



Cisco Unified IP Phone Services Application Development Notes, Cisco Unified Communications Manager Release 9.1(1) and Later

First Published: January 22, 2014

Last Modified: July 15, 2016

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

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The following information is for FCC compliance of Class A devices: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

The following information is for FCC compliance of Class B devices: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If the equipment causes interference to radio or television reception, which can be determined by turning the equipment off and on, users are encouraged to try to correct the interference by using one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Modifications to this product not authorized by Cisco could void the FCC approval and negate your authority to operate the product

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: <http://www.cisco.com/go/trademarks>. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

© 2016 Cisco Systems, Inc. All rights reserved.



CONTENTS

Preface

Preface xi

Overview xi

Audience xi

Organization xii

Related Documentation xiii

Cisco Unified SIP Phone 3905 Documentation xiii

Cisco Unified IP Phone 6900 Series Documentation xiii

Cisco IP Phone 7800 Series Documentation xiii

Cisco Unified IP Phone 7900 Series Documentation xiii

Cisco IP Phone 8800 Series Documentation xiii

Cisco Wireless IP Phone 882x Series Documentation xiv

Cisco Unified IP Phone 8900 Series Documentation xiv

Cisco Unified IP Phone 9900 Series Documentation xiv

Cisco Unified Communications Manager Documentation xiv

Cisco Business Edition 5000 Documentation xiv

Guide Conventions xv

Cisco Developer Network xvi

Documentation, Support, and Security Guidelines xvi

Cisco Product Security Overview xvi

CHAPTER 1

Custom Client Services Overview 1

Services and Directories 1

Restrictions and Limitations 3

Custom Application Delays 3

Deprecated Endpoints 3

CHAPTER 2

New and Changed Information 5

New Information July 2016	5
New Information December 2015	6
New Information July 2015	6
New Information May 2015	7
New Information September 2014	7
New Information for Cisco Unified Communications Manager 10.5 (July 2014)	8
New Information for Cisco Unified Communications Manager 10.0	8
New Information for Cisco Unified Communications Manager 9.1(1)	9
New Information for Cisco Unified Communications Manager 8.5(1)	9
New Information for Cisco Unified Communications Manager 8.0(1)	10

CHAPTER 3

CiscoIPPhone XML Objects 11

Object Behavior	11
XML Object Support	11
Cisco Unified IP Phone 6900 Series XML Object Support	12
Cisco IP Phone 7800 Series XML Object Support	13
Cisco Unified IP Phone 7900 Series and Cisco IP Communicator XML Object Support	14
Cisco Unified Wireless IP Phone 7920 Series XML Object Support	15
Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support	16
XML Object Definitions	18
CiscoIPPhoneMenu	18
CiscoIPPhoneMenu Definition	18
CiscoIPPhoneText	18
CiscoIPPhoneText Definition	19
CiscoIPPhoneInput	19
CiscoIPPhoneInput Definition	20
CiscoIPPhoneDirectory	22
CiscoIPPhoneDirectory Definition	22
Custom Directories	22
CiscoIPPhoneImage	23
CiscoIPPhoneImage Definition	23
CiscoIPPhoneImage Example 1	25
CiscoIPPhoneImage Example 2	25
CiscoIPPhoneImageFile	27
CiscoIPPhoneImageFile Definition	29

CiscoIPPhoneImageFile Examples	30
CiscoIPPhoneGraphicMenu	31
CiscoIPPhoneGraphicMenu Definition	31
CiscoIPPhoneGraphicMenu WindowMode Examples	32
CiscoIPPhoneGraphicFileMenu	33
CiscoIPPhoneGraphicFileMenu Definition	34
CiscoIPPhoneGraphicFileMenu WindowMode Examples	35
CiscoIPPhoneIconMenu	36
CiscoIPPhoneIconMenu Definition	37
CiscoIPPhoneIconFileMenu	37
CiscoIPPhoneIconFileMenu Definition	38
Enhanced Icon Menu Support Feature	38
Supported IP Phones and Codecs	38
CiscoIPPhoneIconFileMenu XML Object Changes	41
Schema Definition	42
CiscoIPPhoneIconFileMenu Example	42
Valid Resource Icon Names	42
Troubleshooting CiscoIPPhoneIconFileMenu XML Objects Using Enhanced Icon Menu Support Feature	43
CiscoIPPhoneStatus	43
CiscoIPPhoneStatus Definition	44
Dynamic Application Status Window Size	45
CiscoIPPhoneStatusFile	47
CiscoIPPhoneStatusFile Definition	47
CiscoIPPhoneExecute	48
CiscoIPPhoneExecute Definition	48
CiscoIPPhoneExecute Example	49
CiscoIPPhoneResponse	49
CiscoIPPhoneResponse Definition	49
CiscoIPPhoneError	49
CiscoIPPhoneError Definition	49
Custom Softkeys	50
SoftKeyItem Definition	50
SoftKeyItem Example 1	50
SoftKeyItem Example 2	51

XML Considerations	51
Mandatory Escape Sequences	51
XML Encoding	52
XML Encoding Example	52
Application Event Handlers	52
Application Event Handler Attributes	53
Event Handler Schema	55
Event Handler Example	55

CHAPTER 4

Component APIs 57

Component API Overview	57
Supported Phone Models	57
Application Management API	60
RTP Streaming API	61
Interaction Rules with Legacy RTP URI Streams	61
Error Schema	61
RTP Streaming API Examples	62
Start Media Example	62
Stop Media Example	62
Errors and Responses	63

CHAPTER 5

Internal URI Features 65

Internal URI Overview	65
Supported URIs by Phone Model	65
Device Control URIs	70
Key	70
Key URI Format	79
Unsupported Key URIs and Alternate Options	80
Display	81
Display URI Format	81
XML Displayable Object URIs	81
SoftKey	82
SoftKey URI Format	82
QueryStringParam	83
QueryStringParam URI Format	83

Example: QueryStringParam URI in CiscoIPPhoneMenu Object	84
Example: Item Selection with Numeric Keypad Calls URI	84
QueryStringParam URI Example Discussion	84
Multimedia URIs	85
RTP Streaming	85
Interaction with Call Streaming	86
RTPRx	86
RTPRx URI Formats	87
RTPTx	87
RTPTx URI Formats	87
RTPMRx	87
RTPMRx URI Format	87
RTPMTx	88
RTPMTx URI Formats	88
Play	88
Play URI Interaction with Incoming Calls	88
Play URI Format	89
XSI Audio Path Control	89
RTP URI Format	89
Vibrate	90
Vibrate URI Format	91
Device	91
Device URI Format	91
Device URI Example	91
Device URI Error and Response	92
Telephony URIs	92
Dial	92
Dial URI Format	93
EditDial	93
EditDial URI Format	94
SendDigits	94
SendDigits URI Format	94
SendDigits Error and Response	95
Application Management URIs	95
Init	95

Init URI Format	95
Notify	96
Notify URI Format	96
Notify URI Examples	97
Application	98
App URI Format	98
App URI Error and Response	100

CHAPTER 6

HTTP Requests and Header Settings 101

HTTP Requests and Header Settings Overview	101
HTTP Client Requests (HTTP GET)	101
HTTP Server Requests (HTTP POST)	102
HTTP Header Settings	102
HTTP Refresh Setting	103
MIME Type and Other HTTP Headers	104
Audio Clips	105
Content Expiration Header Setting	105
Set-Cookie Header Setting	106
HTTP Encoding Header Setting	106
Accept Language	106
Accept Charset	107
HTTP Response Headers: Content-Type	107
IP Phone Client Capability Identification	108
x-CiscoIPPhoneModelName	108
x-CiscoIPPhoneDisplay	108
x-CiscoIPPhoneSDKVersion	108
Accept Header	109
IP Phone Information Access	109

CHAPTER 7

Troubleshooting Cisco Unified IP Phone Service Applications 111

Troubleshooting Tips	111
XML Parsing Errors	111
Error Messages	112

CHAPTER 8

Cisco IP Phone Services Software Development Kit (SDK) 115

SDK Overview	115
SDK Components	115
Sample Services Requirements	117

CHAPTER 9

IP Phone Service Administration and Subscription 119

Administration and Subscription Overview	119
Phone Service Administration Access	119
Phone Service Addition	120
IP Phone Service Parameters Definition	122
User Service Subscription	123

CHAPTER 10

DeviceListX Report 125

DeviceListX Report Overview	125
Benefits	126
Restrictions	126
Integration Considerations and Interoperability	126
Performance and Scalability	127
Security	127
Related Features and Technologies	127
Supported Platforms	127
Prerequisites	127
Message and Interface Definitions	127
DeviceList XML Object	128
DeviceList Object Example with Data	129
Troubleshooting DeviceListX Reports	129
Error Codes	129
Determine Interface Problems	130

APPENDIX A

CiscoIPPhone XML Object Quick Reference 131

APPENDIX B

Cisco Unified IP Phone Services XML Schema File 135

Updated XML Parser and Schema Enforcement	135
CiscoIPPhone.xsd	136

APPENDIX C

Device Capability Query via CTI Feature 149

Feature Description	149
Supported IP Phones and Codecs	149
XML Object Changes	152
Schema Definition	152
Request and Response Examples for getDeviceCaps	153
Troubleshooting	154
Error Handling	154



Preface

- [Overview, page xi](#)
- [Audience, page xi](#)
- [Organization, page xii](#)
- [Related Documentation, page xiii](#)
- [Guide Conventions, page xv](#)
- [Cisco Developer Network, page xvi](#)
- [Documentation, Support, and Security Guidelines, page xvi](#)

Overview

You can use this document to develop and deploy customized client services for the Cisco Unified IP Phones that support Cisco Unified Phone services.

Because of the complexity of a communications network, this guide does not provide complete and detailed information for procedures that you need to perform in Cisco Unified Communications Manager or other network devices.

For information about how to use or administer the phones, see the appropriate phone user guide, phone administration guide, and Cisco Unified Communications Manager documentation.

Related Topics

[Related Documentation, on page xiii](#)

Audience

This document provides the information needed for eXtensible Markup Language (XML) and X/Open System Interface (XSI) programmers and system administrators to develop and deploy new services.

Organization

This document contains the following sections:

Chapter	Description
Custom Client Services Overview, on page 1	Provides an overview of the Cisco Unified IP Phone services for developers.
New and Changed Information, on page 5	Provides details on the new and changed information in the XML service interface for the latest release of Cisco Unified Communication Manager.
CiscoIPPhone XML Objects, on page 11	Describes the general behavior and usage of each XML object.
Component APIs, on page 57	Describes additional application programming interfaces (API) available to the Cisco Unified IP Phones.
Internal URI Features, on page 65	Describes how to implement embedded features on Cisco Unified IP Phones.
HTTP Requests and Header Settings, on page 101	Provides a procedure on handling HTTP client requests, definitions for HTTP header elements, identifies the capabilities of the requesting IP phone client, and defines the Accept header.
Troubleshooting Cisco Unified IP Phone Service Applications, on page 111	Provides troubleshooting tips, XML parsing errors, and error messages.
Cisco IP Phone Services Software Development Kit (SDK), on page 115	Provides a list of the components used in the Cisco Unified IP Services Software Development Kit (SDK) and the sample services requirements.
IP Phone Service Administration and Subscription, on page 119	Describes how to add and administer Cisco Unified IP Phone Services through Cisco Unified Communications Manager Administration.
DeviceListX Report, on page 125	Describes how the report provides a list of the services-capable devices along with basic information about the device to identify or classify the devices based on specific criteria.
CiscoIPPhone XML Object Quick Reference, on page 131	Provides a quick reference of the CiscoIPPhone XML objects and the definitions that are associated with each object.
Cisco Unified IP Phone Services XML Schema File, on page 135	Provides the CiscoIPPhone XML Schema.

Chapter	Description
Device Capability Query via CTI Feature, on page 149	Provides information on the Device Capability Query via CTI feature.

Related Documentation

Use the following sections to obtain related information.

Cisco Unified SIP Phone 3905 Documentation

Refer to publications that are specific to your language, phone model and Cisco Unified Communications Manager release. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-sip-phone-3900-series/tsd-products-support-series-home.html>

Cisco Unified IP Phone 6900 Series Documentation

Refer to publications that are specific to your language, phone model and Cisco Unified Communications Manager release. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-6900-series/tsd-products-support-series-home.html>

Cisco IP Phone 7800 Series Documentation

Refer to publications that are specific to your language, phone model, and call control system. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-7800-series/tsd-products-support-general-information.html>

Cisco Unified IP Phone 7900 Series Documentation

See the publications that are specific to your language, phone model, and Cisco Unified Communications Manager release. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-7900-series/tsd-products-support-general-information.html>

Cisco IP Phone 8800 Series Documentation

Refer to publications that are specific to your language, phone model, and call control system. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-8800-series/tsd-products-support-series-home.html>

The Design Guides are located at the following URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-8800-series/products-implementation-design-guides-list.html>

Cisco Wireless IP Phone 882x Series Documentation

Refer to publications that are specific to your language, phone model, and call control system. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-8800-series/tsd-products-support-series-home.html>

Cisco Unified IP Phone 8900 Series Documentation

Refer to publications that are specific to your language, phone model, and Cisco Unified Communications Manager release. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-8900-series/tsd-products-support-series-home.html>

Cisco Unified IP Phone 9900 Series Documentation

Refer to publications that are specific to your language, phone model, and Cisco Unified Communications Manager release. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phones-9900-series/tsd-products-support-series-home.html>

Cisco Unified Communications Manager Documentation

See the *Cisco Unified Communications Manager Documentation Guide* and other publications that are specific to your Cisco Unified Communications Manager release. Navigate from the following documentation URL:

<http://www.cisco.com/c/en/us/support/unified-communications/unified-communications-manager-callmanager/tsd-products-support-series-home.html>

Cisco Business Edition 5000 Documentation

See the *Cisco Business Edition 5000 Documentation Guide* and other publications that are specific to your Cisco Business Edition 5000 release. Navigate from the following URL:

<http://www.cisco.com/c/en/us/support/unified-communications/business-edition-5000/tsd-products-support-series-home.html>

Guide Conventions

This document uses the following conventions:

Convention	Description
boldface font	Commands and keywords are in boldface .
<i>italic font</i>	Arguments for which you supply values are in <i>italics</i> .
[]	Elements in square brackets are optional.
{ x y z }	Alternative keywords are grouped in braces and separated by vertical bars.
[x y z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
screen font	Terminal sessions and information the system displays are in <code>screen font</code> .
input font	Information you must enter is in <code>input font</code> .
<i>italic screen font</i>	Arguments for which you supply values are in <i>italic screen font</i> .
^	The symbol ^ represents the key labeled Control - for example, the key combination ^D in a screen display means hold down the Control key while you press the D key.
< >	Nonprinting characters such as passwords are in angle brackets.



Note

Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.



Caution

Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Warnings use the following convention:

**Attention****IMPORTANT SAFETY INSTRUCTIONS**

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device. Statement 1071

SAVE THESE INSTRUCTIONS

Cisco Developer Network

The Cisco Developer Network (CDN) portal provides access to multiple Cisco technology developer interfaces and collaborative support communities. CDN also provides formalized support services for these interfaces to enable developers, customers, and partners to accelerate their development. The formalized process provides access to CDN Engineers who are an extension of the product technology engineering teams. CDN Engineers have access to the resources necessary to provide expert support in a timely manner.

The Cisco Developer Network Program is designed for businesses (IHVs and ISVs) interested in going to market with Cisco. The CDN Program enables members to develop compelling solutions that unify data, voice, video, and mobile communications on Cisco's powerful communications platform. The program also allows members to take advantage of Cisco's brand, market leadership position, and installed base to help drive positive business results for themselves and their customers.

For additional information about the CDN Program and CDN support services go to <http://developer.cisco.com/web/devservices>.

Documentation, Support, and Security Guidelines

For information on obtaining documentation, obtaining support, providing documentation feedback, reviewing security guidelines, and also recommended aliases and general Cisco documents, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

<http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html>

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS Version 2.0.

Cisco Product Security Overview

This product contains cryptographic features and is subject to United States and local country laws governing import, export, transfer, and use. Delivery of Cisco cryptographic products does not imply third-party authority to import, export, distribute, or use encryption. Importers, exporters, distributors, and users are responsible for compliance with U.S. and local country laws. By using this product you agree to comply with applicable laws and regulations. If you are unable to comply with U.S. and local laws, return this product immediately.

Further information regarding U.S. export regulations may be found at <http://www.bis.doc.gov/index.php/regulations/export-administration-regulations-ear>.



Custom Client Services Overview

- [Services and Directories, page 1](#)
- [Restrictions and Limitations, page 3](#)

Services and Directories

You can use Cisco Unified IP Phones to deploy customized client services that users can interact with using the phone keypad and display. Services deploy using HTTP from standard web servers.

Users access client services using the Services and Directories buttons or menu options (availability varies by phone model). When a user presses the Services button (or chooses the Services menu item), a menu of configured services displays. The user then chooses a service from the list, and the phone displays the service.

The following list gives typical services that might be supplied to a phone:

- Weather
- Stock information
- Contact information
- Company news
- To-do lists
- Daily schedule

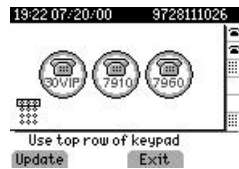
The following figure shows a sample text menu.

Figure 1: Cisco Unified IP Phone Text Menu Sample



Cisco Unified IP Phones can also display graphic menus, as shown in the following figure.

Figure 2: Graphic Menu on a Cisco Unified IP Phone Sample



Phone users can navigate a text menu using the Navigation button followed by the Select softkey, or by using the numeric keypad to enter a selection directly. Graphic menus currently do not support cursor-based navigation; users simply enter a number using the DTMF keypad.

When a menu selection is made, the Cisco Unified IP Phone acts on it by using the HTTP client to load a specific URL. The return type from this URL can be plain text or one of the CiscoIPPhone XML objects. The object loads and the user interacts with the object.

The following figures show typical displays that result from selecting a service. The first figure shows a stock quote that was generated using plain text, and the second figure displays a graphic image.

Figure 3: Plain Text Display Example



Figure 4: Graphic Image Display Example



Cisco Unified Communications Manager limits Cisco Unified IP Phone service activity to a specific Services pane in the Cisco Unified IP Phone display. A service cannot modify the top line of the phone display, which contains the time, date, and primary extension. A service cannot overwrite the bottom line of the display, which contains softkey definitions. The pane that displays the service sits flush with the left side of the display, and enough of the right side of the display remains intact to ensure that users can see the status of phone lines.



Note

HTML Disclaimer: Phone service developers must take into consideration that the phone is not a web browser and cannot parse HTML. Although content is delivered to the phone through HTTP messages using a web server, keep in mind that the content is not HTML. All content comes to the phone either as plain text or packaged in proprietary XML wrappers.

Restrictions and Limitations

Custom Application Delays

When users interact with custom phone applications, they may experience unusually long phone response delays under the following conditions:

- Heavy data usage when there are concurrent phone calls or other HTTP services (for example, Extension Mobility or Extension Mobility Cross Cluster).
- Repeated pushing of large files to the phones (for example, pushing large image files every second).

**Note**

The response time also varies between different phone models due to internal processing limitations.

Administrators should configure the external services for the best application performance. For more information, see [IP Phone Service Administration and Subscription](#), on page 119.

Deprecated Endpoints

As of Cisco Unified Communications Manager Firmware Release 11.5, the following phones are not supported:

- Cisco IP Phone 12 SP+ and related models
- Cisco IP Phone 30 VIP and related models
- Cisco Unified IP Phone 7902
- Cisco Unified IP Phone 7905
- Cisco Unified IP Phone 7910
- Cisco Unified IP Phone 7910SW
- Cisco Unified IP Phone 7912
- Cisco Unified Wireless IP Phone 7920
- Cisco Unified IP Conference Station 7935

If you use any of these phone models on an older release of Cisco Unified Communications Manager and you upgrade to Release 11.5, the phone will not work after the upgrade completes.



New and Changed Information

- [New Information July 2016, page 5](#)
- [New Information December 2015, page 6](#)
- [New Information July 2015, page 6](#)
- [New Information May 2015, page 7](#)
- [New Information September 2014, page 7](#)
- [New Information for Cisco Unified Communications Manager 10.5 \(July 2014\), page 8](#)
- [New Information for Cisco Unified Communications Manager 10.0, page 8](#)
- [New Information for Cisco Unified Communications Manager 9.1\(1\), page 9](#)
- [New Information for Cisco Unified Communications Manager 8.5\(1\), page 9](#)
- [New Information for Cisco Unified Communications Manager 8.0\(1\), page 10](#)

New Information July 2016

The following updates were made.

Section	Reason
CiscoIPPhoneInput Definition, on page 20	Updated to indicate that the Cisco IP Phone 7800 Series supports the HTTP POST method.
SoftKeyItem Definition, on page 50	Updated to give the expected contents of the Position field.
CiscoIPPhoneStatus, on page 43 CiscoIPPhoneStatusFile, on page 47	Updated to indicate the object is not supported for the Cisco Unified Wireless IP Phone 7925G, 7925G-EX, and 7926G.

Section	Reason
Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support , on page 16 CiscoIPPhoneImage Definition , on page 23 CiscoIPPhoneImageFile , on page 27 CiscoIPPhoneGraphicFileMenu , on page 33 Supported IP Phones and Codecs , on page 38 CiscoIPPhoneStatus , on page 43 CiscoIPPhoneStatusFile , on page 47 SoftKeyItem Definition , on page 50 Supported Phone Models , on page 57 Supported URIs by Phone Model , on page 65 Key , on page 70 XSI Audio Path Control , on page 89 Updated XML Parser and Schema Enforcement , on page 135 Supported IP Phones and Codecs , on page 149	Updated for Cisco Wireless IP Phone 8821.
Deprecated Endpoints , on page 3	Added list of phones deprecated for Cisco Unified Communications Manager Release 11.5. All sections in the book that mention deprecated phones were also updated.
Cisco Unified IP Phone 7900 Series and Cisco IP Communicator XML Object Support , on page 14	Moved the 7920 Series phone information into a new section: Cisco Unified Wireless IP Phone 7920 Series XML Object Support , on page 15

New Information December 2015

The following updates were made.

- The section [CiscoIPPhoneInput Definition](#), on page 20 was updated to indicate that the Cisco IP Phone 8800 Series support HTTP Post.
- The section [Key](#), on page 70 was updated to indicate that the Cisco IP Phone 8800 Series supports Key:Directores starting with Firmware Release 11.0

New Information July 2015

Added information for the Cisco IP Phone 8845 and 8865 in the following sections:

- [Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support](#), on page 16
- [CiscoIPPhoneImageFile](#), on page 27
- [Supported IP Phones and Codecs](#), on page 38
- [Dynamic Application Status Window Size](#), on page 45
- [Supported Phone Models](#), on page 57
- [Supported URIs by Phone Model](#), on page 65
- [Key](#), on page 70
- [Content Expiration Header Setting](#), on page 105
- [Supported IP Phones and Codecs](#), on page 149

New Information May 2015

Added information for the Cisco IP Phone 7811 and Cisco IP Phone 8851NR in the following sections:

- [Cisco IP Phone 7800 Series XML Object Support](#), on page 13
- [Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support](#), on page 16
- [CiscoIPPhoneImageFile](#), on page 27
- [Supported IP Phones and Codecs](#), on page 38
- [Dynamic Application Status Window Size](#), on page 45
- [Supported Phone Models](#), on page 57
- [Supported URIs by Phone Model](#), on page 65
- [Key](#), on page 70
- [Dial URI Format](#), on page 93
- [Content Expiration Header Setting](#), on page 105
- [Updated XML Parser and Schema Enforcement](#), on page 135
- [Supported IP Phones and Codecs](#), on page 149

New Information September 2014

Added information for the Cisco IP Phone 8811 in the following sections:

- [Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support](#), on page 16
- [CiscoIPPhoneImageFile](#), on page 27
- [Supported IP Phones and Codecs](#), on page 38
- [Dynamic Application Status Window Size](#), on page 45
- [Supported Phone Models](#), on page 57

- [Supported URIs by Phone Model, on page 65](#)
- [Key, on page 70](#)
- [Content Expiration Header Setting, on page 105](#)
- [Supported IP Phones and Codecs, on page 149](#)

Updated the following sections to correct the Cisco Unified IP Phone 8941 and 8945 support:

- [Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support, on page 16](#)
- [Supported URIs by Phone Model, on page 65](#)

New Information for Cisco Unified Communications Manager 10.5 (July 2014)

Added information on Cisco IP Phones 8841, 8851, and 8861 in the following sections:

- [XML Object Support, on page 11](#)
- [Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support, on page 16](#)
- [CiscoIPPhoneImageFile, on page 27](#)
- [Dynamic Application Status Window Size, on page 45](#)
- [Supported IP Phones and Codecs, on page 38](#)
- [Supported Phone Models, on page 57](#)
- [Supported URIs by Phone Model, on page 65](#)
- [Key, on page 70](#)
- [Supported IP Phones and Codecs, on page 149](#)

Added the following new section:

- [Restrictions and Limitations, on page 3](#)

Updated the following sections to address other deficiencies:

- [Key, on page 70](#)
- [Unsupported Key URIs and Alternate Options, on page 80](#)
- [Content Expiration Header Setting, on page 105](#)

New Information for Cisco Unified Communications Manager 10.0

Added information on Cisco IP Phones 7821, 7841, and 7861 in the following sections:

- [XML Object Support](#), on page 11
- [CiscoIPPhoneImageFile](#), on page 27
- [Supported Phone Models](#), on page 57
- [Key](#), on page 70
- [Supported IP Phones and Codecs](#), on page 38

New Information for Cisco Unified Communications Manager 9.1(1)

- Added information on Cisco Unified IP Phones 8941 and 8945 in the following sections:
 - [Object Behavior](#), on page 11
 - [CiscoIPPhoneImageFile](#), on page 27
 - [Supported IP Phones and Codecs](#), on page 38
 - [Dynamic Application Status Window Size](#), on page 45
 - [Supported Phone Models](#), on page 57
 - [Supported URIs by Phone Model](#), on page 65
 - [Key](#), on page 70
- Added information about the “s” parameter for the RTPRx and the RTPMRx URIs.
 - [RTPRx](#) , on page 86
 - [RTPMRx](#) , on page 87

New Information for Cisco Unified Communications Manager 8.5(1)

- Added information on WindowMode attribute in the following sections:
 - [CiscoIPPhoneImage](#), on page 23
 - [CiscoIPPhoneImageFile](#), on page 27
 - [CiscoIPPhoneGraphicMenu](#), on page 31
 - [CiscoIPPhoneGraphicFileMenu](#), on page 33
- Added information on the new Cisco Unified IP Phone 6945.

New Information for Cisco Unified Communications Manager 8.0(1)

- Added information on support for CiscoIPPhoneStatus and CiscoIPPhoneStatusFile XML objects in [Object Behavior](#), on page 11 on Cisco Unified IP Phones 8961, 9951, and 9971.
- Added information on the new [Device](#), on page 91.
- Added information on Cisco Unified IP Phone 7937 in the following sections:
 - [Object Behavior](#), on page 11
 - [Application Event Handlers](#), on page 52
 - [Component APIs](#), on page 57
 - [Supported URIs by Phone Model](#), on page 65
- Added information on [Enhanced Icon Menu Support Feature](#), on page 38.
- Added information on [Device Capability Query via CTI Feature](#), on page 149
- Added information on Cisco Wireless IP Phone 7925G-EX and 7926G in the following sections:
 - [Object Behavior](#), on page 11
 - [CiscoIPPhoneImageFile](#), on page 27
 - [Enhanced Icon Menu Support Feature](#), on page 38
 - [Application Event Handlers](#), on page 52
 - [Component APIs](#), on page 57
 - [Supported URIs by Phone Model](#), on page 65
 - [Device Control URIs](#), on page 70
 - [Vibrate](#), on page 90
 - [Updated XML Parser and Schema Enforcement](#), on page 135



CiscoIPPhone XML Objects

- [Object Behavior, page 11](#)
- [XML Object Support, page 11](#)
- [XML Object Definitions, page 18](#)
- [Custom Softkeys, page 50](#)
- [XML Considerations , page 51](#)
- [Application Event Handlers, page 52](#)

Object Behavior

You can create interactive service applications when you understand the XML objects that are defined for Cisco Unified IP Phones and the behavior that each object generates.

When a phone loads an XML page, the phone does not have any concept of a service state. IP phones can use HTTP to load a page of content in many different places, starting when the user presses the Services button. Regardless of what causes the phone to load a page, the phone always behaves appropriately after it loads a page.

Appropriate behavior depends solely on the type of data that has been delivered in the page. The web server must deliver the XML pages with a MIME type of text/xml. However, the exact mechanism required varies according to the type of web server that you use and the server-side mechanism that you use to create your pages (for example, if you use static files, JavaScript, or CGI).

Related Topics

[HTTP Requests and Header Settings, on page 101](#)

XML Object Support

The following sections describe the supported XML objects by phone model families. Before creating a service for a particular phone model, check to make sure that the XML object you want to use is supported.

Cisco Unified IP Phone 6900 Series XML Object Support

The following table shows the supported XML objects for the Cisco Unified IP Phone 6900 Series.


Note

Cisco Unified IP Phones 6901 and 6911 do not support XML objects.

Table 1: XML Objects Supported by Cisco Unified IP Phone Services SDK for Cisco Unified IP Phone 6900 Series

XML object	6901, 6911	6921, 6941, 6945, 6961
CiscoIPPhoneMenu	Not supported	Supported
CiscoIPPhoneText	Not supported	Supported
CiscoIPPhoneInput	Not supported	Supported
CiscoIPPhoneDirectory	Not supported	Supported
CiscoIPPhoneImage	Not supported	Not supported (see note 1)
CiscoIPPhoneImageFile	Not supported	Not supported (see note 1)
CiscoIPPhoneGraphicMenu	Not supported	Not supported (see note 1)
CiscoIPPhoneGraphicFileMenu	Not supported	Not supported (see note 1)
CiscoIPPhoneIconMenu	Not supported	Supported (see note 2)
CiscoIPPhoneIconFileMenu	Not supported	Supported
CiscoIPPhoneStatus	Not supported	Not supported
CiscoIPPhoneStatusFile	Not supported	Not supported
CiscoIPPhoneExecute	Not supported	Supported
CiscoIPPhoneResponse	Not supported	Supported
CiscoIPPhoneError	Not supported	Supported

**Note**

- 1 The Cisco Unified IP Phones 6921, 6941, 6945, and 6961 do not support CiscoIPPhoneGraphicFileMenu, CiscoIPPhoneGraphicMenu, CiscoIPPhoneImageFile, and CiscoIPPhoneImage because these phones use a monochrome LCD.
- 2 The Cisco Unified IP Phones 6921, 6941, 6945, and 6961 do not support icons; therefore, all icons are ignored and do not display.

