	No	Yes	Played after the caller chooses the <i>yes</i> option. If not specified, no audio will be played when this option is chosen.

Folder and Class Information

Studio Element Folder Name	Class Name
Number Capture	com.audium.server.voiceElement.phone.MBasicPhoneWithConfirm

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 39

POD_Add

Use the `POD_Add` custom action element to create Piece of Data (POD). You can associate the POD with a customer by using the Customer ID field. The contributor of the POD is the VXML Server hostname.

Note If the `POD_Add` element execution is successful, the customer's phone number is automatically populated in the `Context_POD_Source_Phone` data element.

- [Settings, on page 149](#)
- [Element Data, on page 150](#)
- [Session Data, on page 150](#)
- [Exit States, on page 151](#)
- [Folder and Class Information, on page 151](#)
- [Events, on page 151](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Customer ID	string	No	true	true	None	The customer identification number.
Tags	string	No	true	true	None	A comma-separated list of tags to be associated with the POD.
Field Sets	string	Yes	true	true	None	A comma-separated list of fieldsets. A fieldset is a grouping of related data elements.

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
<DATA_ELEMENT>	string	No	false	true	None	<p>User-defined data element that contains data about a POD.</p> <p>To add additional data elements, perform the following steps:</p> <ol style="list-style-type: none"> 1. Right-click Field Sets setting name or the area below. 2. Choose Add Data Element. <p>You can add, delete, or update the data elements by using these options:</p> <ul style="list-style-type: none"> • Add Data Element • Delete Data Element • Update Name

Element Data

Name	Type	Notes
pod_id	string	Contains the unique ID for the POD that was created.

Session Data

Name	Type	Notes
PodId	string	<p>Contains the unique ID for the POD if the POD creation is successful.</p> <p>When a subdialog returns, IVR subsystem populates the <code>POD.ID ECC</code> variable with <code>PodId</code>. The Call Server sends the <code>POD.ID ECC</code> variable to Unified ICM.</p>

Exit States

Name	Notes
done	The custom action element is added.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Context Service	com.audium.server.action.contextservice.AddPodAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 40

POD_Read

Use the `POD_Read` element to read PODs that were created for a customer.

- [Settings, on page 153](#)
- [Element Data, on page 154](#)
- [Exit States, on page 154](#)
- [Folder and Class Information, on page 154](#)
- [Events, on page 155](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
ID Type	String	Yes	True	False	Customer	This is a mandatory field. User can select the type of id that is used for searching the POD.
ID	String	Yes	True	True	None	This is a mandatory field. User can specify the ID to search the POD with.

Element Data

Name	Type	Notes
context_notes	string	Contains the Context_Notes data element associated with the POD.
context_pod_activity_link	string	Contains the Context_POD_Activity_Link data element associated with the POD.
context_pod_source_cust_name	string	Contains the Context_POD_Source_Cust_Name data element associated with the POD.
context_pod_source_email	string	Contains the Context_POD_Source_Email data element associated with the POD.
context_pod_source_phone	string	Contains the Context_POD_Source_Phone data element associated with the POD.
media_type	string	Contains the mediaType associated with the POD.
pod_id	string	In case of a POD_Read by Customer ID, there might be multiple PODs matching the search criteria. In that case, this contains the POD ID of the last updated POD.
search_result_as_json	string	Contains details of all the PODs that match the search criteria in JSON format.
state	string	Contains the state of the POD.
tags	string	Contains the tags associated with the POD. Multiple tags are separated by spaces.

Exit States

Name	Notes
done	The element execution is complete and the read POD operation is successful.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Context Service	com.audium.server.action.contextservice.ReadPodAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 41

POD_Update

Use the `POD_Update` custom action element to update a POD. You can update a POD by providing the `pod_id`. The update contributor of the POD is the VXML Server hostname.

If you update the tags, fieldsets, or user-defined data elements with new values, the new values are appended.

- [Settings, on page 157](#)
- [Element Data, on page 158](#)
- [Exit States, on page 158](#)
- [Folder and Class Information, on page 158](#)
- [Events, on page 159](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
POD ID	String	Yes	True	True	None	The unique ID for the POD.
Customer ID	String	No	True	True	None	An optional setting, to update the Customer ID in the POD_Update element.
Tags	String	No	True	True	None	A comma-separated list of tags to be associated with the POD.
Field Sets	String	No	True	True	None	A comma-separated list of fieldsets. A fieldset is a grouping of related data elements.

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
<DATA_ELEMENT>	String	No	False	True	None	<p>User-defined data element that contains data about a POD.</p> <p>To add more data elements, perform the following steps:</p> <ol style="list-style-type: none"> 1. Right-click Field Sets setting name or the surrounding area. 2. Choose Add Data Element. <p>You can add, delete, or update the data elements by using these options:</p> <ul style="list-style-type: none"> • Add Data Element • Delete Data Element • Update Name

Element Data

Name	Type	Notes
pod_id	string	Contains the unique ID for the POD that was updated.

Exit States

Name	Notes
done	The custom action element is updated.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco > Context Service	com.audium.server.action.contextservice.UpdatePodAction

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 42

Record

The `Record` voice element makes a recording of the caller's voice. A prompt is played to the caller then the voice element records the caller's voice until a termination key is inputted, the recording time limit has been reached, or (if the configuration specifies so) the caller hung up. An audio cue (beep) may be activated to signal to the caller that the system is ready to record the caller's voice. Different voice browsers may have varying default maximum lengths for voice recording.

The recording is sent to the `Record` element by the voice browser and is stored in an audio file in the location specified by the developer. Any pre-existing file with the same name will be overwritten. The element can be configured to produce a non-repeating filename so all recordings can be retained. The format for this filename is `audioNR.wav` where `N` is the number of milliseconds since midnight January 1, 1970 (GMT) and `R` is a random number between 1 to 1000. All recordings are saved in the WAV format.

- [Settings, on page 161](#)
- [Element Data, on page 165](#)
- [Exit States, on page 165](#)
- [Audio Groups, on page 166](#)
- [Folder and Class Information, on page 166](#)
- [Events, on page 166](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
<code>noinput_timeout</code> (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a <code>noinput</code> event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.

max_noinput_count (Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during input capture. 0 = infinite noinputs allowed.
start_with_beep (Start With Beep)	boolean	Yes	true	true	true	Whether or not to play a beep before recording begins.
terminate_on_dtmf (Terminate On DTMF)	boolean	Yes	true	true	true	Whether or not the caller can end the recording by pressing a touchtone key.
keep_recording_on_hangup (Keep Recording On Hangup)	boolean	Yes	true	true	false	Whether or not the recording is stored if the caller hung up while making the recording. Default = false
max_record_time (Max Record Time)	string	Yes	true	true	180s	<p>The maximum time (in seconds) the recording is allowed to last. Possible values are standard time designations including a positive integer followed by s (for seconds), for example, 30s. Default = 180s.</p> <p>Note Special consideration must be taken for the "ivr record memory session" setting on the gateway and the configured values for the "Max Record Time" settings in the Record element. Depending on the combination of these settings, a caller may exhaust all available memory on the gateway for their session. At which point the gateway will drop the call.</p>

						Note To prevent calls from being dropped while using the Record element, the following formula should be adhered to: "Max Record Time" in seconds * audio codec bitrate in kilobytes/second < "ivr record memory session" setting, in kilobytes. Testing should be done by increasing values for the gateway's "ivr record memory session" setting until an acceptable amount of recorded audio is accepted without exhausting the gateway's session memory (dropped calls).
final_silence (Final Silence)	string	Yes	true	true	4s	<p>The interval of silence (in seconds or milliseconds) that indicates the end of speech. Possible values are standard time designations including both a positive integer and a time unit identifier, for example, 3s (for 3 seconds) or 300ms (for 300 milliseconds). Default = 4s.</p> <p>Note For silence detection to work, you must enable Voice Activity Detection (VAD) in the gateway dial-peers. Manually remove NO VAD from the configuration script and replace it with VAD.</p> <p>Note CUBE does not support silence detection.</p>
filename (Filename)	string	No	true	true	None	The filename of the recording (without extension). If left blank, an auto-generated filename will be used.
file_type (File Type)	string enum	Yes	true	true	wav	This specifies the audio type of the file that will hold the recording. Possible values are: wav vox au other.

mime_type (Mime Type)	string	Yes	true	true	None	This specifies the MIME type of the file that will hold the recording, if file_type is set to other.
file_extension (File Extension)	string	No	true	true	None	This specifies the file extension to use for the recorded file. A file extension different from the file type can be used. For example, with a mime type of <i>vox</i> , the file extension could be set to <i>ulaw</i> .
path (Path)	string	No	true	true	None	The path to the file that will hold the recording. Either the path, ftp host, or both must be specified.
ftp_host (FTP Host)	string	No	true	true	None	The domain name of the host to ftp the recording. Either the path, ftp host, or both must be specified.
Secure (Secure)	boolean	Yes	true	true	false	Whether or not to enable Secure File Transfer protocol (SFTP). Default = false, indicates file transfer happens over FTP by default.
ftp_user (FTP User)	string	Yes	true	true	None	The user name to use while FTPing the recording, if ftp_host is set.
ftp_password (FTP Password)	string	Yes	true	true	None	The password to use while FTPing the recording, if ftp_host is set.
ftp_path (FTP Path)	string	No	true	true	None	The directory in which to FTP the recording, if ftp_host is set.
ftp_in_background (FTP In Background)	boolean	Yes	true	true	true	Whether or not the FTP is to be performed in the background, if ftp_host is set.

