CHAPTER 4

Internal URI Features

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

These sections provide details about the available internal URIs:

- Supported URIs by Phone Model
- Device Control URIs
- XML Displayable Object URIs
- Multimedia URIs
- Telephony URIs
- Application Management URIs

Supported URIs by Phone Model

Table 4-1 lists the URIs that are supported for Release 7.0(1).

Table 4-1 URIs Supported for Release Cisco Unified IP Phone Services SDK

URI	7905G 7912G	7906G 7911G 7931G	7920G	7921G	7940G 7960G	7941G/7941G-GE, 7961G/7961G-GE, 7942G, 7962G, 7945G, 7965G, IP Communicator	7970G 7971G-GE 7975G
Key	X	X	X	X	X	X	X
Softkey	X	X	X	X	X	X	X
Init	X	X	X	X	X	X	X
Dial, EditDial	X	X	X	X	X	X	X
Play	X	X	X	X	X	X	X
QueryStringParam	X	X	X	X	X	X	X
Unicast RTP	X	X	X[1]	X	X	X	X
Multicast RTP	X	X	X	X	X	X	X
Display	—	—	—	—	—	—	X
Vibrate	—	—	X	X	—	—	
Notify[2]	—	X	—	X	—	X	X
SendDigits[2]	—	X	—	X	—	X	X
Application[2]	—	X	—	X	—	X	X

1. Only supports one incoming and one outgoing unicast stream and does not support the Volume parameter for RTP Receive streams.
2. Requires Cisco Unified IP Phone firmware version 8.3(2) or later, which contains an updated XML parser. See the "Updated XML Parser and Schema Enforcement" section on page B-1.

Device Control URIs

These sections describe the device control URIs:

- Key
- Display

Key

The Key URI allows a programmer 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.

Verify available softkeys by using the QA web pages that the phones web server provides to indicate the active softkey set.

URI Format

```
Key:n
```

Where

n = a Key name.

The following is a complete listing of the Key URIs:

- Key:Line1 to Key:Line34
- Key:KeyPad0 to Key:KeyPad9
- Key:Soft1 to Key:Soft5
- Key:KeyPadStar
- Key:KeyPadPound
- Key:VolDwn
- Key:VolUp
- Key:Headset

- Key:Speaker
- Key:Mute
- Key:NavLeft
- Key:NavRight
- Key:NavSelect
- Key:Info
- Key:Messages
- Key:Services
- Key:Directories
- Key:Settings
- Key:NavUp
- Key:NavDwn
- Key:AppMenu
- Key:Hold

Display

The Display URI is available only on those Cisco Unified IP Phones that have a color backlight on the phone display, including the Cisco Unified IP Phone 7970G and 7971G-GE. 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. As such, various phone states (such as phone startup, incoming and active calls, or other user input states) override the Display URI settings.

URI Format

```
Display:State:Interval
```

Where

State = whether the phone display is turned on (**`On`**) or off (**`Off`**) or set to default (**`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).

For example:

- `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 off 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
- QueryStringParam

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.

URI Format

`SoftKey:n`

Where

n = one of the following softkey names:

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

Table 4-2 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 4-2 Valid Softkey Actions for CiscoIPPhoneObject Types

IPPhoneObject[1]	Select	Exit	Update	Submit	Search	<<	Cancel	Next	Dial	Edit Dial
CiscoIPPhoneMenu	X	X								
CiscoIPPhoneIconMenu	X	X								
CiscoIPPhoneText		X	X							
CiscoIPPhoneImage		X	X							
CiscoIPPhoneGraphicMenu		X	X							
CiscoIPPhoneInput				X	X[2]	X	X			
CiscoIPPhoneDirectory							X	X	X[3]	X[3]

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. For more details, see the "Telephony URIs" section on page 4-15.

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.

URI Format

```
QueryStringParam:d
```

Where

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

Example 4-1 QueryStringParam URI in a 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 4-1 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 MenuItem(s) 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 you highlight the first MenuItem and press the Read softkey, the phone generates the following URL:

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

Example 4-2 Selecting an Item with Numeric Keypad Calls the URL

```
<CiscoIPPhoneMenu>
    <Title>Message List</Title>
    <Prompt>Two Messages</Prompt>
    <MenuItem>
        <Name>Messae One</Name>
        <URL>http://server/messages.asp?message=1</URL>
    </MenuItem>
    <MenuItem>
        <Name>Messae 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>
```

The Cisco Unified IP Phones allow you to implement the QueryStringParam URI in either manner although Example 4-2 is not as efficient as Example 4-1. Choose the best way to perform the action based on your applications needs.

Example 4-2 does have a slight advantage in that if the user chooses an item with the numeric keypad, the URL gets called. This 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 not having to repeat redundant portions of a URL in every MenuItem.

Multimedia URIs

These sections describe the multimedia URIs:

- RTP Streaming
- Play
- Vibrate

RTP Streaming

You can invoke RTP streaming via 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. See the "RTP Streaming API" section on page 3-2.

The supported format of the RTP stream is as follows:

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

The following list gives these possible CiscoIPPhoneError codes:

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

Interaction with Call Streaming

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

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

- Existing Rx URI streams will be terminated if a new call begins or an existing call is resumed.

  The user has no explicit mechanism for terminating the Rx URI stream independent of the call. Thus, if the Rx 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 URI stream requests received during an active call will be accepted (whisper), but the volume parameter of the URI will be ignored.

  If the Rx URI request was done via 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 URI via an application, then the user likely is not concerned about having the audio mixed with the current call. However, they should also be presented with an option to stop the application, when needed.
- For the Rx 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, and
  - There is at least one active receive stream

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

- If a Rx or Tx 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` will be 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.

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 playout. 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.

URI Formats

```
RTPTx:i:p
RTPTx:Stop
```

Where

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

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 transmitted on channel one.

RTPMRx

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

URI Format

```
RTPMRx:i:p:v
```

Where

i = the Multicast IP Address from which to receive an RTP stream.

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 playout. 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.

URI Formats

```
RTPTx:i:p
```

Where

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

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 MWI light, it will be flashing 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 via HTTP, the HTTP response (CiscoIPPhoneResponse object) is not returned until after the playback has completed.

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.

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.

Vibrate

The Vibrate URI is available on the Cisco Unified IP Phones 7920G and 7921G wireless phone models, and it enables third-party applications to invoke the phone's 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 will execute its default vibrate sequence.

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.

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

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

For example:

- `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.

Telephony URIs

These sections describe the telephony URIs:

- Dial
- EditDial
- SendDigits

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:

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

URI Format

```
Dial:n
```

Where

n = the number dialed (such as **Dial:1000**).

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:

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

URI Format

```
EditDial:n
```

Where

n = the number dialed (such as **EditDial: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.

URI Format

```
SendDigits:dtmfSequence:audibleFeedback::applicationId
```

Where

dtmfSequence = the sequence of DTMF digits to be sent. Value must contain only 0123456789#*ABCD

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.

For 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>
```

Error and Response

When the SendDigits URI is invoked via an Execute object, it will use 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
- Notify
- Application

Init

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

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.

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.

For example:

- 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, see the "Application Event Handlers" section on page 2-29 for details.

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 **appId** 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.

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 (see the "Application Event Handlers" section on page 2-29) 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 (see the "Application Event Handlers" section on page 2-29).
  - 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 (see the "Application Event Handlers" section on page 2-29).
  - 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 (see the "Application Event Handlers" section on page 2-29).
- 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.

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 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
Request made to change state of an application that is not current active	6 (Internal error)	Application is not Active