Cisco IP Phone 7800 Series XML Object Support

The following table shows the supported XML objects for the Cisco IP Phone 7800 Series.

Table 2: XML Objects Supported by Cisco Unified IP Phone Services SDK for Cisco IP Phone 7800 Series

XML object	7811, 7821, 7841, and 7861
CiscoIPPhoneMenu	Supported
CiscoIPPhoneText	Supported
CiscoIPPhoneInput	Supported
CiscoIPPhoneDirectory	Supported
CiscoIPPhoneImage	Not supported
CiscoIPPhoneImageFile	Not supported
CiscoIPPhoneGraphicMenu	Not supported
CiscoIPPhoneGraphicFileMenu	Not supported
CiscoIPPhoneIconMenu	Supported
CiscoIPPhoneIconFileMenu	Supported
CiscoIPPhoneStatus	Not supported
CiscoIPPhoneStatusFile	Not supported
CiscoIPPhoneExecute	Supported
CiscoIPPhoneResponse	Supported
CiscoIPPhoneError	Supported

Cisco Unified IP Phone 7900 Series and Cisco IP Communicator XML Object Support

The following table shows the supported XML objects for the Cisco Unified IP Phone 7900 Series and the Cisco IP Communicator.


Note

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Table 3: XML Objects Supported by Cisco Unified IP Phone Services SDK for Cisco Unified IP Phone 7900 Series and Cisco IP Communicator

XML object	7905G, 7906G, 7911G, 7912G, 7931G	7937G	7940G, 7960G	7941G, 7941G-GE, 7942G, 7945G, 7961G, 7961G-GE, 7962G, 7965G, 7970G, 7971G-GE, 7975G, IP Communicator
CiscoIPPhoneMenu	Supported	Supported	Supported	Supported
CiscoIPPhoneText	Supported	Supported	Supported	Supported
CiscoIPPhoneInput	Supported	Supported	Supported	Supported
CiscoIPPhoneDirectory	Supported	Supported	Supported	Supported
CiscoIPPhoneImage	Not supported	Supported	Supported	Supported
CiscoIPPhoneImageFile	Not supported	Not supported	Not supported	Supported
CiscoIPPhoneGraphicMenu	Not supported	Supported	Not supported	Supported
CiscoIPPhoneGraphicFileMenu	Not supported	Not supported	Not supported	Supported
CiscoIPPhoneIconMenu	Supported (see note 1)	Supported	Supported	Supported
CiscoIPPhoneIconFileMenu	Not supported	Not supported	Not supported	Supported (see note 2)
CiscoIPPhoneStatus	Not supported	Not supported	Supported	Supported
CiscoIPPhoneStatusFile	Not supported	Not supported	Not supported	Supported (see note 3)
CiscoIPPhoneExecute	Supported	Supported	Supported	Supported

XML object	7905G, 7906G, 7911G, 7912G, 7931G	7937G	7940G, 7960G	7941G, 7941G-GE, 7942G, 7945G, 7961G, 7961G-GE, 7962G, 7965G, 7970G, 7971G-GE, 7975G, IP Communicator
CiscoIPPhoneResponse	Supported	Supported	Supported	Supported
CiscoIPPhoneError	Supported	Supported	Supported	Supported

**Note**

- 1 The Cisco Unified IP Phones 7905G and 7912G do not support CIP images; therefore, all icons get ignored and do not display.
- 2 The Cisco Unified IP Phones 7970G and 7971G-GE require firmware version 7.1(2) or higher to support this object, and Cisco IP Communicator requires software version 2.01 or higher.

Related Topics

[Deprecated Endpoints, on page 3](#)

Cisco Unified Wireless IP Phone 7920 Series XML Object Support

The following table shows the supported XML objects for the Cisco Unified Wireless IP Phone 7920 Series.

**Note**

The Cisco Unified Wireless IP Phone 7920 is deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phone still works on previous versions of Cisco Unified Communications Manager

Table 4: XML Objects Supported by Cisco Unified IP Phone Services SDK for Cisco Unified Wireless IP Phone 7920 Series

XML object	7920	7921G, 7925G, 7925G-EX, 7926G
CiscoIPPhoneMenu	Supported	Supported
CiscoIPPhoneText	Supported	Supported
CiscoIPPhoneInput	Supported	Supported
CiscoIPPhoneDirectory	Supported	Supported
CiscoIPPhoneImage	Supported (see note 1)	Supported
CiscoIPPhoneImageFile	Not supported	Supported

XML object	7920	7921G, 7925G, 7925G-EX, 7926G
CiscoIPPhoneGraphicMenu	Supported (see note 1)	Supported
CiscoIPPhoneGraphicFileMenu	Not supported	Supported
CiscoIPPhoneIconMenu	Supported	Supported
CiscoIPPhoneIconFileMenu	Not supported	Supported
CiscoIPPhoneStatus	Not supported	Not supported
CiscoIPPhoneStatusFile	Not supported	Not supported
CiscoIPPhoneExecute	Supported (see note 2)	Supported
CiscoIPPhoneResponse	Supported	Supported
CiscoIPPhoneError	Supported	Supported

**Note**

- 1 The Cisco Unified IP Phone 7920 has only a 128-by-59 display with 2 grayscale images clipping the graphic equally on both sides and providing vertical scrolling. When an image with 4 grayscale settings occurs (<Depth>2</Depth>), the phone equally splits them into 2 grayscale settings (0-1 get treated as 0 and 2-3 get treated as 1).
- 2 The Cisco Unified IP Phone 7920 does not support Priority 1 when on a call.

Related Topics

[Deprecated Endpoints, on page 3](#)

Cisco Unified IP Phone 8800, 8900, and 9900 Series XML Object Support

The following table shows the supported XML objects for the Cisco Unified IP Phone 8800, 8900, and 9900 Series.

Table 5: XML Objects Supported by Cisco Unified IP Phone Services SDK for Cisco Unified IP Phone 8800, 8900, and 9900 Series

XML object	8821	8831	8811, 8841, 8845, 8851, 8851NR, 8861, 8865 (see note 4)	8941, 8945	8961, 9951, 9971
CiscoIPPhoneMenu	Supported	Supported	Supported	Supported	Supported

XML object	8821	8831	8811, 8841, 8845, 8851, 8851NR, 8861, 8865 (see note 4)	8941, 8945	8961, 9951, 9971
CiscoIPPhoneText	Supported	Supported	Supported	Supported	Supported
CiscoIPPhoneInput	Supported	Supported	Supported	Supported	Supported
CiscoIPPhoneDirectory	Supported	Supported	Supported	Supported	Supported
CiscoIPPhoneImage	Supported	Not supported	Supported	Supported	Supported
CiscoIPPhoneImageFile	Supported	Not supported	Supported	Supported (see note 1)	Supported
CiscoIPPhoneGraphicMenu	Supported	Not supported	Supported	Supported	Supported
CiscoIPPhoneGraphicFileMenu	Not supported	Not supported	Supported	Supported	Supported
CiscoIPPhoneIconMenu	Supported	Supported	Supported	Supported	Supported
CiscoIPPhoneIconFileMenu	Supported	Not supported	Supported	Supported (see note 2)	Supported
CiscoIPPhoneStatus	Not supported	Not supported	Supported	Supported (see note 3)	Supported
CiscoIPPhoneStatusFile	Not supported	Not supported	Supported	Supported (see note 3)	Supported
CiscoIPPhoneExecute	Supported	Supported	Supported	Not supported	Supported
CiscoIPPhoneResponse	Supported	Supported	Supported	Supported	Supported
CiscoIPPhoneError	Supported	Supported	Supported	Supported	Supported

**Note**

- 1 The Cisco Unified IP Phones 8941 and 8945 supports display image sizes and color depths: 498x289x24 color.
- 2 The Cisco Unified IP Phones 8941 and 8945 do not support Enhanced Icon Menu.
- 3 The Cisco Unified IP Phones 8941 and 8945 require Firmware Release 9.3(1) or later.
- 4 The Cisco IP Phones 8841, 8851, and 8861 require Firmware Release 10.2(1) or later. The Cisco IP Phone 8811 requires Firmware Release 10.2(2) or later. The Cisco IP Phone 8851NR requires Firmware Release 10.3(1) or later. The Cisco IP Phone 8845 and 8865 require Firmware Release 10.3(2) or later.

XML Object Definitions

The following sections provide definitions and descriptions of each CiscoIPPhone XML object.

CiscoIPPhoneMenu

A menu on the phone contains a list of text items, one per line. Users choose individual menu items using the same mechanisms that are used for built-in menus in the phone.

When a menu loads, the phone behaves the same as for built-in phone menus. The user navigates through the list of menu items and eventually chooses one using either the Select softkey or the DTMF keys.

After the user chooses a menu option, the phone generates an HTTP request for the page with the URL or executes the uniform resource identifiers (URIs) that are associated with the menu item.

Related Topics

[Custom Client Services Overview, on page 1](#)

[CiscoIPPhoneGraphicMenu, on page 31](#)

CiscoIPPhoneMenu Definition

```
<CiscoIPPhoneMenu>
<Title>Title text goes here</Title>
<Prompt>Prompt text goes here</Prompt>
<MenuItem>
<Name>The name of each menu item</Name>
<URL>The URL associated with the menu item</URL>
</MenuItem>
</CiscoIPPhoneMenu>
```



Note

- The *Name* field under the *<MenuItem>* supports a maximum of 64 characters. This field can also accept two carriage returns to allow the MenuItem name to span three lines on the display.
- The Cisco Unified IP Phone 6900 Series do not display the *Title* and *Prompt* menu fields at the same time. If both *Title* and *Prompt* fields are defined at the same time, then these phones display only the *Prompt* field.

The XML format allows you to specify a *Title* and *Prompt* that are used for the entire menu, followed by a sequence of *<MenuItem>* objects. IP phones allow a maximum of 100 MenuItems. Each *<MenuItem>* includes a *Name* and an associated *URL*.

CiscoIPPhoneText

The *CiscoIPPhoneText* XML object displays ordinary 8-bit ASCII text on the phone display. The *<Text>* message must not contain any control characters, except for carriage returns, line feeds, and tabs. The IP phone firmware controls all other pagination and word wrap issues.

**Note**

Cisco Unified IP Phones support the full ISO 8859-1 (Latin 1) and Shift_JIS character sets.

CiscoIPPhoneText Definition

```
<CiscoIPPhoneText>
<Title>Title text goes here</Title>
<Prompt>The prompt text goes here</Prompt>
<Text>The text to be displayed as the message body goes here</Text>
</CiscoIPPhoneText>
```

**Note**

The Cisco Unified IP Phone 6900 Series do not display the *Title* and *Prompt* menu fields at the same time. If both *Title* and *Prompt* fields are defined at the same time, then these phones display only the *Prompt* field.

Two optional fields can appear in the XML message:

- The first optional field, *Title*, defines text that displays at the top of the display page. If a *Title* is not specified, the *Name* field of the last chosen *MenuItem* displays in the *Title* field.
- The second optional field, *Prompt*, defines text that displays at the bottom of the display page. If a *Prompt* is not specified, Cisco Unified Communications Manager clears the prompt area of the display pane.

Many XML objects that are described in this document also have *Title* and *Prompt* fields. These fields normally behave identically to behavior described in this section.

CiscoIPPhoneInput

When an IP phone receives an XML object of type *CiscoIPPhoneInput*, it constructs an input form and displays it. The user enters data into each input item and sends the parameters to the target URL. The following figure shows a sample display that is receiving input from a user.

Figure 5: Sample user input display



Many XML objects that are described in this document also have *Title* and *Prompt* fields. These fields normally behave identically to behavior described in this section.

**Note**

Non-XML Text: This document only describes the supported CiscoIPPhone XML objects. You can also deliver plain text using HTTP. Pages that are delivered as MIME type text/html behave exactly the same as XML pages of type *CiscoIPPhoneText*. One important difference is that you cannot include a title or prompt.

**Note**

Keypad navigation: IP phones allow navigation to a specific line in a menu by pressing numeric DTMF keys. When a menu is on the display, the number for selecting the menu is on the left.

When normal text displays, the numbers do not display on the left side of the screen, but the navigation capability still exists. A carefully written text service display can take advantage of this capability.

During text entry, the phones display softkeys to assist users with text entry. Users can navigate between fields with the vertical scroll button that is used to navigate menus.

CiscoIPPhoneInput Definition

```
<CiscoIPPhoneInput>
<Title>Directory title goes here</Title>
<Prompt>Prompt text goes here</Prompt>
<URL method="post">The target URL for the completed input goes here</URL>
<InputItem>
<DisplayName>Name of the input field to display</DisplayName>
<QueryStringParam>The parameter to be added to the target URL</QueryStringParam>
<DefaultValue>The default display name</DefaultValue>
<InputFlags>The flag specifying the type of allowable input</InputFlags>
</InputItem>
</CiscoIPPhoneInput>
```

**Note**

The Cisco Unified IP Phone 6900 Series do not display the *Title* and *Prompt* menu fields at the same time. If both *Title* and *Prompt* fields are defined at the same time, then these phones display only the *Prompt* field.

The *Title* and *Prompt* tags in the object define text that are used in the same way as the identical fields in the other CiscoIPPhone XML objects.

The *URL* tag defines the URL to which the input results are sent. The actual HTTP request sent to this server specifies the URL with a list of parameters that are appended to it as a query string. The parameters include Name/Value pairs, one for each input item.

**Note**

The Cisco IP Phone 7800 and 8800 Series are the only phones that support the HTTP POST method.

The *InputItem* tag defines each item in the list. The number of *InputItems* must not exceed five. Each input item includes a *DisplayName*, which is the prompt that is written to the display for that particular item. Each item also has a *QueryStringParam*, which is the name of the parameter that is appended to the URL when it is sent out after input is complete. Each input item can also use the *DefaultValue* tag to set the default value to be displayed.

The final attribute for each input item comprises a set of *InputFlags*. The following table describes the input types that are currently defined.

Table 6: InputFlag Definitions

InputFlag	Description	Notes
A	Plain ASCII text	Use the DTMF keypad to enter text that consists of uppercase and lowercase letters, numbers, and special characters.
T	Telephone number	Enter only DTMF digits for this field. The acceptable input includes numbers, #, and *.
N	Numeric	Enter numbers as the only acceptable input.
E	Equation	Enter numbers and special math symbols.
U	Uppercase	Enter uppercase letters as the only acceptable input.
L	Lowercase	Enter lowercase letters as the only acceptable input.
P	Password field	<p>Enter individual characters using the standard keypad-repeat entry mode. The system automatically converts accepted characters into an asterisk, keeping the entered value private.</p> <p>Note P specifies the only <i>InputFlag</i> that works as a modifier. For example, specify a value of “AP” in the <i>InputFlag</i> field to use plain ASCII as the input type and to mask the input as a password by using an asterisk (*).</p>

CiscoIPPhoneDirectory

The *CiscoIPPhoneDirectory* XML object supports the Directory operation of IP phones. The following figure shows how an XML *CiscoIPPhoneDirectory* object displays on the phone.

Figure 6: CiscoIPPhoneDirectory Object Display Sample



CiscoIPPhoneDirectory Definition

```
<CiscoIPPhoneDirectory>
<Title>Directory title goes here</Title>
<Prompt>Prompt text goes here</Prompt>
<DirectoryEntry>
<Name>The name of the directory entry</Name>
<Telephone>The telephone number for the entry</Telephone>
</DirectoryEntry>
</CiscoIPPhoneDirectory>
```



Note

For the directory listing, the IP phone displays the appropriate softkeys that are needed to dial the numbers that are listed on the display. The softkeys include the Edit Dial softkey, which allows users to insert access codes or other necessary items before dialing.

The *Title* and *Prompt* tags in the XML object have the usual semantics. A single *CiscoIPPhoneDirectory* object can contain a maximum of 32 *DirectoryEntry* objects. If more than 32 entries must be returned, use multiple *CiscoIPPhoneDirectory* objects in subsequent HTTP requests.



Note

The Cisco Unified IP Phone 6900 Series do not display the *Title* and *Prompt* menu fields at the same time. If both *Title* and *Prompt* fields are defined at the same time, then these phones display only the *Prompt* field.

Custom Directories

You can use the Cisco Unified Communications Manager enterprise URL Directories parameter and CiscoIPPhone XML objects to display custom directories. The URL Directories parameter points to a URL that returns a *CiscoIPPhoneMenu* object to extend the directories menu. The request for URL Directories must return a valid *CiscoIPPhoneMenu* object, even if the object has no *DirectoryEntry* objects.

To create a custom directory, use the following optional objects in the order in which they are listed:

- 1 Use the *CiscoIPPhoneInput* XML object to collect search criteria.
- 2 Use the *CiscoIPPhoneText* XML object to display status messages or errors.

- 3 Use the *CiscoIPPhoneDirectory* XML object to return a list of directory entries that can be dialed.

You can omit the *CiscoIPPhoneInput* or *CiscoIPPhoneText* objects. You can display multiple *CiscoIPPhoneDirectory* objects by specifying an HTTP refresh header that points to the URL of the next individual directory object, which the user accesses by pressing the Next softkey on the phone.

CiscoIPPhoneImage

The *CiscoIPPhoneImage* provides a bitmap display with a 133 x 65 pixel pane (irrespective of the window mode being normal width or wide width), that is available to access services. Each pixel includes four grayscale settings. A value of three (3) displays as black, and a value of zero (0) displays as white.



Note

The phone uses an LCD display, which inverts the palette.

The *CiscoIPPhoneImage* XML type lets you use the IP phone display to present graphics to the user.

Related Topics

[CiscoIPPhoneGraphicMenu](#), on page 31

CiscoIPPhoneImage Definition

```
<CiscoIPPhoneImage WindowMode="XSI window width mode">
<Title>Image title goes here</Title>
<Prompt>Prompt text goes here</Prompt>
<LocationX>Position information of graphic</LocationX>
<LocationY>Position information of graphic</LocationY>
<Width>Size information for the graphic</Width>
<Height>Size information for the graphic</Height>
<Depth>Number of bits per pixel</Depth>
<Data>Packed Pixel Data</Data>
<SoftKeyItem>
<Name>Name of the soft key</Name>
<URL>URL of soft key</Name>
<Position>Numerical position of the soft key</Position>
</SoftKeyItem>
</CiscoIPPhoneImage>
```

The *WindowMode* attribute is an optional attribute that is used to set the width of an XSI application window. This attribute is supported on the Cisco Unified IP Phones 7941, 7942, 7945, 7961, 7962, 7965, 7970, 7971, and 7975.

The *WindowMode* attribute accepts either of the following values:

Normal:

(Default value) The application window is in the normal-width mode. See the following figure.

Wide:

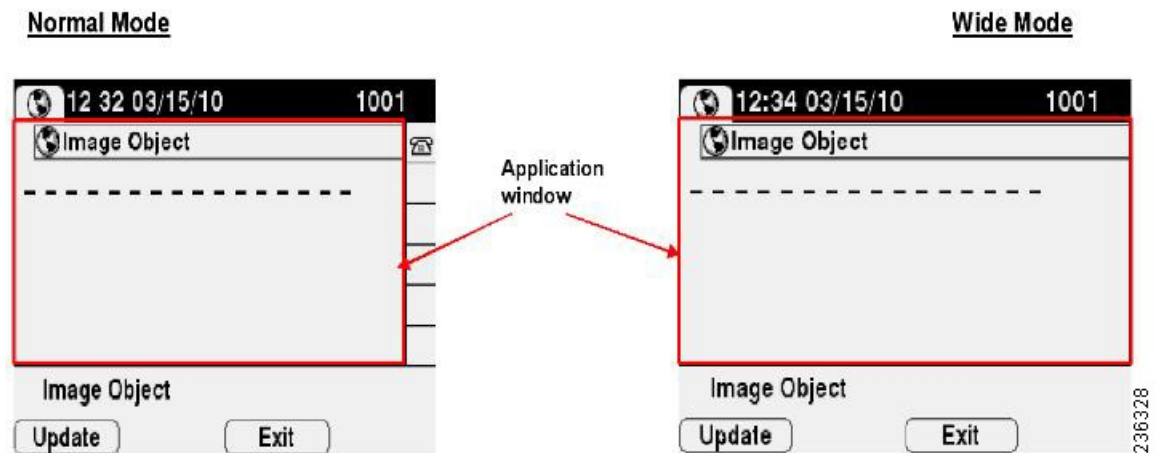
The application window is in the full-width mode, that is the window expands to the complete phone screen width. The wide mode supports a maximum width of 320 pixels for an image. See the following figure.

The *WindowMode* attribute name and value are case sensitive. If the attribute name is wrong, the name is ignored and the default window width is used. If the attribute value is wrong, the parser reports an XML parse error and the object is rejected.

The *WindowMode* attribute is ignored on phones that does not support this feature. In these cases, the window remains the original width. In phones that support this attribute, the absence of the attribute means that the phone uses Normal mode.

For examples on the use of the *WindowMode* attribute, see [CiscoIPPhoneImage Example 2](#), on page 25.

Figure 7: WindowMode



The *Title* and *Prompt* elements serve the same purpose as they do in the other CiscoIPPhone XML objects. The *Title* displays at the top of the page, and the *Prompt* displays at the bottom.

Use *LocationX* and *LocationY* to position the graphic on the phone display. Position the upper, left corner of the graphic at the pixel defined by these two parameters. Setting the X and Y location values to (0, 0) positions the graphic at the upper, left corner of the display. Setting the X and Y location values to (-1, -1) centers the graphic in the services pane of the phone display.

When you use CiscoIPPhoneImage with the Cisco Wireless IP Phone 8821, the phone ignores *LocationX* and *LocationY*. The image will always be placed in the center.

Use *Width* and *Height* to size the graphic. If the values do not match with the pixel stream specified in the *Data* field, results will be unpredictable or incorrect.

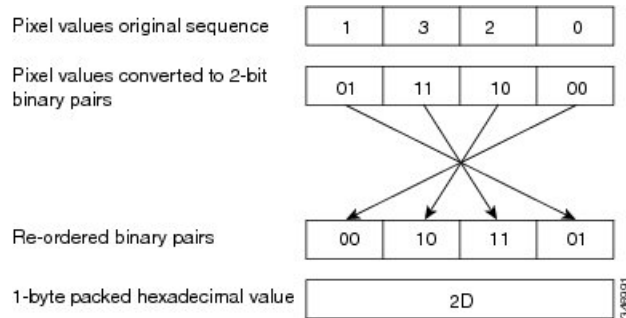
Depth specifies the number of bits per pixel. IP phones support a maximum value of 2 bits per pixel. A bit depth of 1 is black and white.

The *Data* tag delimits a string of hexadecimal digits that contain the packed value of the pixels in the display. In the IP phone, each pixel has only four possible values, which means that you can pack four pixels into a single byte. A pair of hexadecimal digits represents each byte.

The following figure provides an example of the mechanics of pixel packing. Scanning from left to right in the display, the illustration shows the process for packing consecutive pixel values of 1, 3, 2, and 0. First, the

pixels get converted to 2-bit binary numbers. Then, the binary pairs get reordered in sets of four to create a single reordered byte, which the two hexadecimal digits represent.

Figure 8: Packed Pixel Translation Example



CiscoIPPhoneImage Example 1

The following XML code defines a *CiscoIPPhoneImage* object that displays the sequence of pixels shown in the above figure as a graphic positioned at the center of the phone display.

```
<CiscoIPPhoneImage>
  <Title/>
  <LocationX>-1</LocationX>
  <LocationY>-1</LocationY>
  <Width>4</Width>
  <Height>1</Height>
  <Depth>2</Depth>
  <Data>2D</Data>
  <Prompt/>
</CiscoIPPhoneImage>
```

The graphic display comprises a contiguous stream of hexadecimal digits, with no spaces or other separators. If the number of pixels to be displayed does not represent an even multiple of four, pad the end of the pixel data with blank (zero value) pixels, so the data is packed correctly. The phone ignores the padded data.



Note

Before displaying a graphic image on an IP phone, the software clears the pane dedicated to services. If a service has text or other information that must be preserved (including the title area), the information must get redrawn as part of the graphic. If the title is to be hidden, the graphic must be large enough to cover it.

CiscoIPPhoneImage Example 2

The following XML code examples show the usage of the *WindowMode* attribute in the *CiscoIPPhoneImage* object.

- *CiscoIPPhoneImage* object with no *WindowMode* attribute. See the following figure.

```
<CiscoIPPhoneImage>
  <Title>Image Object</Title>
  <LocationX>0</LocationX>
  <LocationY>20</LocationY>
  <Width>133</Width>
  <Height>45</Height>
```

```

    <Depth>1</Depth>
    <Data>f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0</Data>
    <Prompt>Image Object</Prompt>
  </CiscoIPPhoneImage>

```

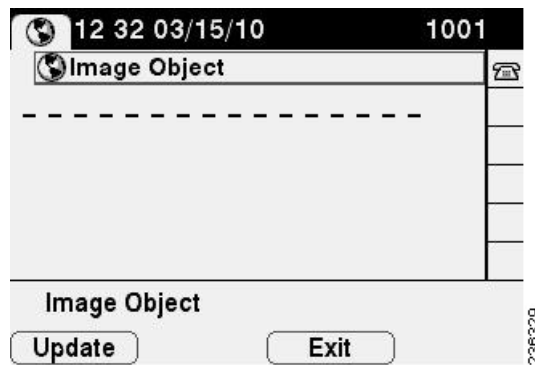
- CiscoIPPhoneImage object with WindowMode set to normal. See the following figure.

```

<CiscoIPPhoneImage WindowMode="Normal">
  <Title>Image Object</Title>
  <LocationX>0</LocationX>
  <LocationY>20</LocationY>
  <Width>133</Width>
  <Height>45</Height>
  <Depth>1</Depth>
  <Data>f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0</Data>
  <Prompt>Image Object</Prompt>
</CiscoIPPhoneImage>

```

Figure 9: CiscoIPPhoneImage Object



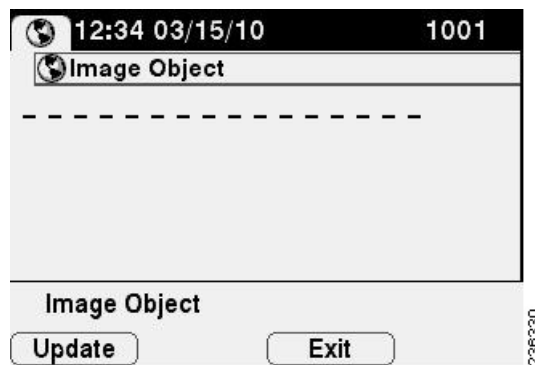
- CiscoIPPhoneImage object with WindowMode set to wide. See the following figure.

```

<CiscoIPPhoneImage WindowMode="Wide">
  <Title>Image Object</Title>
  <LocationX>0</LocationX>
  <LocationY>20</LocationY>
  <Width>133</Width>
  <Height>45</Height>
  <Depth>1</Depth>
  <Data>f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0</Data>
  <Prompt>Image Object</Prompt>
</CiscoIPPhoneImage>

```

Figure 10: CiscoIPPhoneImage Object with WindowMode Wide



CiscoIPPhoneImageFile

The latest generation of IP phones have higher-resolution displays with more color depth. The Cisco Unified IP Phone 7970G, for example, has a display area of 298x168 pixels available to the Services pane and renders images in 12-bit color.

To support these more advanced displays, the XML object allows the use of color PNG images in addition to the grayscale *CiscoIPPhoneImage* objects. The *CiscoIPPhoneImageFile* object behaves like the *CiscoIPPhoneImage* object, except for the image data. Instead of using the <Data> tag to embed the image data, the <URL> tag points to the PNG image file.

The web server must deliver the PNG image to the phone with an appropriate MIME Content-Type header, such as image/png, so that the phone recognizes the content as a compressed, binary PNG image. The PNG image can be either palettized or RGB, and the maximum image size and color depth are model dependent (see the following table).


Note

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Model	Resolution (see note 1) (width x height)	Resolution in wide mode (width x height)	Color, Grayscale, Monochrome	Color depth (bits)
Cisco Unified IP Phones 6921, 6961	396 x 81	N/A	Monochrome	—
Cisco Unified IP Phones 6941, 6945	396 x 162	N/A	Monochrome	—
Cisco IP Phones 7811, 7821, 7841, 7861	N/A	N/A	Monochrome	—
Cisco Unified IP Phones 7905G, 7906G, 7911G, 7912G (see note 2), 7931G	N/A	N/A	Grayscale	1
Cisco Unified IP Phone 7920	128 x 59	N/A	Grayscale	1
Cisco Unified Wireless IP Phones 7921G, 7925G, 7926G, 7925G-EX	176 x 140	N/A	Color	16

Model	Resolution (see note 1) (width x height)	Resolution in wide mode (width x height)	Color, Grayscale, Monochrome	Color depth (bits)
Cisco Unified IP Phone 7937G	255 x 128	N/A	Grayscale	2
Cisco Unified IP Phones 7940G, 7960G	133 x 65	N/A	Grayscale	2
Cisco Unified IP Phones 7941G, 7941G-GE, 7942G, 7961G, 7961G-GE, 7962G	298 x 144	320 x 144	Grayscale	4
Cisco Unified IP Phones 7945G, 7965G	298 x 156	320 x 156	Color	16
Cisco Unified IP Phone 7970G, 7971G	298 x 168	320 x 168	Color	12
Cisco Unified IP Phone 7975G	298 x 168	320 x 168	Color	16
Cisco IP Communicator	298 x 168	N/A	Color	24
Cisco Unified IP Conference Station 8831	396 x 162	N/A	Monochrome	—
Cisco IP Phone 8811	559 x 265	N/A	Monochrome	0-10
Cisco IP Phone 8841, 8845, 8851, 8851NR, 8861, and 8865	559 x 265	N/A	Color	24
Cisco Wireless IP Phone 8821	240 x 215			
Cisco Unified IP Phones 8941, 8945	498 x 289	N/A	Color	24

Model	Resolution (see note 1) (width x height)	Resolution in wide mode (width x height)	Color, Grayscale, Monochrome	Color depth (bits)
Cisco Unified IP Phones 8961, 9951, 9971	498 x 289	N/A	Color	24

**Note**

- 1 Resolution represents the size of the display that is accessible by Services; not the full resolution of the physical display.
- 2 The Cisco Unified IP Phones 7905 and 7912 have pixel-based displays, but they do not support XML images.