Note For recording, use these procedures:

- Nomatch events cannot be thrown in this voice element. Since all audio is recorded (except DTMF key presses), there is no reaction on spoken commands (including hotlinks).
- A noinput event is possible if the voice browser detects no audio once recording has started. If the input timeout has been reached, the noinput event is thrown.
- The path setting does not require a trailing slash. The voice element will determine the appropriate destination. The path may be specified in operating system specific format (for example, on Windows it might be specified as C:\directory\subdirectory\ and on UNIX it might be /usr/local/directory/).
- For a recording to be stored, you can choose either to store it locally or remotely. For locally on the VXML server itself, configure only the filename (*myfile*) and the path (*c:/recordings/*). For remotely on a ftp server, configure the filename (*myfile*) and the FTP details such as: host, user, path, and password. Once your record element is configured, determine the url to access the recording from an external system. Run a simple test by playing the recording from your web browser. Make use of the url: `http://<ftpserver>/<ftppath>/filename` . Find the correct path to play the audio file and use the same url in the audio element settings.
- If *terminate_on_DTMF* is *false* or off, recording will stop only after the voice browser reaches the input timeout.
- Some voice browsers may not accept all options provided for the *file_type* and *mime_type* settings. Check your voice browser documentation for information on supported audio types.
- It is important to ensure that VXML Server has permission to save audio files to the specified path.
- VVB silence detection while recording is not supported currently.

Element Data

Name	Type	Notes
filename	string	This stores the filename of the recording (without the path).
filepath	string	This stores the path to the file holding the recording (including the filename).
hungUpWhileRecording	boolean	This stores a <i>true</i> if the caller hung up while making the recording, <i>false</i> if not.

Exit States

Name	Notes
max_noinput	The maximum number of noinput events has occurred. If the <i>max_noinput</i> count is 0, this exit state will never occur.
done	The message was recorded.

Audio Groups

Record Capture

Name (Label)	Req'd	Max1	Notes
initial_audio_group (Initial)	Yes	Yes	Played when the voice element first begins.
noinput_audio_group (No Input)	No	No	Played when a noinput event occurs.

Folder and Class Information

Studio Element Folder Name	Class Name
Record	com.audium.server.voiceElement.record.MRecord

Events

Name (Label)	Notes
Event Handler	You can select either VXML Event or Java Exception as event handler type from the drop-down list.



CHAPTER 43

Record_With_Confirm

The `Record_With_Confirm` voice element combines the functionality of the `Record` voice element with that of the `MenuYesNo` voice element. The voice element records the caller's voice, then prompts the caller to confirm that the recording is acceptable. The caller can then accept or reject the confirmation or ask to have the message replayed. If the caller accepts the recording, the voice element saves the file just as the `Record` voice element does. This voice element contains all settings and audio groups from both the `Record` and `MenuYesNo` voice elements, however audio groups that are found in both voice elements (nomatch, noinput, and help) are now named differently for them to be distinguished.

- [Settings, on page 167](#)
- [Element Data, on page 171](#)
- [Exit States, on page 171](#)
- [Audio Groups, on page 172](#)
- [Folder and Class Information, on page 173](#)
- [Events, on page 173](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	true	both	The type of entry allowed for input during confirmation. Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.

record_max_noinput_count (Record Max NoInput Count)	int \geq 0	Yes	true	true	3	The maximum number of noinput events allowed during input capture. 0 = infinite noinputs allowed.
confirm_max_noinput_count (Confirm Max NoInput Count)	int \geq 0	Yes	true	true	3	The maximum number of noinput events allowed during confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch Count)	int \geq 0	Yes	true	true	3	The maximum number of nomatch events allowed during confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int \geq 0	Yes	true	true	3	<p>The maximum number of times a caller is allowed to reject a recording. 0 = infinite disconfirmations allowed.</p> <p>Note Special consideration must be taken for the "ivr record memory session" setting on the gateway: Each time a caller "disconfirms" a recording made while using the Record_With_Confirm element, the disaffirmed recording(s) remain in memory on the gateway. Depending on the "ivr record memory session" setting on the gateway and the configured values for the "Max Disconfirmed Count" and "Max Record Time" settings in the Record_With_Confirm element, a caller may exhaust all available memory on the gateway for their session. At which point the gateway will drop the call.</p>

						Note In general, to prevent calls from being dropped while using the Record_With_Confirm element, the following formula should be adhered to: ("Max Record Time" in seconds * audio codec bitrate in kilobytes/second) * "Max Disconfirmed Count" < "ivr record memory session" setting, in kilobytes. Testing should be done by increasing values for the gateway's "ivr record memory session" setting until an acceptable amount of audio/retries are accepted without exhausting the gateway's session memory (dropped calls).
confirm_confidence_level (Confirm Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.50	The confidence level threshold to use for the confirmation.
start_with_beep (Start With Beep)	boolean	Yes	true	true	true	Whether or not to play a beep before recording begins.
terminate_on_dtmf (Terminate On DTMF)	boolean	Yes	true	true	true	Whether or not the caller can end the recording by pressing a touchtone key.
keep_recording_on_hangup (Keep Recording On Hangup)	boolean	Yes	true	true	false	Whether or not the recording is stored if the caller hung up while making the recording or during the confirmation menu. Default = false.
max_record_time (Max Record Time)	string	Yes	true	true	180s	The maximum time (in seconds) the recording is allowed to last. Possible values are standard time designations including a positive integer followed by s (for seconds), for example, 30s. Default = 180s.
final_silence (Final Silence)	string	Yes	true	true	4s	The interval of silence (in seconds or milliseconds) that indicates the end of speech. Possible values are standard time designations including both a positive integer and a time unit

						<p>identifier, for example, 3s (for 3 seconds) or 300ms (for 300 milliseconds). Default = 4s.</p> <p>Note For silence detection to work, you must enable Voice Activity Detection (VAD) in the gateway dial-peers. Manually remove NO VAD from the configuration script and replace it with VAD.</p>
replay (Replay)	boolean	Yes	true	true	false	Adds an option to replay the confirm initial audio groups.
filename (Filename)	string	No	true	true	None	The filename of the recording (without extension). If left blank, an auto-generated filename will be used.
file_type (File Type)	string enum	Yes	true	true	wav	This specifies the audio type of the file that will hold the recording. Possible values are: wav vox au other.
mime_type (Mime Type)	string	Yes	true	true	None	This specifies the MIME type of the file that will hold the recording, if file_type is set to other.
file_extension (File Extension)	string	No	true	true	None	This specifies the file extension to use for the recorded file. A file extension different from the file type can be used. For example, with a mime type of vox, the file extension could be set to ulaw.
path (Path)	string	No	true	true	None	The path to the file that will hold the recording. Either the path, ftp host, or both must be specified.
ftp_host (FTP Host)	string	No	true	true	None	The domain name of the host to FTP the recording. Either the path, ftp host, or both must be specified.
Secure (Secure)	boolean	Yes	true	true	false	Whether or not to enable Secure File Transfer protocol (SFTP). Default = false, indicates file transfer happens over FTP by default.
ftp_user (FTP User)	string	Yes	true	true	None	The user name to use while FTPing the recording, if ftp_host is set.

ftp_password (FTP Password)	string	Yes	true	true	None	The password to use while FTPing the recording, if ftp_host is set.
ftp_path (FTP Path)	string	No	true	true	None	The directory in which to FTP the recording, if ftp_host is set.
ftp_in_background (FTP In Background)	boolean	Yes	true	true	true	Whether or not the FTP is to be performed in the background, if ftp_host is set.

