

XSI Application Management

XML Service Interface Application Note



Target Audience: Unified Communication XML Application developers

Concept: This document was created by the Cisco Voice Technology Group as part of a series of documents to provide information and implementation guidance on new XML application interface objects as they become available.

The XML Service Interface documentation and SDK are updated as new versions of Cisco Unified Communication Manager are released. In between releases of Cisco Unified Communication Manager, new releases of IP phone firmware are also released. Some of these IP phone firmware releases may contain enhancements to the XML Service Interface. This series of XML service interface applications notes has been produced as a way of making these enhancements available to the developer community as early as possible.

The information contained in this application note will be incorporated into the next release of the XML application interface documentation and the SDK. At that time, the information in this document will be deprecated, but until that time this document acts as the reference documentation for the XML Service Interface API being described.



XSI Application Management

Table of Contents

Introduction	2
Application URI	2
Event Handlers	4
Syntax	7
Event Handler Attributes.....	7
Event Schema	7
Application URI	7
Sample Usage	9
Event Handlers	9
Application URI	9
Pre-requisites	9
Supported Platforms	10
Error and Response.....	10
Application URI	10
Debug / Logging.....	10
Caveats	10
Additional References	10

Introduction

As many application developers are aware, there is little in the way of state notification and application management in the current XML Services implementation. This can lead to a lack of context when an IP phone is in application mode, and a new call arrives at the phone. In order to provide a more elegant handoff between call mode and application mode, a new API, the Application Management API has been implemented.

Application URI

The Application Management API includes a new Application URI which allows application to request changes to their application/window state. Applications can request to lose/gain focus or to be minimized or closed.



The Application URI also includes priority and idle timer attributes which allow an application to better control the relative importance of the application state change request. The priority attribute essentially allows application to request a state change ‘immediately’, ‘when idle’, or ‘if idle’, and the idle timer value allows the application to specify the idle timeout interval used by the priority attribute. For the “ReleaseFocus”, “Minimize”, and “Close” actions, the idle timer value is defined as the amount of time since the user last interacted with that specific application window/context. For the “GainFocus” action, the idle timer value is the amount of time since the user last interacted with **any** application on the phone. Any pending application state change timers are automatically cancelled any time the displayable XSI object changes for a given application context. Note that state change timers are maintained on a per-application-context basis, so displayable object changes in one context (say Services) does not affect timers running in a different context (say Directories).

Anytime an Application URI request is made, it explicitly has a particular 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 ‘appId’ parameter **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 as they were interacting with it. 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.

Application URI	Description
App:RequestFocus	<p>Make a request to the application manager to bring the application context (window) containing this application into focus (maximize).</p> <p>If the requested application is Open, but not currently Active, this request will not succeed (error response) .</p> <p>If the application already has focus, the request has no effect.</p> <p>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.</p>
App:ReleaseFocus	<p>Make a request to the application manager to relinquish focus to another application context – essentially a “move to back” request.</p>



	<p>If the application doesn't have focus, the request has no effect.</p> <p>If there are no other applications open (available to receive focus) then this application will retain focus. Applications must use onAppFocusLost event handlers to know when focus is actually lost.</p>
App:Minimize	<p>Make a request to the application manager to minimize (tabify) the application context containing this application.</p> <p>If the requested application is Open, but not currently Active, this request will not succeed (error response).</p> <p>If the application is already minimized, the request has no effect.</p> <p>This request will always result 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.</p>
App:Close	<p>Make a request to the application manager to close the application context containing this application.</p> <p>If the requested application is Open, but not currently Active, this request will not succeed (error response).</p> <p>This request will result in the application context (and all applications within that context) being closed.</p> <p>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).</p>

Event Handlers

The Application Manager API includes Event Handlers which can be attached to any XML displayable object by specifying the following attributes:

Attribute	Description
-----------	-------------



appId	The application to which this displayable XSI 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 is free to assign any type of globally unique identifier.
onAppFocusLost	<p>Event handler invoked whenever the application loses focus.</p> <p>Loss of focus can occur in 2 ways:</p> <ol style="list-style-type: none">1. The application’s context has lost focus2. The application was navigated away from, either directly by the user, or programmatically by a refresh header or HTTP push. <p>If a Notify URI is used as the event handler, the default ‘data’ sent in the notification will be:</p> <pre><notifyApplicationEvent appId=”appId” type=”focusLost”/></pre>
onAppFocusGained	<p>Event handled invoked whenever the application gains focus.</p> <p>Focus can be gained in 2 ways:</p> <ol style="list-style-type: none">1. The application is Active and the application’s context has gained focus2. The application was navigated to, either directly by the user, or by a refresh header or HTTP push. <p>If a Notify URI is used as the event handler, the default ‘data’ sent in the notification will be:</p> <pre><notifyApplicationEvent appId=”appId” type=”focusGained”/></pre>
onAppMinimized	<p>Event handler invoked whenever the application is minimized.</p> <p>Currently, an application can only be minimized programmatically by a call to App:Minimize, but this invocation could occur by direct action of the user (from a softkey invocation, for example) or from the application via a push request.</p> <pre><notifyApplicationEvent appId=”appId” type=”minimized”/></pre>
onAppClosed	<p>Event handler invoked whenever the application closes.</p> <p>The application will be closed when:</p>



	<ol style="list-style-type: none">1. The application's context is closed which will, in turn, close all applications in its stack.2. The application no longer exists on the context's URL stack because it was navigated out of, or because it was pruned from the URL stack (stack size exceeded). <p>This event handler cannot contain HTTP or HTTPS URLs.</p> <p>If a Notify URI is used as the event handler, the default 'data' sent in the notification will be:</p> <pre><notifyApplicationEvent appId="appId" type="closed"/></pre>
--	--

NOTE: An App URI with Priority=0 is not allowed in any App Mgmt Event Handlers.