If the number of colors in the image is not reduced to match the phone capabilities, the image will be dithered by the phone and yield less than desirable results in most cases. To reduce the number of colors in a graphics editing program, such as Adobe Photoshop, use the *Posterize* command. The Posterize command takes one value as input for the number of color tones per color channel. For example, using the value of 16 (4-bits per channel = 16 tones per channel) correctly dithers the color palette of the image for the best display results on the Cisco Unified IP Phone 7970G.

The following figure shows a CiscoIPPhoneImageFile object on a Cisco Unified IP Phone 7970G display.

Figure 11: Cisco Unified IP Phone 7970G Image File Display

**Related Topics**

[Deprecated Endpoints, on page 3](#)

CiscoIPPhoneImageFile Definition

```
<CiscoIPPhoneImageFile WindowMode="Width Mode of XSI window">
  <Title>Image Title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <LocationX>Horizontal position of graphic</LocationX>
  <LocationY>Vertical position of graphic</LocationY>
```

```
<URL>Points to the PNG image</URL>
</CiscoIPPhoneImageFile>
```

For the description on WindowMode attribute, see [CiscoIPPhoneImage](#), on page 23

For examples of the use of the WindowMode attribute, see [CiscoIPPhoneImageFile Examples](#), on page 30.

CiscoIPPhoneImageFile Examples

The following XML code shows the usage of WindowMode attribute in CiscoIPPhoneImageFile object.

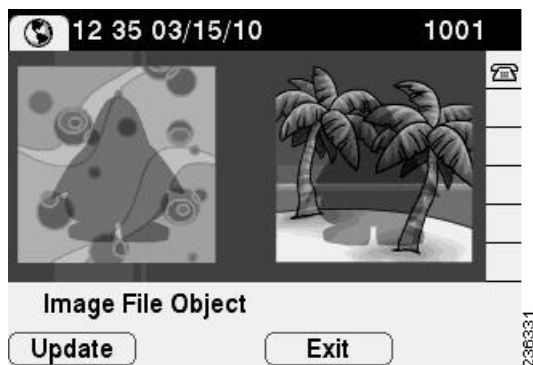
- Without the WindowMode attribute. See the following figure.

```
<CiscoIPPhoneImageFile>
  <Title>Image File Object</Title>
  <Prompt>Image File Object</Prompt>
  <LocationX>0</LocationX>
  <LocationY>0</LocationY>
  <URL>http://10.74.63.74:8080/xsi/normal1.png</URL>
</CiscoIPPhoneImageFile>
```

- With WindowMode attribute set to Normal. See the following figure.

```
<CiscoIPPhoneImageFile WindowMode="Normal">
  <Title>Image File Object</Title>
  <Prompt>Image File Object</Prompt>
  <LocationX>297</LocationX>
  <LocationY>0</LocationY>
  <URL>http://10.74.63.74:8080/xsi/normal1.png</URL>
</CiscoIPPhoneImageFile>
```

Figure 12: CiscoIPPhoneImageFile Object



- WindowMode attribute set to Wide and point the URL to a larger png image file. See the following figure.

```
<CiscoIPPhoneImageFile WindowMode="Wide">
  <Title>Image File Object</Title>
  <Prompt>Image File Object</Prompt>
  <LocationX>319</LocationX>
  <LocationY>0</LocationY>
```

```
<URL>http://10.74.63.74:8080/xsi/wide1.png</URL>
</CiscoIPPhoneImageFile>
```

Figure 13: WindowMode Attribute Set to Wide



CiscoIPPhoneGraphicMenu

Graphic menus serve the same purpose as text menus: they allow a user to select a URL from a list. Use graphic menus in situations when the items may not be easy to display in a text list.

For example, users might prefer to have their choices presented in a non-ASCII character set such as Kanji or Arabic. When using non-ASCII character sets, the system presents the information as a bitmap graphic. To select a menu, the user enters a number from 1 to 12 using the numeric keypad (* and # are not active).

Related Topics

[CiscoIPPhoneImage](#), on page 23

[CiscoIPPhoneMenu](#), on page 18

CiscoIPPhoneGraphicMenu Definition

```
<CiscoIPPhoneGraphicMenu WindowMode="Width Mode of XSI window">
  <Title>Menu title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <LocationX>Position information of graphic</LocationX>
  <LocationY>Position information of graphic</LocationY>
  <Width>Size information for the graphic</Width>
  <Height>Size information for the graphic</Height>
  <Depth>Number of bits per pixel</Depth>
  <Data>Packed Pixel Data</Data>
  <MenuItem>
    <Name>The name of each menu item</Name>
    <URL>The URL associated with the menu item</URL>
  </MenuItem>
</CiscoIPPhoneGraphicMenu>
```

For the description on *WindowMode* attribute, see [CiscoIPPhoneImage](#), on page 23.

For examples of the use of the *WindowMode* attribute, see [CiscoIPPhoneGraphicFileMenu WindowMode Examples](#), on page 35.

Menu items in the graphic menu have a name, like the text menu counterparts. Although the name does not display to the user, it still performs a function. The name of the menu item provides the default title that is used when the URL for the chosen item loads. If the loaded page has a title of its own, the phone uses the defined title instead.

The XML tags in *GraphicMenu* use the tag definitions for *CiscoIPPhoneImage* and *CiscoIPPhoneMenu*. Although the semantics of the tags are identical, you can have only 12 *MenuItem* objects in a *CiscoIPPhoneGraphicMenu* object.

Related Topics

[CiscoIPPhoneMenu](#), on page 18

[CiscoIPPhoneImage](#), on page 23

CiscoIPPhoneGraphicMenu WindowMode Examples

The following XML code shows the usage of WindowMode attribute in CiscoIPPhoneGraphicMenu object.

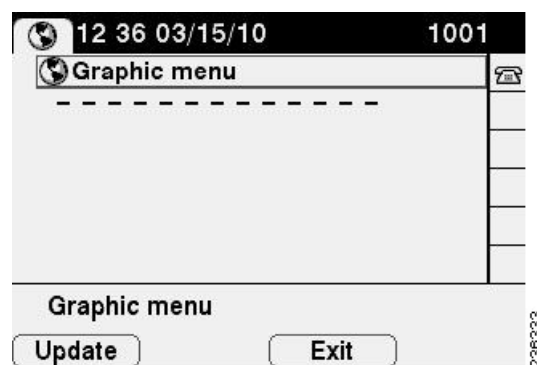
- Without WindowMode attribute. See the following figure.

```
<CiscoIPPhoneGraphicMenu>
  <Title>Graphic menu</Title>
  <Prompt>Graphic menu</Prompt>
  <LocationX>10</LocationX>
  <LocationY>15</LocationY>
  <Width>133</Width>
  <Height>45</Height>
  <Depth>1</Depth>
  <Data>f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0</Data>
  <MenuItem>
    <Name>dial_1000</Name>
    <URL>Dial:1000</URL>
  </MenuItem>
</CiscoIPPhoneGraphicMenu>
```

- With WindowMode attribute set to Normal. See the following figure.

```
<CiscoIPPhoneGraphicMenu WindowMode="Normal">
  <Title>Graphic menu</Title>
  <Prompt>Graphic menu</Prompt>
  <LocationX>10</LocationX>
  <LocationY>15</LocationY>
  <Width>133</Width>
  <Height>45</Height>
  <Depth>1</Depth>
  <Data>f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0</Data>
  <MenuItem>
    <Name>dial_1000</Name>
    <URL>Dial:1000</URL>
  </MenuItem>
</CiscoIPPhoneGraphicMenu>
```

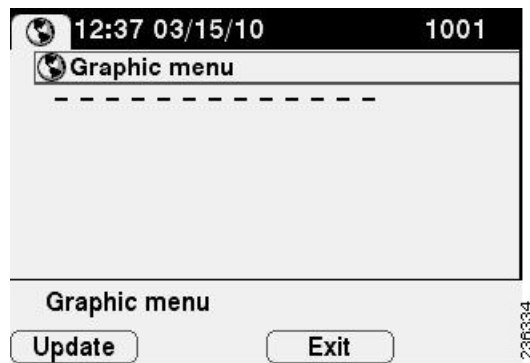
Figure 14: CiscoIPPhoneGraphicMenu Object



- With WindowMode attribute set to Wide. See the following figure.

```
<CiscoIPPhoneGraphicMenu WindowMode="Wide">
  <Title>Graphic menu</Title>
  <Prompt>Graphic menu</Prompt>
  <LocationX>10</LocationX>
  <LocationY>15</LocationY>
  <Width>133</Width>
  <Height>45</Height>
  <Depth>1</Depth>
  <Data>f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0</Data>
  <MenuItem>
    <Name>dial_1000</Name>
    <URL>Dial:1000</URL>
  </MenuItem>
</CiscoIPPhoneGraphicMenu>
```

Figure 15: WindowMode Attribute set to Wide



CiscoIPPhoneGraphicFileMenu

Some of the Cisco Unified IP Phone models, such as the Cisco Unified IP Phone 7970G and Cisco IP Communicator, have pointer devices. The Cisco Unified IP Phone 7970G uses a touchscreen overlay on the display, and the PC-based Cisco IP Communicator uses the standard Windows mouse pointer.

Because these devices can receive and process “pointer” events, a CiscoIPPhoneGraphicFileMenu object exposes the capability to application developers. The CiscoIPPhoneGraphicFileMenu behaves similar to the CiscoIPPhoneGraphicMenu, in that a group of options are presented by an image. When one of those objects is selected, a URL action initiates. However, the FileMenu does not use the keypad, but uses rectangular touch areas. This rectangular touch area, <TouchArea>, is defined by coordinates relative to the upper-left corner of the Services display. The (X1,Y1) points specify the upper-left corner of the <TouchArea>, and (X2,Y2) specify the lower-right corner of the <TouchArea>.

The following figure shows the display of the CiscoIPPhoneGraphicFileMenu.

Figure 16: CiscoIPPhoneGraphicFileMenu



If the coordinates that are supplied in the `<TouchArea>` tag exceed the dimensions of the phone display, the `<TouchArea>` rectangle will be clipped to fit.

The `<TouchArea>` rectangles can overlap, and the first match is always taken. This allows a sense of Z-order for images where smaller touchable objects can be overlaid on top of larger ones. In this case, the smaller object `<MenuItem>` must appear before the larger one in the `<CiscoIPPhoneGraphicFileMenu>` object.

The requirements for the PNG image referenced by the `<URL>` tag match those that the *CiscoIPPhoneImageFile* object uses.



Note

The Cisco Wireless IP Phone 8821 ignores the parameters. It centers and scales the image.

Related Topics

[CiscoIPPhoneImageFile](#), on page 27

[CiscoIPPhoneGraphicMenu](#), on page 31

CiscoIPPhoneGraphicFileMenu Definition

```
<CiscoIPPhoneGraphicFileMenu WindowMode="Width Mode of XSI window">
  <Title>Image Title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <LocationX>Horizontal position of graphic</LocationX>
  <LocationY>Vertical position of graphic</LocationY>
  <URL>Points to the PNG background image</URL>
  <MenuItem>
    <Name>Same as CiscoIPPhoneGraphicMenu</Name>
    <URL>Invoked when the TouchArea is touched</URL>
    <TouchArea X1="left edge" Y1="top edge" X2="right edge" Y2="bottom edge"/>
  </MenuItem>
</CiscoIPPhoneGraphicFileMenu>
```

For the description on WindowMode attribute, see [CiscoIPPhoneImage](#), on page 23.

For examples of the use of the WindowMode attribute, see [CiscoIPPhoneGraphicFileMenu WindowMode Examples](#), on page 35.

CiscoIPPhoneGraphicFileMenu WindowMode Examples

The following XML code shows the usage of WindowMode attribute in CiscoIPPhoneGraphicFileMenu object.

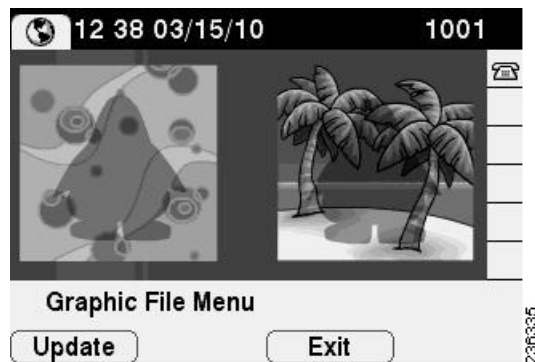
- Without WindowMode attribute. See the following figure.

```
<CiscoIPPhoneGraphicFileMenu>
  <Title>Graphic File Menu</Title>
  <Prompt>Graphic File Menu</Prompt>
  <LocationX>0</LocationX>
  <LocationY>0</LocationY>
  <URL>http://10.74.63.74:8080/xsi/normal1.png</URL>
  <MenuItem>
    <Name>dial_1000</Name>
    <URL>Dial:1000</URL>
    <TouchArea X1="0" Y1="0" X2="160" Y2="168"/>
  </MenuItem>
</CiscoIPPhoneGraphicFileMenu>
```

- With WindowMode attribute set to Normal. See the following figure.

```
<CiscoIPPhoneGraphicFileMenu WindowMode="Normal">
  <Title>Graphic File Menu</Title>
  <Prompt>Graphic File Menu</Prompt>
  <LocationX>0</LocationX>
  <LocationY>0</LocationY>
  <URL>http://10.74.63.74:8080/xsi/normal1.png</URL>
  <MenuItem>
    <Name>dial_1000</Name>
    <URL>Dial:1000</URL>
    <TouchArea X1="0" Y1="0" X2="160" Y2="168"/>
  </MenuItem>
</CiscoIPPhoneGraphicFileMenu>
```

Figure 17: CiscoIPPhoneGraphicFileMenu



- With WindowMode attribute set to Wide. See the following figure

```
<CiscoIPPhoneGraphicFileMenu WindowMode="Wide">
  <Title>Graphic File Menu</Title>
  <Prompt>Graphic File Menu</Prompt>
  <LocationX>0</LocationX>
  <LocationY>0</LocationY>
  <URL>http://10.74.63.74:8080/xsi/wide3.png</URL>
  <MenuItem>
    <Name>dial_1000</Name>
    <URL>Dial:1000</URL>
    <TouchArea X1="0" Y1="0" X2="160" Y2="168"/>
  </MenuItem>
</CiscoIPPhoneGraphicFileMenu>
```

```

    </MenuItem>
  </CiscoIPPhoneGraphicFileMenu>

```

Figure 18: WindowMode Attribute set to Wide



CiscoIPPhoneIconMenu

Icon menus serve the same purpose as text menus: they allow a user to select a URL from a list. Use icon menus in situations when you want to provide additional visual information to the user to show the state or category of an item. For example, you include a read and unread icon in a mail viewer. You can use the icons to convey the message state.

Icons in the *CiscoIPPhoneIconMenu* object have a maximum width of 16 pixels and a maximum height of 10 pixels.

The following figure shows an IconMenu on an IP phone.

Figure 19: CiscoIPPhoneIconMenu on a Cisco Unified IP Phone Sample



The system presents the information as a bitmap graphic to the left of the menu item text. The user selects menu items in the same way as a *CiscoIPPhoneMenu* object.

Related Topics

[CiscoIPPhoneMenu, on page 18](#)

[CiscoIPPhoneImage, on page 23](#)

CiscoIPPhoneIconMenu Definition

```
<CiscoIPPhoneIconMenu>
  <Title>Title text goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <MenuItem>
    <IconIndex>Indicates what IconItem to display</IconIndex>
    <Name>The name of each menu item</Name>
    <URL>The URL associated with the menu item</URL>
  </MenuItem>
  <SoftKeyItem>
    <Name>Name of softkey</Name>
    <URL>URL or URI of softkey</URL>
    <Position>Position information of the softkey</Position>
  </SoftKeyItem>
  <IconItem>
    <Index>A unique index from 0 to 9</Index>
    <Height>Size information for the icon</Height>
    <Width>Size information for the icon</Width>
    <Depth>Number of bits per pixel</Depth>
    <Data>Packed Pixel Data</Data>
  </IconItem>
</CiscoIPPhoneIconMenu>
```



Note

The Cisco Unified IP Phone 6900 Series do not display the *Title* and *Prompt* menu fields at the same time. If both *Title* and *Prompt* fields are defined at the same time, then these phones display only the *Prompt* field.

The XML tags in *CiscoIPPhoneIconMenu* use the tag definitions for *CiscoIPPhoneImage* and *CiscoIPPhoneMenu*. Although the semantics of the tags are identical, you can have only 32 *MenuItem* objects in a *CiscoIPPhoneIconMenu* object.

Related Topics

[CiscoIPPhoneMenu, on page 18](#)

[CiscoIPPhoneImage, on page 23](#)

CiscoIPPhoneIconFileMenu



Note

The *CiscoIPPhoneIconFileMenu* object is updated to support new attributes. For details, see [Enhanced Icon Menu Support Feature, on page 38](#).

This icon menu is similar to *CiscoIPPhoneMenu*, but it uses color PNG icons rather than grayscale CIP icons. Use icon menus in situations when you want to provide additional visual information to the user to show the state or category of an item. For example, you can use icons to indicate priority (see the following figure).

Icons in the *CiscoIPPhoneIconFileMenu* object have a maximum width of 18 pixels and a maximum height of 18 pixels. Instead of using the `<Data>` tag to embed the image data into the `<IconItem>` tag, this object uses a `<URL>` tag to point to the PNG image file to be used for that icon.

Figure 20: CiscoIPPhoneIconFileMenu Object Display Sample



CiscoIPPhoneIconFileMenu Definition

```
<CiscoIPPhoneIconFileMenu>
  <Title>Title text goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <MenuItem>
    <IconIndex>Indicates what IconItem to display</IconIndex>
    <Name>The name of each menu item</Name>
    <URL>The URL associated with the menu item</URL>
  </MenuItem>
  <IconItem>
    <Index>A unique index from 0 to 9</Index>
    <URL>location of the PNG icon image</URL>
  </IconItem>
</CiscoIPPhoneIconFileMenu>
```



Note

The Cisco Unified IP Phone 6900 Series do not display the *Title* and *Prompt* menu fields at the same time. If both *Title* and *Prompt* fields are defined at the same time, then these phones display only the *Prompt* field.

Enhanced Icon Menu Support Feature

The Enhanced Icon Menu Support feature extends the existing *CiscoIPPhoneIconFileMenu* XML object by allowing:

- An icon in its `<Title>` element.
- Internal phone firmware icons, like security state or call state icons, in its `<MenuItems>` and `<Title>` elements.

Supported IP Phones and Codecs

The following table lists the IP phone models that support the Enhanced Icon Menu Support feature.

**Note**

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Phone model	Support	Firmware supported
Cisco Unified IP Phone 9900 Series		
9971	Supported	9.0(1) and later
9951	Supported	9.0(1) and later
Cisco Unified IP Phone 8900 Series		
8941, 8945	Not supported	—
8961	Supported	9.0(1) and later
Cisco IP Phone 8800 Series		
8811	Not supported	—
8841	Not supported	—
8845	Not supported	—
8851	Not supported	—
8851NR	Not supported	—
8861	Not supported	—
8865	Not supported	—
Cisco IP Conference Phone		
8831	Not supported	—
Cisco Wireless IP Phone 8820 Series		
8821	Not supported	—
Cisco Unified IP Phone 7900 Series		
7905	Not supported	—
7906	Supported	8.4(1) and later

Phone model	Support	Firmware supported
7911	Supported	8.4(1) and later
7912	Not supported	—
7931	Supported	8.4(1) and later
7937	Not supported	—
7940	Not supported	—
7941	Supported	8.4(1) and later
7942	Supported	8.4(1) and later
7945	Supported	8.4(1) and later
7960	Not supported	—
7961	Supported	8.4(1) and later
7962	Supported	8.4(1) and later
7965	Supported	8.4(1) and later
7970	Supported	8.4(1) and later
7971	Supported	8.4(1) and later
7975	Supported	8.4(1) and later
7985	Not supported	—
Cisco Unified Wireless IP Phone 7900 Series		
7920	Not supported	—
7921G	Not supported	—
7925G, 7925G-EX	Not supported	—
7926G	Not supported	—
Cisco IP Phone 7800 Series		
7811	Not supported	—
7821	Not supported	—

Phone model	Support	Firmware supported
7841	Not supported	—
7861	Not supported	—
Cisco Unified IP Phone 6900 Series		
6921	Not supported	—
6941	Not supported	—
6945	Not supported	—
6961	Not supported	—
Other devices		
Cisco IP Phone Communicator	Not supported	—

**Note**

Cisco recommends the use of latest firmware. The firmware can be downloaded from the following location (requires login or service contract):

<http://software.cisco.com/download/navigator.html?i=!mmd>

Although several codecs are listed within the schema, only the codecs G711, G729, and G722 are currently supported.

Related Topics

[Deprecated Endpoints, on page 3](#)

CiscoIPPhoneIconFileMenu XML Object Changes

The following changes have been made in the CiscoIPPhoneIconFileMenu XML object for the Enhanced Icon Menu Support feature:

- The CiscoIPPhoneIconFileMenu schema is updated to allow an *IconIndex* attribute in the *<Title>* element.
- A Resource URI attribute is available for the *<URL>* element of the *<IconItem>* element. This Resource URI can be used in place of the HTTP URL.
- The Resource URI identifies the icons in the *<IconItems>*. When a phone parses the *<URL>* element in *<IconItem>*, the phone looks for the Resource URI.
 - If the Resource URI is present, the phone validates the URI against the valid Resource Icon values. If the validation is successful, the phone uses the icon specified by the Resource URI.
 - If the Resource URI is not present or if the URI fails the validation against a recognized Icon value, then a default unknown-icon image displays.

Related Topics

[Valid Resource Icon Names, on page 42](#)

Schema Definition

The definition of the CiscoIPPhoneIconFileMenu schema remains the same except for the *<Title>* element and the *IconIndex* attribute specified as follows:

```
<xsd:complexType name="Title">
  <xsd:attribute name="IconIndex"
    type="xsd:unsignedShort"
    use="optional"/>
</xsd:complexType>
```

CiscoIPPhoneIconFileMenu Example

The following is an example of the CiscoIPPhoneIconFileMenu object with IconIndex attribute in *<Title>* element and Resource URI attribute in *<IconItem>* element:

```
<CiscoIPPhoneIconFileMenu>
  <Title IconIndex="2">Conference List</Title>
  <IconItem>
    <Index>1</Index>
    <URL>Resource:Icon.SecureCall</URL>
  </IconItem>
  <IconItem>
    <Index>2</Index>
    <URL>Resource:Icon.Connected</URL>
  </IconItem>
  <IconItem>
    <Index>3</Index>
    <URL>Resource:AnimatedIcon.Ringin</URL>
  </IconItem>
  <MenuItem>
    <Name>Schmo, Joe</Name>
    <IconIndex>1</IconIndex>
    <URL>http://192.168.1.12:8080/details?user=jschmo</URL>
  </MenuItem>
  <MenuItem>
    <Name>Blow, Joe</Name>
    <IconIndex>2</IconIndex>
    <URL>http://192.168.1.12:8080/details?user=jblow</URL>
  </MenuItem>
  <MenuItem>
    <Name>Joining, Just Now</Name>
    <IconIndex>3</IconIndex>
    <URL>http://192.168.1.12:8080/details?user=jjoining</URL>
  </MenuItem>
</CiscoIPPhoneIconFileMenu>
```

Valid Resource Icon Names

The following are the valid Resource Icon names:

- Icon.Connected
- Icon.AuthenticatedCall
- Icon.SecureCall
- Icon.OnHook
- Icon.OffHook

- Icon.Messages
- Icon.InUse
- Icon.Headset
- Icon.Handset
- Icon.Speaker
- Icon.Locked
- Icon.UnLocked
- Icon.Checked
- Icon.UnChecked
- Icon.RadioButtonOn
- Icon.RadioButtonOff
- AnimatedIcon.Ringin
- AnimatedIcon.Hold
- AnimatedIcon.MessageWaiting
- AnimatedIcon.StreamingRx
- AnimatedIcon.StreamingTx
- AnimatedIcon.StreamRxTx
- AnimatedIcon.Throbber

Troubleshooting CiscoIPPhoneIconFileMenu XML Objects Using Enhanced Icon Menu Support Feature

The following errors and conditions may occur in the Enhanced Icon Menu Support feature:

- If the CiscoIPPhoneIconFileMenu object is invalid, a parsing error is generated and a CiscoIPPhoneError object (with Number="1") is returned as the response.
- If the Resource URI does not specify a recognized Icon resource, then a default unknown-icon image is displayed.

Error Handling

Standard XML services debugging techniques are applied to the Enhanced Icon Menu Support feature. The root cause for any parsing errors displays in the phone console logs. For HTTP requests and responses, sniffer traces and web server debug can be used to examine the CiscoIPPhoneIconFileMenu object to ensure that the object conforms to the schema.

CiscoIPPhoneStatus

The *CiscoIPPhoneStatus* object is also a displayable object, but differs from other objects in that it displays on the Call plane of the phone rather than the Services plane. The CiscoIPPhoneStatus object hovers above

the Call plane and is typically used in conjunction with CTI applications to present application status to the user.

The Status object cannot be closed or cleared by the user (for example, by pressing Services) because the Status object is only present on the Call plane. In order to clear the object, the phone must execute the Init:AppStatus URI. This would typically occur as the result of an application server pushing an Execute object to the phone that contains the Init:AppStatus URI.

**Note**

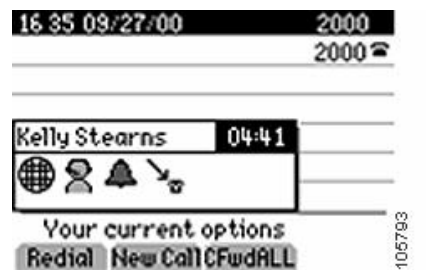
The CiscoIPPhoneStatus object can only be pushed (HTTP POST) to the phone; the object cannot be pulled (HTTP GET).

The *CiscoIPPhoneStatus* object can be refreshed or replaced at any time. It is not necessary to clear an existing Status object before sending a new Status object. The new object simply replaces the old object.

The following figure shows the CiscoIPPhoneStatus object that contains the following visual elements:

- 106 x 21 graphics area for displaying CIP images (same image format as CiscoIPPhoneImage)
- Seedable, free-running timer (optional)
- Single-line text area (optional)

Figure 21: IconMenu on a CiscoIPPhoneStatus sample



The Cisco Unified Wireless IP Phone 7925G, 7925G-EX, and 7926G and the Cisco IP Phone 8821 do not support CiscoIPPhoneStatus.

CiscoIPPhoneStatus Definition

```
<CiscoIPPhoneStatus>
  <Text>This is the text area</Text>
  <Timer>Timer seed value in seconds</Timer>
  <LocationX>Horizontal alignment</LocationX>
  <LocationY>Vertical alignment</LocationY>
  <Width>Pixel width of graphic</Width>
  <Height>Pixel height of graphic</Height>
  <Depth>Color depth in bits</Depth>
  <Data>Hex binary image data</Data>
</CiscoIPPhoneStatus>
```

**Note**

The Cisco Unified IP Phone 6900 Series do not display the *Title* and *Prompt* menu fields at the same time. If both *Title* and *Prompt* fields are defined at the same time, then these phones display only the *Prompt* field.

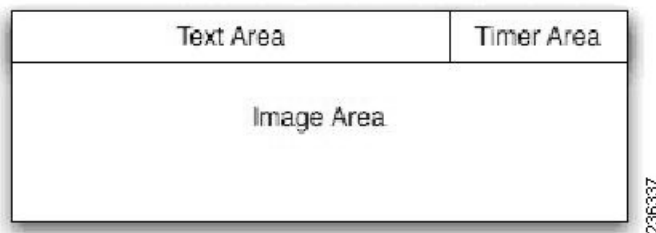
Dynamic Application Status Window Size

You can enable applications to dynamically adjust their window sizes based on the displayed content. The minimum size requirements limit the windows size so that it is large enough to stand out from the Overview content. For example, using a smaller window for an application allows more content from the Overview to be displayed. Sizing the window occurs when the phone receives a CiscoIPPhoneStatus or CiscoIPPhoneStatusFile object with its associated PNG file.

The Application Status window contains three main areas (see the following figure):

- Text Area
- Timer Area
- Image Area

Figure 22: Application Status Window Elements



Note

Selfterminating XML elements, undeclared or missing elements, and elements with the default values are all considered unconfigured elements.

To allow dynamic sizing, do not configure the Text and Timer areas with any value other than the default used by the XML parser. If both elements are not configured, you can proceed, but must follow these rules:

- Do not display the Text Area and Timer Area sections of the Application Status window.
- If the LocationX element is not configured or is set to centered, and the image provided is less than the maximum width allowed, the Image Area can be resized.
- If the image provided is smaller than the minimum width, the minimum allowed window width should be used.
- If the width of the image provided is between the minimum and maximum sizes of the window, the window should be sized to display the image as well as the standard surrounding borders.
- The image height should never change.