Note For settings, for Record_With_Confirm, follow these procedures:

- The path setting does not require a trailing slash. The voice element will determine the appropriate destination. The path may be specified in operating system specific format (for example, on Windows it might be specified as `C:\directory\subdirectory\` and on UNIX it might be `/usr/local/directory/`).
- For a recording to be stored, you can choose either to store it locally or remotely. For locally on the VXML server itself, configure only the filename (`myfile`) and the path (`c:/recordings/`). For remotely on a ftp server, configure the filename (`myfile`) and the FTP details such as: host, user, path, and password. Once your record element is configured, determine the url to access the recording from an external system. Run a simple test by playing the recording from your web browser. Make use of the url: `http://<ftpserver>/<ftppath>/filename`. Find the correct path to play the audio file and use the same url in the audio element settings.
- If `terminate_on_DTMF` is *false* or off, recording will stop only after the voice browser reaches the input timeout.
- Some voice browsers may not accept all options provided for the `file_type` and `mime_type` settings. Check your voice browser documentation for information on supported audio types.
- It is important to ensure that VXML Server has permission to save audio files to the specified path.

Element Data

Name	Type	Notes
filename	string	This stores the filename of the recording (without the path).
filepath	string	This stores the path to the file holding the recording (including the filename).
confirm_confidence	float	This is the confidence value of the utterance for the confirmation menu.
hungUpWhileRecording	boolean	This stores a <i>true</i> if the caller hung up while making the recording or the confirmation menu, <i>false</i> if not.

Exit States

Name	Notes
------	-------

max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
max_disconfirmed	The maximum number of disconfirmations has occurred. If the max disconfirmed count is set to 0, this exit state will never occur.
done	The recorded message was confirmed.

Audio Groups

Record Capture

Name (Label)	Req'd	Max1	Notes
record_initial_audio_group (Record Initial)	Yes	Yes	Played when the voice element first begins.
record_noinput_audio_group (Record NoInput)	No	No	Played when a noinput event occurs during recording.

Record Confirm

Name (Label)	Req'd	Max1	Notes
before_confirm_audio_group (Before Confirm)	No	Yes	Played before the recording is played back. The recording will be played back after this audio group is done playing.
after_confirm_audio_group (After Confirm)	No	Yes	Played after the recording is played back. At least one of the two confirm prompts must be specified.
confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during confirmation.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during confirmation.
confirm_help_audio_group (Confirm Help)	No	No	Played when the caller asks for help during the confirmation menu. If not specified, help is treated as a nomatch by default.
max_disconfirmed_audio_group	No	Yes	Played after the caller disconfirms the recorded entry, upon reaching the <code>max_disconfirmed_count</code> .

(Max Disconfirmed)			The prompt should be about exiting with the <code>max_disconfirmed</code> exit state.
--------------------	--	--	---

Folder and Class Information

Studio Element Folder Name	Class Name
Record	com.audium.server.voiceElement.record.MRecordWithConfirm

Events

Name (Label)	Notes
Event Handler	You can select either VXML Event or Java Exception as event handler type from the drop-down list.



CHAPTER 44

Rest_Client Element

- [Rest_Client](#), on page 175

Rest_Client

The Rest_Client element provides a flexible interface in order to interact with REST endpoints. The communication between the REST client and server is made completely secure using two-way Secure Sockets Layer (SSL). The Rest_Client element permits users to send GET, POST, PUT, or DELETE requests to application servers.

For more information about Secure Socket Layer Authentication, see the *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio* at <http://www.cisco.com/c/en/us/support/customer-collaboration/unified-customer-voice-portal/products-user-guide-list.html>.

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Endpoint URL	Boolean	Yes	true	false	Blank	This settings specifies whether SNMP alert to be generated.
HTTP method		Yes			GET	Supported HTTP methods: GET (Read); PUT (Update/Replace); POST (Create) DELETE (Delete)

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Parameters		No			<i>Blank</i>	Any additional parameters will be passed along with URL. (such as specifying the response format or the amount returned). Header parameters Path parameters Query string parameters Request body parameters. Example: 'Authentication type': 'Preemptive'
Ignore Certificate Validation		Yes			true	The SSL security setting gets enabled when flag is set to false.
Require HTTP authentication		Yes			false	The http authentication (options true/false)
User Name		Yes			Blank	Username of REST end point to be accessed (available if Require HTTP auth is true).
Password		Yes			Blank	Password of REST end point to be accessed (available if Require HTTP auth is true).

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Headers		No			<i>Blank</i>	

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
						<p>The meta-data associated with the API request and response.</p> <p>Options:</p> <p>Authorization: Carries credentials containing the authentication information of the client for the resource being requested.</p> <p>WWW-Authenticate: This is sent by the server if it needs a form of authentication before it can respond with the actual resource being requested. Often sent along with a response code of 401, which means 'unauthorized'.</p> <p>Accept-Charset: This is a header which is set with the request and tells the server about which character sets are acceptable by the client.</p> <p>Content-Type: Indicates the media type of the response. Values:</p> <p>text/html - - Indicates that the request body format is HTML</p> <p>application/json - Indicates that the request body format</p>

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
						<p>is JSON.</p> <p>application/xml - Indicates that the request body format is XML.</p> <p>application/urldata - Indicates that the request body is URL encoded.</p> <p>Cache-Control: This is the cache policy defined by the server for this response, a cached response can be stored by the client and re-used till the time defined by the Cache-Control header.</p>
Body		No			Blank	
Use Proxy		Yes			false	Enable/Disable the proxy server (true/false)
Use Host		Yes			false	IP address or hostname of the HTTP proxy server
Use Port		Yes			false	Port of the HTTP proxy server

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
XPath / JSONPath		No			<i>Blank</i>	<p>XPath expressions are used in JavaScript to return the values from the XML.</p> <p>JSONPath expressions are used in JavaScript to return the values from the JSON(JavaScript Object Notation).</p> <p>For more information about Xpath/JSONPath Expression, see the User Guide for <i>Cisco Unified CVP VXML Server and Cisco Unified Call Studio</i></p>
Connect Timeout		Yes			3000 msec	HTTP request timeout
Read Timeout					5000 msec	

Element Data

Name	Type	Notes
response_body	string	This element data carries the REST response that is received from the REST end point.
status_code	string	This element data carries the REST response code received for the REST operation performed.

Exit States

Name	Notes
done	The element execution is complete.

Events

Name (Label)	Notes
Event Type	Java Exception event handler type can be selected.



CHAPTER 45

ReqICMLabel

The `ReqICMLabel` element allows a Call Studio script to pass caller input, Call Peripheral Variables, and Expanded Call Context (ECC) variables to an ICM script. The `ReqICMLabel` must be inserted into a Call Studio script as a decision element. In Call Studio, the returned ICM label contains a result, which can be used by other elements in the same application, such as the Transfer or Audio element.

After the `ReqICMLabel` exits its done path, you can retrieve the values set by the ICM script by selecting the Element Data tab for the `ReqICMLabel` element. The element data value is `{Data.Element.ReqICMLabelElement.result}`. `ReqICMLabelElement` is the name of the `ReqICMLabel` element in the Studio script. The default name for this element is `ReqICMLabel_<n>`, where `<n>` is a number. The first `ReqICMLabel` you add to the script is named `ReqICMLabel_01`, the second is named `ReqICMLabel_02`, etc. For example, if you changed `ReqICMLabel` to `GetICMLabel`, the value returned from ICM would be `{Data.Element.GetICMLabel.result}`, where `result` is the variable of the `ReqICMLabel` element that contains the ICM label.

For more information on using the `ReqICMLabel`, refer to the [Configuration Guide for Cisco Unified Customer Voice Portal](#).

- [Settings, on page 183](#)
- [Element Data, on page 184](#)
- [Session Data, on page 185](#)
- [Exit States, on page 185](#)
- [Folder and Class Information, on page 185](#)
- [Events, on page 185](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Call Peripheral Variables 1 – 10 (callvar1 – callvar10)	string	No	true	true	None	Call Peripheral Variables passed by the Studio script to the ICM Server. Each of these settings can be a maximum length of 210 characters. The ICM Server returns a name-value pair for up to 10 Call Peripheral Variables in a result. Any value that is placed

						in callvar<n> from a Call Studio script is returned unchanged, if the ICM Script does not change it.
Call Peripheral Variables Return 1 – 10 (callvarReturn1 – callvarReturn10)	string	No	true	true	None	Call Peripheral Variables created upon the return of the ICM Label request, regardless of whether or not these variables are filled by the ICM Script. The reason we need two sets of these variables is to keep reporting the To ICM Call Peripheral Variables separate from what is returned from the ICM.
FromExtVXML0 - 3 (External VXML 0 – External VXML 3)	string array	No	true	true	None	Expanded Call Context (ECC) variables passed by the Studio script to the ICM Server. Each variable is a string of name-value pairs, separated by semicolons, for up to 4 external VXML variables. Each of these settings can be a maximum length of 210 characters.
ToExtVXML0 - 4 (External VXML 0 – External VXML 4)	string array	No	true	true	None	Expanded Call Context (ECC) variables received from the ICM script. The ICM Server returns a string of name-value pairs, separated by semicolons, for up to 5 external VXML variables.
Timeout	integer	Yes	true	true	3000 (ms)	The number of milliseconds the transfer request waits for a response from the ICM Server before timing out. Note: This value can only be increased or decreased by increments of 500 ms.
caller_input (Caller Input)	string	No	true	true	None	This setting can be a maximum length of 210 characters. The value of this setting will be sent from VXML Server to ICM at runtime. Should a response from ICM be needed, the Call Peripheral Variables or ToExtVXML settings should be used.

