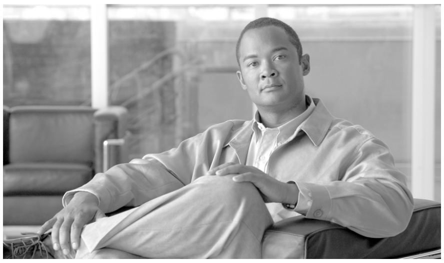


CHAPTER 2

CiscoIPPhone XML Objects

The following sections describe the general behavior and use of XML objects:

- Understanding Object Behavior
- XML Object Definitions
- Custom Softkeys
- XML Considerations
- Application Event Handlers

Understanding Object Behavior

Creating interactive service applications is relatively easy when you understand the XML objects that are defined for Cisco Unified IP Phones and the behavior that each object generates.

Regarding services, the phone does not have any concept of a state when it loads an XML page. Cisco Unified IP Phones can use HTTP to load a page of content in many different places, starting when the **services** button is pressed. 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 are using and the server side mechanism that you are using to create your pages (for example, static files, JavaScript, CGI, and so on). See Chapter 6, "HTTP Requests and Header Settings" for more information.

Table 2-1 shows the supported XML objects for this release.

Table 2-1 XML Objects Supported for Release 7.0(1) Cisco Unified IP Phone Services SDK

Phone Model XML Object	**7905G 7906G 7911G 7912G 7931G**	**7920G**	**7921G**	**7940G 7960G**	**7941G/7941G-GE 7942G, 7945G, 7961G/7961G-GE, 7962G, 7965G, 7970G/ 7971G-GE, 7975G, IP Communicator**
CiscoIPPhoneMenu	X	X	X	X	X
CiscoIPPhoneText	X	X	X	X	X
CiscoIPPhoneInput	X	X	X	X	X
CiscoIPPhoneDirectory	X	X	X	X	X
CiscoIPPhoneImage		X[1]	X	X	X
CiscoIPPhoneImageFile			X		X
CiscoIPPhoneGraphicMenu		X[1]	X	X	X
CiscoIPPhoneGraphicFileMenu			X		X
CiscoIPPhoneIconMenu	X[2]	X	X	X	X
CiscoIPPhoneIconFileMenu			X		X[3]
CiscoIPPhoneStatus				X	X
CiscoIPPhoneStatusFile			X		X[3]
CiscoIPPhoneExecute	X	X[4]	X	X	X
CiscoIPPhoneResponse	X	X	X	X	X
CiscoIPPhoneError	X	X	X	X	X

1. The Cisco Unified IP Phone 7920G 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 Phones 7905G and 7912G do not support CIP images; therefore, all icons get ignored and do not display.
3. The Cisco Unified IP Phones 7970G and 7971G-GE require firmware version 7.01 or higher to support this object, and Cisco IP Communicator requires software version 2.01 or higher.
4. The Cisco Unified IP Phone 7920G does not support Priority 1 when on a call.

XML Object Definitions

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

- CiscoIPPhoneMenu
- CiscoIPPhoneText
- CiscoIPPhoneInput
- CiscoIPPhoneDirectory
- CiscoIPPhoneImage
- CiscoIPPhoneImageFile
- CiscoIPPhoneGraphicMenu
- CiscoIPPhoneGraphicFileMenu
- CiscoIPPhoneIconMenu
- CiscoIPPhoneIconFileMenu
- CiscoIPPhoneStatus
- CiscoIPPhoneStatusFile
- CiscoIPPhoneExecute
- CiscoIPPhoneResponse
- CiscoIPPhoneError

CiscoIPPhoneMenu

A menu on the phone comprises a list of text items, one per line. Users choose individual menu items by using the same mechanisms that are used for built-in menus in the phone as described in Chapter 1, "Overview".

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 XML format allows you to specify a title and prompt that are used for the entire menu, followed by a sequence of `MenuItem` objects.
Cisco Unified IP Phones allow a maximum of 100 `MenuItems`. Each `MenuItem` includes a `Name` and an associated `URL`.

When a menu is loaded, the phone behaves the same as for built-in phone menus. The user navigates through the list of menu items and eventually chooses one by 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.

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 Cisco Unified IP Phone firmware controls all other pagination and wordwrap issues.