See the following table for an overview of the maximum and minimum image area sizes by phone model. Most phone models support all sizes between the minimum and maximum. An exception is allowed for the Cisco Unified IP Phones 7940G and 7960G due to resource constraints. For these phones, you should implement both the maximum size and minimum size windows, ignoring all of the intermediate sizes.

Table 7: Application Status Window Allowable Image Sizes

Phone models	Maximum image area width	Minimum image area width	Maximum image area height
7937G	133	21	65
7940G, 7960G	106	21	21
7941G, 7941G-GE, 7942G, 7945G, 7961G, 7961G-GE, 7962G, 7965G	252	50	50
7970G, 7971G-GE, 7975G, IP Communicator	262	50	50
8811, 8841, 8845, 8851, 8851NR, 8861, 8865	414	70	70
8831			
8941, 8945	342	73	73
8961, 9951, 9971	342	73	73

The following table shows an overview of the text and timer area sizes by phone model.

Table 8: Application Status Window Allowable Text and Timer Sizes

Phone models	Text area size (WxH)	Timer area size (WxH)	Text area size No timer (WxH)
7940G, 7960G	76x11	30x11	106x11
7941G, 7941G-GE, 7942G, 7945G, 7961G, 7961G-GE, 7962G, 7965G,	192x20	60x20	252x20
7970G, 7971G-GE, 7975G, IP Communicator	202x20	60x20	262x20
8831			
8811, 8841, 8845, 8851, 8851NR, 8861, 8865	300x36	100x36	414x36
8941, 8945	278x23	52x23	342x23

Phone models	Text area size (WxH)	Timer area size (WxH)	Text area size No timer (WxH)
8961, 9951, 9971	278x23	52x23	342x23

CiscoIPPhoneStatusFile

The behavior of this object is identical to the *CiscoIPPhoneStatus* object, except it uses a color PNG image instead of a grayscale CIP image for the graphics area.

The maximum image size is 262 x 50 pixels for the Cisco Unified IP Phone 7970G, but differs for other phone models.

The following figure shows how an XML *CiscoIPPhoneStatusFile* object displays on a phone.

Figure 23: CiscoIPPhoneStatusFile Object Display Sample



The Cisco Unified Wireless IP Phone 7925G, 7925G-EX, and 7926G and the Cisco IP Phone 8821 do not support CiscoIPPhoneStatusFile.

Related Topics

[CiscoIPPhoneStatus](#), on page 43

CiscoIPPhoneStatusFile Definition

```
<CiscoIPPhoneStatusFile>
<Text>This is the text area</Text>
<Timer>Timer seed value in seconds</Timer>
<LocationX>Horizontal alignment</LocationX>
<LocationY>Vertical alignment</LocationY>
```

```
<URL>location of the PNG image</URL>
</CiscoIPPhoneStatusFile>
```

Note that instead of using the `<Data>` tag to embed the image data, this object uses a `<URL>` tag to point to the PNG image file to be used for the graphics area.

Related Topics

[Dynamic Application Status Window Size, on page 45](#)

CiscoIPPhoneExecute

The *CiscoIPPhoneExecute* object differs from the other CiscoIPPhone objects. It is not a displayable object for providing user interaction. The purpose of this object is to deliver (potentially multiple) execution requests to the phone.

Like the other XML objects, the CiscoIPPhoneExecute can be either pushed (HTTP POST) or pulled (HTTP GET). Upon receiving a CiscoIPPhoneExecute object, the phone begins executing the specified ExecuteItems. Order of execution is not guaranteed, so ExecuteItems will likely not execute in the order in which they are listed in the CiscoIPPhoneExecute object.



Note

Limit the requests to three ExecuteItems: only one can be a URL and two URIs per *CiscoIPPhoneExecute* object, or you can send three URIs with no URL.

CiscoIPPhoneExecute Definition

```
<CiscoIPPhoneExecute>
  <ExecuteItem URL="the URL or URI to be executed"/>
</CiscoIPPhoneExecute>
```

The `<ExecuteItem>` tag of the *CiscoIPPhoneExecute* object includes an optional attribute called *Priority*. The Priority attribute is used to inform the phone of the urgency of the execute request and to indicate whether the phone should be interrupted to perform the request. The Priority levels determine whether the phone must be idle to perform the requested action. The Idle Timer (along with an optional Idle URL) is defined globally in the Cisco Unified Communications Manager Administration Enterprise Parameters and can be overridden on an individual phone basis in the Cisco Unified Communications Manager Device configuration.

The following table lists the Priority levels and their behavior.

Priority	Behavior	Description
0	Execute Immediately	The URL executes regardless of the state of the phone. If the Priority attribute does not get specified in the <code><ExecuteItem></code> , the default priority gets set to zero for backward compatibility.
1	Execute When Idle	The URL gets delayed until the phone goes idle, then it executes.
2	Execute If Idle	The URL executes on an idle phone; otherwise, it does not get executed (it does not get delayed).

**Note**

The Priority attribute is only used for HTTP URLs. Internal URIs always execute immediately.

CiscoIPPhoneExecute Example

The following CiscoIPPhoneExecute object results in the phone playing an alert chime, regardless of the state of the phone, but waits until the phone goes idle before displaying the specified XML page.

```
<CiscoIPPhoneExecute>
  <ExecuteItem Priority="0" URL="Play:chime.raw"/>
  <ExecuteItem Priority="1" URL="http://server/textmessage.xml"/>
</CiscoIPPhoneExecute>
```

CiscoIPPhoneResponse

The *CiscoIPPhoneResponse* objects provide messages and information resulting from a *CiscoIPPhoneExecute*. As a result, a *ResponseItem* exists for each *ExecuteItems* that you send. The order differs based on completion time, and the execution order is not guaranteed.

The URL attribute specifies the URL or URI that was sent with the request. The Data attribute contains any special data for the item. The Status attribute specifies a status code. Zero indicates that no error occurred during processing of the ExecuteItem. If an error occurred, the phone returns a *CiscoIPPhoneError* object.

CiscoIPPhoneResponse Definition

```
<CiscoIPPhoneResponse>
  <ResponseItem Status="the success or failure of the action"
    Data="the information returned with the response"
    URL="the URL or URI specified in the Execute object"/>
</CiscoIPPhoneResponse>
```

CiscoIPPhoneError

The following list gives possible CiscoIPPhoneError codes:

- Error 1 = Error parsing CiscoIPPhoneExecute object
- Error 2 = Error framing CiscoIPPhoneResponse object
- Error 3 = Internal file error
- Error 4 = Authentication error

CiscoIPPhoneError Definition

```
<CiscoIPPhoneError Number="x"/> optional error message <CiscoIPPhoneError>
```

The text value of the *CiscoIPPhoneError* object may contain an optional error message to further describe the nature of the error condition.

Custom Softkeys

IP Phones can use custom softkeys with any of the displayable CiscoIPPhone XML objects, with the following exceptions:

- *CiscoIPPhoneStatus* object, which cannot control softkeys
- *CiscoIPPhoneExecute* object, which is not displayable

Softkeys can have either URL or URI actions associated with them. The *SoftkeyItem* can define separate actions to be taken when the softkey is pressed and released. The standard UI behavior is to execute an action when a key is released, and this action is defined by the `<URL>` tag. An action can also be taken when the softkey is initially pressed by including the optional `<URLDown>` tag. For example, you might use `<URLDown>` for a press-to-talk application in which pressing the button starts audio streaming and releasing the button stops it.



Note

The `<URLDown>` tag can only contain Internal URIs: it cannot contain an HTTP URL. The “URL” in the name “URLDown” does not signify that an HTTP URL can be used.

SoftKeyItem Definition

```
<SoftKeyItem>
  <Name>Displayed sofkey label</Name>
  <URL>URL or URI action for softkey RELEASE event</URL>
  <URLDown>URL or URI action for softkey PRESS event</URLDown>
  <Position>position of softkey</Position>
</SoftKeyItem>
```

Position

- -1 designates the Application button
- Cisco Unified Wireless IP Phone 792x Series: 1 to 8 designates the softkeys
- Cisco Wireless IP Phone 8821: 1 to 8 designates the softkeys
- Other phones: 1 to 16 designates the softkeys

The *SoftKeyItem* in the -1 position does not display on the phone screen. If the user pressed the application button while in an XSI application, the action associated with the -1 position executes.

For the Cisco Wireless IP Phone 8821, the softkey position 1 corresponds to the right softkey. All the other positions correspond to the left softkey.

SoftKeyItem Example 1

In this example, a *CiscoIPPhoneText* object has a single custom softkey defined.

```
<CiscoIPPhoneText>
  <Text>This object has one softkey named "Custom"</Text>
  <SoftKeyItem>
```

```

<Name>Custom</Name>
<URL>http://someserver/somepage</URL>
<Position>4</Position>
</SoftKeyItem>
</CiscoIPPhoneText>

```

If any custom softkeys are defined in the XML object, then all default softkeys are removed from that object. To retain default softkey behavior, you must explicitly define the softkeys in the XML object using a `<SoftKeyItem>` tag. The internal Softkey URIs can be used in the `<URL>` tag of `<SoftKeyItem>` to invoke default softkey actions from custom softkeys.

**Note**

If there are no custom softkeys and there is no default softkey placed in position 1, either a Next or Update softkey is assigned automatically. If the URL is a Refresh URL, the Next softkey is assigned. If not, the Update softkey is assigned.

Related Topics

[Internal URI Features, on page 65](#)

SoftKeyItem Example 2

The following softkey definitions would provide the Custom softkey, without losing the default Select softkey behavior.

```

<SoftKeyItem>
  <Name>Select</Name>
  <URL>SoftKey:Select</URL>
  <Position>1</Position>
</SoftKeyItem>
<SoftKeyItem>

```

XML Considerations

The XML parser in the IP Phones does not function as a fully-capable XML parser. Do not include any tags other than those defined in your XML display definitions.

**Note**

All CiscoIPPhone element names and attribute names are case sensitive.

Mandatory Escape Sequences

By XML convention, the XML parser also requires that you provide escape values for a few special characters. The following table lists characters and their escape values.

Table 9: Escape Sequences for Special Characters

Character	Name	Escape sequence
&	Ampersand	&

Character	Name	Escape sequence
"	Quote	"
'	Apostrophe	'
<	Left angle bracket	<
>	Right angle bracket	>

Escaping text can be tedious, but some authoring tools or scripting languages can automate this task.

XML Encoding

Because the phone firmware can support multiple encodings, the XML encoding should always be set in the XML header.

If the XML encoding header is not specified, the phone will default to the encoding specified by the current user locale.



Note

This behavior is NOT compliant with XML standards, which specify UTF-8 as the default encoding, so any UTF-8 encoded XML object must have the encoding explicitly set for the phone to parse it correctly.

The encoding value specified in the XML header must match one of the encodings provided by the IP Phone in its Accept-Charset HTTP request header, as shown in [XML Encoding Example](#), on page 52

XML Encoding Example

The following examples illustrate UTF-8 and ISO-8859-1 encoding, respectively:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<?xml version="1.0" encoding="iso-8859-1" ?>
```

Related Topics

[HTTP Encoding Header Setting](#), on page 106

Application Event Handlers

The Application Manager API includes an Application Management Event Handler, which is supported by any displayable object, as noted in the following table. The unsupported objects are not contained in a standard application context and are handled differently by the Application Manager API.

Table 10: Application Event Handler Support

Supported	Unsupported
CiscoIPPhoneMenu	CiscoIPPhoneStatus
CiscoIPPhoneText	CiscoIPPhoneStatusFile
CiscoIPPhoneInput	
CiscoIPPhoneDirectory	
CiscoIPPhoneImage	
CiscoIPPhoneImageFile	
CiscoIPPhoneGraphicMenu	
CiscoIPPhoneGraphicFileMenu	
CiscoIPPhoneIconMenu	
CiscoIPPhoneIconFileMenu	

**Note**

Support for the Application Event Handlers requires an updated XML Parser.

Related Topics

[Application](#), on page 98

[Supported Phone Models](#), on page 57

[Updated XML Parser and Schema Enforcement](#), on page 135

Application Event Handler Attributes

The Application Event Handlers can be attached to a supported object by specifying the attributes described in the following table.

**Note**

An Application URI with Priority=0 is not allowed in the Application Event Handlers.

Table 11: Application Event Handler Attributes

Attribute	Description
appID	Identifies the application to which this displayable object belongs. The format of the appID attribute should be in the format vendor/product, such as Cisco/Unity, but this syntax is not enforced, and the application can assign any unique identifier.
onAppFocusLost	<p>Invoked when the application loses focus, if one of the following conditions occurs:</p> <ul style="list-style-type: none"> • The application context loses focus • The application was navigated away from, either directly by the user, or programmatically by a refresh header or HTTP push <p>Note If a Notify URI is used as the event handler, a notification is sent with this default data: <code><notifyApplicationEvent appId="appId" type="focusLost"/></code></p>
onAppFocusGained	<p>Invoked when the application gains focus, if one of the following conditions occurs:</p> <ul style="list-style-type: none"> • The application is Active and the application context gains focus • The application was navigated to, either directly by the user, or by a refresh header or HTTP push <p>Note If a Notify URI is used as the event handler, a notification is sent with this default data: <code><notifyApplicationEvent appId="appId" type="focusGained"/></code></p>
onAppMinimized	<p>Invoked when the application is minimized.</p> <p>An application can only be minimized in a program by a call to App:Minimize, but this invocation could occur by direct action of the user (for example, from a softkey invocation) or from the application using a push request. <code><notifyApplicationEvent appId="appId" type="minimized"/></code></p>
onAppClosed	<p>Invoked when the application closes, if one of the following conditions occur:</p> <ul style="list-style-type: none"> • The application context is closed which will, in turn, close all applications in its stack • The application no longer exists on the context URL stack because it was navigated away from, or because it was pruned from the URL stack (stack size exceeded) <p>Note This event handler cannot contain HTTP or HTTPS URLs.</p> <p>Note If a Notify URI is used as the event handler, a notification is sent with this default data: <code><notifyApplicationEvent appId="appId" type="closed"/></code></p>

Related Topics[Application, on page 98](#)

Event Handler Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="notifyApplicationEvent">
    <xs:complexType>
      <xs:attribute name="appId" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
            <xs:maxLength value="64"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="type" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="closed"/>
            <xs:enumeration value="minimized"/>
            <xs:enumeration value="focusLost"/>
            <xs:enumeration value="focusGained"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Event Handler Example

```

<CiscoIPPhoneImage appId="Cisco/Unity"
  onAppFocusLost="RTPTx:Stop; RTPTx:Stop; Notify:http:server:80:path"
  onAppFocusGained="http://server/mainpage/updateUI"
  onAppClosed="Notify:http:server:80:eventlistener/appClosed">
  ...
</CiscoIPPhoneImage>

```




Component APIs

- [Component API Overview, page 57](#)
- [Supported Phone Models, page 57](#)
- [Application Management API, page 60](#)
- [RTP Streaming API, page 61](#)
- [Errors and Responses, page 63](#)

Component API Overview

In addition to the primary phone XSI API, the following two additional component APIs are available:

- Application Management API
- RTP Streaming API

Supported Phone Models

The following table lists the Cisco Unified IP Phone models that support the component APIs.



Note

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Table 12: Phone Models that Support the Component APIs

Phone model	Supported, not supported	Firmware supported (see note 1)
Cisco Unified IP Phone 9900 Series		
9971	Supported	9.1(1) or later

Phone model	Supported, not supported	Firmware supported (see note 1)
9951	Supported	9.1(1) or later
Cisco Unified IP Phone 8900 Series		
8941, 8945	Supported (see note 2)	9.3(1) or later
8961	Supported	9.1(1) or later
Cisco IP Phone 8800 Series		
8811	Supported	10.2(2) or later
8841	Supported	10.2(1) or later
8845	Supported	10.3(2) or later
8851	Supported	10.2(1) or later
8851NR	Supported	10.3(1) or later
8861	Supported	10.2(1) or later
8865	Supported	10.3(2) or later
Cisco IP Conference Phone		
8831	Not supported	—
Cisco Wireless IP Phone 8820 Series		
8821	Not supported	—
Cisco Unified IP Phone 7900 Series		
7905G	Not supported	—
7906G	Supported	8.3(2) or later
7911G	Supported	8.3(2) or later
7912G	Not supported	—
7931G	Supported	8.3(2) or later
7937G	Not supported	—
7940G	Not supported	—

Phone model	Supported, not supported	Firmware supported (see note 1)
7941G, 7941G-GE	Supported	8.3(2) or later
7942G	Supported	8.3(2) or later
7945G	Supported	8.3(2) or later
7960G	Not supported	—
7961G, 7961G-GE	Supported	8.3(2) or later
7962G	Supported	8.3(2) or later
7965G	Supported	8.3(2) or later
7970G	Supported	8.3(2) or later
7971G-GE	Supported	8.3(2) or later
7975G	Supported	8.3(2) or later
7985G	Not supported	—
Cisco Unified Wireless IP Phone 7900 Series		
7920	Not supported	—
7921G	Not supported	—
7925G, 7925G-EX	Not supported	—
7926G	Not supported	—
Cisco IP Phone 7800 Series		
7811	Not supported	—
7821	Not supported	—
7841	Not supported	—
7861	Not supported	—
Cisco Unified IP Phone 6900 Series		
6921	Not supported	—
6941	Not supported	—

Phone model	Supported, not supported	Firmware supported (see note 1)
6945	Not supported	—
6961	Not supported	—
Other devices		
Cisco IP Phone Communicator	Supported	7.0 or later

**Note**

- 1 Cisco recommends the use of latest firmware. The firmware can be downloaded from the following location (requires login or service contract):
<http://software.cisco.com/download/navigator.html?i=!mmd>
- 2 Cisco Unified IP Phones 8941 and 8945 support RTP Streaming API in firmware 9.3(1) or later. The Cisco Unified IP Phones 8941 and 8945 do not support Application Event Handlers (appID, onAppFocusLost, onAppFocusGained, onAppMinimized, onAppClosed)

Related Topics

[Deprecated Endpoints](#), on page 3

Application Management API

To address the limited application management, the Application Management API provides a smoother handoff between the call mode and the application mode. The Application API consists of two primary components:

- Application URI
- Application Event Handlers

**Note**

Support for the Application Management API requires an updated XML Parser.

Related Topics

[Application Event Handlers](#), on page 52

[Application](#), on page 98

[Updated XML Parser and Schema Enforcement](#), on page 135

RTP Streaming API

This XML-based RTP Streaming API allows applications to initiate and observe RTP audio streams. This API extends capabilities beyond the legacy RTP streaming URIs by providing support for stream start and stop event listeners and the ability to specify other extended stream attributes, such as codec type.



Note

Support for the RTP Streaming API requires an updated XML Parser.

The event handlers typically use the standard Notification framework, but they can also invoke most other URIs, with the exception of HTTP URLs.

Related Topics

[Updated XML Parser and Schema Enforcement](#), on page 135

[Notify](#), on page 96

Interaction Rules with Legacy RTP URI Streams

The RTP Streaming API allows a full-duplex stream (mode=sendReceive) to be set up as a single stream request, which simplifies the usage of the API. However, in some cases, this API creates some interoperability issues with the legacy RTP URIs because the legacy RTP URIs send and receive streams separately. The interaction rules between legacy RTP URI streams and the new RTP Streaming API are:

- If an RTP Stop URI is invoked, and an RTP Streaming API stream is currently streaming in that same direction, then the entire RTP Streaming API stream is stopped.

For example, if a full-duplex stream is set up through the RTP Streaming API (mode=sendReceive) and then an RTPTx:Stop URI is invoked, the stream will be stopped in both the send and receive directions (and the onStopped event handler will be called, if present).

- If the stopMedia request (from the RTP Streaming API) does not specify a stream ID, then the request will stop all services RTP streams, in any direction (send or receive) and of any type (multicast and unicast). This allows applications using the RTP Streaming API to stop media streams which may have been started by the legacy RTP URIs or by other applications for which a stream ID is not known.

Error Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="errorResponse">
    <xs:complexType>
      <xs:all>
        <xs:element name="type">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="InvalidURL"/>
              <xs:enumeration value="InvalidResource"/>
              <xs:enumeration value="InvalidResourceID"/>
              <xs:enumeration value="UnavailableResource"/>
              <xs:enumeration value="InvalidXML"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:all>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```

    </xs:element>
    <xs:element name="data" nillable="true">
      <xs:simpleType>
        <xs:restriction base="xs:string"/>
      </xs:simpleType>
    </xs:element>
  </xs:all>
</xs:complexType>
</xs:element>
</xs:schema>

```

RTP Streaming API Examples

The following examples show how to work with the RTP Streaming API.

Start Media Example

- Request

```

HTTP POST /CGI/Execute
<startMedia>
  <mediaStream
    onStopped="Notify:http:server:80:path/page"
    receiveVolume="50">
    <type>audio</type>
    <codec>G.729</codec>
    <mode>sendReceive</mode>
    <address>239.1.1.3</address>
    <port>20480</port>
  </mediaStream>
</startMedia>

```

- Response

```

HTTP200 OK
<mediaStream id="abc123"/>

```

Stop Media Example

- Request

```

HTTP POST CGI/Execute
<stopMedia>
  <mediaStream id="abc123"/>
</stopMedia>

```

- Response

```

HTTP 200 OK

```

If the user terminates the media stream by placing the active audio path on-hook, the following notification is sent:

```

HTTP POST /server/path/page
DATA=<notifyMediaEvent type="stopped" origin="user">
  <mediaStream id="abc123"/>
</notifyMediaEvent>

```

Errors and Responses

The following table describes error conditions and responses for the RTP Streaming API.

Table 13: RTP Streaming API Error Conditions and Responses

Condition	Applicable method	HTTP result code	Type	Data
Authorization failed	all	401 (Authorization Failed)	N/A	N/A
Request object does not comply with the API's XML schema	all	400 (BadRequest)	InvalidXML	<parser error description>
Media cannot be started because no DSP resources is available to handle the media	startMedia	400 (BadRequest)	Unavailable Resource	No Media Resource Available
Media cannot be stopped because the specified stream ID does not exist	stopMedia	400 (BadRequest)	InvalidResourceID	Unknown Media Stream ID: <streamID>



Internal URI Features

- [Internal URI Overview, page 65](#)
- [Supported URIs by Phone Model, page 65](#)
- [Device Control URIs, page 70](#)
- [XML Displayable Object URIs, page 81](#)
- [Multimedia URIs, page 85](#)
- [Telephony URIs, page 92](#)
- [Application Management URIs, page 95](#)

Internal URI Overview

Internal uniform resource identifiers (URIs) provide access to embedded phone features such as placing calls, playing audio files, and invoking built-in object features.

Supported URIs by Phone Model

The following tables list the URIs that are supported for Release 8.0(1) and later. The notes mentioned in the tables follow the final table.

Table 14: URIs Supported by Cisco Unified IP Phone Services SDK for Cisco Unified IP Phone 6900 Series

URI	6921, 6941, 6945, 6961
Key	Supported
Softkey	Supported
Init	Supported
Dial, EditDial	Supported

URI	6921, 6941, 6945, 6961
Play	Supported
QueryStringParam	Supported
Unicast RTP	Supported
Multicast RTP	Supported
Display	Not supported
Vibrate	Not supported
Notify (see note 6)	Not supported
SendDigits (see note 6)	Not supported
Application (see note 6)	Not supported
Device	Not supported

Table 15: URIs Supported by Cisco IP Phone Services SDK for Cisco Unified IP Phone 7800 Series

URI	7811, 7821, 7841, 7861
Key	Supported
Softkey	Supported
Init	Supported
Dial, EditDial	Supported
Play	Supported
QueryStringParam	Supported
Unicast RTP	Supported
Multicast RTP	Supported
Display	Not supported
Vibrate	Not supported
Notify (see note 6)	Not supported
SendDigits (see note 6)	Not supported

URI	7811, 7821, 7841, 7861
Application (see note 6)	Not supported
Device	Not supported

**Note**

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Table 16: URIs Supported by Cisco Unified IP Phone Services SDK for Cisco Unified IP Phone 7900 Series and Cisco IP Communicator

URI	7905G, 7912G	7906G,7911G, 7931G	7920	7921G, 7925G, 7925G-EX, 7926G	7937G	7940G 7960G	7941G, 7941G-GE, 7961G, 7961G-GE, 7942G, 7962G, 7945G, 7965G, IP Communicator	7970G, 7971G-GE, 7975G
Key	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Softkey	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Init	Supported	Supported	Supported	Supported	Supported (see note 2)	Supported	Supported	Supported
Dial, EditDial	Supported (see note 3)	Supported	Supported	Supported (see note 4)	Supported	Supported	Supported	Supported
Play	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
QueryStringParam	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Unicast RTP	Supported	Supported	Supported (see note 5)	Supported	Supported	Supported	Supported	Supported
Multicast RTP	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Display	Not supported	Not supported	Not supported	Supported	Not supported	Not supported	Supported	Supported

URI	7905G, 7912G	7906G,7911G, 7931G	7920	7921G, 7925G, 7925G-EX, 7926G	7937G	7940G 7960G	7941G, 7941G-GE, 7961G, 7961G-GE, 7942G, 7962G, 7945G, 7965G, IP Communicator	7970G, 7971G-GE, 7975G
Vibrate	Not supported	Not supported	Supported	Supported	Not supported	Not supported	Not supported	Not supported
Notify (see note 6)	Not supported	Supported	Not supported	Not supported	Not supported	Not supported	Supported	Supported
SendDigits (see note 6)	Not supported	Supported	Not supported	Not supported	Not supported	Not supported	Supported	Supported
Application (see note 6)	Not supported	Supported	Not supported	Not supported	Not supported	Not supported	Supported	Supported
Device	Not supported	Not supported	Not supported	Supported	Not supported	Not supported	Not supported	Not supported

Table 17: URIs Supported by Cisco Unified IP Phone Services SDK for Cisco Unified IP Phone 8800, 8900, and 9900 Series

URI	8831	8811, 8841, 8845, 8851, 8851NR, 8861, 8865	8941, 8945	8961, 9951, 9971
Key	Supported	Supported	Supported	Supported (see note 1)
Softkey	Supported	Supported	Supported	Supported
Init	Supported	Supported	Supported	Supported
Dial, EditDial	Supported	Supported	Supported	Supported
Play	Supported	Supported	Supported	Supported
QueryStringParam	Supported	Supported	Supported	Supported
Unicast RTP	Supported	Supported	Supported	Supported
Multicast RTP	Supported	Supported	Supported	Supported
Display	Supported	Supported	Supported	Supported

URI	8831	8811, 8841, 8845, 8851, 8851NR, 8861, 8865	8941, 8945	8961, 9951, 9971
Vibrate	Not supported	Not supported	Not supported	Not supported
Notify (see note 6)	Supported	Supported	Supported	Supported
SendDigits (see note 6)	Supported	Supported	Supported	Supported
Application (see note 6)	Not supported	Supported	Not supported	Supported
Device	Not supported	Supported	Not supported	Not supported

Table 18: URIs Supported by Cisco Unified IP Phone Services SDK for Cisco Unified Wireless IP Phone 7920 Series and Cisco Wireless IP Phone 8820 Series

URI	7920	7921G, 7925G, 7925G-EX, 7926G	8821
Key	Supported	Supported	Supported
Softkey	Supported	Supported	Supported
Init	Supported	Supported	Supported
Dial, EditDial	Supported	Supported (see note 4)	Supported (see note 4)
Play	Supported	Supported	Supported
QueryStringParam	Supported	Supported	Supported
Unicast RTP	Supported (see note 5)	Supported	Supported
Multicast RTP	Supported	Supported	Supported
Display	Not supported	Supported	Supported
Vibrate	Supported	Supported	Supported
Notify (see note 6)	Not supported	Not supported	Not supported
SendDigits (see note 6)	Not supported	Not supported	Not supported

URI	7920	7921G, 7925G, 7925G-EX, 7926G	8821
Application (see note 6)	Not supported	Not supported	Not supported
Device	Not supported	Supported	Supported

**Note**

- 1 Key:Info, Key:Services, Key:Directories, Key:Settings, Key:AppMenu, and Key:Hold are not supported by Cisco Unified IP Phones 8961, 9951, and 9971.
- 2 Cisco Unified IP Phones 7905G and 7912G support only Dial:N format.
- 3 Only Init:CallHistory is supported. Does not support Init:Services, Init:Messages, and Init:Directories.
- 4 Cisco Unified Wireless IP Phones 7921G, 7925G, 7925G-EX, and 7926G and the Cisco Wireless IP Phone 8821 support only the Dial:N format.
- 5 Only supports one incoming and one outgoing unicast stream and does not support the Volume parameter for RTP Receive streams.
- 6 Requires Cisco Unified IP Phone firmware version 8.3(2) or later, which contains an updated XML parser.

Related Topics

[Deprecated Endpoints, on page 3](#)

[Updated XML Parser and Schema Enforcement, on page 135](#)

Device Control URIs

These sections describe the device control URIs.

Key

The *Key* URI allows a program to send an event that a key has been pressed. The system initiates the event as if the button was physically pressed.

Note that when buttons are pressed with this method, if the button is not present on the phone (hard button) or not available (softkey) when the URI is processed, the event is discarded.

If the softkey set is changing and disabled while the event is being processed, the request is discarded.

The following tables list the *Key* URIs and the phone models in which these softkeys are supported. The notes mentioned in the tables follow the final table.