Element Data

Name	Type	Notes
result	string	ICM Label returned from an ICM server.
callvar<n>	string	Call Peripheral Variables that the Studio scripts passes to the ICM Server. Valid Call Peripheral Variables are callvar1 – callvar10.
callvarReturn<n>	string	Call Peripheral Variables that the ICM script returns to the VXML Server. Valid Call Peripheral Variables are callvarReturn1 – callvarReturn10.

		<p>For example, if an ICM script contains call peripheral variable 3 with the string value “CompanyName=Cisco Systems, Inc”, you can access the value of CompanyName that is returned by the ICM script by using:</p> <pre>Data.Element.ReqICMLabelElement.callvarReturn3.</pre> <p>The returned value is <i>Cisco Systems, Inc.</i></p>
--	--	--

Session Data

Name	Type	Notes
name	string	<p>Value for a name-value pair contained in a <code>ToExtVXML</code> variable returned in the ICM label. You must know which name-value pairs are set in the ICM script to retrieve the correct value from the Call Studio script.</p> <p>For example, if an ICM script contains a <code>user.microapp.ToExtVXML0</code> variable with the string value <i>CustomerName=Mantle</i>, specify <code>Data.Session.CustomerName</code>. If the same ICM script contains a <code>user.microapp.ToExtVXML0</code> variable with the string value <i>BusinessType=Manufacturing</i>, you can access the customer business type returned by the ICM script by using <code>Data.Session.BusinessType</code>.</p>

Exit States

Name	Notes
done	The element execution is complete and the value was successfully retrieved.
error	The element failed to retrieve the value.

Folder and Class Information

Studio Element Folder Name	Class Name
Cisco	com.cisco.cvp.vxml.custelem.ReqICMLabel

Events

Name (Label)	Notes
Event Type	You can select Java Exception as event handler type.

The output of the Customer_Lookup element can be in JSON format . To know more about parsing the JSON Data refer to "Parsing JOSN Data" section in *User Guide for Cisco Unified CVP VXML Server and Cisco Unified Call Studio*.



CHAPTER 46

Subdialog Invoke

The `Subdialog Invoke` element initiates a subdialog invocation to another VoiceXML application, and handles passing data to and from the application. For the entire duration while a subdialog application is handling a call, the calling application waits in a dormant state for the subdialog to return. The goal of the Subdialog Invoke element is to allow voice applications to be invoked across multiple servers, as well as giving temporary control of the call to a voice application (such as flat VoiceXML and JSPs) created outside Call Studio.

- [Settings, on page 187](#)
- [Exit States, on page 188](#)
- [Folder and Class Information, on page 188](#)
- [Events, on page 188](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
subdialog_uri (Subdialog URI)	string	Yes	true	true	<i>None</i>	Specifies the URI of the subdialog to invoke. This may either be a relative or absolute URI, but must be accessible to the voice browser at runtime.
local_application (Local Application)	boolean	Yes	true	true	<i>None</i>	Specifies whether or not the subdialog application is running on the same application server as the application in which the current element appears.
parameter (Parameter)	string	No	false	true	<i>None</i>	Holds the name and value of a parameter to pass to the subdialog. The format is the name of the parameter followed by an equal sign (=) followed by the value of the parameter. For example: <i>name=John Doe</i> . The element will use the text up to the first

						equal sign as the name of the parameter and the remaining text as the value .
return_value (Return Value)	string	No	false	true	<i>None</i>	Holds the name of a return value from the subdialog. For example: <i>result</i> . The names specified here must match the variable names returned by the subdialog. Return values will be stored as element data, in a variable of the name specified here.

Exit States

Name	Notes
done	The element execution is complete.

Folder and Class Information

Studio Element Folder Name	Class Name
General	com.audium.server.voiceElement.internal.SubdialogInvoke

Events

Name (Label)	Notes
Event Type	The VXML Event handler type is available for this element.



CHAPTER 47

Subdialog Return

In most situations, the CVP Subdialog Return element (see [CVP Subdialog Return](#)) **should be used instead of this one**, to offer full compatibility with ICM. However, there is **one exception** to this. If the voice application will *only* be called by a Subdialog Invoke element (that is, never by ICM), then the Subdialog Start and Subdialog Return elements may be used instead. In this scenario, using this element allows an arbitrary number of return values to be retrieved from the subdialog, whereas the CVP Subdialog Return element allows only four.

- [Settings, on page 189](#)
- [Exit States, on page 190](#)
- [Folder and Class Information, on page 190](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
return_value (Return Value)	string	No	false	true	None	Optional return argument that holds a name/value pair to be returned to the calling application. The format should be: the name of the argument followed by an equal sign and the value of the argument. For example; <i>name=John Doe</i> . The element will take the text up to the first equal sign to be the name of the argument and the text following the equal sign to the value.



Note The following characters are not allowed in the return argument:

< > ' ' &

Exit States

Name	Notes
done	The element execution is complete.

Folder and Class Information

Studio Element Folder Name	Class Name
General	com.audium.server.voiceElement.internal.DefaultSubdialogReturnElement



CHAPTER 48

Subdialog Start

In most situations, the `CVP Subdialog Start` element (see [CVP Subdialog Start](#)) **should be used instead of this one**, to offer full compatibility with ICM. However, there is **one exception** to this. If the voice application will *only* be called by a `Subdialog Invoke` element (that is, never by ICM), then the `Subdialog Start` and `Subdialog Return` elements may be used instead.

Data can be passed to the VoiceXML application either as HTTP parameters or VoiceXML parameters (using the `<param>` tag). In the first case (that is, as HTTP parameters), Call Services will automatically create session data using the name of the data received. In the second case (that is, as VoiceXML parameters), the `Subdialog Start` element must be configured appropriately in order for the data to be available as element or session data for the duration of the call session. For each data passed as a VoiceXML parameter, the repeatable `Parameter` setting must be configured with the same exact name as the data.

- [Settings, on page 191](#)
- [Exit States, on page 192](#)
- [Folder and Class Information, on page 192](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Substitution Allowed	Default	Notes
Parameter (Parameter)	string	No	false	true	<i>None</i>	Holds the name of a parameter passed as input to the subdialog. It must match the exact value specified in the calling dialog. This is a repeatable setting, so multiple values can be specified.
Store As (Store As)	string	No	false	false	Session Data	Set to <code>Session Data</code> to store the listed parameters in Session data, or to <code>Element Data</code> to store them in Element data.

Exit States

Name	Notes
done	The element execution is complete.

Folder and Class Information

Studio Element Folder Name	Class Name
General	com.audium.server.voiceElement.internal.DefaultSubdialogStartElement



CHAPTER 49

Time

The `Time` voice element captures a time input from the caller. The time input can be entered using spoken inputs (including hours and minutes) or DTMF inputs (in the HHMM format). The captured value will be stored in element data as a five character string in the format HHMMX, where X is one of four possible values: “a” for AM, “p” for PM, “h” for a military time, or “?” for an ambiguous time. Using speech input, the time input may be spoken in natural language.

- [Settings, on page 193](#)
- [Element Data, on page 194](#)
- [Exit States, on page 195](#)
- [Audio Groups, on page 195](#)
- [Folder and Class Information, on page 196](#)
- [Events, on page 196](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: voice dtmf both.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
collect_max_noinput_count (Time Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during time input capture. 0 = infinite noinputs allowed.

collect_max_nomatch_count (Time Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during time input capture. 0 = infinite nomatches allowed.
collect_confidence_level (Time Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use during time capture.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Time element will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example nbestUtterance1_secureLogging,*****.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the Element Data table for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
Value	string	The number captured and stored as a whole or decimal number with an optional minus sign.
value_confidence	float	This is the confidence value of the captured utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.

nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
done	The time capture was completed.