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

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>
```

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.

Note **Non-XML Text**: This document only describes the supported CiscoIPPhone XML objects. You can also deliver plain text via 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.

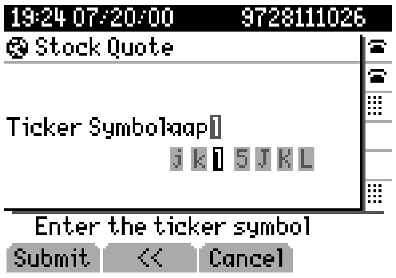
Note **Keypad navigation**: Cisco Unified IP Phones allow navigation to a specific line in a menu by pressing numeric DTMF keys. When a menu is on the display, the number for selecting the menu is on the left.

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

CiscoIPPhoneInput

When a Cisco Unified IP Phone receives an XML object of type `CiscoIPPhoneInput`, it constructs an input form and displays it. The user then enters data into each input item and sends the parameters to the target URL. Figure 2-1 shows a sample display that is receiving input from a user.

Figure 2-1 Sample User Input Display



19:24 07/20/00 9728111026
Stock Quote
Ticker Symbol:aapl
Enter the ticker symbol

Definition

```
<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 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>
```

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

The `URL` tag delimits 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 CiscoIPPhoneInput objects do not use the HTTP POST method.

The `InputItem` tag delimits 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.

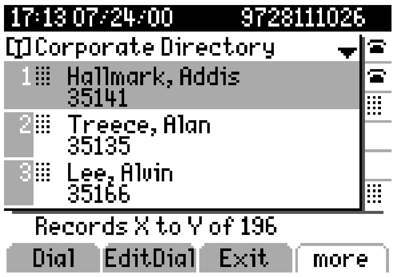
InputFlag	Description
A	Plain ASCII text—use the DTMF keypad to enter text that consists of uppercase and lowercase letters, numbers, and special characters.
T	Telephone number—enter only DTMF digits for this field. The acceptable input includes numbers, #, and *.
N	Numeric—enter numbers as the only 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—enter individual characters using the standard keypad-repeat entry mode. The system automatically converts accepted characters into an asterisk, keeping the entered value private. **Note** P specifies the only `InputFlag` that works as a modifier. For example, specify a value of "AP" in the `InputFlag` field to use plain ASCII as the input type and to mask the input as a password by using an asterisk (*).

During text entry, Cisco Unified IP 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, and so on.

CiscoIPPhoneDirectory

The phone originally incorporated the `CiscoIPPhoneDirectory` XML object to support the Directory operation of Cisco Unified IP Phones, but it is available for your development purposes also. Figure 2-2 shows how an XML `CiscoIPPhoneDirectory` object displays on the phone.

Figure 2-2 CiscoIPPhoneDirectory Object Display Sample


17:13 07/24/00 9728111026
Corporate Directory
1 Hallmark, Addis
35141
2 Treece, Alan
35135
3 Lee, Alvin
35166
Records X to Y of 196

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 Cisco Unified 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.

Custom Directories

You can use the Cisco Unified Communications Manager enterprise parameter, “URL Directories” and CiscoIPPhone XML objects to display custom directories. The “URL Directories” points to a URL that returns a `CiscoIPPhoneMenu` object that extends the **directories** menu. The request for “URL Directories” must return a valid `CiscoIPPhoneMenu` object, even if 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 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 Cisco Unified IP Phone display to present graphics to the user.

Definition