**Note**

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Table 19: Key URIs with Supported Phone Models: Cisco Unified IP Phone 6900 Series

Key URIs	6921, 6941, 6945, 6961
Key:Applications	Not supported
Key:AppMenu	Not supported
Key:Contacts	Supported
Key:Directories	Not supported
Key:Feature1 to Key:Feature120	Supported
Key:FixedFeature1 to 3	Supported
Key:Headset	Supported
Key:Hold	Not supported
Key:Info	Not supported
Key:KemPage	Not supported
Key:KeyPad0 to Key:KeyPad9	Supported
Key:KeyPadPound	Supported
Key:KeyPadStar	Supported
Key:Line1 to Key:Line120	Supported
Key:Messages	Supported
Key:Mute	Supported
Key:NavBack	Not supported
Key:NavDwn	Supported
Key:NavLeft	Not supported
Key:NavRight	Not supported

Key URIs	6921, 6941, 6945, 6961
Key:NavSelect	Supported
Key:NavUp	Supported
Key:Offhook	Not supported
Key:Onhook	Not supported
Key:PTT	Not supported
Key:Release	Not supported
Key:Services	Not supported
Key:Session1 to Key:Session6	Not supported
Key:Settings	Supported
Key:Soft1 to Key:Soft5	Supported (see note 7)
Key:Speaker	Supported
Key:VolDwn	Supported
Key:VolUp	Supported

Table 20: Key URIs with Supported Phone Models: Cisco IP Phone 7800 Series

Key URIs	7811, 7821, 7841, 7861 (See Note 10)
Key:Applications	Supported
Key:AppMenu	Not supported
Key:Contacts	Not supported
Key:Directories	Supported
Key:Feature1 to Key:Feature120	Not supported
Key:FixedFeature1 to 3	Not supported
Key:Headset	Supported
Key:Hold	Supported

Key URIs	7811, 7821, 7841, 7861 (See Note 10)
Key:Info	Supported
Key:KemPage	Not supported
Key:KeyPad0 to Key:KeyPad9	Supported
Key:KeyPadPound	Supported
Key:KeyPadStar	Supported
Key:Line1 to Key:Line120	Supported
Key:Messages	Supported
Key:Mute	Supported
Key:NavBack	Not supported
Key:NavDwn	Supported
Key:NavLeft	Not supported
Key:NavRight	Not supported
Key:NavSelect	Supported
Key:NavUp	Supported
Key:Offhook	Not supported
Key:Onhook	Not supported
Key:PTT	Not supported
Key:Release	Not supported
Key:Services	Supported
Key:Session1 to Key:Session6	Not supported
Key:Settings	Supported
Key:Soft1 to Key:Soft4	Supported
Key:Speaker	Supported
Key:VolDwn	Supported

Key URIs	7811, 7821, 7841, 7861 (See Note 10)
Key:VolUp	Supported

Table 21: Key URIs with Supported Phone Models: Cisco Unified IP Phone 7900 Series and Cisco IP Communicator

Key URIs	7905G, 7912G	7906G, 7911G, 7931G	7937G	7940G, 7960G	7941G, 7941G-GE, 7942G, 7945G, 7961G, 7961G-GE, 7962G, 7965G, IP Communicator	7970G, 7971G-GE, 7975G
Key:Applications	Not supported	Supported	Not supported	Not supported	Not supported	Supported
Key:AppMenu	Not supported	Supported	Not supported	Supported	Supported	Supported
Key:Contacts	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
Key:Directories	Not supported	Not supported	Supported	Supported	Supported	Supported
Key:Feature1 to Key:Feature120	Not supported	Supported	Not supported	Supported	Supported	Supported
Key:FixedFeature1 to 3	Not supported	Supported	Not supported	Supported	Supported	Supported
Key:Headset	Not supported	Supported	Not supported	Supported	Supported	Supported
Key:Hold	Supported	Supported	Supported	Supported	Supported	Supported
Key:Info	Supported	Supported	Supported	Supported	Supported	Supported
Key:KemPage	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
Key:KeyPad0 to Key:KeyPad9	Supported	Supported	Supported	Supported	Supported	Supported
Key:KeyPadPound	Supported	Supported	Supported	Supported	Supported	Supported
Key:KeyPadStar	Supported	Supported	Supported	Supported	Supported	Supported
Key:Line1 to Key:Line120	Not supported	Supported	Supported	Supported	Supported	Supported
Key:Messages	Not supported	Supported	Not supported	Supported	Supported	Supported
Key:Mute	Not supported	Supported	Supported	Supported	Supported	Supported

Key URIs	7905G, 7912G	7906G, 7911G, 7931G	7937G	7940G, 7960G	7941G, 7941G-GE, 7942G, 7945G, 7961G, 7961G-GE, 7962G, 7965G, IP Communicator	7970G, 7971G-GE, 7975G
Key:NavBack	Not supported	Supported	Supported	Supported	Supported	Supported
Key:NavDwn	Supported	Supported	Supported	Not supported	Not supported	Supported
Key:NavLeft	Not supported	Supported	Supported	Not supported	Not supported	Supported
Key:NavRight	Not supported	Supported	Supported	Not supported	Not supported	Supported
Key:NavSelect	Not supported	Supported	Supported	Not supported	Not supported	Supported
Key:NavUp	Supported	Supported	Supported	Supported	Supported	Supported
Key:Offhook	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
Key:Onhook	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
Key:PTT	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
Key:Release	Not supported	Supported	Not supported	Supported	Supported	Supported
Key:Services	Supported	Not supported	Not supported	Supported	Supported	Supported
Key:Session1 to Key:Session6	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
Key:Settings	Not supported	Supported	Not supported	Supported	Supported	Supported
Key:Soft1 to Key:Soft5	Supported	Supported	Supported (see note 6)	Supported	Supported	Supported
Key:Speaker	Not supported	Supported	Supported	Supported	Supported	Supported
Key:VolDwn	Supported	Supported	Supported	Supported	Supported	Supported
Key:VolUp	Supported	Supported	Supported	Supported	Supported	Supported

Table 22: Key URIs with Supported Phone Models: Cisco Unified IP Phone 8800, 8900, and 9900 Series

Key URIs	8831	8811, 8841, 8845, 8851, 8851NR, 8861, 8865	8941, 8945	8961, 9951, 9971
Key:Applications	Not supported	Supported	Supported (see note 1)	Supported
Key:AppMenu	Not supported	No supported	Not supported	Not supported
Key:Contacts	Not supported	Supported	Not supported	Not supported
Key:Directories	Not supported	Supported (see note 11)	Not supported	Not supported
Key:Feature1 to Key:Feature120	Not supported	Supported (see note 9)	Supported (see note 2)	Supported
Key:FixedFeature1 to 3	Not supported	Supported	Supported	Supported
Key:Headset	Not supported	Supported	Supported	Supported
Key:Hold	Not supported	Supported	Not supported	Not supported
Key:Info	Not supported	Not supported	Not supported	Not supported
Key:KemPage	Not supported	Supported	Not supported	Not supported
Key:KeyPad0 to Key:KeyPad9	Supported	Supported	Supported	Supported
Key:KeyPadPound	Supported	Supported	Supported	Supported
Key:KeyPadStar	Supported	Supported	Supported	Supported
Key:Line1 to Key:Line120	Supported	Supported (see note 9)	Supported (see note 3)	Supported
Key:Messages	Not supported	Supported	Supported	Supported
Key:Mute	Not supported	Supported	Supported	Supported
Key:NavBack	Not supported	Not supported	Supported (see note 4)	Supported
Key:NavDwn	Not supported	Supported	Supported	Supported
Key:NavLeft	Not supported	Supported	Not supported	Supported

Key URIs	8831	8811, 8841, 8845, 8851, 8851NR, 8861, 8865	8941, 8945	8961, 9951, 9971
Key:NavRight	Not supported	Supported	Not supported	Supported
Key:NavSelect	Not supported	Supported	Supported	Supported
Key:NavUp	Not supported	Supported	Supported	Supported
Key:Offhook	Not supported	Not supported	Not supported	Not supported
Key:Onhook	Not supported	Not supported	Not supported	Not supported
Key:PTT	Not supported	Not supported	Not supported	Not supported
Key:Release	Not supported	Supported	Not supported	Supported
Key:Services	Not supported	Supported	Not supported	Not supported (See note 8)
Key:Session1 to Key:Session6	Not supported	Supported (see note 9)	Not supported	Supported
Key:Settings	Not supported	Supported	Not supported (see note 5)	Not supported
Key:Soft1 to Key:Soft5	Supported	Supported	Supported (see note 7)	Supported
Key:Speaker	Supported	Supported	Supported	Supported
Key:VolDwn	Supported	Supported	Supported	Supported
Key:VolUp	Supported	Supported	Supported	Supported

Table 23: Key URIs with Supported Phone Models: Cisco Unified Wireless IP Phone 7920 Series, Cisco Wireless IP Phone 8820 Series

Key URIs	7920	7921G, 7925G, 7925G-EX, 7926G	8821
Key:Applications	Supported	Not supported	Not supported
Key:AppMenu	Supported	Not supported	Not supported
Key:Contacts	Not supported	Not supported	Not supported
Key:Directories	Supported	Not supported	Not supported
Key:Feature1 to Key:Feature120	Supported	Not supported	Not supported

Key URIs	7920	7921G, 7925G, 7925G-EX, 7926G	8821
Key:FixedFeature1 to 3	Supported	Not supported	Not supported
Key:Headset	Supported	Not supported	Not supported
Key:Hold	Supported	Supported	Supported
Key:Info	Supported	Supported	Supported
Key:KemPage	Not supported	Not supported	Not supported
Key:KeyPad0 to Key:KeyPad9	Supported	Supported	Supported
Key:KeyPadPound	Supported	Supported	Supported
Key:KeyPadStar	Supported	Supported	Supported
Key:Line1 to Key:Line120	Supported	Not supported	Not supported
Key:Messages	Supported	Not supported	Not supported
Key:Mute	Supported	Supported	Supported
Key:NavBack	Supported	Not supported	Not supported
Key:NavDwn	Supported	Supported	Supported
Key:NavLeft	Supported	Supported	Supported
Key:NavRight	Supported	Supported	Supported
Key:NavSelect	Supported	Supported	Supported
Key:NavUp	Supported	Supported	Supported
Key:Offhook	Not supported	Supported	Supported
Key:Onhook	Not supported	Supported	Supported
Key:PTT	Not supported	Supported	Supported
Key:Release	Supported	Not supported	Not supported
Key:Services	Supported	Not supported	Not supported
Key:Session1 to Key:Session6	Not supported	Not supported	Not supported
Key:Settings	Supported	Not supported	Not supported

Key URIs	7920	7921G, 7925G, 7925G-EX, 7926G	8821
Key:Soft1 to Key:Soft5	Supported See note 12.	Supported See note 12.	Supported See note 12.
Key:Speaker	Supported	Supported	Supported
Key:VolDwn	Supported	Supported	Supported
Key:VolUp	Supported	Supported	Supported

**Note**

- 1 Cisco Unified IP Phones 8941 and 8945 support Key:Applications in firmware 9.3(1) and later.
- 2 Cisco Unified IP Phones 8941 and 8945 support four features.
- 3 Cisco Unified IP Phones 8941 and 8945 supports four lines.
- 4 Cisco Unified IP Phones 8941 and 8945 support Key:NavBack in firmware 9.3(2) and later, and only if a back softkey is on the screen.
- 5 Cisco Unified IP Phones 8941 and 8945 support Key:Settings for firmware 9.2(3) and before.
- 6 Cisco Unified IP Phone 7937 supports four softkeys.
- 7 These phones support four softkeys.
- 8 Cisco Unified IP Phones 8961, 9951, and 9971 do not support the Key:Services URI because the phones do not have a Services button. Applications must use the Init:Services and App:Close URIs. See [Unsupported Key URIs and Alternate Options, on page 80](#)
- 9 The Cisco IP Phone 8811, 8841, and 8845 support 5 lines, 5 sessions, and 5 features. The Cisco IP Phones 8851 and 8851NR support 77 lines, 5 sessions, and 77 features. The Cisco IP Phone 8861 and 8865 supports 113 lines, 5 sessions, and 113 features.
- 10 The Cisco IP Phone 7811 supports 1 line. It does not have feature buttons.
- 11 Key:Directories is supported on the Cisco IP Phone 8800 series starting with Firmware Release 11.0
- 12 The wireless phones have only 2 softkeys.

Related Topics

[Deprecated Endpoints, on page 3](#)

Key URI Format

Key:n

Where

n = a Key name

Unsupported Key URIs and Alternate Options

This section describes the unsupported *Key* URIs in the phone models and provides alternative options, if any, for the unsupported URIs.

Table 24: Unsupported Key URIs and Alternative Options

Phone models	Unsupported URI	Description and alternatives
8961, 9951, 9971	Key:Services	<p>The Cisco Unified IP Phones 8961, 9951, and 9971 do not have a Services button. Therefore, the <i>Key:Services</i> URI is not supported in these phones.</p> <p>The application must use the <i>Init:Services</i> URI and the <i>App:Close</i> URI to close the last XSI application launched from the application. If there is no application open, then the request has no effect.</p> <p>Additionally, the Exit softkey takes the application to the previous screen, and if the application is at the top level, the Exit softkey closes the application.</p>
8961, 9951, 9971	Key:Info	The Cisco Unified IP Phones 8961, 9951, and 9971 do not have a standalone help application. Help is provided within the context of each application.
8961, 9951, 9971	Key:Directories	In the Cisco Unified IP Phones 8961, 9951, and 9971, the <i>Key:Contacts</i> URI replaces the <i>Key:Directories</i> URI. You can use <i>Key:Contacts</i> to invoke the new contacts application in these phones.
8961, 9951, 9971	Key:Settings	The Cisco Unified IP Phones 8961, 9951, and 9971 do not have a single monolithic settings application. Therefore the <i>Key:Settings</i> URI is not supported in these phones.
8961, 9951, 9971	Key:AppMenu	The Cisco Unified IP Phones 8961, 9951, and 9971 do not support the <i>Key:AppMenu</i> URI. All applications are accessed using their individual <i>Key</i> URIs like Applications, Contacts, and Messages.

Phone models	Unsupported URI	Description and alternatives
6921, 6941, 6945, 6961, 8961, 9951, 9971	Key:Hold	<p>The Cisco Unified IP Phone 6900 Series and Cisco Unified IP Phones 8961, 9951, and 9971 do not support the <i>Key:Hold</i> URI.</p> <p>The <i>Key</i> URI equivalents for invoking the standard fixed features are:</p> <ul style="list-style-type: none"> • To invoke transfer, use <i>Key:FixedFeature1</i>. • To invoke conference, use <i>Key:FixedFeature2</i>. • To invoke hold, use <i>Key:FixedFeature3</i>. <p>The fixed feature keys in Cisco Unified IP Phones 8961, 9951, and 9971 are field replaceable.</p>

Display

The *Display* URI is available only on those Cisco Unified IP Phones that have a color backlight on the phone display. Using the *Display* URI, you can control how long the backlight remains on or off.

Note, however, that other administrator-controlled or user-indicated display settings take precedence over the *Display* URI. Therefore, various phone states (such as phone startup, incoming and active calls, or other user input states) override the *Display* URI settings.

Display URI Format

Display:State:Interval

Where

State = whether the phone display is turned on or off, or set to default to return the display to its specified state.

Interval = duration (in minutes) in which the phone state remains in the specified state (unless activated by automated or user input). Value must be an integer ranging from 0-1440 minutes. If the value is set to 0, the display remains in the indicated state indefinitely (unless activated by automated or user input).

Examples

Display:Off:60 turns the phone display off for 1 hour (60 minutes).

Display:On:10 turns the phone display on for 10 minutes.

Display:Off:0 turns off the display until activated.

Display:Default returns the display to its specified state for that time.

XML Displayable Object URIs

These sections describe the XML displayable object URIs.

SoftKey

You can execute native softkey functionality when the phone executes a *SoftKey* URI. The *SoftKey* URI allows developers to customize softkey names and layout in the Services and Directories windows while retaining the functionality that the softkeys provide.

SoftKey URIs work in menu items and in softkey items in the XML objects for which they natively occur on the phone.


Note

The Softkey URI is not supported in the Execute object.

SoftKey URI Format

SoftKey:n

Where

n = one of the following softkey names:

- Back
- Cancel
- Exit
- Next
- Search
- Select
- Submit
- Update
- Dial
- EditDial
- <<

The following table contains valid softkey actions for each XSI object type follow. The URI invokes the native functionality that each key possesses in the given object context.

Table 25: Valid softkey actions for CiscoIPPhoneObject types


IPPhoneObject (see note 1)	Select	Exit	Update	Submit	Search	<<	Cancel	Next	Dial	EditDial
CiscoIPPhoneMenu	X	X								
CiscoIPPhoneIconMenu	X	X								
CiscoIPPhobneText		X	X							

IPPhoneObject (see note 1)	Select	Exit	Update	Submit	Search	<<	Cancel	Next	Dial	EditDial
CiscoIPPhoneImage		X	X							
CiscoIPPhoneGraphicMenu		X	X							
CiscoIPPhoneInput				X	X (see note 2)	X	X			
CiscoIPPhoneDirectory							X	X	X (see note 3)	X (see note 3)

**Note**

- 1 The *SoftKey* URI is not allowed in an Execute object.
- 2 Only when used under the Directories button.
- 3 The *SoftKey:Dial* and *SoftKey:EditDial* URIs can be used only for Directory objects, but the *Dial:xxx* and *EditDial:xxx* URIs can be used as the URL of any *SoftKeyItem* or *MenuItem*.

The Cisco Unified IP Phones 8961, 9951, and 9971 have the following enhancements to their display:

- The positions of the softkeys have been changed. Moving from left to right, the Exit is the first softkey followed by the Submit, Select, Update, or Next softkey, and finally the Delete softkey.
- In the submenu screens, the back arrow icon () replaces the << or Exit softkeys, and it is placed in the first (extreme left) position.
- The phone displays error messages, like XML Parse error or HTTP failures, in a new window.

Related Topics

[Telephony URIs, on page 92](#)

QueryStringParam

The *QueryStringParam* URI allows an application developer to collect more information from the user with less interaction. When the user performs an action with a softkey, you can either append a query string parameter to the URL of the highlighted *MenuItem* or append the query string parameter from the *MenuItem* to the URL of the softkey.

QueryStringParam URI Format

QueryStringParam:d

Where

d = the data to be appended to a corresponding URL.

Example: QueryStringParam URI in CiscoIPPhoneMenu Object

```
<CiscoIPPhoneMenu>
<Title>Message List</Title>
<Prompt>Two Messages</Prompt>
<MenuItem>
  <Name>Message One</Name>
  <URL>QueryStringParam:message=1</URL>
</MenuItem>
<MenuItem>
  <Name>Message Two</Name>
  <URL>QueryStringParam:message=2</URL>
</MenuItem>
<SoftKeyItem>
  <Name>Read</Name>
  <URL>http://server/read.asp</URL>
</SoftKeyItem>
<SoftKeyItem>
  <Name>Delete</Name>
  <URL>http://server/delete.asp</URL>
</SoftKeyItem>
</CiscoIPPhoneMenu>
```

Example: Item Selection with Numeric Keypad Calls URI

The following example shows how to use the *QueryStringParam* URI in a *CiscoIPPhoneMenu* object. The *CiscoIPPhoneMenu* object includes two *MenuItems* with *QueryStringParam* URIs. If the user chooses the *MenuItems* with the numeric keypad, the cursor moves to that entry, but nothing executes because the values are *QueryStringParam* URIs.

If the user presses either custom softkey, the currently highlighted *MenuItem* URI value gets appended to the softkey URL that was pressed and requested from the web server.

If the user highlights the first *MenuItem* and press the Read softkey, the phone generates the following URL:

http://server/read.asp?message=1

```
<CiscoIPPhoneMenu>
<Title>Message List</Title>
<Prompt>Two Messages</Prompt>
<MenuItem>
  <Name>Message One</Name>
  <URL>http://server/messages.asp?message=1</URL>
</MenuItem>
<MenuItem>
  <Name>Message Two</Name>
  <URL>http://server/messages.asp?message=2</URL>
</MenuItem>
<SoftKeyItem>
  <Name>Read</Name>
  <Position>1</Position><URL>QueryStringParam:action=read</URL>
</SoftKeyItem>
<SoftKeyItem>
  <Name>Delete</Name>
  <Position>2</Position><URL>QueryStringParam:action=delete</URL>
</SoftKeyItem>
</CiscoIPPhoneMenu>
```

QueryStringParam URI Example Discussion

The Cisco Unified IP Phones allow you to implement the *QueryStringParam* URI in either manner although [Example: Item Selection with Numeric Keypad Calls URI](#), on page 84 is not as efficient as [Example:](#)

[QueryStringParam URI in CiscoIPPhoneMenu Object, on page 84](#). Choose the best way to perform the action based on your applications needs.

The Item selection example has a slight advantage in that if the user chooses an item with the numeric keypad, the URL gets called. This action would allow you to invoke some default behavior, such as to read the message in the example. By highlighting the first message and pressing the Read softkey, the phone creates the following URL: *http://server/messages.asp?message=1&action=read*

Using the *QueryStringParam* URI reduces the size of the XML objects that you generate by removing redundant portions of a URL in every *MenuItem*.

Multimedia URIs

These sections describe the multimedia URIs.

RTP Streaming

You can invoke RTP streaming using URIs in services. You can instruct the phone to transmit or receive an RTP stream with the following specifications:

- RTPRx
- RTPTx
- RTPMRx
- RTPMTx

**Note**

For some Cisco Unified IP Phone models, the RTP Streaming URIs have been deprecated by the RTP Streaming API.

The supported format of the RTP stream is:

- The codec is G.711 mu-law.
- The packet size is 20 ms.

The possible CiscoIPPhoneError codes are:

- Error 1 = Error parsing CiscoIPPhoneExecute object
- Error 2 = Error framing CiscoIPPhoneResponse object
- Error 3 = Internal file error
- Error 4 = Authentication error

Related Topics

[RTP Streaming API, on page 61](#)

Interaction with Call Streaming

- Existing Tx or MTx URI streams are terminated if a new call begins or an existing call resumes.
- Tx or MTx URI stream requests received when a call is active are rejected with an errorNo=4 unauthorized. If a call is in a Held state (connected but not actively streaming), the Tx or MTx URI request is accepted, but will terminate if the call resumes.



Note Returning errorNo=4 allows the application to distinguish this error from the normal errorNo=1 busy response.

- Existing Rx or MRx URI streams are terminated if a new call begins or an existing call resumes.
The user has no explicit mechanism for terminating the Rx or MRx URI stream independent of the call. Thus, if the Rx or MRx stream is not terminated automatically, it would continue to play. For example, a user is listening to Internet radio feed and gets an incoming call. The user answers the call, which either closes or minimizes the Internet radio XSI application. Otherwise, the user has no intuitive way to stop the music stream.
- New Rx or MRx URI stream requests received during an active call are accepted (whisper), but the volume parameter of the URI is ignored.

If the Rx or MRx URI request was done using push, then the associated application is responsible for using push Priority attributes and for stopping and starting the stream.

If the user initiates the Rx or MRx URI using an application, then the user likely is not concerned about having the audio mixed with the current call. However, the user should also be presented with an option to stop the application, when needed.

- For the Rx or MRx URI, the Mute indicator light is only lit when both these conditions are met:
 - There are no active transmit streams from either a call or an XML services stream.
 - There is at least one active receive stream.

For example, if an active call is ended or put on hold while a Rx or MRx URI stream is active, the Mute indicator will light.

- If a Rx or MRx or Tx or MTx URI request is received and there is already an active XML services stream in that direction, then a response with errorNo=1 Tx/Rx is already active is returned. The previous stream must be terminated (either by the user or by an RTP Stop URI) before a new stream can be started.

This response provides visibility to the application if the phone is currently busy. It then allows the application to decide whether or not to terminate the existing stream and start a new one, rather than being controlled by the phone firmware.

RTPRx

The *RTPRx* URI instructs the phone to receive a Unicast RTP stream or to stop receiving Unicast or Multicast RTP streams.

RTPRx URI Formats

RTPRx:i:p:v

RTPRx:Stop

Where

i = the IP Address from which the stream is coming.

p = the UDP port on which to receive the RTP stream. Ensure that this is an even port number within the decimal range of 20480 to 32768. If no port is specified, the phone chooses a port and returns it when initiated by a push request.

Stop = the parameter that will stop any active RTP stream from being received on channel one

v = the optional volume setting that controls the volume of stream play out. The supplied value is a percentage of the maximum volume level of the device and must be in the range 0-100. The phone converts the specified percentage into the closest device-supported volume level setting and uses it. After the initial volume level gets set and the stream starts, you can manually change the volume level as needed. If the optional volume parameter does not get included, the current volume setting on the phone gets used as the default.

RTPTx

Use the *RTPTx* URI to instruct the phone to transmit a Unicast RTP stream or to stop transmitting Unicast or Multicast RTP streams.

RTPTx URI Formats

RTPTx:i:p

RTPTx:Stop

Where

i = the IP Address to which an RTP stream is transmitted.

p = the UDP port on which to transmit the RTP stream. Ensure that this is an even port number within the decimal range of 20480 to 32768.

Stop = the parameter that will stop any active RTP stream from being received on channel one

RTPMRx

The *RTPMRx* URI instructs the phone to receive a Multicast RTP.

RTPMRx URI Format

RTPMRx:i:p:v

Where

i = the Multicast IP Address from which to receive an RTP stream. For information on selecting a Multicast IP Address, see the *Cisco Unified Communications System SRND*, the *IANA guidelines*, and your local network administration policies.

p = the Multicast UDP port from which to receive the RTP stream. Ensure that this is an even port number within the decimal range of 20480 to 32768.

v = the optional volume setting that controls the volume of stream play out. The supplied value is a percentage of the maximum volume level of the device and must be in the range 0-100. The phone converts the specified percentage into the closest device-supported volume level setting and uses it. After the initial volume level gets set and the stream starts, you can manually change the volume level as needed. If the optional volume parameter does not get included, the current volume setting on the phone gets used as the default.

RTPMTx

The *RTPMTx* URI instructs the phone to transmit a Multicast RTP stream.

RTPMTx URI Formats

RTPMTx:i:p

Where

i = the Multicast IP Address to which an RTP stream is transmitted. For information on selecting a Multicast IP Address, see the Cisco Unified Communications System SRND, the IANA guidelines, and your local network administration policies.

p = the Multicast UDP port on which to transmit the RTP stream. Ensure that this is an even port number within the decimal range of 20480 to 32768.

Play

The *Play* URI downloads an audio file from the TFTP server and plays through the phone speaker. This same mechanism also plays ring files, and the format of the files is the same. You could use the *Play* URI to play files that are in the Ringlist.xml or those that are not. If the phone is equipped with an message waiting light, the light will flash while the audio file is playing, providing a visual alert as well.



Note

The *Play* URI is a synchronous request. If the request is pushed to the phone using HTTP, the HTTP response (*CiscoIPPhoneResponse* object) is not returned until after the playback has completed.

Play URI Interaction with Incoming Calls

The *Play* URI and incoming calls (ringing) have equal priority access to the DSP ringer resources resulting in the following interactions:

- If a *Play* URI is currently playing, an incoming call (ringing) will not preempt the *Play* URI; the *Play* URI will finish playing first.
- If the phone is ringing and a *Play* URI request is sent to the phone, the execution of the *Play* URI defers until the phone stops ringing (the DSP ringer resource becomes available) and then the *Play* URI will play.

Play URI Format

Play:f

Where

f = the filename of a raw audio file in the TFTP path (such as Play:Classic2.raw).

The audio files for the rings must meet the following requirements for proper playback on Cisco Unified IP Phones:

- Raw PCM (no header)
- 8000 samples per second
- 8 bits per sample
- uLaw compression
- Maximum ring size: 16080 samples
- Minimum ring size: 240 samples
- Number of samples in the ring is evenly divisible by 240.
- Ring starts and ends at the zero crossing.

To create PCM files for custom phone rings, you can use any standard audio editing packages that support these file format requirements.

XSI Audio Path Control

The XSI Audio Path Control feature enables XSI calls to specify if the audio is played on the speakerphone or handset speaker of the phone. The feature is available on the following phones:

- Cisco Unified Wireless IP Phones 7921G, 7925G, 7925G-EX, and 7926G with release 1.4(4) and later

**Note**

In releases prior to 1.4(4), by default the audio path is set to speakerphone unless a headset is connected.

- Cisco Wireless IP Phone 8821

The XSI Audio Path Control feature utilizes the RTP URI which has been extended to give the administrator this option to specify whether audio received via XSI is played through the speaker phone or handset speaker of the Cisco IP Phone.

RTP URI Format

RTPRx:i:p:v:s or RTPMRx:i:p:v:s

Where

i = equals IP address (x.x.x.x).

p = equals UDP port (20480-32768).

v = volume (0-100).

s = specifies where the audio for an XSI call should be played.

- If $s = 0$, then the audio for the XSI call will be played to the speaker phone.
- If $s = 1$, then the audio for the XSI call will be played to the handset speaker or headset.
- If $s = 2$, then the audio for the XSI call will be played to the current audio path.
- If s is not present, then the audio for the XSI call is played to the speaker phone.

Examples

XSI Audio Path	Stream Type	RTP URI Example
Speakerphone	Unicast	RTPRx:10.0.0.10:20500 RTPRx:10.0.0.10:20500::0 RTPRx:10.0.0.10:20500:100:0
Handset/Headset	Unicast	RTPRx:10.0.0.10:20500::1 RTPRx:10.0.0.10:20500:100:1
Speakerphone	Multicast	RTPMRx:10.0.0.10:20500 RTPMRx:10.0.0.10:20500::0 RTPMRx:10.0.0.10:20500:100:0
Handset/Headset	Multicast	RTPMRx:10.0.0.10:20500::1 RTPMRx:10.0.0.10:20500:100:1

Vibrate

The *Vibrate* URI is available on the Cisco Unified Wireless IP Phones 7920, 7921G, 7925G, 7925G-EX, and 7926G, and it enables third-party applications to invoke the phone vibration capabilities for silent alerts, similar to the way in which the Play URI plays audible alerts. If the *Vibrate* parameters are not specified or if the device is unable to support custom *Vibrate* sequences, the device executes the default vibrate sequence.



Note

The Cisco Unified Wireless IP Phone 7920 is deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Related Topics

[Deprecated Endpoints, on page 3](#)

Vibrate URI Format

Vibrate:vibrateDuration:silenceDuration:count

Where

vibrateDuration = duration (in milliseconds) in which the vibrate state remains on. Value must be an integer ranging from 0-65536 milliseconds (ms).

silenceDuration = duration (in milliseconds) in which the vibrate state remains off. Value must be an integer ranging from 0-65536 ms.

count = number of times to repeat the vibrate on and off sequence.