Audio Groups

Time Capture

Name (Label)	Req'd	Max1	Notes
collect_initial_audio_group (Time Initial)	Yes	Yes	Played when the voice element first begins.
collect_noinput_audio_group (Time NoInput)	No	No	Played when a noinput event occurs. The noinput event count corresponds to the audio group count.

collect_nomatch_audio_group (Time NoMatch)	No	No	Played when a nomatch event occurs. The nomatch event count corresponds to the audio group count.
collect_help_audio_group (Time Help)	No	No	Played when a help event occurs. The help event count corresponds to the audio group count. If not specified, a help event is treated as nomatch.

End

Name (Label)	Req'd	Max 1	Notes
done_audio_group (Done)	No	Yes	Played after the time capture is completed. If not specified, no audio will be played.

Folder and Class Information

Studio Element Folder Name	Class Name
Date & Time	com.audium.server.voiceElement.time.MBasicTime

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 50

Time_With_Confirm

The `Time_With_Confirm` voice element captures a time input from the caller, and presents a confirmation menu allowing the caller to either accept their entry or re-enter the time. The time input can be entered using spoken inputs (including hours and minutes) or DTMF inputs (in the HHMM format). The captured value will be stored in element data as a five character string in the format HHMMX, where X is one of four possible values: “a” for AM, “p” for PM, “h” for a military time, or “?” for an ambiguous time. Using speech input, the time input may be spoken in natural language.

- [Settings, on page 197](#)
- [Element Data, on page 199](#)
- [Exit States, on page 200](#)
- [Audio Groups, on page 200](#)
- [Folder and Class Information, on page 201](#)
- [Events, on page 201](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allowed	Default	Notes
inputmode (Input Mode)	string enum	Yes	true	false	both	The type of entry allowed for input. Possible values are: <code>voice</code> <code>dtmf</code> <code>both</code> .
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
collect_max_noinput_count (Time Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during time input capture. 0 = infinite noinputs allowed.

collect_max_nomatch_count (Time Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during time input capture. 0 = infinite nomatches allowed.
confirm_max_noinput_count (Confirm Max NoInput Count)	int ≥ 0	Yes	true	true	3	The maximum number of noinput events allowed during time input confirmation. 0 = infinite noinputs allowed.
confirm_max_nomatch_count (Confirm Max NoMatch Count)	int ≥ 0	Yes	true	false	3	The maximum number of nomatch events allowed during time input confirmation. 0 = infinite nomatches allowed.
max_disconfirmed_count (Max Disconfirmed Count)	int ≥ 0	Yes	true	false	3	The maximum number of times a caller is allowed to disconfirm a captured input. 0 = infinite disconfirmations allowed.
collect_confidence_level (Time Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.40	The confidence level threshold to use during time capture.
confirm_confidence_level (Confirm Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.50	The confidence level threshold to use during confirmation.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	If set to true, only the grammars of the current Time_With_Confirm element (the builtin time and boolean grammars) will be enabled for the duration of the element. Otherwise all active grammars will be enabled.
secure_logging (Secure Logging)	boolean	Yes	true	true	false	If set to true, user DTMF input for the element is considered secure and the attributes utterance, interpretation, value, nbestUtteranceX and nbestInterpretationX are masked in VXML server logs. The format used to render secure element attributes is to add a <i>_secureLogging</i> suffix. For example nbestUtterance1_secureLogging, ****.
maxnbest (Maxnbest)	int ≥ 1	Yes	true	true	1	The maximum number of speech recognition results that can be generated per voice input.

Refer to the Element Data table for information about nbestUtteranceX and nbestInterpretationX.

Element Data

Name	Type	Notes
Value	string	The number captured and stored as a whole or decimal number with an optional minus sign.
value_confidence	float	This is the confidence value of the captured number utterance. When n-best recognition is enabled, this stores the confidence score of the top hypothesis in the n-best list.
confirm_confidence	float	This is the confidence value of the captured confirm utterance.
nbestLength	int ≥ 1	This stores the number of n-best hypotheses generated by the speech engine.
nbestUtterance1 nbestUtterance2 ... nbestUtteranceX	string	This set of element data stores the captured n-best utterances. While the maximum number of nbestUtteranceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestUtterance1 holds the utterance of the top hypothesis in the n-best list and nbestUtteranceX holds the utterance of the last hypothesis.
nbestInterpretation1 nbestInterpretation2 ... nbestInterpretationX	string	This set of element data stores the interpretations of captured n-best utterances. While the maximum number of nbestInterpretationX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestInterpretation1 holds the interpretation of the top hypothesis in the n-best list and nbestInterpretationX holds the interpretation of the last hypothesis.
nbestConfidence1 nbestConfidence2 ... nbestConfidenceX	float	This set of element data stores the confidence scores of captured n-best utterances. While the maximum number of nbestConfidenceX values is equal to the maxnbest setting value, the actual number of these values available is determined by speech recognition at runtime, where nbestConfidence1 holds the confidence score of the top hypothesis in the n-best list and nbestConfidenceX holds the confidence score of the last hypothesis.
nbestInputmode1 nbestInputmode2 ... nbestInputmodeX	string	This set of element data stores the input modes of captured n-best utterances.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
max_disconfirmed	The maximum number of disconfirmations has occurred. If the max_disconfirmed_count is set to 0, this exit state will never occur.
done	The time captured is confirmed.

Audio Groups

Time Capture

Name (Label)	Req'd	Max1	Notes
collect_initial_audio_group (Time Initial)	Yes	Yes	Played when the voice element first begins.
collect_noinput_audio_group (Time NoInput)	No	No	Played when a noinput event occurs during time input. The noinput event count corresponds to the audio group count.
collect_nomatch_audio_group (Time NoMatch)	No	No	Played when a nomatch event occurs during time input. The nomatch event count corresponds to the audio group count.
collect_help_audio_group (Time Help)	No	No	Played when a help event occurs during time input. The help event count corresponds to the audio group count. If not specified, a help event throws a nomatch event.

Time Confirm

Name (Label)	Req'd	Max1	Notes
confirm_initial_audio_group (Confirm Initial)	Yes	Yes	Played when confirmation of the captured time first begins.

confirm_nomatch_audio_group (Confirm NoMatch)	No	No	Played when a nomatch event occurs during time confirmation. The nomatch event count corresponds to the audio group count.
confirm_noinput_audio_group (Confirm NoInput)	No	No	Played when a noinput event occurs during time confirmation. The noinput event count corresponds to the audio group count.
confirm_help_audio_group (Confirm Help)	No	No	Played when a help event occurs during time confirmation. The help event count corresponds to the audio group count. If not specified, by default help throws a nomatch.
disconfirmed_audio_group (Disconfirmed)	No	No	Played after the caller disconfirms a time entry captured.

End

Name (Label)	Req'd	Max 1	Notes
yes_audio_group (Yes)	No	Yes	Played after the caller chooses the <i>yes</i> option. If not specified, no audio will be played when this option is chosen.

Folder and Class Information

Studio Element Folder Name	Class Name
Date & Time	com.audium.server.voiceElement.time.MBasicTimeWithConfirm

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.



CHAPTER 51

Transfer

The `Transfer` voice element performs a call transfer to a phone number specified by a configuration setting. Depending on how the voice browser is configured, the call transfer can be a bridge transfer or a blind transfer. For a bridge transfer, the voice browser makes an outbound call while maintaining the original call and acts as a bridge between the two calls. The advantage of this is that once the secondary call ends, the original call can still continue with the IVR. The disadvantage is that two separate phone lines are used. For a blind transfer, the voice browser makes an outbound call and when connected, links the original call to the new caller through the use of a telephony switch. At this point, the voice browser (and as a result VXML Server) is no longer in control of the call. Blind transfers involve only one line.

The `Transfer` element defines exit states for the different ways bridge transfers can end such as the person being called hung up, there was no answer, there was a busy signal, or some other phone-related error occurred. Since blind transfers take the call away from the voice browser and VXML Server, a `Transfer` element performing a blind transfer would never return an exit state. Instead, an special event would be thrown by the voice browser, caught in the root document for the call, and VXML Server would terminate the session by interrupting the `Transfer` element.

The number to transfer to can be any phone number allowed by the voice browser telephony provider (some may place restrictions on outbound dialing). Please note that different voice browsers may or may not accept certain kinds of phone numbers. Check your voice browser documentation for specific requirements and restrictions for call transfer.