```
<CiscoIPPhoneImage>
  <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 softkey</Name>
    <URL>URL of softkey</Name>
    <Position>Numerical position of the softkey</Position>
   </SoftKeyItem>
</CiscoIPPhoneImage>
```

The `CiscoIPPhoneImage` object definition includes two familiar elements: `Title` and `Prompt`. These 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.

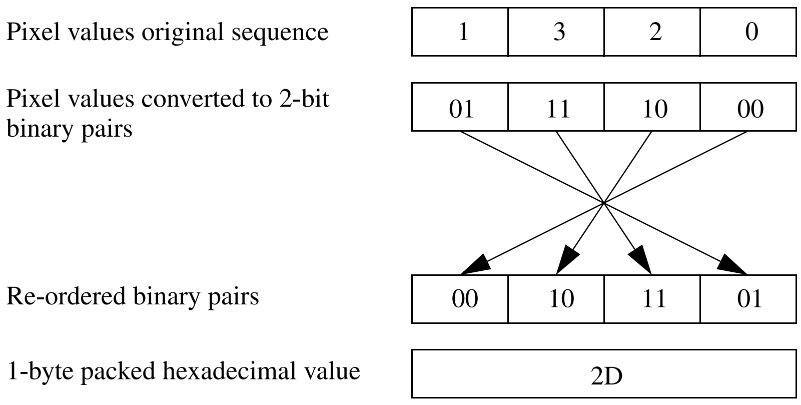
Use `Width` and `Height` to size the graphic. If the values do not match with the pixel stream specified in the `Data` field, results will be unpredictable incorrect.

`Depth` specifies the number of bits per pixel. Cisco Unified IP Phones support a maximum value of 2. 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 Cisco Unified 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.

Figure 2-3 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 re-ordered in sets of four to create a single re-ordered byte, which two hexadecimal digits represent.

Figure 2-3 Packed Pixel Translation Example


Pixel values original sequence
1
3
2
0
Pixel values converted to 2-bit binary pairs
01
11
10
00
Re-ordered binary pairs
00
10
11
01
1-byte packed hexadecimal value
2D

Example

The following XML code defines a `CiscoIPPhoneImage` object that displays the sequence of pixels shown in Figure 2-3 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.

Tip Before displaying a graphic image on a Cisco Unified 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.

CiscoIPPhoneImageFile

The latest generation of Cisco Unified 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, a new 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 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 Table 2-2).

Table 2-2 Cisco Unified IP Phones Display Image Sizes and Color Depths

Model	Resolution[1] (width x height)	Color/Grayscale	Color Depth (bits)
Cisco Unified IP Phones 7905G, 7906G, 7911G, 7912G[2], 7931G	N/A	Grayscale	1
Cisco Unified IP Phone 7920	128 x 59	Grayscale	1
Cisco Unified IP Phone 7921G	176 x 140	Color	16
Cisco Unified IP Phones 7940G/60G	133 x 65	Grayscale	2
Cisco Unified IP Phones 7941G, 7941G-GE, 7942G, 7961G, 7961G-GE, 7962G	298 x 144	Grayscale	4

Table 2-2 Cisco Unified IP Phones Display Image Sizes and Color Depths (continued)

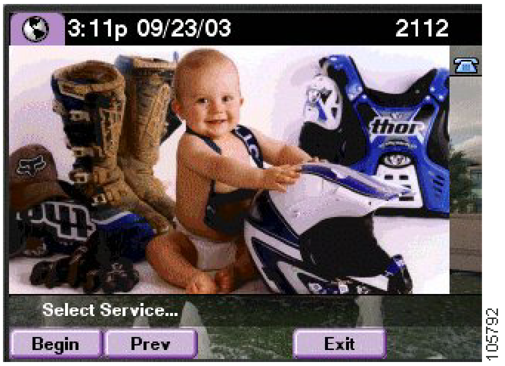
Model	Resolution[1] (width x height)	Color/Grayscale	Color Depth (bits)
Cisco Unified IP Phones 7945G, 7965G	298 x 156	Color	16
Cisco Unified IP Phone 7970G/7971G	298 x 168	Color	12
Cisco Unified IP Phone 7975G	298 x 168	Color	16
Cisco IP Communicator	298 x 168	Color	24

1. 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) will correctly dither the color palette of the image for the best display results on the Cisco Unified IP Phone 7970G.

Figure 2-4 shows a CiscoIPPhoneImageFile object on a Cisco Unified IP Phone 7970G display.

Figure 2-4 Cisco Unified IP Phone 7970G Image File Display



Definition

```
<CiscoIPPhoneImageFile>
  <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

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).

Definition

```
<CiscoIPPhoneGraphicMenu>
  <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>
```

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 is loaded. If the loaded page has a title of its own, the phone uses that 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. See "CiscoIPPhoneMenu" and "CiscoIPPhoneImage" for detailed descriptions.

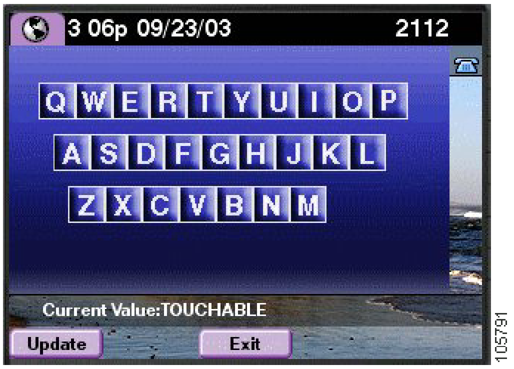
CiscoIPPhoneGraphicFileMenu

Some of the latest 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 new 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>`.

Figure 2-5 shows the display of the CiscoIPPhoneGraphicFileMenu.