Examples

Vibrate:1000:0:1 initiates a single vibrate for 1 second.

Vibrate:500:1500:5 initiates five vibrations, each lasting for 500 ms, followed by 1500 ms of silence.

Device

The *Device* URI instructs the device to automatically unlock the input or display interface without the user unlocking the device manually.

The *Device* URI accepts only the Unlock command.

If the device is configured to automatically lock the input or display interface, the normal idle timeout behavior applies and the device is automatically locked again.

Device URI Format

Device:{command}

Where

command = The command the device follows:

- Type: Enum
- Valid Value: Unlock
- Default-value: N/A

Device URI Example

This alert example performs the following actions:

- 1 Plays a tone on the phone
- 2 Unlocks the phone
- 3 Displays an alarm message on the phone

```
<CiscoIPPhoneExecute>
  <ExecuteItem URL="Device:Unlock"/>
```

```
<ExecuteItem URL="Play:alert.wav"/>
</CiscoIPPhoneExecute>
```

On processing the above command, the following response is sent:

```
<CiscoIPPhoneText>
<Title>Alert</Title>
<Prompt>Urgent</Prompt>
<Text>
Please go to room 1234.
</Text>
<SoftKeyItem>
<Name>Accept</Name>
<URL>http://<ip>/AlertResponse.jsp?reason=accept</URL>
</SoftKeyItem>
<SoftKeyItem>
<Name>Busy</Name>
<URL>http://<ip>/AlertResponse.jsp?reason=busy</URL>
</SoftKeyItem>
</CiscoIPPhoneText>
```

Device URI Error and Response

When the *Device* URI is invoked from an Execute object, it uses the standard URI Status and Data values in the *ResponseItems*.

Condition	Status	Data
Executed successfully	0 (Success)	Success
URI syntax is invalid	1 (Parse error)	Invalid URI
URI is not supported	6 (Internal error)	URI not found

Telephony URIs

These sections describe the telephony URIs.

Dial

The *Dial* URI initiates a new call to a specified number. The *Dial* URI invokes when it is contained in a menu item, the menu item is highlighted, and the device is taken off hook.

Activate the *Dial* URI by one of the following methods:

- Line button
- Speaker button
- Headset button
- Handset hook switch
- Normal menu item
- Softkey item selection

Dial URI Format

Dial:*{dialSequence}**[:{useAppUI}:{applicationId}[:audibleFeedback]]*

Where

dialSequence = The sequence of DTMF digits to be dialed. Commas represent 1 second pauses.

- Value Type: String
- Values: minLength=0, no maxLength, can only contain 0123456789#*ABCD and comma (,)
- Default value: N/A

useAppUI = Specifies whether or not this application will be used as the user interface for this call. A value of true will cause the application to keep UI focus when the call is made instead of switching to the Call UI application. The *appld* must be specified or this parameter will have no effect: it will always be false. This optional field is supported only on the Cisco Unified IP Phone 7900, 8800, 8900, and 9900 Series.

- Value Type: boolean
- Values: 0 or 1 (0=false 1=true)
- Default value: 0

applicationId = The unique name of the XSI web application requesting this call. This optional field is supported only on the Cisco Unified IP Phone 7900, 8800, 8900, and 9900 Series.

- Value Type: String
- Values: minLength=1, no maxLength, cannot contain semicolons – should be in the format Company/Product.
- Default value: Nil, which means this dial request will not be associated with any application

audibleFeedback = Whether or not to provide audible feedback to the user when the DTMF digits are dialed. This optional field is supported only on the Cisco Unified IP Phone 7900, 8800, 8900, and 9900 Series.

- Value Type: Boolean
- Values: 0, 1 (0=false 1=true)
- Default value: 1

EditDial

The *EditDial* URI initiates a new call to a specified number. The *EditDial* URI invokes when it is contained in a menu item and the menu item is highlighted.

Activate the *EditDial* URI by one of the following methods:

- Line button
- Speaker button
- Headset button
- Handset hook switch

- Normal menu item
- Softkey item selection

EditDial URI Format

EditDial:n

Where

n = the number dialed

Example

EditDial:1000 initiates a call to the phone with DN 1000.

SendDigits

The *SendDigits* URI instructs the phone to send a specified sequence of DTMF digits in-band within the media stream of the current active (streaming) call.

Audible feedback to the user can be enabled or disabled and an optional application ID can be specified to ensure that the DTMF digits will only be sent to the call which is associated with a specific application.

SendDigits URI Format

SendDigits:dtmfSequence:audibleFeedback::applicationId

Where

dtmfSequence = the sequence of DTMF digits to be sent. Value must contain only 0123456789#*ABCD and comma (.). The comma represents a one second pause.

audibleFeedback = indicates whether to provide audible feedback to the user as the DTMF digits are entered. Values can be 0 (false) or 1 (true).

applicationId = optional identifier of the application associated with the call which must receive the DTMF digits. Value must be 0-64 and cannot contain colons. The default value is null indicating that the active call should receive the DTMF digits, regardless of any application association.

Example

Make a call using a calling card service that implements these steps:

- 1 Connects to a 800 calling card service (using the *Dial* URI).
- 2 Application waits to give call time to connect.
- 3 Dials the destination number, ensuring that the digits can only be dialed from this application.
- 4 Pauses 2 seconds.
- 5 Dials the calling card number.
- 6 Pauses 1 second.

7 Dials the pin number.

```
<CiscoIPPhoneExecute>
  <ExecuteItem URL="Dial:918005551212:1:Cisco/Dialer"/>
</CiscoIPPhoneExecute>
<CiscoIPPhoneExecute>
  <ExecuteItem URL="SendDigits:6185551212,,987654321,1234:1:Cisco/Dialer"/>
</CiscoIPPhoneExecute>
```

SendDigits Error and Response

When the *SendDigits* URI is invoked from an *Execute* object, it uses the standard URI Status and Data values in *ResponseItems*.

Condition	Status	Data
Executed successfully	0 (Success)	Success
URI syntax is invalid	1 (Parse error)	Invalid URI
URI is not supported	6 (Internal error)	URI not found
Unable to execute URI because there currently is no active (streaming) call	6 (Internal error)	No Active Call
Unable to execute URI because the current active (streaming) call is not associated with the specified application	6 (Internal error)	No Active Call for Application
Phone is temporarily unable to execute URI due to some other transient issue	6 (Internal error)	<Failure>

Application Management URIs

These sections describe the application management URIs.

Init

The *Init* URI allows an application to initialize a feature or data with the argument that is passed with the URI.

Init URI Format

Init:o

Where

o = the Object name.

Valid object name:

- *CallHistory*: When the phone encounters an *Init:CallHistory* URI, it clears the internal call history logs that are stored in the phone. This action initializes Missed Calls, Received Calls, and Placed Calls.
- *Services*: When the phone encounters an *Init:SERVICES* URI, it closes the Services application. If Services is not currently open, it has no effect.
- *Messages*: When the phone encounters an *Init:Messages* URI, it closes the Messages application. If Messages is not currently open, it has no effect.
- *Directories*: When the phone encounters an *Init:Directories* URI, it closes the Directories application. If Directories is not currently open, it has no effect.

Notify

The *Notify* URI generates network notifications to back-end applications. This feature is most useful for XSI objects that support action handlers (such as displayable XSI objects and RTP streams). For example, use the *Notify* URI to deliver notifications to back-end applications when an XSI application is closed or when an RTP stream is terminated.

You can also specify the *Notify* URI in place of most fields that accept a generic URI, including softkeys and menu items. For example, you can call the *Notify* URI from a softkey or menu item to trigger a back-end event that does not require an interface change, such as manipulating the state of audio streams or other non-visual resources. The *Notify* URI also works in conjunction with the *QueryStringParam* URI, such that the exact contents of the *QueryStringParam* data will be used as the *Notify* URI data.

The *Notify* URI is not made in the context of an XSI application session and does not contain any HTTP cookie or session information. Thus, the back-end application cannot rely on HTTP cookies or session information to uniquely identify the client or application. Instead, the application must embed any necessary information in the *Notify* path and data fields, or leave the data field empty and rely on any default information provided by the specific event handler.



Note

The *Notify* URI is not supported in the *Execute* object.

Notify URI Format

Notify:protocol:host:port:path:credentials:data

Where

protocol = network protocol to use for the *Notify* connection; http is the only supported protocol.

host = network host designated to receive the notification. Value must be entered as a hostname or IP address.

port = network port to use for the *Notify* connection. Value must be a number from 1-65535.

path = protocol-specific information. Value cannot contain colons or semicolons.

credentials = optional protocol-specific credentials used to authenticate to the server. For HTTP, this is a base64-encoded version of *userid:password*. Value cannot contain colors or semicolons. If the credentials

parameter is not specified or if it is null, no Authorization header will be included in the request. The HTTP notification service will retry the request 3 times before failing and logging an error message.

data = optional application-specific event data. Value cannot contain semicolons.

Notify URI Examples

- Called from RTP *onStreamStopped* Event Handler, no credentials, with data:

```
Notify:http:myserver:8080:path/streamhandler?event=stopped:
:myStreamStoppedData

HTTP POST /path/streamhandler?event=stopped HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset="UTF-8"
Host: myserver:8080
Content-Length: 23

DATA=myStreamStoppedData
```

- Called from RTP *onStreamStopped* Event Handler, no credentials, no data:

```
Notify:http:server:8080:path/streamhandler?event=stopped

HTTP POST /path/streamhandler?event=stopped HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset="UTF-8"
Host: myserver:8080
Content-Length: 40

DATA=<notifyStreamStopped id="stream1"/>
```

- Called from *SoftKey*, with credentials, with data:

```
Notify:http:myserver:8080:path/streamhandler?event=stopped:
8fh4hf7s7dhf :myStreamStoppedData

HTTP POST /path/streamhandler?event=stopped HTTP/1.1
Accept: */*
Authorization: Basic 8fh4hf7s7dhf
Content-Type: application/x-www-form-urlencoded; charset="UTF-8"
Host: myserver:8080
Content-Length: 23
```

- Called from *SoftKey*, no credentials, no data

```
Notify:http:server:8080:path/streamhandler?event=stopped

HTTP POST /path/streamhandler?event=stopped HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset="UTF-8"
Host: myserver:8080
Content-Length: 5
```

- Called from *SoftKey* with *QueryStringParam* URI:

```
<CiscoIPPhoneMenu>
  <MenuItem>
    <Name>Voicemail1</Name>
    <URL>QueryStringParam:id=1</URL>
  </MenuItem>
  <MenuItem>
    <Name>Voicemail2</Name>
    <URL>QueryStringParam:id=2</URL>
  </MenuItem>
  <SoftKeyItem>
    <Name>Play</Name>
    <URL>Notify:http:vmailSrvr:8080:path/play</URL>
  </SoftKeyItem>
</CiscoIPPhoneMenu>
```

If the Voicemail2 menu item was selected when the Play softkey was pressed, the following notification would be sent:

```
HTTP POST /path/play HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset="UTF-8"
Host: vmailSrvr:8080
Content-Length: 9

DATA=id=2
```

Application

The *Application* URI is a component of the Application Management API, which provides an improved hand-off between call mode and application mode. The *Application* URI allows applications to request changes to their application or window state. Applications can request to change focus, to be minimized, or to be closed.



Note

The other component of the Application Management API is the Application Management Event Handler.

When an *Application* URI request is made, it has a specific application associated with it (not just the application context) and that action can only be taken on that specific application. The application specified in the *applId* parameter (of the displayable XML object) must be active at the time the action is requested, or an error will be returned.

This prevents open, but not active, applications which are buried on the application “stack” from closing the entire application context which would also close the active application, potentially disrupting the user's interaction with the application. This also means that if an application closes or becomes non-active (for example, if user navigates out of an application, or a new application is pushed to the context) any pending *Application* URI requests are immediately cancelled.



Note

The Cisco Unified IP Phone 6900 Series cannot add phone service under application due to hard key mapping.

Related Topics

[Application Event Handlers](#), on page 52

App URI Format

App:action:priority:idleTimer:applicationId

Where

action = action to be taken with the application. Values include:

- *RequestFocus*: Makes a request to the application manager to bring the application context (window) containing this application into focus (maximize). This is a request, not a demand, as higher priority applications may prevent the application from actually gaining focus. Applications must use *onAppFocusGained* event handlers to know when focus is actually gained.

- If the requested application is Open, but not currently Active, this request will not succeed (error response).
- If the application already has focus, the request has no effect.
- *ReleaseFocus*: Makes a request to the application manager to relinquish focus to another application context (essentially, a “move-to-back” request). Applications must use *onAppFocusLost* event handlers to know when focus is actually lost.
 - If the application does not have focus, the request has no effect.
 - If there are no other applications open (available to receive focus) then this application will retain focus.
- *Minimize*: Makes a request to the application manager to minimize the application context containing this application. This request always results in the application (eventually) being minimized. If the application has focus when this URI executes, the *onAppFocusLost* event handler will be invoked first, then the *onAppMinimize* handler.
 - If the requested application is Open, but not currently Active, this request will not succeed (error response).
 - If the application is already minimized, the request has no effect.
- *Close*: Makes a request to the application manager to close the application context containing this application.
 - If the requested application is open, but not currently active, this request will not succeed (error response). This request will result in the application context (and all applications within that context) being closed.
 - If the application has focus when this URI executes, the *onAppFocusLost* event handler will be invoked prior to the *onAppClosed* event handler (which will always be invoked).

priority = priority at which the action should be take. Values include:

- 0: Do immediately, even if user is interacting with the phone. This priority is unavailable if the *Application* URI is contained within an Application Management Event Handler.
- 1: Do when user is done interacting with the phone.
- 2: Do only if the user is not interacting with the phone.

idleTimer = duration of time (in seconds) the phone or application must be idle before the action should be taken. Values must range from 10-86400 (seconds); default is 60 seconds. The *idleTimer* value has no effect on *priority=0* requests. Any pending timers are automatically cancelled when the displayable object changes for an application context.

applicationId = optional identifier of the application on which the action should be taken. Values must range in length from 1-64 string characters and cannot contain colons. The default value is the application of the displayable object in which the URI is defined.

**Note**

If the *Application* URI is used in an *ExecuteItem*, you must specify the *applicationId* because the application context of the request cannot be inferred.

App URI Error and Response

All *Application* URI requests are asynchronous, so the only return value indicates that the URI was successfully parsed and that the specified application was valid and currently active in its context. The application is notified of the actual state change asynchronously using the event handlers.

Condition	Status	Data
Executed successfully	0 (Success)	Success
URI syntax is invalid	1 (Parse error)	Invalid URI
Unknown application ID	6 (Internal error)	Unknown Application ID
Request made to change state of an application that is not current active	6 (Internal error)	Application is not Active



CHAPTER

6

HTTP Requests and Header Settings

- [HTTP Requests and Header Settings Overview, page 101](#)
- [HTTP Client Requests \(HTTP GET\), page 101](#)
- [HTTP Server Requests \(HTTP POST\), page 102](#)
- [HTTP Header Settings, page 102](#)
- [IP Phone Client Capability Identification , page 108](#)
- [Accept Header, page 109](#)
- [IP Phone Information Access, page 109](#)

HTTP Requests and Header Settings Overview

Cisco Unified IP Phones use HTTP to communicate to external applications. The phone firmware includes an HTTP client for making requests and an HTTP server for receiving requests. This chapter describes the capabilities of the HTTP interface.

HTTP Client Requests (HTTP GET)

The following description designates how HTTP client requests are handled:

- 1 The phone HTTP client performs an HTTP GET for a specified URL.
- 2 The HTTP server processes request and returns an XML object or plain text.
- 3 The phone processes the supported HTTP headers.
- 4 The phone parses the XML object if *ContentType* is text/xml.
- 5 The phone presents data and options to the user, or in the case of a *CiscoIPPhoneExecute* object, begins executing the URIs.

HTTP Server Requests (HTTP POST)

The following description designates how an HTTP server request is made to the phone using an HTTP POST operation:

- 1 The server performs an HTTP POST in response to a case-sensitive URL of the phone with this format: *http://x.x.x.x/CGI/Execute*, where *x.x.x.x* represents the IP address of the destination Cisco Unified IP Phone.

The form that is posted should have a case-sensitive form field name called “XML” that contains the desired XML object. For any HTTP POST operation, the server must provide basic HTTP authentication information with the POST. The provided credentials must be of a user in the global directory with a device association with the target phone.

If the credentials are invalid, or the Authentication URL is not set properly in the Cisco Unified Communications Manager Administration, the phone will return a *CiscoIPPhoneError* with a value of 4 (Authentication Error) and processing will stop.

- 2 The phone processes the supported HTTP headers
- 3 The phone parses and validates the XML object
- 4 The phone presents data and options to the user, or in the case of a *CiscoIPPhoneExecute* object, begins executing the URIs.

Any HTTP POST object is limited to 512 bytes in size. Larger objects (such as images) can only be delivered to the phone using HTTP GET. To push large objects to the phone, the server application must take an indirect approach, by pushing an Execute object to the phone that contains an *ExecuteItem* that points to the URL of the large object.



Note

JTAPI can push an XML object directly to an IP phone, with the added benefit of not requiring authentication (because the JTAPI connection itself is already authenticated). This option works particularly well for adding XML services interfaces to existing CTI applications (where the overhead of the CTI connection is already a requirement). Objects pushed using JTAPI are also limited to a maximum size of 512 bytes. For more information, see the *Cisco Unified Communications Manager JTAPI Developer Guide*.

HTTP Header Settings

The following list provides definitions for HTTP header elements for Cisco Unified IP Phone Services:

- Refresh: sets the refresh time (in seconds) and URL
 - If no time is set or it is zero, the refresh gets set to manual.
 - If no URL is set, the current URL gets used.
- ContentType: notifies the phone of the MIME type that was sent.
- Expires: sets the Date/Time in GMT when the page is to expire.

Pages that have expired before being loaded do not get added to the URL stack in the phone. The phone does not cache content.

- Set Cookie
- HTTP encoding header

HTTP Refresh Setting

The HTTP headers that are sent with any page from an HTTP server can include a Refresh setting. This setting comprises two parameters: a time in seconds and a URL. These two parameters direct the recipient to wait the time given in the seconds parameter and then get the data to which the URL points.

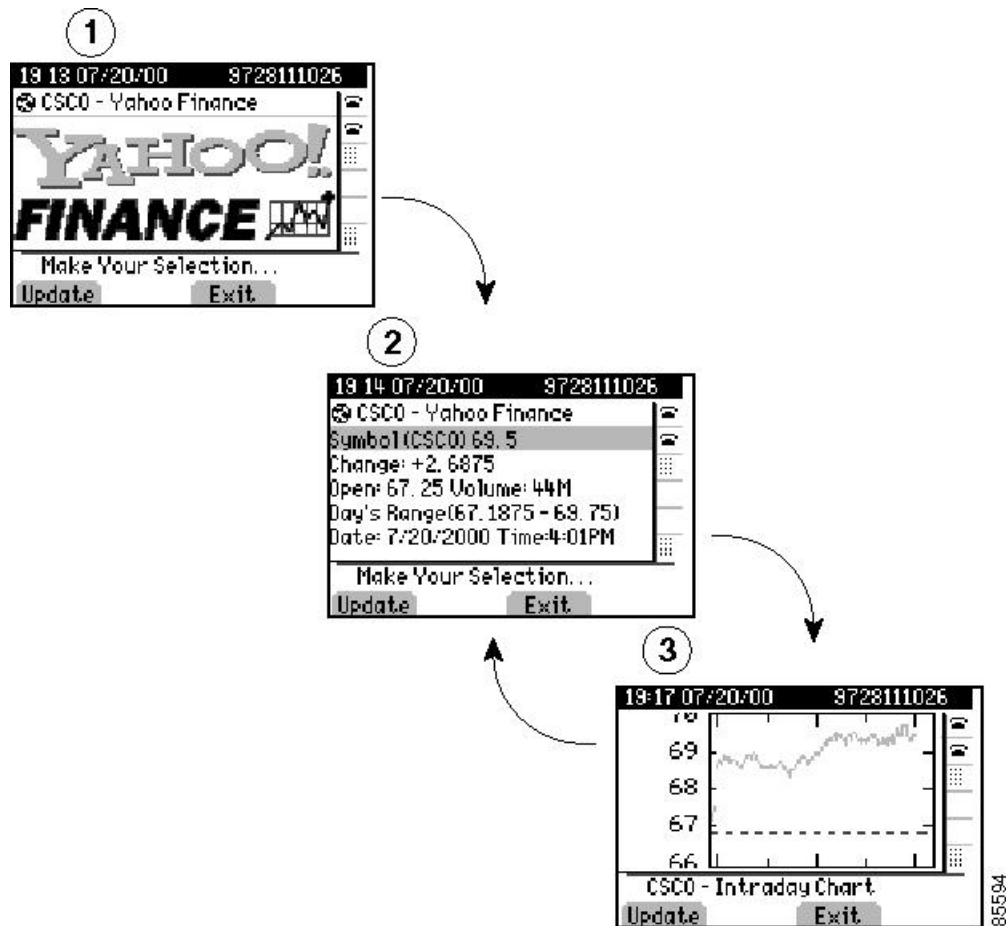
The Cisco Unified IP Phone HTTP client properly supports this setting, which gives a great deal of power to service developers. It means that a new page can replace any XML object that displays after a fixed time.

The following figure shows an example of how to use the refresh setting. This sample page shows the user the current value of Cisco stock.

- A splash screen that displays the Yahoo logo.
- After a very short time, the screen displays the numeric Cisco stock parameters.

- Finally, the screen shows a graph of Cisco intraday stock performance. The display repeatedly cycles between the final two views.

Figure 24: Refresh Display Sample



Refreshing the display can occur without user intervention, because the display automatically cycles if a timer parameter is specified. On any given screen, however, the user can force an immediate reload by pressing the Update softkey. Also, if a timer parameter of 0 was sent in the header, the page never automatically reloads. In this case, the display will move to the next page only when the Update softkey is pressed. If no refresh URL is specified, the current page gets reloaded.

MIME Type and Other HTTP Headers

Although delivering pages with the proper MIME type and other formatting items is not difficult, you require a moderately in-depth knowledge of your web server. The following code excerpt, written in JavaScript and used with Microsoft IIS and ASP, sets these values in a few lines:

```
<%@ Language=JavaScript %>
<%
Response.AddHeader( "Refresh",
                    "3; url=http://services.cisco.com/s/q.asp");
Response.ContentType = "text/xml";
```



```
//
// Additional page content here
//
%>
```

Usually, you can set the MIME type for pages in any web server by simply performing an association to the .xml file extension. Your web server documentation should explain how to accomplish this association. This action allows you to serve static pages without the need for writing script.

If you want to deliver dynamic content by using the other supported HTTP headers, you need to understand how to generate the HTTP headers using the desired programming language and have common gateway interface (CGI) or script access on the target web server.

Audio Clips

You can serve audio clips to the phone from a web server by using the "audio/basic" MIME type setting. When this MIME type is used, the body of the response should contain raw audio data in the same format that is used for custom IP phone rings. For more information, see the "Custom Phone Rings" chapter in the *Cisco Unified Communications Manager System Guide* (also available in the online help) and the phone administration guide.



Note

The audio file should not be longer than five seconds.

Use the following ASP sample script to set the MIME type and to serve the file that is specified in the #include command:

```
<%@ Language=JavaScript%>
<%
Response.ContentType = "audio/basic";
%><!--#include file="filename.raw" --><% Response.End();%>
```

Using a script to generate the MIME header when playing a sound provides an advantage because you may also include a refresh header to take the phone to a subsequent URL. Usually, you can set the MIME type for pages in any web server by simply performing an association to the .xml or .raw file extension. Your web server documentation should explain how to accomplish this. This action allows you to serve static pages without the need for writing script.

Content Expiration Header Setting

The expiration header can control which URLs are added to the phone URL history. This behavior differs slightly from traditional web browsers but is implemented to perform the same function. Disable the Back button functionality to avoid calling a URL twice.

This functionality allows you to expire the content of any page that is sent to the phone. When a user presses the Exit softkey, the user goes back to the last URL that did not expire when it was loaded. This action differs from traditional browsers by not considering the current freshness of the data but the freshness of the data when the URL was requested. This functionality requires you to have a page expire when it is first loaded and to not set a time and date in the future.

The following example shows how to have content on IIS expire by using Active Server Page (ASP):

```
<%@ Language=JavaScript %>
<%
Response.ContentType = "text/xml";
Response.Expires = -1;
%>
```

The “Expires” property specifies the number of minutes to wait for the content to expire. Setting this value to -1 subtracts 1 minute from the request time and returns a date and time that have already passed.

**Note**

The Cisco IP Phone 8800 Series requires that the expire date be after 1970/1/1.

Set-Cookie Header Setting

A *cookie* is the term for a mechanism that the Web server uses to give the client a piece of data and have the client return the data with each request. The two traditional uses for cookies are:

For Web sites to store a unique identifier or other information on the client's file system. The information is available to the Web server on subsequent visits.

To track a unique identifier for state management. The client returns the cookie with each request and the server uses this identifier to index information about the current session. The identifier is commonly referred to as a session ID. Most Web servers have a built-in session management layer that uses this second type of cookie, which is commonly referred to as a session cookie.

The following example shows the Set-Cookie header that is returned to the browser when a request method is used:

```
Set-Cookie: ASPSESSIONIDGQGGRSL=OCNMLFDBJIPNIOOKFNFMOL; path=/
```

The Cisco Unified IP Phone can receive and use a total of four cookies per host per session and can store information for up to eight sessions at once. Each cookie can be up to 255 bytes in size. These cookies are available until the server terminates the session or the client session has been idle for more than 30 minutes. On many phones which are capable of running multiple applications concurrently, the session state is also cleared when the application window closes. This behavior is consistent with PC-based browsers and provides better security because anyone attempting to reopen a secure application would be forced to authenticate. If the client is connecting to a new server and all session resources are in use, the client clears and reuses the session with the longest inactivity time.

When using ASP on IIS, the default server configuration automatically generates a session cookie and sends the cookie to the client using the Set-Cookie header. This cookie enables you to use the Session object from within ASP to store and retrieve data spanning multiple requests for the life of the session. When using JSP on Tomcat, the default configuration generates and issues a session cookie.

HTTP Encoding Header Setting

The encoding header controls language and character settings related to localization.

Accept Language

Cisco Unified IP Phones populate the Accept-Language HTTP request header in compliance with the HTTP specification.

For example, the Accept-Language value advertised by a phone configured for the English_United_States user locale would look like:

```
Accept-Language: en-US
```

Accept Charset

The phones are capable of handling UTF-8 encoding and, depending on phone model, some degree of Unicode support.

Some phone models (such as the Cisco Unified IP Phones 7905, 7912, 7940, and 7960) can handle UTF-8 encoding, but will only recognize characters which can be represented by the default encoding of the phone's current user locale. For example, if the phone is currently configured to use the English_United_States locale, then it will only be able to display UTF-8 characters which map to the ISO-8859-1 character set.

Other phone models (such as the Cisco Unified IP Phones 7911, 7941, 7961, 7970, and 7971) provide UTF-8 and true Unicode support. These phones provide support for more multi-byte character sets and user locales like Japanese and Chinese.

**Note**

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

In addition to the character set for the currently configured user locale, the phone models also support ISO-8859-1 characters in their font files.

All phones advertise their supported encodings using the standard HTTP Accept-Charset header. According to the HTTP standard, q-values are used to specify preferred encodings. The older phone models, with more limited UTF-8 support, specify a lower q-value for UTF-8 than the default user locale encoding.

For example, an older phone model configured with the English_United_States user locale would include an Accept-Charset header similar to the following:

```
Accept-Charset: iso-8859-1, utf-8;q=0.8
```

A newer phone model with Unicode support would advertise an Accept-Charset similar to the following:

```
Accept-Charset: utf-8, iso-8859-1;q=0.8
```

Related Topics

[Deprecated Endpoints, on page 3](#)

HTTP Response Headers: Content-Type

Because the phones are capable of supporting multiple character encodings, HTTP responses returned to the phones should include the charset parameter on the HTTP Content-Type header. The following are examples of responses including the charset parameter:

```
Content-Type: text/xml; charset=ISO-8859-1  
Content-Type: text/xml; charset=UTF-8  
Content-Type: text/plain; charset=Shift_JIS
```

HTTP standards state that if the encoding is not explicitly specified, ISO-8859-1 is the default. Cisco Unified IP Phones are typically compatible with this spec, but not fully compliant.

If charset is not specified, the phones use the default encoding for the currently configured user locale. To avoid possible problems where the default encoding may not be ISO-8859-1, the web server should explicitly set the Content-Type charset to match one of the Accept-Charset values specified by the phone.

IP Phone Client Capability Identification

XML services are supported on many Cisco Unified IP Phones, so web application servers must identify the capabilities of the requesting IP phone to optimize the content returned to the phone. For example, if the requesting phone is a Cisco Unified IP Phone 7960, which cannot support color PNG images, the application server must be able to identify this limitation and return a gray scale CIP image instead.

The IP phone client request to send the relevant information from the IP phone to the web server application includes these HTTP headers:

- x-CiscoIPPhoneModelName
- x-CiscoIPPhoneDisplay
- x-CiscoIPPhoneSDKVersion

x-CiscoIPPhoneModelName

This Cisco-proprietary header contains the Cisco manufacturing Model Name of the device, which can typically be found by going to **Settings > Model Information**, but varies between different models. Some examples of manufacturing Model Names are CP-7905G, CP-7940G, CP-7960, CP-7960G, and CP-7970G.

x-CiscoIPPhoneDisplay

This Cisco-proprietary header contains the display capabilities of the requesting device with the following parameters (listed in the order in which they appear):

- Width (in pixels)
- Height (in pixels)
- Color depth (in bits)
- A single character indicating whether the display is color ("C") or gray scale ("G")

These parameters get separated by commas as shown in the following example of a Cisco Unified IP Phone 7970 header:

```
x-CiscoIPPhoneDisplay: 298, 168, 12, C
```

**Note**

The pixel resolutions advertised by the device define the area of the display accessible by the phone services; not the actual resolution of the display.

x-CiscoIPPhoneSDKVersion

This Cisco-proprietary header contains the version of the IP Phone Services SDK that the requesting phone supports. The HTTP header does not specify which URIs are supported. Therefore, you must check the Supported URIs matrix in the IP Phone Services SDK to determine which URIs are supported based on the Phone Model Name and supported SDK version.