- [Settings, on page 203](#)
- [Element Data, on page 205](#)
- [Exit States, on page 205](#)
- [Audio Groups, on page 205](#)
- [Folder and Class Information, on page 206](#)
- [Events, on page 206](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allowed	Default	Notes

transfer_destination (Transfer Destination)	string	Yes	true	true	None	The phone number to transfer to. It may contain non-numerical characters to allow support for phone extensions. If the destination_type is sip, make sure that the value for transfer_destination is in the SIP URI (number@domain) format.
destination_type (Destination Type)	string	No	true	true	tel	The type of transfer destination to which the voice element is to connect. Possible values are: tel sip.
connect_timeout (Connect Timeout)	string	Yes	true	true	60s	The maximum time (in seconds) that voice element is allowed to wait for an answer, before exiting with a noanswer exit state. Possible values are standard time designations including both a positive integer and a time unit s, for example, 10s (for 10 seconds). Default = 60s.
max_transfer_time (Max Transfer Time)	string	Yes	true	true	0s	The maximum duration (in seconds) that the transfer is allowed to last. Possible values are standard time designations including both a non-negative integer and a time unit s, for example, 30s (for 30 seconds). Default = 0s (means no limit). This setting only applies when bridge is set to true.
bridge (Bridge)	binary	Yes	true	true	false	Determines whether the application remains connected to the caller after the transfer is initiated. Possible values are: true false. Default = false. When set to false (that is, a blind transfer), the application redirects the caller to the callee without remaining in the connection; the transfer outcome is completely unsupervised. When set to true (that is, a bridge transfer), the application stays connected to the caller and adds the callee to the connection for the duration of the transferred call.
transfer_audio (Transfer Audio)	string	No	true	true	None	The URI location of the audio file to be played while connecting the call.
aai (Application-to-application Information)	string	No	true	true	None	A string containing Application-to-Application Information data to be sent to an application on the far-end.

Element Data

Name	Type	Notes
result	string	The value returned by the transfer field. This is dependent on the voice browser.

Exit States

Name	Notes
busy	The number was busy.
noanswer	There was no answer.
phone_error	There was some sort of phone-related error.
done	The call transfer completed successfully.

- Note** Hosting voice browsers may disable call transfers for developer accounts. You should verify with your provider that transfer is enabled for your application.
- Note** Some voice browsers use a code to indicate which call transfers will be allowed. This code appears before the phone number.
- Note** Some voice browsers support the inclusion of an extension in the phone number so that the system can transfer to a particular extension. It is up to the developer to pass this voice element a string containing the appropriate format. Check the platform specific documentation for support of extension dialing in transfer.

Audio Groups

Transfer Audio

Name (Label)	Req'd	Max1	Notes
initial_audio_group (Initial)	No	Yes	Played to introduce the transfer. If there is none, the transfer occurs immediately.
busy_audio_group (Busy)	No	Yes	Played when there is a busy signal, right before the voice element exits with the "busy" exit state.

noanswer_audio_group (No Answer)	No	Yes	Played when there is no answer, right before the voice element exits with the <i>noanswer</i> exit state.
phone_error_audio_group (Phone Error)	No	Yes	Played when there is some kind of phone-related error, right before the voice element exits with the <i>phone_error</i> exit state.

End

Name (Label)	Req'd	Max 1	Notes
done_audio_group (Done)	No	Yes	Played when the call transfer completes with the party called hanging up and the caller staying on the line.

Folder and Class Information

Studio Element Folder Name	Class Name
Call Control	com.audium.server.voiceElement.transfer.MTransfer

Events

Name (Label)	Notes
Event Handler	You can select either VXML Event or Java Exception as event handler type from the drop-down list.



VideoConnect

The VideoConnect element plays a specific video file (identified using the dialed number) from the video media server and collect digits during the video file playback.

This chapter contains the following topics:

- [Settings, on page 207](#)
- [Element Data, on page 208](#)
- [Exit States, on page 208](#)
- [Events, on page 208](#)
- [Others, on page 208](#)

Settings

Name (Label)	Type	Required	Single Setting Value	Substitution Allowed	Default	Notes
Video Media Server DN	String	Yes	True	True	None	Video Media Server Destination Number. Example: 5000. Must be a valid dialed number on Cisco UBE and the Video Media Server.
Digit Match Pattern	String	No	True	True	None	Pattern to use for matching incoming digit collection. Leave blank for no digit collection. Example: 600. Must be a valid pattern for Cisco IOS gateway. The Pattern format is same as the destination-pattern format used in IOS gateway dial-peers.
No-input Timeout	String	No	True	True	No timeout	Maximum time (seconds) to wait for caller input. Example: 15.

**Note**

- If you enter the DTMF that do not match the configured pattern. It results in an automatic retry for digit collection, so unmatched patterns does not cause the video element to exit.
- If the intent is to explicitly trap no-matches, then you can collect any single digit and return to the application.

Element Data

Element data is created *only* when the exit state is not set to *done*.

Name	Type	Notes
callerdtmf	String	The digit string value captured.
result	String	Video call outcome.

Exit States

State	Description
End_of_media	The Video played to completion and the video server gets disconnected.
Caller_input	The Caller entered a DTMF string that matched the specified digit collection pattern.
No_input	A digit collection pattern was specified, but no input was received before the input timeout occurred.
Error	This exit state is used when an error occurs and for all other unexpected termination reasons.

Events

Name (Label)	Notes
Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.

Others

Studio Element Folder	Video
Class Name	com.cisco.cvp.vxml.custelem.VideoConnect



CHAPTER 53

Web Service Elements

Along with Action and Decision elements, another way to perform backend interactions and obtain real-time data is via the Web Service element. This element leverages industry standards, such as the Web Service Definition Language (WSDL) for service definitions and SOAP for message encapsulation to provide simple, seamless interaction with remote web services.

Unlike one-off web service implementations using custom code, this element provides an intuitive graphical interface that dynamically adjusts to match each of your web services. It uses WSDL to discover required and optional settings, setting dependencies, and even valid enumerated values. Like other elements in @audiumstudio.field@, it ensures that the values you enter are of the right type, while still allowing the use of Substitution throughout.

Web Service elements provides a dynamic graphical interface for embedding web service interactions into the call flow.

This element is designed to work with the following technologies:

- WSDL 1.1 (using namespace <http://schemas.xmlsoap.org/wsdl/>)
 - Binding Styles
 - RPC/encoded
 - RPC/literal
 - Document/literal
 - Document/literal (wrapped)
- SOAP 1.1 encoding (using namespace <http://schemas.xmlsoap.org/soap/encoding/>)
 - Includes built-in support for 1-dimensional SOAP-encoded arrays that do not use href references for array items.
 - To parse n-dimensional SOAP-encoded arrays (where n is greater than 1) or href references in web service response messages, use the "Store Full Response XML" option and process the response with custom code.
- XML schemas (using namespace <http://www.w3.org/2001/XMLSchema>)
 - Includes built-in support for 1-dimensional arrays (that is, sequences).
 - To parse n-dimensional arrays (where n is greater than 1) in web service response messages, use the "Store Full Response XML" option and process the response with custom code.

Note The earlier application that contains Web Service element has to imported again to Call Studio latest version before deploying in new VXML server.

- [Exit States, on page 210](#)
- [Element Data, on page 210](#)
- [Settings, on page 211](#)
- [Configuring Request Parameters, on page 216](#)
- [Configuring Response Parameters, on page 217](#)

Exit States

Name	Description
done	This exit state is followed when the web service was successfully invoked at runtime, and responded within the time specified in the "Connection Timeout" setting.
Java Exception-error	This exit state is followed when the element encounters any error at runtime. Some examples include a web service that cannot be reached, the web service taking too long (more than the value specified in the "Connection Timeout" setting) to respond, or receiving unexpected data from the service. If this exit state is followed, refer to the @audiumcallservices.field@ logs for additional information about the cause.
fault	This exit state is only present when the loaded WSDL specifies a possible fault message for the selected operation. This exit state is followed when the web service is successfully contacted at runtime, but it responds with its fault message.

Element Data

response_xml	Only created if the "Store Full Response XML" checkbox has been checked. Holds the full XML response from the web service at runtime, for later processing by custom code or for debugging purposes.
--------------	--



Note This element may also create numerous other element or session data variables (with user-specified names), depending on the settings specified in the "Configure Response Parameters" dialog.

Settings

The Web Service element has just one Element Configuration tab, named "General". Refer to the image below and description of each setting for more information.

Figure 1: Element Configuration Tab

The screenshot shows the 'Element Configuration' dialog box for a 'Web Service' element. The 'General' tab is selected. The 'Name' field is set to 'Web Service_01'. The 'Load WSDL' section includes a 'URI' dropdown, a text input field, a 'Browse' button, and a 'Load' button. The 'Configure Web Service Call' section includes fields for 'Service', 'Port', and 'Operation', along with 'Request' and 'Response' configuration buttons. The 'Runtime Settings' section includes a 'Connect Timeout' field (set to 3), a 'Requires HTTP Authentication' checkbox, 'Username' and 'Password' fields with a 'Uses Substitution' checkbox, and 'Proxy Host' and 'Proxy Port' fields.