Syntax

Event Handler Attributes

The event handler attributes can be used on any displayable XSI object:

Menu, IconMenu, IconFileMenu, GraphicMenu, GraphicFileMenu, Image, ImageFile, Text, Input, Directory.

They cannot be used on the Status or StatusFile objects because those objects are not contained in a standard application context and they are handled differently by the Application Manager.

Event 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>
```

Application URI

App:{action}:{priority}[:{idleTimer}[:{applicationId}]]

**{action}**

Description: The action to be taken with the application

Type: Enumeration

Values/Range: 'RequestFocus', 'ReleaseFocus', 'Minimize', 'Close'

Default-value: N/A

{priority}

Description: Priority at which the action should be taken

Type: Enumeration

Values/Range:

'0' = Do immediately, even if user is interacting with the phone

'1' = Do when user is done interacting with the phone

'2' = Do only if user is not interacting with the phone

Default-value: N/A

NOTE: Priority 0 is not allowed if the App URI is contained in an App Mgmt Event Handler.

{idleTimer}

Description: How long the phone or application must be idle (in seconds) before this action should be taken.

Type: Integer

Values/Range: min=10, max=86400 (24 hours)

Default-value: 60

NOTE: The idleTimer value has no effect for Priority=0 requests.

NOTE: Any pending timers are automatically cancelled whenever the displayable object changes for an application context.

{applicationId} (Optional)

Description: The ID of the application that the action should be taken on

Type: String

Values/Range: minLength = 1, maxLength=64; cannot contain colons

Default-value: The application of the displayable object in which this URI is defined.

NOTE: If the Application URI is used in an ExecuteItem, the 'applicationId' parameter must be specified since the application context of the request cannot be inferred.



Sample Usage

Event Handlers

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

→ Notify

```
<notifyApplicationEvent appId="Cisco/Unity" type="focusLost"/>
```

Application URI

```
<CiscoIPPhoneImage appId="Cisco/Unity"  
  onAppFocusLost="App:Close:1:30"  
  ...  
</CiscoIPPhoneImage>  
  
<CiscoIPPhoneExecute>  
  <ExecuteItem URL="App:Close:0::Cisco/Unity"/>  
</CiscoIPPhoneExecute>
```

Pre-requisites

This XML Service Interface API, requires IP phone firmware 8.3(2) or later. For the availability of the 8.3(2) phone firmware, please consult www.cisco.com or contact the Developer Support team in order to obtain an early release of this phone firmware.

http://www.cisco.com/en/US/products/svcs/ps3034/ps5408/ps5418/serv_home.html

NOTE: Phone Firmware 8.3(2) contains an updated XML parser, with more rigid enforcement of the XML schema. XML applications that have been written to earlier versions of the XML parser, may encounter exceptions if they do not conform to the XML schema. It is recommended that any existing XML applications be validated with the new XML parser before trying to enhance the XML application to use this new XML Service Interface API.



For details on the new XML parser, please consult the XML Service Interface Application note entitled **IP Phone Services XML Schema Enforcement**

Supported Platforms

The following models of IP phones will support this XML Service Interface API, 7906, 7911, 7931, 7941, 7961, 7970, and 7971

The 7905, 7912, 7920, 7940, 7960 and 7985 will not support this XML Service Interface API

Error and Response

Application URI

All Application URI requests are asynchronous, so the only return value will be to indicate 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 via 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
If a request is made to change the state of an application which is not currently Active.	6 (Internal error)	Application is not Active

Debug / Logging

Any failures to deliver a notification will be logged to the phone console (and web console logs) to allow debugging of undelivered events.

Caveats

An App URI with Priority=0 is not allowed in any App Mgmt Event Handlers.

Additional References

Cisco IP phone Services along with developer resources such as the SDK to assist developers in getting started with applications and documentation on the various URIs and XML Objects that are supported in the IP phones are available at the link posted below.



http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_usage_guide_book09186a00807a34b9.html

In recent years, Cisco has introduced a new generation of desktop IP phone models, the 7906, 7911, 7931, 7941, 7961, 7970 and 7971. These new models of phones have been introduced as a platform in which new features can be introduced, and on which new applications can be built. The introduction of these new generation IP phones has also provided an opportunity to increase the range of XML objects and URIs that are supported. The introduction of enhancements to IP phone services is also the time to remind application developers, of the XML schema that is defined for the IP phone services. This schema is published at the following location, and has been since initial release

http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_usage_guide_chapter09186a00807a352c.html

Please contact the developer support team and gain more information about Developer services and access to early deployment phone loads.

http://www.cisco.com/en/US/products/svcs/ps3034/ps5408/ps5418/serv_home.html