**Note**

Beginning with the IP Phone Services SDK 3.3(3), the SDK version number matches the minimum Cisco Unified Communications Manager software that is required to support it. For example, SDK version 3.3(4) gets supported only on Cisco Unified Communications Manager version 3.3(4) or later.

Related Topics

[Supported URIs by Phone Model](#), on page 65

Accept Header

The Accept header represents a standard HTTP header that is used to inform web servers about the content-handling capabilities of the client.

Cisco Unified IP Phones include proprietary content-types to indicate which XML objects are supported. These proprietary content-types all begin with x-CiscoIPPhone, to indicate Cisco Unified IP Phone XML objects, followed by a slash "/", followed by either a specific XML object or a "*" to indicate all objects.

For example, x-CiscoIPPhone/* indicates that all XML objects defined in the specified version of the SDK are supported, and x-CiscoIPPhone/Menu specifies that the <CiscoIPPhoneMenu> object gets supported.

As the Menu example illustrates, the name of the XML object can be derived directly from the content-type by appending the sub-type (the part after the slash) onto CiscoIPPhone. The content-type can also include an optional version to indicate support for a particular SDK version of that object. If a version is not specified, then the x-CiscoIPPhoneSDKVersion is implied. The syntax of the version number may vary, but, in general, will be as follows:

<major version>.<minor version>.<maintenance version>

Here are some examples of typical content-types:

- x-CiscoIPPhone/*;version=3.3.3
- x-CiscoIPPhone/Text
- x-CiscoIPPhone/Menu;version=3.3.4

IP Phone Information Access

Cisco Unified IP Phones have an embedded web server to provide a programming interface for external applications, and a debugging and management interface for system administrators.

You can access the administrative pages using a standard web browser and pointing to the IP address of the phone with: /http://<phoneIP>/, where phoneIP is the IP address of the specific phone.

These device information pages are available in either HTML format for manual debugging purposes, or in XML format for automation purposes. The following table lists the available URLs and their purpose.

Table 26: Device Information URLs

HTML URL	XML URL	Description
/DeviceInformation	/DeviceInformationX	General device information

HTML URL	XML URL	Description
/NetworkConfiguration	/NetworkConfigurationX	Network configuration information
/EthernetInformation	/EthernetInformationX	Ethernet counters
/PortInformation? <i>n</i>	/PortInformationX? <i>n</i>	Detailed port information, where <i>n</i> is a model-specific ethernet port identifier, typically in the range 1-3.
/DeviceLog? <i>n</i>	/DeviceLogX? <i>n</i>	Device logging, debug, and error messages, where <i>n</i> is a model-specific log number, typically in the range 0 - 2.
/StreamingStatistics? <i>n</i>	/StreamingStatisticsX? <i>n</i>	Current RTP streaming stats, where <i>n</i> is model-specific RTP stream identifier, typically in the range 1-3.
/CGI/Execute (password-protected CGI script)		The target URL of a phone push (HTTP POST) request.
/CGI/Screenshot (password-protected CGI script)		Returns an exact snapshot of the current phone display. The size and format of the image returned is model-specific



Troubleshooting Cisco Unified IP Phone Service Applications

- [Troubleshooting Tips](#), page 111
- [XML Parsing Errors](#), page 111
- [Error Messages](#) , page 112

Troubleshooting Tips

The following tips apply to troubleshooting Cisco Unified IP Phone service applications:

- Microsoft Internet Explorer 5 or higher can display the XML source with its default style sheet.
- Understand that standard IP troubleshooting techniques are important for HTTP errors.
- Externally verify name resolution (Phone has DNS set).
- If DNS is suspected, use IP addresses in URLs.
- Browse the URL in question with Microsoft Internet Explorer or download and verify with another web browser.
- Use a logged telnet session to verify that the desired HTTP headers are returned (telnet to the server on port 80, and then enter: get /path/page).

Related Topics

[Troubleshooting CiscoIPPhoneIconFileMenu XML Objects Using Enhanced Icon Menu Support Feature](#), on page 43

XML Parsing Errors

The following tips apply to troubleshooting XML parsing errors in Cisco Unified IP Phone services applications:

- Verify the object tags (the object tags are case sensitive).

- Verify that “&” and the other four special characters are used according to the restrictions while inside the XML objects.
- Validate XML applications developed prior to Cisco Unified IP Phone Firmware Release 8.3(2) against the more recent XML parser. Some of examples of the types of errors you might encounter include:
 - CiscoIPPhoneMenu Object: If the field `<Name>` is missing for a `<MenuItem>`, the original parser would stop rendering from that `<MenuItem>` onwards. The new parser will display a blank line in the menu list and continue to render any subsequent `<MenuItem>` definitions.
 - CiscoIPPhoneDirectory Object: If the field `<Name>` is not present, the old original parser would not display the directory entry, the new parser will display the directory entry, but there will be no `<Name>` associated with it.
 - CiscoIPPhoneInput Object: The `URL` and `QueryStringParam` fields are mandatory. The original parser would not report an error on the missing URL and on submit request would display a “Host not Found” message. If the `QueryStringParam` field is missing, the updated parser will report an error.
 - SoftKeyItem: The `Position` field is mandatory. If the `Position` field is not present, the updated XML parser will report an error.

Related Topics

[Mandatory Escape Sequences, on page 51](#)

[CiscoIPPhone XML Objects, on page 11](#)

[Updated XML Parser and Schema Enforcement, on page 135](#)

Error Messages

The following error messages may appear on the prompt line of the Cisco Unified IP Phone display:

- XML Error[4] = XML Parser error (Invalid Object)
- XML Error[5] = Unsupported XML Object (not supported by this phone model)
- HTTP Error[8] = Unknown HTTP Error
- HTTP Error[10] = HTTP Connection Failed

The Cisco Unified IP Phone 6900 Series supports the following error messages:

Text	Description
Services Unavailable	cfg file directoryURL or servicesURL is empty
Host Not Found	DNS query fails
Server Busy!	Server response 503
Connection failed	Socket cannot be created or the connection fails

Text	Description
XML Error [4]: Parse Error	Does not match XML schema
Data too large!	Downloaded content is over 196608 bytes
No services configured	HTTP message body is empty
Filename too long!	file name length is over 127 characters.
File Not Found	Server response 404
HTTP connection failed	Server response 500
Unknown Error	Other errors



Cisco IP Phone Services Software Development Kit (SDK)

- [SDK Overview, page 115](#)
- [SDK Components, page 115](#)
- [Sample Services Requirements, page 117](#)

SDK Overview

The Cisco IP Phone Services Software Development Kit (SDK) contains everything that you require to create XML applications, including necessary documentation and sample applications. Contact Cisco Developer Services to obtain the SDK at:

<http://developer.cisco.com/web/ipps>



Note

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Related Topics

[Deprecated Endpoints, on page 3](#)

SDK Components

The SDK contains the following components.

Documentation

- *Cisco IP Services Development Notes* (PDF format)
- *Cisco URL Proxy Guide* (Rich Text Format)

- *Cisco LDAP Programming Guide* (Microsoft Word format)
- *Cisco CIP Image Release Notes* (Microsoft Word format)
- *Cisco IP Applications Samples* (Microsoft Word format)

Development Tools

- Cip.8bi: Adobe Photoshop plug-in that allows .cip extensions to be viewed and saved.
- Cip2Gif.exe: DOS-based program that converts .cip files to .gif.
- Gif2Cip.exe: DOS-based program that converts .gif files to .cip.
- ImageViewer.exe: Windows application that displays .cip graphic files.
- Cisco CIPImage: used for converting images to and from CIP images (automatically installed)
- Cisco URL Proxy: Proxy server that is needed to use the sample services (automatically installed).
- Cisco LDAP Search: Service that is installed to do LDAP searches (automatically installed).
- Microsoft XML Parser (MXSML) 3.0: Used for parsing XML data (automatically installed)
- Cisco Unified IP Phone Services ASP/Javascript Library (automatically installed)
- Cisco Unified IP Phone Services Java Library: Used by the JSP apps (manually installed; see *JSP Install* readme)
- CallManager Simulator: Used for developing Phone Services without a Cisco Unified Communications Manager server
- Cisco Unified IP Phone XML Schema (.xsd) file: Used with an XML editor to validate XML syntax

Sample Services

- Weather forecast lookup for any city (ASP)
- Currency Exchange Rates and Converter (ASP)
- UPS Rates & tracking (ASP)
- World Clock (ASP)
- Measurement conversions (ASP)
- US White pages/Yellow Pages search (ASP)
- Calendar (ASP)
- Stock Ticker (ASP)
- Stock Chart (ASP)
- Push2Phone (ASP and JSP)
- Click2Dial (ASP and JSP)
- IdleURL (ASP) - Not supported on Cisco Unified IP Phones 7905G and 7912G
- MConference (JSP)
- Hootie (ASP)

- InterCom (ASP)
- JPEGViewer (ASP)
- Logo (ASP)
- Clock (ASP)
- Personal Service (ASP)
- WaterMark (ASP)
- Extension Mobility Controller (JSP)
- Speed Dials (JSP)
- Group MWI (JSP)
- AutoDialer (JSP)
- PhotoDirectory (JSP)
- CallerInfo (JSP)
- PushAuthenticate (ASP)
- ScreenShot (ASP)
- Integrating RS-232 devices with IP Telephony Applications (OtherApps)
- PNGViewer (ASP)
- Keyboard (ASP)
- MultiDirectory (ASP)
- Phone Push Step and Subsystem (Cisco Unified Contact Center Express / CRS)

Sample Services Requirements

The following list contains the items that are required for the sample services to work properly:

- Microsoft IIS 4.0 or later (for ASP sample services)
- Sun J2SE 1.4.2 or later and Tomcat 4.0 or later (for JSP sample services)
- Internet Connection to external websites like Yahoo.com, Cnn.com etc.
- Cisco Unified Communications Manager 4.1(2) or later.
- Cisco Unified IP Phones that supports XML services

The setup program installs a CiscoServices web project to c:\CiscoIpServices directory. The sample services are copied to c:\CiscoIpServices\Services subdirectory, and IIS and WSH example codes are provided. The web server already senses these services and you do not require further administration. You can view or edit all the source code with any text editor. For additional documentation, go to this directory: c:\CiscoIpServices\Documentation. Find tools to help develop services in c:\CiscoIpServices\Tools.



IP Phone Service Administration and Subscription

- [Administration and Subscription Overview, page 119](#)
- [Phone Service Administration Access, page 119](#)
- [Phone Service Addition, page 120](#)
- [IP Phone Service Parameters Definition, page 122](#)
- [User Service Subscription, page 123](#)

Administration and Subscription Overview

Cisco Unified Communications Manager administrators maintain the list of services to which users can subscribe. Administrators must use Cisco Unified Communications Manager Administration to add and administer Cisco Unified IP Phone services.



Note

This chapter provides a brief overview about managing IP Phone services. For detailed up-to-date instructions, refer to the *Cisco Unified Communications Manager Administration Guide* available at the following URL:

<http://www.cisco.com/c/en/us/support/unified-communications/unified-communications-manager-callmanager/products-maintenance-guides-list.html>

Phone Service Administration Access

To access phone service administration, open Cisco Unified Communications Manager Administration and choose **Device > Device Settings > Phone Services**:

- Phone services can have any number of parameters associated with them.
- You can specify phone service parameters as optional or required, depending on how the phone service application defines them.

- Users can subscribe to any service configured in their cluster, using their User Options web pages.
- Service subscriptions currently occur on a device basis.

A URL constitutes the core of each service. When a service is chosen from the menu, the URL gets requested using HTTP, and a server somewhere provides the content. The Service URL field shows this URL entry. For the services to be available, the phones in the Cisco Unified Communications Manager cluster must have network connectivity to the server.

Example

`http://<servername>/ccmuser/sample/sample.asp`

Where

`<servername>` designates a fully qualified domain name or an IP address.

Phone Service Addition

To access phone service administration, open Cisco Unified Communications Manager Administration and choose **Device > Device Settings > Phone Services**:

The Cisco Unified Services Configuration page in Cisco Unified Communications Manager Administration contains the fields as shown in the following table.

Table 27: IP Phone Service Configuration Settings

Field	Description
Service Information	
Service Name	Enter the name of the service as it will display on the menu of available services in Cisco Unified CM User Options. Enter up to 32 characters for the service name.
ASCII Service Name	Enter the name of the service to display if the phone cannot display Unicode.
Service Description	Enter a description of the content that the service provides.

Field	Description
Service URL	<p>Enter the URL of the server where the IP phone services application is located. Make sure that this server remains independent of the servers in your Cisco Unified Communications Manager cluster. Do not specify a Cisco Unified Communications Manager server or any server that is associated with Cisco Unified Communications Manager (such as a TFTP server or directory database publisher server).</p> <p>For the services to be available, the phones in the Cisco Unified Communications Manager cluster must have network connectivity to the server.</p> <p>When defining the service URL, you can embed a special #DEVICENAME# substitution tag within the URL. This tag provides a convenient method for IP phones to pass their device name to a web application server. For example, if a service URL was defined in Cisco Unified Communications Manager Administration as: <code>http://myserver/myscript?name=#DEVICENAME#</code>, when a phone actually makes the HTTP request for the service, the requested URL will appear as: <code>http://myserver/myscript?name=SEP000123456789</code></p>
Secure-Service URL	<p>Enter the secure URL of the server where the Cisco Unified IP Phone services application is located. Make sure that this server remains independent of the servers in your Cisco Unified Communications Manager cluster. Do not specify a Cisco Unified Communications Manager server or any server that is associated with Cisco Unified Communications Manager (such as a TFTP server or publisher database server).</p> <p>For the services to be available, the phones in the Cisco Unified Communications Manager cluster must have network connectivity to the server.</p> <p>Note If you do not provide a Secure-Service URL, the device uses the nonsecure URL. If you provide both a secure URL and a nonsecure URL, the device chooses the appropriate URL, based on its capabilities.</p>
Service Category	Select a service application type.
Service Type	Select whether the service will be provisioned to the Services, Directories, or Messages button.
Service Vendor	For XML services, you can leave this field blank.
Service Version	For XML services, you can leave this field blank.
Enable	<p>Select this check box to enable the service, or clear the check box to disable the service without deleting it.</p> <p>Note You cannot delete default services. Use this field if a default service exists, but you do not want to make it available for subscription.</p>

Field	Description
Enterprise Subscriptions	<p>Select this check box to automatically provision the new service to all devices in the enterprise without requiring individual subscription. If this option is selected, the service automatically gets provisioned and does not get presented for user subscription.</p> <p>Note Be aware that this check box is available for selection only when the service is created. You cannot modify it.</p>

IP Phone Service Parameters Definition

Each service can have a list of parameters. You can use these parameters, which are appended to the URL when they are sent to the server, to personalize a service for an individual user. Examples of parameters include stock ticker symbols, city names, or user IDs. The service provider defines the semantics of a parameter.

The Cisco Unified IP Phone Service Parameter Configuration page in Cisco Unified Communications Manager Administration contains the fields as described in the following table.

Table 28: IP Phone Service Parameter Settings

Field	Description
Service Parameter Information	
Parameter Name	Enter the exact query string parameter to use when you build the subscription URL; for example, symbol.
Parameter Display Name	Enter a descriptive parameter name to display to the user in Cisco Unified CM User Options; for example, Ticker Symbol.
Default Value	Enter the default value for the parameter. This value displays to the user when a service is being subscribed to for the first time; for example, CSCO.
Parameter Description	Enter a description of the parameter. The user can access the text that is entered here while the user is subscribing to the service. The parameter description should provide information or examples to help users input the correct value for the parameter.
Parameter is Required	If the user must enter data for this parameter before the subscription can be saved, check the Parameter is Required check box.

Field	Description
Parameter is a Password (mask contents)	You can mask entries in Cisco Unified CM User Options, so asterisks display rather than the actual user entry. You may want to do this for parameters such as passwords that you do not want others to be able to view. To mask parameter entry, select the Parameter is a Password (mask contents) check box in the Configure IP phone service Parameter window in Cisco Unified Communications Manager Administration.

**Note**

If you change the service URL, remove a Cisco Unified IP Phone service parameter, or change the Parameter Name of a phone service parameter for a phone service to which users are already subscribed, be sure to click Update Subscriptions to update all currently subscribed users with the changes. If you do not update subscriptions, users must resubscribe to the service to rebuild the URL correctly.

User Service Subscription

End users can configure service subscriptions using the Cisco Unified CM User Options pages. After users log in and choose a device, a list of services that are assigned to the phone display. The user can configure these services, adding additional ones or removing unused services. These password-protected windows are authenticated using the LDAP directory.

Users can personalize their services using the User Options pages to:

- Customize the name of the service.
- Enter any available service parameters.
- Review the description of each parameter.

After all the required fields are set, the user clicks Subscribe to add the services. A custom URL gets built and stored in the database for this subscription. The service then appears on the device services list.



DeviceListX Report

- [DeviceListX Report Overview, page 125](#)
- [Benefits, page 126](#)
- [Restrictions, page 126](#)
- [Integration Considerations and Interoperability, page 126](#)
- [Performance and Scalability, page 127](#)
- [Security, page 127](#)
- [Related Features and Technologies, page 127](#)
- [Supported Platforms, page 127](#)
- [Prerequisites, page 127](#)
- [Message and Interface Definitions, page 127](#)
- [DeviceList XML Object, page 128](#)
- [Troubleshooting DeviceListX Reports, page 129](#)

DeviceListX Report Overview



Note

The DeviceListX Report is no longer supported as of Cisco Unified Communications Manager Release 5.0. Retrieving real-time information from Cisco Unified Communications Manager is now supported using the Cisco Unified Communications Manager AXL Serviceability API.

The DeviceListX Report provides a list of the services-capable devices along with basic information about the device to identify or classify the devices based on specific criteria. The report also includes the current device status and the IP address information that is obtained from the Real-Time Information Service.

**Note**

DeviceListX does not support all devices. If you have a device that you need to support, contact Cisco Developer Support to verify whether it is supported:

<http://developer.cisco.com/web/ipps>

When a third-party developer initiates an *HTTP GET* request for the DeviceListX.asp report page, the system retrieves the following information about phones that are registered to a Cisco Unified Communications Manager server from the database:

- Device Type
- Device Name
- Device Description
- Calling Search Space
- Device Pool
- IP Address
- Real-Time Information

The completed list of data gets formatted into a simple XML object and gets returned in the HTTP Response to the developer.

Benefits

DeviceListX provides access to critical real-time data that was previously unavailable to third-party developers. In particular, the ability to list currently registered devices along with their IP address allows developers to easily build push, broadcast, and CTI-type applications.

Restrictions

Only users with administrative privileges to the Cisco Unified Communications Manager Administration can access the report.

**Note**

To minimize processing overhead on the Cisco Unified Communications Manager server, access to the DeviceListX report gets rate-limited to once per minute. Any attempt to pull the report more frequently will fail. In practice, the developer application should pull and cache the DeviceListX report, refreshing only as often as required, typically every few hours or daily.

Integration Considerations and Interoperability

The interface allows HTTP 1.1 or HTTP 1.0 *GET* requests for the report. The report returns data that is encapsulated by using XML version 1.0.

Performance and Scalability

You can run this report on the largest supported Cisco Unified Communications Manager cluster size for the targeted release without impacting core features, such as delaying dial tone. On multiserver Cisco Unified Communications Manager clusters, the report can access only from the publisher server. In large clusters where the publisher is not a Cisco Unified Communications Manager server, no possibility exists of impacting the system performance as perceived by a user.

This report is not intended for use during real time, so this interface should provide a mechanism for developers to poll for the data on a daily or hourly basis. Give consideration to the frequency of polling and the time of day to prevent unnecessary burden on the system during peak usage times.

Security

This report, which is within the Cisco Unified Communications Manager Administration, inherits its security from that web site, so no security issues directly relate to this report. If the Cisco Unified Communications Manager Administration changes how it implements security with additions, such as SSL, this report benefits from that enhancement.

Related Features and Technologies

DeviceListX acts as an independent interface, which is a real-time complement to the XML-Layer Database API (AXL), where AXL provides access to static, persisted data, and DeviceListX provides access to dynamic, volatile information.

Supported Platforms

For the DeviceListX.asp page to function requires Cisco Unified Communications Manager Administration reporting infrastructure. The following releases support DeviceListX.asp:

- Cisco CallManager Release 3.2(3)SPB
- Cisco Unified Communications Manager Release 4.0(1) and later

Prerequisites

You can access this feature when devicelistX.asp resides in the C:\ciscoWebs\Admin\reports directory of the Cisco Unified Communications Manager publisher server.

Message and Interface Definitions

Use the following URL to access the report using HTTP:

<http://x.x.x.x/CCMAdmin/reports/devicelistx.asp>

where

x.x.x.x can either be the IP address or hostname of the Cisco Unified CallManager system that contains the report.



Note

Beginning with Cisco Unified CallManager 4.1 release, the DeviceListX report can only be accessed using secure HTTP (HTTPS), so the URL must begin with "https:" rather than "http:".

DeviceList XML Object

Third-party applications that reside elsewhere on the network commonly use the interface. The application makes an HTTP request for the report and gets a response that contains a DeviceList XML object. The XML object follows:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<DeviceList>
<Device t="" n="" d="" c="" p="" i="" s="" />
</DeviceList>
```

Table 29: DeviceList XML Object Attributes

Attribute name	Field name	Description
t	Device Type	Numeric enumeration value that is specified in the database.
n	Device Name	String value that specifies the device name.
d	Device Description	String value that is specified in the database.
c	Device Calling Search Space	String value that is specified in the database.
p	Device Pool	String value that is specified in the database.
i	Device IP Address	Last known IP address as reported by the Real-Time Information Service "" = No known IP address "x.x.x.x" = Last known IP address

Attribute name	Field name	Description
s	Device Status	Numeric enumeration for the current device status as reported by the Real-Time Information Service "" = Device not found "1" = Device registered "2" = Device found but not currently registered

DeviceList Object Example with Data

```
<?xml version="1" encoding="iso-8859-1"?>
<DeviceList>
<Device t="35" n="SEP000123456789" d="Auto 2010" c="" p="Default" i="10.1.1.1" s="1"/>
</DeviceList>
```

Troubleshooting DeviceListX Reports

These sections can assist you in troubleshooting DeviceListX Reports.

Error Codes

The error codes that are specific to this report interface are:

Error message 1001

Message

Error Message 1001 Too many simultaneous requests for Device List. Please wait at least 60 seconds and try again.

Explanation

When two or more clients attempt to get the list at the same time, or if the list is long, overlapping requests can result (first request is processing when the second request attempts processing).

Recommended action

Request information only as often as necessary.



Note

Cisco recommends that you wait longer than 60 seconds between requests.

Error message 1002**Message**

Error Message 1002 Too many consecutive requests for Device List.
Please wait at least 60 seconds and try again.

Explanation

Because the system is busy, it cannot process a Device List.

Recommended action

Request information only as often as necessary. Because the real-time status of every device gets checked, Device List represents a CPU-intensive process.

**Note**

Cisco recommends that you wait longer than 60 seconds between requests.

Determine Interface Problems

Use the following procedure to determine whether a problem exists with the interface and determine the root cause of the problem.

Procedure

-
- Step 1** Check the Windows NT Event Logs for error messages that pertain to the IIS server and the SQL server:
Start > Programs > Administrative Tools > Event Viewer
- Step 2** Check for error messages or successful completion of a request in the IIS log files, which are typically located in
C:\WINNT\System32\LogFiles\W3SVC1
The date of the log provides part of the log name. All times in the log files specify GMT for noted events. The IIS logs appear in chronological order and can easily be searched by specific query event.
- Step 3** Use a web browser, such as Internet Explorer, to request the URL of the devicelistx.asp web page. A successful request yields a well-formed XML object of all the device information.
- Step 4** Use a Sniffer trace to view the HTTP GET request and response transaction between the third-party application and the report.
-



APPENDIX A

CiscoIPPhone XML Object Quick Reference

The following sections provide a quick reference of the CiscoIPPhone XML objects and the definitions that are associated with each object.

CiscoIPPhoneMenu

```
<CiscoIPPhoneMenu>
  <Title>Title text goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <MenuItem>
    <Name>The name of each menu item</Name>
    <URL>The URL associated with the menu item</URL>
  </MenuItem>
</CiscoIPPhoneMenu>
```

CiscoIPPhoneText

```
<CiscoIPPhoneText>
  <Title>Title text goes here</Title>
  <Prompt>The prompt text goes here</Prompt>
  <Text>Text to display as the message body goes here</Text>
</CiscoIPPhoneText>
```

CiscoIPPhoneInput

```
<CiscoIPPhoneInput>
  <Title>Directory title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <URL>The target URL for the completed input goes here</URL>
  <InputItem>
    <DisplayName>Name of input field to display</DisplayName>
    <QueryStringParam>The parameter to be added to the target URL</QueryStringParam>
    <DefaultValue>Value</DefaultValue>
    <InputFlags>The flag specifying the type of allowable input</InputFlags>
  </InputItem>
</CiscoIPPhoneInput>
```

CiscoIPPhoneDirectory

```
<CiscoIPPhoneDirectory>
  <Title>Directory title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <DirectoryEntry>
    <Name>The name of the directory entry</Name>
    <Telephone>The telephone number for the entry</Telephone>
  </DirectoryEntry>
</CiscoIPPhoneDirectory>
```

CiscoIPPhoneImage

```
<CiscoIPPhoneImage WindowMode="XSI window width mode">
  <Title>Image title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <LocationX>Position information of graphic</LocationX>
  <LocationY>Position information of graphic</LocationY>
  <Width>Size information for the graphic</Width>
  <Height>Size information for the graphic</Height>
  <Depth>Number of bits per pixel</Depth>
  <Data>Packed Pixel Data</Data>
  <SoftKeyItem>
    <Name>Name of the soft key</Name>
    <URL>URL of soft key</Name>
    <Position>Numerical position of the soft key</Position>
  </SoftKeyItem>
</CiscoIPPhoneImage>
```

CiscoIPPhoneImageFile

```
<CiscoIPPhoneImageFile WindowMode="Width Mode of XSI window">
  <Title>Image Title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <LocationX>Horizontal position of graphic</LocationX>
  <LocationY>Vertical position of graphic</LocationY>
  <URL>Points to the PNG image</URL>
</CiscoIPPhoneImageFile>
```

CiscoIPPhoneGraphicMenu

```
<CiscoIPPhoneGraphicMenu WindowMode="Width Mode of XSI window">
  <Title>Menu title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <LocationX>Position information of graphic</LocationX>
  <LocationY>Position information of graphic</LocationY>
  <Width>Size information for the graphic</Width>
  <Height>Size information for the graphic</Height>
  <Depth>Number of bits per pixel</Depth>
  <Data>Packed Pixel Data</Data>
  <MenuItem>
    <Name>The name of each menu item</Name>
    <URL>The URL associated with the menu item</URL>
  </MenuItem>
</CiscoIPPhoneGraphicMenu>
```

CiscoIPPhoneGraphicFileMenu

```
<CiscoIPPhoneGraphicFileMenu WindowMode="Width Mode of XSI window">
  <Title>Image Title goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <LocationX>Horizontal position of graphic</LocationX>
  <LocationY>Vertical position of graphic</LocationY>
  <URL>Points to the PNG background image</URL>
  <MenuItem>
    <Name>Same as CiscoIPPhoneGraphicMenu</Name>
    <URL>Invoked when the TouchArea is touched</URL>
    <TouchArea X1="left edge" Y1="top edge" X2="right edge" Y2="bottom edge"/>
  </MenuItem>
</CiscoIPPhoneGraphicFileMenu>
```

CiscoIPPhoneIconMenu

```
<CiscoIPPhoneIconMenu>
  <Title>Title text goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <MenuItem>
    <IconIndex>Indicates what IconItem to display</IconIndex>
    <Name>The name of each menu item</Name>
    <URL>The URL associated with the menu item</URL>
  </MenuItem>
```

```

<SoftKeyItem>
  <Name>Name of softkey</Name>
  <URL>URL or URI of softkey</URL>
  <Position>Position information of the softkey</Position>
</SoftKeyItem>
<IconItem>
  <Index>A unique index from 0 to 9</Index>
  <Height>Size information for the icon</Height>
  <Width>Size information for the icon</Width>
  <Depth>Number of bits per pixel</Depth>
  <Data>Packed Pixel Data</Data>
</IconItem>
</CiscoIPPhoneIconMenu>

```

CiscoIPPhoneIconFileMenu

```

<CiscoIPPhoneIconFileMenu>
  <Title>Title text goes here</Title>
  <Prompt>Prompt text goes here</Prompt>
  <MenuItem>
    <IconIndex>Indicates what IconItem to display</IconIndex>
    <Name>The name of each menu item</Name>
    <URL>The URL associated with the menu item</URL>
  </MenuItem>
  <IconItem>
    <Index>A unique index from 0 to 9</Index>
    <URL>location of the PNG icon image</URL>
  </IconItem>
</CiscoIPPhoneIconFileMenu>

```

CiscoIPPhoneStatus

```

<CiscoIPPhoneStatus>
  <Text>This is the text area</Text>
  <Timer>Timer seed value in seconds</Timer>
  <LocationX>Horizontal alignment</LocationX>
  <LocationY>Vertical alignment</LocationY>
  <Width>Pixel width of graphic</Width>
  <Height>Pixel height of graphic</Height>
  <Depth>Color depth in bits</Depth>
  <Data>Hex binary image data</Data>
</CiscoIPPhoneStatus>

```

CiscoIPPhoneStatusFile