Element Configuration Outline

Web Service

General

Name: Web Service_01

Load WSDL

URI Browse

Load

Configure Web Service Call

Service:

Port:

Operation:

Request:

Response: ☐ Store Full Response

Runtime Settings

Connect Timeout:

☐ Requires HTTP Authentication

Username:

Password: ☐ Uses Substitution

☐ Use Proxy

Proxy Host:

Proxy Port:

Group	Name	Description
Load WSDL	WSDL Location	<p>In order for the Web Service element to be configurable, a WSDL file defining the desired web service must first be loaded. First, choose either "URI" or "File" from the drop-down, then either browse for a local file or enter a remote URI where the WSDL can be retrieved, the URI can be HTTP or HTTPS. Then, click the "Load" button to initiate @audiumstudio.field@'s download, caching, and parsing of the WSDL. Once WSDL is loaded, the other configuration options become available.</p>

Group	Name	Description
Configure Web Service Call	Service	This drop-down allows you to select which service you would like this element to invoke at runtime. Generally, WSDL files only define a single service so this list may have just one item. Each service's namespace is listed alongside it in parenthesis.
	Port	This drop-down allows you to specify which port you would like to use to connect to the web service at runtime. Each port has a name, and may define completely different connection properties than other ports. Please refer to your web service's documentation, or the WSDL file, for information about what each port represents. Note that this port list is dependent on which service is selected, and so it will update as the service is changed.
	Operation	This drop-down allows you to specify which operation you would like to execute against the previously-selected service. Note that this operation list is dependent on which port is selected, and so it will update as the port is changed.
	Request	Click the "Configure" button next to the "Request" label to bring up the "Configure Request Parameters" dialog. Using that dialog, you can specify which values to send to the web service as inputs at runtime.
	Response	Click the "Configure" button next to the "Response" label to bring up the "Configure Response Parameters" dialog. Using that dialog, you can specify in which element or session data variable each potential return value from the web service should be stored at runtime.
	Store Full Response XML	

Group	Name	Description
		Check this box if you would like the full XML response from the web service to be stored in element data at runtime, for later processing by your own custom code, or for debug purposes. Note that checking this box may be memory intensive if the response XML documents are large. Even if this checkbox has been selected, response parameter storage settings from the "Configure Response Parameter" dialog will still be used.

Group	Name	Description
Runtime Settings	Connect Timeout	<p>This setting allows you to specify how many seconds @audiumcallservices.field@ should wait for the web service socket connection to get established at runtime, before timing-out and following the "error" exit state.</p> <p>Note This is the tcp connect timeout (sun.net.client.defaultConnectTimeout) which specifies the timeout (in milliseconds) to establish the connection to the host. Depending on which service it uses it (http or ftp), it is used for the connection set up and not for the session timeout.</p>
	Requires HTTP Authentication	Check this box if you would like HTTP authentication to be used when accessing the web service at runtime.
	Username	Only available if the "Requires HTTP Authentication" checkbox has been selected. This field allows you to specify the username to use for HTTP authentication when accessing the web service at runtime.
	Password	Only available if the "Requires HTTP Authentication" checkbox has been selected. This field allows you to specify the password to use for HTTP authentication when accessing the web service at runtime.
	Use Proxy	Check this box if you would like a proxy to be used when accessing the web service at runtime.
	Proxy Host	Only available if the "Use Proxy" checkbox has been selected. This field allows you to specify the proxy host to use to access the web service at runtime.
	Proxy Port	

Group	Name	Description
		Only available if the "Use Proxy" checkbox has been selected. This field allows you to specify the proxy port to use to access the web service at runtime.

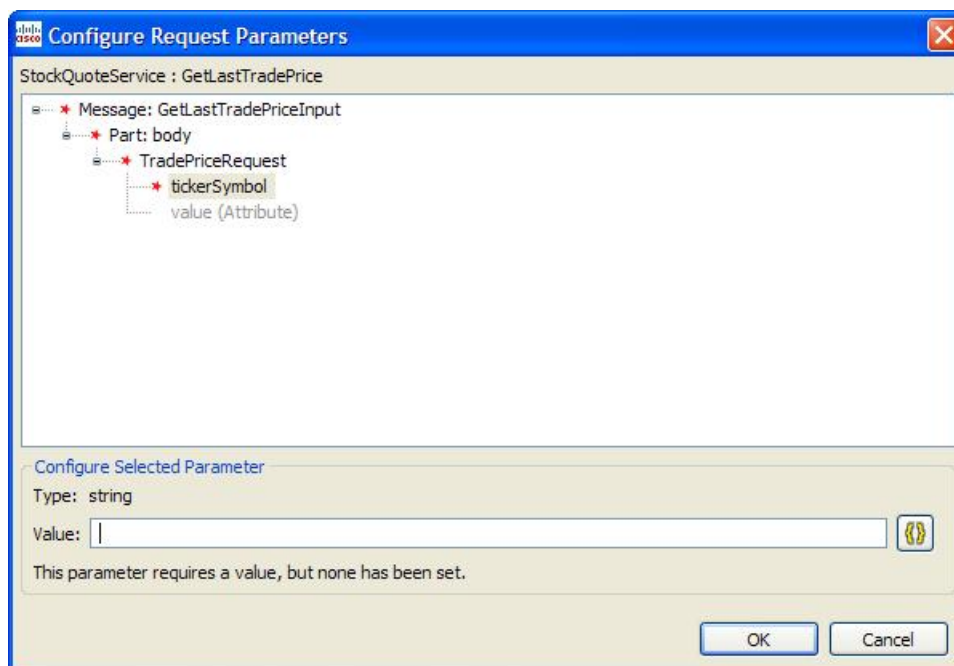
Configuring Request Parameters



Note Unified CVP Call Studio does not support SOAP Encode Schema. For all request and response parameters use the XMLSchema namespace format as listed in the [XML Schema document](#).

When the "Configure" button for Request Parameters is clicked, the following dialog is displayed:

Figure 2: Configure Request Parameters



Its contents are pre-populated with parameters that the loaded WSDL specifies. These parameters are displayed in a tree format, and use the same symbols for required and repeatable that the settings of other elements use. If a setting is optional it is greyed-out by default (like "value" in the image above), and can be added by right-clicking on it and choosing "Add PARAM_NAME":

Figure 3: Add Parameter



Each parameter has a type, such as string, integer, or float. Some parameters cannot hold a value (they will show "N/A" as their type), because they are intended to either only contain child parameters, or to act as markers. An example of a marker parameter might be "disable_logging"; if it is defined, then no logging will be performed on the service end. Only variables with a type can hold a value. The value you enter will be validated as you type it (a warning message may be displayed below the value field), and also when you validate the entire project before deploying.

If a setting is repeatable it will have its index in the list in brackets, such as the "item" parameter in the following example:

Figure 4: Repeatable Parameters



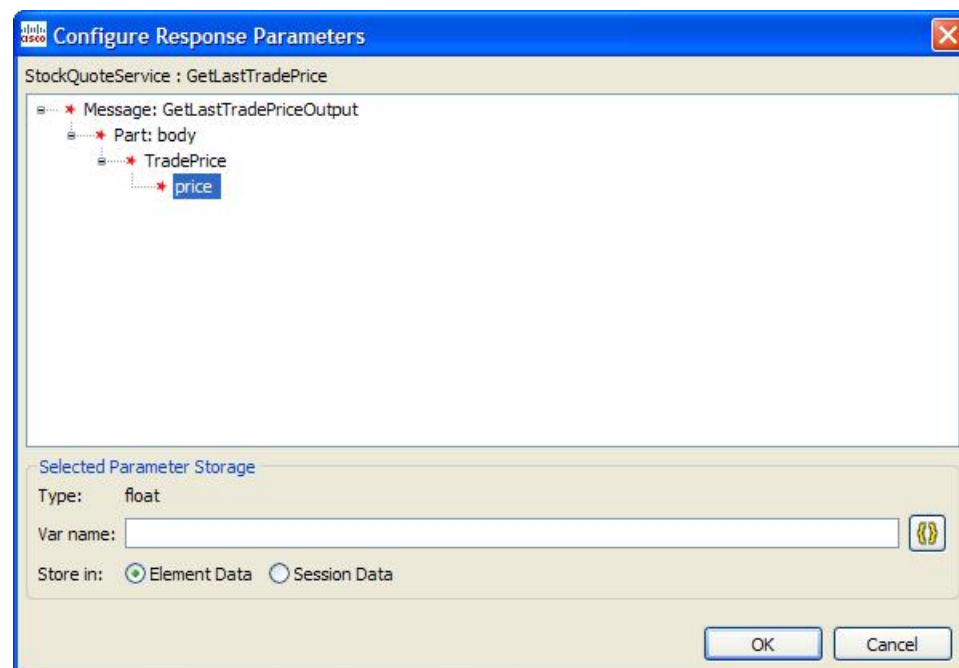
You can add additional parameters to the list by right-clicking on any list item and choosing "Add PARAM_NAME". To remove a parameter from the list, right-click on it and choose "Delete PARAM_NAME". This same functionality can be used to disable (gray-out) an optional parameter, regardless of whether it is repeatable or not.