Figure 2-5 CiscoIPPhoneGraphicFileMenu



If the coordinates that are supplied in `<TouchArea>` tag exceed the dimensions of the phone display, the `<TouchArea>` rectangle will be "clipped" to fit. See Table 2-2, "Cisco Unified IP Phones Display Image Sizes and Color Depths" for a listing of usable display resolutions for each phone model.

The `<TouchArea>` rectangles are allowed to 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.

Definition

```
<CiscoIPPhoneGraphicFileMenu>
  <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>
```

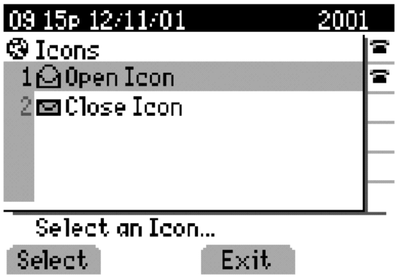
CiscoIPPhoneIconMenu

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 a read and unread icon in a mail viewer. You can use the icons can to convey the message state.

Icons in the `CiscoIPPhoneMenu` object have a maximum width of 16 pixels and a maximum height of 10 pixels.

Figure 2-6 shows an IconMenu on a Cisco Unified IP Phone.

Figure 2-6 IconMenu 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.

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>
```

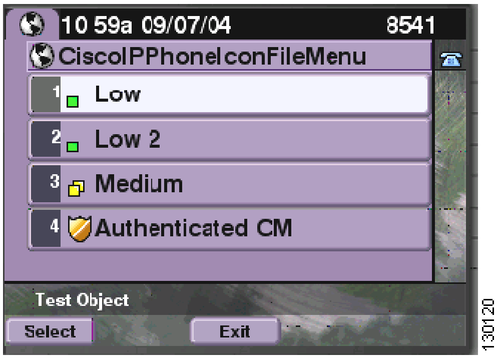
The XML tags in IconMenu use the tag definitions for CiscoIPPhoneImage and CiscoIPPhoneMenu. Although the semantics of the tags are identical, you can have only 32 MenuItem objects in a `CiscoIPPhoneIconMenu` object. See "CiscoIPPhoneMenu" and "CiscoIPPhoneImage" for detailed descriptions.

CiscoIPPhoneIconFileMenu

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 Figure 2-7).

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 2-7 CiscoIPPhoneIconFileMenu Object Display Sample



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>
```

CiscoIPPhoneStatus

The CiscoIPPhoneStatus object is also a displayable object, but differs from the preceding 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.

Because the Status object is only present on the Call plane, the object cannot be closed or cleared by the user (for example, by pressing Services). 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.

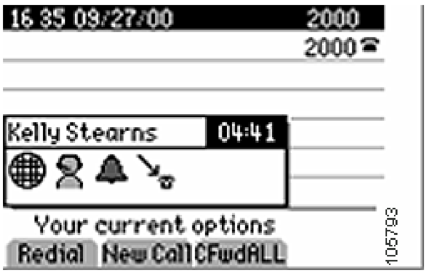
Note The CiscoIPPhoneStatus object can only be pushed (HTTP POST) to the phone; it cannot be pulled (HTTP GET).

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.

Figure 2-8 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 2-8 IconMenu on a CiscoIPPhoneStatus Sample