```

<CiscoIPPhoneStatusFile>
  <Text>This is the text area</Text>
  <Timer>Timer seed value in seconds</Timer>
  <LocationX>Horizontal alignment</LocationX>
  <LocationY>Vertical alignment</LocationY>
  <URL>location of the PNG image</URL>
</CiscoIPPhoneStatusFile>

```

CiscoIPPhoneExecute

```

<CiscoIPPhoneExecute>
  <ExecuteItem URL="The URL or URI to be executed"/>
</CiscoIPPhoneExecute>

```

CiscoIPPhoneError

```

<CiscoIPPhoneError Number="x"/>

```

CiscoIPPhoneResponse

```

<CiscoIPPhoneResponse>
  <ResponseItem Status="the success or failure of the action"Data="the information associated

```

```
with the request" URL="the URL or URI specified in the Execute object"/>  
</CiscoIPPhoneResponse>
```



Cisco Unified IP Phone Services XML Schema File

- [Updated XML Parser and Schema Enforcement, page 135](#)
- [CiscoIPPhone.xsd, page 136](#)

Updated XML Parser and Schema Enforcement

In order to provide a stable and consistent platform upon which to build enhancements to IP phones services, Cisco released an updated XML parser beginning with Firmware Release 8.3(2). The Cisco Unified IP Phones 6921, 6941, 6945, and 6961 supports the XML parser from Firmware Release 9.1(1) onwards. As a result, many Cisco Unified IP Phones now contain this updated XML parser which provides a more rigid enforcement of the XML schema. This updated parser provides more error logging information when it encounters XML schema violations, and it enables developers to debug their applications more efficiently.

Cisco recommends that developers verify that their existing applications conform to the XML schema to avoid incompatibilities with any XML enhancements, particularly if you want to incorporate new URIs.

The following Cisco IP Phones implement this XML parser: 6921, 6941, 6945, 6961, 7906G, 7911G, 7921G, 7925G, 7925G-EX, 7926G, 7931G, 7941G, 7941G-GE 7942G, 7945G, 7961G, 7961G-GE, 7962G, 7965G, 7970G, 7971G-GE, 7975G, 8800 Series, 8821, 8961, 9951, and 9971.



Note

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

The XML parser behavior for specific phone models are noted in the following table:

Phone model	XML parser behavior
7937	The Cisco Unified IP Phone 7937 does not strictly follow the boundary conditions. When there is a parser error, the 7937 may not return error messages. But in certain cases it may reboot to correct a parsing error.

Phone model	XML parser behavior
7921G, 7925G, 7925G-EX, 7926G	The Cisco Unified Wireless IP Phones 7921G, 7925G, 7925G-EX, and 7926G XML parser returns error for unsupported XML objects and syntax errors. For values longer than specified length, the values are truncated.
8821	The Cisco Wireless IP Phone 8821 XML parser returns errors for unsupported XML objects and syntax errors.

Related Topics

[Deprecated Endpoints, on page 3](#)

CiscoIPPhone.xsd



Note

Cisco Unified IP Phones 6921, 6941, 6945, and 6961 do not support:

- NavLeft, NavRight, NavBack, and PushToTalk Key attributes
- CiscoIPPhoneKeyType

The following code is the schema file.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.4 U (http://www.xmlspy.com) by Cisco Systems, Inc. (Cisco Systems, Inc.) -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified" version="3.3.4">
  <xsd:complexType name="CiscoIPPhoneExecuteItemType">
    <xsd:attribute name="Priority" use="optional">
      <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedByte">
          <xsd:minInclusive value="0"/>
          <xsd:maxInclusive value="2"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="URL" use="required">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="256"/>
          <xsd:minLength value="1"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
  </xsd:complexType>
  <xsd:complexType name="CiscoIPPhoneResponseItemType">
    <xsd:sequence>
      <xsd:element name="Status" type="xsd:short"/>
      <xsd:element name="Data">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element name="URL">
        <xsd:simpleType>
```



```

        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="256"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneTouchAreaMenuItemType">
  <xsd:sequence>
    <xsd:element name="Name" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="0"/>
          <xsd:maxLength value="32"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="URL" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="0"/>
          <xsd:maxLength value="256"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="TouchArea" type="CiscoIPPhoneTouchAreaType" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneTouchAreaType">
  <xsd:attribute name="X1" type="xsd:unsignedShort" use="required"/>
  <xsd:attribute name="Y1" type="xsd:unsignedShort" use="required"/>
  <xsd:attribute name="X2" type="xsd:unsignedShort" use="required"/>
  <xsd:attribute name="Y2" type="xsd:unsignedShort" use="required"/>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneDirectoryEntryType">
  <xsd:sequence>
    <xsd:element name="Name" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="32"/>
          <xsd:minLength value="0"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Telephone" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="32"/>
          <xsd:minLength value="0"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneInputItemType">
  <xsd:sequence>
    <xsd:element name="DisplayName" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="32"/>
          <xsd:minLength value="0"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="QueryStringParam">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="32"/>
          <xsd:minLength value="1"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="InputFlags">

```

```

<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="A"/>
    <xsd:enumeration value="T"/>
    <xsd:enumeration value="N"/>
    <xsd:enumeration value="E"/>
    <xsd:enumeration value="U"/>
    <xsd:enumeration value="L"/>
    <xsd:enumeration value="AP"/>
    <xsd:enumeration value="TP"/>
    <xsd:enumeration value="NP"/>
    <xsd:enumeration value="EP"/>
    <xsd:enumeration value="UP"/>
    <xsd:enumeration value="LP"/>
    <xsd:enumeration value="PA"/>
    <xsd:enumeration value="PT"/>
    <xsd:enumeration value="PN"/>
    <xsd:enumeration value="PE"/>
    <xsd:enumeration value="PU"/>
    <xsd:enumeration value="PL"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name="DefaultValue" minOccurs="0">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="32"/>
      <xsd:minLength value="0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneMenuItemType">
  <xsd:sequence>
    <xsd:element name="Name" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="0"/>
          <xsd:maxLength value="64"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="URL" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="256"/>
          <xsd:minLength value="0"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneIconItemType">
  <xsd:sequence>
    <xsd:element name="Index" type="xsd:unsignedShort"/>
    <xsd:element name="Width">
      <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedShort">
          <xsd:minInclusive value="1"/>
          <xsd:maxInclusive value="16"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Height">
      <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedShort">
          <xsd:minInclusive value="1"/>
          <xsd:maxInclusive value="10"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Depth">

```

```

        <xsd:simpleType>
          <xsd:restriction base="xsd:unsignedShort">
            <xsd:minInclusive value="1"/>
            <xsd:maxInclusive value="2"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    <xsd:element name="Data" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:hexBinary">
          <xsd:maxLength value="40"/>
          <xsd:minLength value="0"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneIconMenuItemType">
  <xsd:sequence>
    <xsd:element name="Name" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="0"/>
          <xsd:maxLength value="64"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="URL" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="256"/>
          <xsd:minLength value="0"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="IconIndex" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:short">
          <xsd:minInclusive value="0"/>
          <xsd:maxInclusive value="9"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneIconFileType">
  <xsd:sequence>
    <xsd:element name="Index">
      <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedShort">
          <xsd:minInclusive value="0"/>
          <xsd:maxInclusive value="9"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="URL">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="256"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneKeyType">
  <xsd:sequence>
    <xsd:element name="Key">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="KeyPad0"/>
          <xsd:enumeration value="KeyPad1"/>
          <xsd:enumeration value="KeyPad2"/>

```

```

        <xsd:enumeration value="KeyPad3"/>
        <xsd:enumeration value="KeyPad4"/>
        <xsd:enumeration value="KeyPad5"/>
        <xsd:enumeration value="KeyPad6"/>
        <xsd:enumeration value="KeyPad7"/>
        <xsd:enumeration value="KeyPad8"/>
        <xsd:enumeration value="KeyPad9"/>
        <xsd:enumeration value="KeyPadStar"/>
        <xsd:enumeration value="KeyPadPound"/>
        <xsd:enumeration value="NavUp"/>
        <xsd:enumeration value="NavDown"/>
        <xsd:enumeration value="NavLeft"/>
        <xsd:enumeration value="NavRight"/>
        <xsd:enumeration value="NavSelect"/>
        <xsd:enumeration value="NavBack"/>
        <xsd:enumeration value="PushToTalk"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name="URL" minOccurs="0">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0"/>
            <xsd:maxLength value="256"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="URLDown" minOccurs="0">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0"/>
            <xsd:maxLength value="256"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CiscoIPPhoneSoftKeyType">
    <xsd:sequence>
        <xsd:element name="Name" minOccurs="0">
            <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                    <xsd:maxLength value="32"/>
                    <xsd:minLength value="0"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="Position">
            <xsd:simpleType>
                <xsd:restriction base="xsd:unsignedShort">
                    <xsd:minInclusive value="1"/>
                    <xsd:maxInclusive value="8"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="URL" minOccurs="0">
            <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                    <xsd:maxLength value="256"/>
                    <xsd:minLength value="0"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="URLDown" minOccurs="0">
            <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                    <xsd:minLength value="0"/>
                    <xsd:maxLength value="256"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

```

```

<xsd:complexType name="CiscoIPPhoneDisplayableType">
  <xsd:sequence>
    <xsd:element name="Title" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="0"/>
          <xsd:maxLength value="32"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Prompt" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="0"/>
          <xsd:maxLength value="32"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="SoftKeyItem" type="CiscoIPPhoneSoftKeyType" minOccurs="0"
maxOccurs="8"/>
    <xsd:element name="KeyItem" type="CiscoIPPhoneKeyType" minOccurs="0" maxOccurs="32"/>
  </xsd:sequence>
  <xsd:attribute name="keypadTarget" use="optional" default="application">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="application"/>
        <xsd:enumeration value="applicationCall"/>
        <xsd:enumeration value="activeCall"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="appId" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="64"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="onAppFocusLost" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="256"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="onAppFocusGained" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="256"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="onAppMinimized" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="256"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="onAppClosed" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="256"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

```

```

<xsd:element name="CiscoIPPhoneExecute">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="ExecuteItem" type="CiscoIPPhoneExecuteItemType" maxOccurs="3"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneResponse">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="ResponseItem" type="CiscoIPPhoneResponseItemType" maxOccurs="3"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneError">
  <xsd:complexType>
    <xsd:attribute name="Number" type="xsd:unsignedShort" use="required"/>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneText">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="Text" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:minLength value="0"/>
                <xsd:maxLength value="4000"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneInput">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="URL">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="256"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="InputItem" type="CiscoIPPhoneInputItemType" minOccurs="0"
maxOccurs="5"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneDirectory">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="DirectoryEntry" type="CiscoIPPhoneDirectoryEntryType"
minOccurs="0" maxOccurs="32"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneImage">
  <xsd:complexType>
    <xsd:complexContent>

```

```

<xsd:extension base="CiscoIPPhoneDisplayableType">
  <xsd:sequence>
    <xsd:element name="LocationX" default="0" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:short">
          <xsd:minInclusive value="-1"/>
          <xsd:maxInclusive value="132"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="LocationY" default="0" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:short">
          <xsd:minInclusive value="-1"/>
          <xsd:maxInclusive value="64"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Width">
      <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedShort">
          <xsd:minInclusive value="1"/>
          <xsd:maxInclusive value="133"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Height">
      <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedShort">
          <xsd:minInclusive value="1"/>
          <xsd:maxInclusive value="65"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Depth">
      <xsd:simpleType>
        <xsd:restriction base="xsd:unsignedShort">
          <xsd:minInclusive value="1"/>
          <xsd:maxInclusive value="2"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Data" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:hexBinary">
          <xsd:maxLength value="2162"/>
          <xsd:minLength value="0"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:extension>
</xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneImageFile">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="LocationX" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">
                <xsd:minInclusive value="-1"/>
                <xsd:maxInclusive value="297"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="LocationY" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">
                <xsd:minInclusive value="-1"/>
                <xsd:maxInclusive value="167"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

```

```

        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="URL">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="256"/>
          <xsd:minLength value="1"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneMenu">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="MenuItem" type="CiscoIPPhoneMenuItemType" minOccurs="0"
maxOccurs="100"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneIconMenu">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="MenuItem" type="CiscoIPPhoneIconMenuItemType" minOccurs="0"
maxOccurs="32"/>
          <xsd:element name="IconItem" type="CiscoIPPhoneIconItemType" minOccurs="0"
maxOccurs="10"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneIconFileMenu">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="MenuItem" type="CiscoIPPhoneIconMenuItemType" minOccurs="0"
maxOccurs="32"/>
          <xsd:element name="IconItem" type="CiscoIPPhoneIconFileItemType" minOccurs="0"
maxOccurs="10"/>
        </xsd:sequence>
        <xsd:attribute name="IconIndex" type="xsd:unsignedShort" use="optional"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneGraphicMenu">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="LocationX" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">
                <xsd:minInclusive value="-1"/>
                <xsd:maxInclusive value="132"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="LocationY" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">

```



```

        <xsd:minInclusive value="-1"/>
        <xsd:maxInclusive value="64"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="Width">
    <xsd:simpleType>
      <xsd:restriction base="xsd:unsignedShort">
        <xsd:minInclusive value="1"/>
        <xsd:maxInclusive value="133"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="Height">
    <xsd:simpleType>
      <xsd:restriction base="xsd:unsignedShort">
        <xsd:minInclusive value="1"/>
        <xsd:maxInclusive value="65"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="Depth">
    <xsd:simpleType>
      <xsd:restriction base="xsd:unsignedShort">
        <xsd:minInclusive value="1"/>
        <xsd:maxInclusive value="2"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="Data" minOccurs="0">
    <xsd:simpleType>
      <xsd:restriction base="xsd:hexBinary">
        <xsd:maxLength value="2162"/>
        <xsd:minLength value="0"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="MenuItem" type="CiscoIPPhoneMenuItemType" minOccurs="0"
maxOccurs="12"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneGraphicFileMenu">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="CiscoIPPhoneDisplayableType">
        <xsd:sequence>
          <xsd:element name="LocationX" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">
                <xsd:minInclusive value="-1"/>
                <xsd:maxInclusive value="297"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="LocationY" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">
                <xsd:minInclusive value="-1"/>
                <xsd:maxInclusive value="167"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="URL">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:maxLength value="256"/>
                <xsd:minLength value="1"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>

```

```

        <xsd:element name="MenuItem" type="CiscoIPPhoneTouchAreaMenuItemType"
minOccurs="0" maxOccurs="32"/>
    </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="CiscoIPPhoneStatus">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="Text" minOccurs="0">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:minLength value="0"/>
                        <xsd:maxLength value="32"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="Timer" minOccurs="0">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:unsignedShort">
                        <xsd:minInclusive value="0"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="LocationX" default="0" minOccurs="0">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:short">
                        <xsd:minInclusive value="-1"/>
                        <xsd:maxInclusive value="105"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="LocationY" default="0" minOccurs="0">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:short">
                        <xsd:minInclusive value="-1"/>
                        <xsd:maxInclusive value="20"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="Width">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:unsignedShort">
                        <xsd:minInclusive value="1"/>
                        <xsd:maxInclusive value="106"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="Height">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:unsignedShort">
                        <xsd:minInclusive value="1"/>
                        <xsd:maxInclusive value="21"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="Depth">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:unsignedShort">
                        <xsd:minInclusive value="1"/>
                        <xsd:maxInclusive value="2"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="Data" minOccurs="0">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:hexBinary">
                        <xsd:minLength value="0"/>
                        <xsd:maxLength value="557"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

```

```

        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="CiscoIPPhoneStatusFile">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="Text" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:minLength value="0"/>
                <xsd:maxLength value="32"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="Timer" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:unsignedShort">
                <xsd:minInclusive value="0"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="LocationX" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">
                <xsd:minInclusive value="-1"/>
                <xsd:maxInclusive value="261"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="LocationY" default="0" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:short">
                <xsd:minInclusive value="-1"/>
                <xsd:maxInclusive value="49"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="URL">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="256"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>

```




APPENDIX

C

Device Capability Query via CTI Feature

- [Feature Description, page 149](#)
- [Supported IP Phones and Codecs, page 149](#)
- [XML Object Changes, page 152](#)
- [Schema Definition, page 152](#)
- [Request and Response Examples for getDeviceCaps, page 153](#)
- [Troubleshooting, page 154](#)

Feature Description

The Device Capability Query via CTI feature was added for Cisco Unified Communications Manager Release 8.0(1).

A backend CTI application that communicates with the phone using the UserData tunnel cannot retrieve information on device capabilities such as XSI feature support on a phone. Due to this lack of information, and to ensure compatibility, only a minimum set of features were generally configured.

The Device Capability Query via CTI feature overcomes this limitation. This feature allows a CTI-based application or a Cisco Unified Communications Manager application to query a registered phone for device capabilities using the UserData tunnel interface of the phone (over SCCP or SIP and RemoteCC).

Applications that have an HTTP interface with a phone do not have this limitation. The HTTP request from such phones include XSI capabilities header, and the DeviceInformationX servlet of such phones can be accessed to retrieve other device information.

Although designed to work using CTI over the UserData tunnel, this feature can also work over HTTP using the POST method.

Supported IP Phones and Codecs

The following table lists the Cisco Unified IP Phone models that support the Device Capability Query via CTI feature.

**Note**

The Cisco Unified IP Phones 7902, 7905, 7910, and 7912, and the Cisco Unified Wireless IP Phone 7920 are deprecated with Cisco Unified Communications Manager 11.5(1) and later. The phones still work on previous versions of Cisco Unified Communications Manager.

Table 30: Phone Models that Support the Device Capability Query via CTI Feature

Phone model	Supported, Not supported	Firmware supported (see note)
Cisco Unified IP Phone 9900 series		
9951	Supported	9.0(1) and later
9971	Supported	9.0(1) and later
Cisco Unified IP Phone 8900 Series		
8941		
8945		
8961	Supported	9.0(1) and later
Cisco Unified IP Phone 8800 Series		
8811	Supported	10.2(2) and later
8841, 8851, 8861	Supported	10.2(1) and later
8851NR	Supported	10.3(1) and later
8845, 8865	Supported	10.3(2) and later
Cisco IP Conference Phones		
8831		
Cisco Wireless IP Phone 8820 series		
8821	Supported	11.0(1) and later
Cisco Unified IP Phone 7900 Series		
7905	Not supported	Not applicable
7906	Supported	8.4(1) and later
7911	Supported	8.4(1) and later

Phone model	Supported, Not supported	Firmware supported (see note)
7912	Not supported	Not applicable
7931	Supported	8.4(1) and later
7937	Not supported	Not applicable
7940	Not supported	Not applicable
7941	Supported	8.4(1) and later
7942	Supported	8.4(1) and later
7945	Supported	8.4(1) and later
7960	Not supported	Not applicable
7961	Supported	8.4(1) and later
7962	Supported	8.4(1) and later
7965	Supported	8.4(1) and later
7970	Supported	8.4(1) and later
7971	Supported	8.4(1) and later
7975	Supported	8.4(1) and later
7985	Not supported	Not applicable
Cisco Unified Wireless IP Phone 7900 Series		
7920	Not supported	Not applicable
7921G	Supported	1.0(3) and later
7925G	Supported	1.3(1) and later
7925G-EX	Supported	1.4(1) and later
7926G	Supported	1.4(1) and later
Cisco IP Phone 7800 Series		
7821	Supported	9.1(1) and later
7841	Supported	9.1(1) and later

Phone model	Supported, Not supported	Firmware supported (see note)
7861	Supported	9.1(1) and later
Cisco Unified IP Phone 6900 Series		
6921	Supported	9.1(1) and later
6941	Supported	9.1(1) and later
6945	Supported	9.1(1) and later
6961	Supported	9.1(1) and later
Other devices		
Cisco IP Phone Communicator	Not supported	Not applicable

**Note**

Cisco recommends the use of latest firmware. The firmware can be downloaded from the following location (requires login or service contract):

<http://software.cisco.com/download/navigator.html?i=!mmd>

Although several codecs are listed within the schema, only the codecs G711, G729, and G722 are currently supported.

Related Topics

[Deprecated Endpoints, on page 3](#)

XML Object Changes

To support this feature, new request and response objects are created. The `<getDeviceCaps>` is the request object and the `<getDeviceCapsResponse>` is the response object.

On receiving the `<getDeviceCaps>` object, the phone returns the `<getDeviceCapsResponse>` object. All elements in the `<getDeviceCapsResponse>` object are required and must not be null.

Schema Definition

The `getDeviceCapsResponse` XML schema is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.example.org/devicecaps"
xmlns:tns="http://www.example.org/devicecaps" xmlns="http://www.w3.org/2001/XMLSchema">
  <element name="getDeviceCapsResponse" type="tns:deviceCapType" nillable="true"/>
  <complexType name="deviceCapType">
    <all>
      <element name="physical" type="tns:physicalCapType" nillable="true"/>
      <element name="services" nillable="true"/>
    </all>
  </complexType>
</schema>
```



```

    <complexType>
      <complexContent>
        <extension base="tns:servicesCapType">
          <attribute name="sdkVersion" type="string" use="required"/>
        </extension>
      </complexContent>
    </complexType>
  </element>
</all>
</complexType>
<complexType name="physicalCapType">
  <all>
    <element name="modelName" nillable="false">
      <simpleType>
        <restriction base="string">
          <maxLength value="32"/>
          <minLength value="1"/>
        </restriction>
      </simpleType>
    </element>
    <element name="display" nillable="true">
      <complexType>
        <attribute name="width" type="unsignedShort" use="required"/>
        <attribute name="height" type="unsignedShort" use="required"/>
        <attribute name="bitDepth" type="unsignedShort" use="required"/>
        <attribute name="isColor" type="boolean" use="required"/>
      </complexType>
    </element>
  </all>
</complexType>
<complexType name="servicesCapType">
  <all>
    <element name="browser" type="tns:browserCapType" nillable="true"/>
  </all>
</complexType>
<complexType name="browserCapType">
  <all>
    <element name="accept" nillable="false"/>
    <element name="acceptLanguage" nillable="false"/>
    <element name="acceptCharset" nillable="false"/>
  </all>
</complexType>
</schema>

```

Request and Response Examples for getDeviceCaps

The following are the request and response examples for a getDeviceCaps object:

Request sent to the phone:

```
<getDeviceCaps/>
```

Response returned from the phone:

```

<getDeviceCapsResponse>
  <physical>
    <modelName>CP-7970</modelName>
    <display width="298" height="168" bitDepth="12" isColor="true"/>
  </physical>
  <services sdkVersion="5.0.3">
    <browser>
    </browser>
  </services>
</getDeviceCapsResponse>

```

Troubleshooting

The following error may occur in this feature:

- If the `getDeviceCaps` object is invalid (misspelled), a parsing error is generated and a `CiscoIPPhoneError` object (with `Number="1"`) is returned as the response.

Error Handling

Standard XML services debugging techniques are applied to this feature.

The root cause for any parsing errors is displayed in the phone console logs. For HTTP requests and responses, sniffer traces and web server debug can be used to examine the `getDeviceCaps` object to ensure that it conforms to the schema.



INDEX

A

accept header, support for [109](#)
 App, used as URI [98](#)
 audio types, supported [105](#)

C

CGI script [109](#)
 Execute [109](#)
 Screenshot [109](#)
 Cisco IP Phone Services Software Development Kit (SDK) [115](#),
 [117](#)
 components [115](#)
 development tools [115](#)
 sample services description [115](#)
 sample services requirements [115](#), [117](#)
 Cisco Unified Communications Manager [120](#), [122](#), [123](#)
 service addition [120](#)
 service parameter definition [122](#)
 service subscription [123](#)
 Cisco Unified IP Phone models [1](#), [11](#), [27](#), [43](#), [45](#), [57](#), [65](#), [70](#), [109](#)
 color depth [27](#)
 Component API support [57](#)
 directories [1](#)
 display and color support [27](#)
 information access [109](#)
 resolution [27](#)
 services [1](#)
 supported displays and colors [27](#)
 supported URIs [65](#), [70](#)
 Key [70](#)
 supported XML objects [11](#), [43](#), [45](#)
 CiscoIPPhone XML objects [11](#), [18](#), [19](#), [22](#), [23](#), [27](#), [31](#), [33](#), [36](#), [37](#), [43](#),
 [47](#), [48](#), [49](#)
 CiscoIPPhoneDirectory [22](#)
 CiscoIPPhoneError [49](#)
 CiscoIPPhoneExecute [48](#)
 CiscoIPPhoneGraphicFileMenu [33](#)
 CiscoIPPhoneGraphicMenu [31](#)
 CiscoIPPhoneIconFileMenu [37](#)

CiscoIPPhone XML objects (*continued*)

 CiscoIPPhoneIconMenu [36](#)
 CiscoIPPhoneImage [23](#)
 CiscoIPPhoneImageFile [27](#)
 CiscoIPPhoneInput [19](#)
 CiscoIPPhoneMenu [18](#)
 CiscoIPPhoneResponse [49](#)
 CiscoIPPhoneStatus [43](#)
 CiscoIPPhoneStatusFile [47](#)
 supported phone models [11](#)
 understanding [11](#)
 client requests using HTTP [101](#)
 content expiration, header setting [105](#)
 cookie behavior [106](#)

D

development tools, included with SDK [115](#)
 device control URI [70](#), [81](#)
 Display [81](#)
 Key [70](#)
 device information URLs [109](#)
 Device, used as URI [91](#)
 DeviceInformation [109](#)
 DeviceList XML Object [128](#)
 attributes [128](#)
 description [128](#)
 DeviceListX report [125](#)
 DeviceLog [109](#)
 Dial, used as URI [92](#)
 directories, custom [22](#)

E

EditDial, used as URI [93](#)
 error codes, description [49](#), [85](#)
 EthernetInformation [109](#)
 execute items, send requests [48](#)
 Execute, CGI script [109](#)

F

features, supported [5](#)

G

graphic menus, creating [31](#)
 graphics [23, 27, 37](#)
 displaying grayscale images [23](#)
 PNG support [27, 37](#)

H

header settings for HTTP [102](#)
 HTML URL [109](#)
 DeviceInformation [109](#)
 DeviceLog?n [109](#)
 EthernetInformation [109](#)
 NetworkConfiguration [109](#)
 PortInformation?n [109](#)
 SteamingStatistics?n [109](#)
 HTTP [101, 102, 103, 104, 105, 106, 108, 109](#)
 client requests [101](#)
 header settings [102, 103, 104, 105, 106, 108, 109](#)
 accept [109](#)
 content expiration [105](#)
 MIME type [104](#)
 Refresh [103](#)
 set-cookie [106](#)
 x-CiscoIPPhoneDisplay [108](#)
 x-CiscoIPPhoneModelName [108](#)
 x-CiscoIPPhoneSDKVersion [108](#)
 how used [101](#)
 server requests [102](#)
 HTTP GET [101](#)
 HTTP POST [102](#)

I

icon menus, creating [36, 37](#)
 color [37](#)
 grayscale [36](#)
 Init, used as URI [95](#)
 input forms [19, 20](#)
 creating [19](#)
 supported input types [20](#)

M

menus [18, 31, 36, 37](#)
 graphic [31](#)
 icon [37](#)
 icon, grayscale [36](#)
 text [18](#)
 MIME type for HTTP [104](#)

N

NetworkConfiguration [109](#)
 Notify, used as URI [96](#)

P

Play, used as URI [88](#)
 PNG images [27, 37](#)
 displaying on screen [27](#)
 using in menus [37](#)
 PortInformation [109](#)

R

refresh setting [103](#)
 for HTTP [103](#)
 RTPMRx, used as URI [87](#)
 RTPMTx, used as URI [88](#)
 RTPRx, used as URI [86](#)
 RTPTx, used as URI [87](#)

S

Screenshot, CGI script [109](#)
 SDK, See [Cisco IP Services Software Development Kit \(SDK\)](#)
 SendDigits, used as URI [94](#)
 server requests using HTTP [102](#)
 services [115, 117, 120, 122, 123](#)
 add to Cisco Unified Communications Manager [120](#)
 define parameters in Cisco Unified Communications Manager [122](#)
 requirements for samples [117](#)
 samples included [115](#)
 subscribing to in Cisco Unified Communications Manager [123](#)
 set-cookie, header setting [106](#)
 softkey [82](#)
 custom [82](#)
 native [82](#)
 special characters [51](#)

status, displaying for applications [47](#)
 SteamingStatistics [109](#)

T

text menus, creating [18](#)
 text messages, displaying [18](#)
 troubleshooting [111](#), [112](#)
 error messages [112](#)
 tips [111](#)
 XML parsing errors [111](#)

U

uniform resource identifiers (URI) [65](#), [82](#), [83](#), [85](#), [86](#), [87](#), [88](#), [90](#), [91](#),
 [92](#), [93](#), [94](#), [95](#), [96](#), [98](#)
 control RTP streaming [85](#), [86](#), [87](#), [88](#)
 RTPMRx [87](#)
 RTPMTx [88](#)
 RTPRx [86](#)
 RTPTx [87](#)
 description [65](#)
 displayable objects [82](#), [83](#)
 QueryStringParam [83](#)
 SoftKey [82](#)
 miscellaneous [88](#), [90](#), [91](#), [92](#), [93](#), [94](#), [95](#), [96](#), [98](#)
 App [98](#)
 Device [91](#)
 Dial [92](#)
 EditDial [93](#)

uniform resource identifiers (URI) (*continued*)
 miscellaneous (*continued*)
 Init [95](#)
 Notify [96](#)
 Play [88](#)
 SendDigits [94](#)
 Vibrate [90](#)
 URI, See [uniform resource identifiers \(URI\)](#)

V

Vibrate, used as URI [90](#)

X

x-CiscoIPPhoneDisplay [108](#)
 x-CiscoIPPhoneModelName [108](#)
 x-CiscoIPPhoneSDKVersion [108](#)
 XML considerations [51](#)
 mandatory escape sequences [51](#)
 XML objects [18](#)
 CiscoIPPhoneText [18](#)
 XML schema file [135](#)
 XML URL [109](#)
 DeviceInformationX [109](#)
 DeviceLogX?n [109](#)
 EthernetInformationX [109](#)
 NetworkConfigurationX [109](#)
 PortInformationX?nX [109](#)
 SteamingStatisticsX?n [109](#)