Similar to element settings, all required parameters must be configured with a value in order for the voice application project to pass validation.

Configuring Response Parameters

Response parameters (data sent back by a web service) are handled in much the same way as request parameters. The "Configure Request Parameters" dialog is also tree-based, and it allows for parameters to be added or deleted as desired.

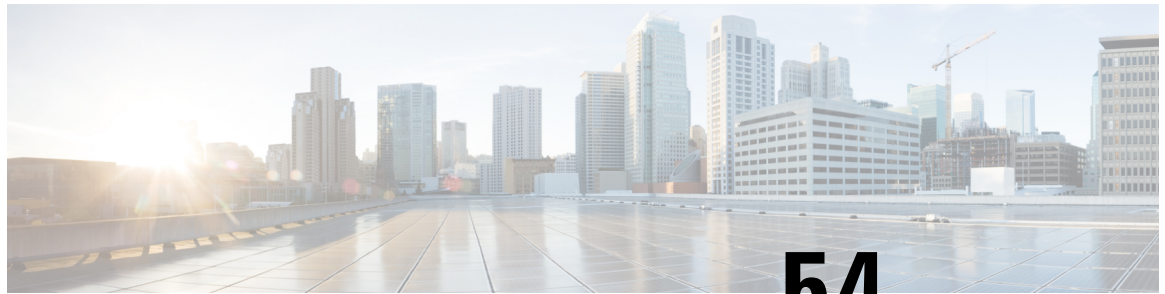
Figure 5: Configure Response Parameters



However, there are a few differences. First, you must specify whether each parameter should be stored in Element or Session data. Additionally, the text input field is used to specify the variable name to create, rather than a value to pass to the service.

No type-checking is performed in this dialog; the response parameter type is listed only for convenience.

The most significant difference between this dialog and the "Configure Request Parameters" dialog is that parameters marked as required do not need to be configured. Any parameter not configured in this dialog will simply not be stored in element or session data at runtime; if it is present in the web service's response, it will be ignored.



CHAPTER 54

Yes_No_Menu

The `Yes_No_Menu` voice element presents a yes/no menu. It can be configured to accept DTMF entry (1 for yes and 2 for no) or spoken input (*yes* or *no* and other synonymous utterances, however this is dependent on the voice browser). There is an optional feature that allows the word *replay* to be spoken (or DTMF button 3) that replays the `initial_audio_group`. The voice element uses the browser specific VoiceXML builtin grammar for the boolean field type. A separate exit state exists for the yes and no choices (there is no exit state for replay since dialog execution is still contained within the confines of the voice element).

- [Settings, on page 219](#)
- [Element Data, on page 220](#)
- [Exit States, on page 220](#)
- [Audio Groups, on page 221](#)
- [Folder and Class Information, on page 221](#)
- [Events, on page 221](#)

Settings

Name (Label)	Type	Req'd	Single Setting Value	Sub. Allowed	Default	Notes
<code>max_noinput_count</code> (Max NoInput Count)	<code>int ≥ 0</code>	Yes	<code>true</code>	<code>true</code>	3	0 = infinite noinputs allowed.
<code>max_nomatch_count</code> (Max NoMatch Count)	<code>int ≥ 0</code>	Yes	<code>true</code>	<code>true</code>	3	0 = infinite nomatches allowed.
<code>inputmode</code> (Input Mode)	<code>string</code> <code>enum</code>	Yes	<code>true</code>	<code>false</code>	<code>both</code>	The type of entry allowed for input (using speech recognition, DTMF entry, or both). Possible values are: <code>voice dtmf both</code> .

replay (Replay)	boolean	Yes	true	true	false	True adds a <i>replay</i> option which replays the initial prompt.
noinput_timeout (Noinput Timeout)	string	Yes	true	true	5s	The maximum time allowed for silence or no keypress before a noinput event is thrown. Possible values are standard time designations including both a non-negative number and a time unit, for example, 3s (for seconds) or 3000ms (for milliseconds). Default = 5s.
confidence_level (Confidence Level)	decimal (0.0 – 1.0)	Yes	true	true	0.50	The confidence level threshold to use.
modal (Disable Hotlinks)	boolean	Yes	true	true	false	Whether or not to temporarily disable all hotlink grammars (global or local) and universal grammars. If set to true, only the boolean builtin grammar will be enabled for the duration of the element. Otherwise all active grammars will be enabled.

Element Data

Name	Type	Notes
value	string	This is the value chosen by the caller. Can be: <i>yes</i> or <i>no</i> .
value_confidence	float	This is the confidence value of the utterance.

Exit States

Name	Notes
max_nomatch	The maximum number of nomatch events has occurred. If the nomatch max count is 0, this exit state will never occur.
max_noinput	The maximum number of noinput events has occurred. If the noinput max count is 0, this exit state will never occur.
yes	The utterance was recognized as <i>yes</i> .
no	The utterance was recognized as <i>no</i> .



Note The replay option, when activated, resets all the event counts (noinput and nomatch).

Audio Groups

Yes / No Capture

Name (Label)	Req'd	Max1	Notes
initial_audio_group (Initial)	Yes	Yes	Played when the voice element first begins.
nomatch_audio_group (NoMatch)	No	No	Played when a nomatch event occurs.
noinput_audio_group (NoInput)	No	No	Played when a noinput event occurs.
help_audio_group (Help)	No	No	Played when the caller asks for help. If not specified, help is treated as a nomatch event by default.

End

Name (Label)	Req'd	Max 1	Notes
yes_audio_group (Yes)	No	Yes	Played when the caller chose the yes option. If not present, no audio will play when this option is chosen.

Folder and Class Information

Studio Element Folder Name	Class Name
Menu	com.audium.server.voiceElement.menu.MYesNoMenu

Events

Name (Label)	Notes
--------------	-------

Event Type	You can select Java Exception , VXML Event , or Hotlink as event handler for this element.
------------	---



CHAPTER 55

Throw

The Throw functionality is part of event handler feature. The `Throw` element is used to raise a custom exception during a call flow execution. It can be used in a main flow or in a subflow. The `Throw` element is used to throw recently caught Java Exceptions, VXML Exception or user defined custom exceptions.

Example, `com.audium.MyException`

- [General, on page 223](#)

General

Name (Label)	Type	Req'd	Default	Notes
Event Code*	String	Yes		This is a mandatory field to be filled if you are using the Throw element in the call flow. You can define the name of the custom event or exception in this field.
Message	String	Yes		You can enter custom exception message and create a substitution tag in this field. For example, <code>{Data.Session.lastException.message}</code> .
Custom Field1 Custom Field 2 Custom Field 3	String	Yes		You can enter the value in this field from the substitutions tag, the last exception session variable will be used for the same. The last exception session variable will hold the last thrown exception.



INDEX

A

Application_Modifier [11](#)
 defined [11](#)

C

Counter [15](#)
 action element defined [15](#)
courtesy callback [29](#)
 set up defaults [29](#)
Currency element [39](#)
 defined with use of different grammars [39](#)
CVP Subdialog Return [55](#)
 when to use and exception [55](#)
CVP Subdialog Start [57](#)
 when it must be used [57](#)

D

Database element [59](#)
 four types of commands [59](#)
Digits element [75](#)
 capture a string of numbers [75](#)

F

Form element [91](#)
 purpose and multiple grammars possible [91](#)

J

JNDI Database connections [62](#)
 how to create in Tomcat [62](#)

M

Math element [118](#)
 list of operators and functions [118](#)
Menu Support element [123](#)
 described [123](#)

N

Numbers element [129](#)
 different from digits [129](#)

P

Phone element [139](#)
 capture a phone number [139](#)

R

Record element [161](#)
 record a caller's voice input [161](#)
ReqICMLabel [183](#)
 purpose [183](#)

S

Subdialog Invoke [187](#)
 purpose [187](#)
Subdialog Return [189](#)
 only time it should be used [189](#)
Subdialog Start [191](#)
 only time it should be used [191](#)

T

Time element [193](#)
 capture time input from a caller [193](#)
Tomcat [62](#)
 JNDI database connection [62](#)
 creating [62](#)
Transfer element [203](#)
 call transfer to a specific number [203](#)

V

VideoConnect [207](#)
Voice input [161](#)
 use Record element [161](#)

Y

Yes_No_Menu [219](#)

Yes_No_Menu (*continued*)
how it is used [219](#)