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>
```

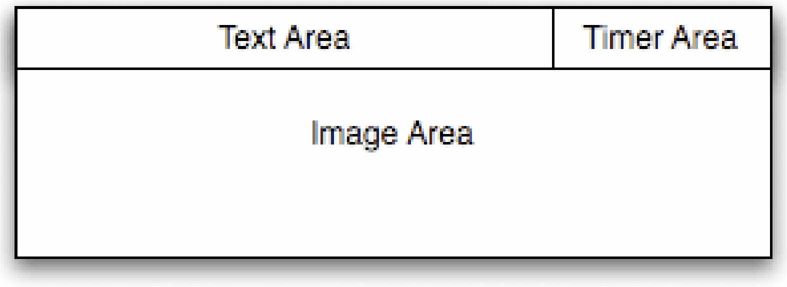
Dynamic Sizing of the Application Status Window

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 a large enough size 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 upon the reception of a CiscoIPPhoneStatus or CiscoIPPhoneStatusFile object with its associated PNG file.

The Application Status window contains three main areas: (see Figure 2-9):

- Text Area
- Timer Area
- Image Area

Figure 2-9 Elements of Application Status Window



Note Self-terminating XML elements, non-declared or missing elements, and elements with the default values are all considered non-configured 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 Table 2-3 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/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 2-3 Application Status Window Allowable Image Sizes

Phone Models	Maximum Image Area Width	Minimum Image Area Width	Maximum Image Area Height
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

See Table 2-4 for an overview of the text and timer area sizes by phone model.

Table 2-4 ***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

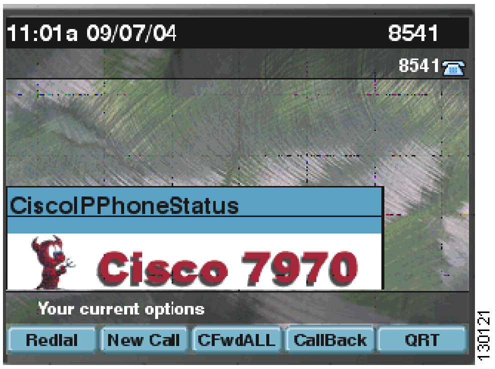
CiscoIPPhoneStatusFile

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. See "Dynamic Sizing of the Application Status Window" section on page 2-20 for details.

Figure 2-10 shows how an XML `CiscoIPPhoneStatusFile` object displays on a phone.

Figure 2-10 ***CiscoIPPhoneStatusFile Object Display Sample***



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.

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 will begin 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.

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 a per phone basis in the Cisco Unified Communications Manager Device configuration.

The following table lists the Priority levels and their behavior.

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 `<ExecuteItem>`, 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.

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` object items provide messages and information resulting from `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.

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

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

Cisco Unified IP Phones can use custom softkeys with any of the displayable CiscoIPPhone XML objects, excluding the `CiscoIPPhoneStatus` object which cannot control softkeys and the `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.

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>
```

Example

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, then you must explicitly define it 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. See Chapter 4, “Internal URI Features” for more information on invoking internal softkey features.

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 softkey will be “Next.” If not, the“Update” softkey is assigned.

Example

The following softkey definitions would provide the custom softkey, without losing the default “Select” behavior:

```
<SoftKeyItem>
   <Name>Select</Name>
  <URL>SoftKey:Select</URL>
  <Position>1</Position>
</SoftKeyItem>
<SoftKeyItem>
  <Name>Custom</Name>
  <URL>http://someserver/somepage</URL>
  <Position>4</Position>
</SoftKeyItem>
```

XML Considerations

The XML parser in Cisco Unified 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. Table 2-5 lists characters and their escape values.

.

Table 2-5 Escape Sequences for Special Characters

Character	Name	Escape Sequence
&	Ampersand	&amp;
"	Quote	&quot;
'	Apostrophe	&apos;
<	Left angle bracket	&lt;
>	Right angle bracket	&gt;

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

XML Encoding

Since 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 the example below.

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" ?>
```

For details on setting HTTP header encoding settings, see the "HTTP Encoding Header Setting" section on page 6-8.

Application Event Handlers

The Application Manager API (see "Application" section on page 4-22) includes an Application Management Event Handler which is supported by any displayable object, which are 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:

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 (see "Updated XML Parser and Schema Enforcement" section on page B-1 for details).

Attributes

The Application Event Handlers can be attached to a supported object by specifying the attributes:

Note An Application URI with Priority=0 is not allowed in the Application Event Handlers (see "Application" section on page 4-22).

Attribute	Description
appID	Identifies 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 can assign any unique identifier.
onAppFocusLost	Invoked when the application loses focus, if: • The application's context has lost focus, or • The application was navigated away from, either directly by the user, or programmatically by a refresh header or HTTP push. **Note** If a Notify URI is used as the event handler, a notification is sent with this default data: `<notifyApplicationEvent appId="appId" type="focusLost"/>`
onAppFocusGained	Invoked when the application gains focus, if: • The application is Active and the application's context has gained focus, or • The application was navigated to, either directly by the user, or by a refresh header or HTTP push. **Note** If a Notify URI is used as the event handler, a notification is sent with this default data: `<notifyApplicationEvent appId="appId" type="focusGained"/>`

Attribute	Description
onAppMinimized	Invoked when the application is minimized. 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. `<notifyApplicationEvent appId="appId" type="minimized"/>`
onAppClosed	Invoked whenever the application closes, if: • The application's context is closed which will, in turn, close all applications in its stack, or • 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). **Note** This event handler cannot contain HTTP or HTTPS URLs. **Note** If a Notify URI is used as the event handler, a notification is sent with this default data: `<notifyApplicationEvent appId="appId" type="closed"/>`

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>
```

Example

```
<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>
```