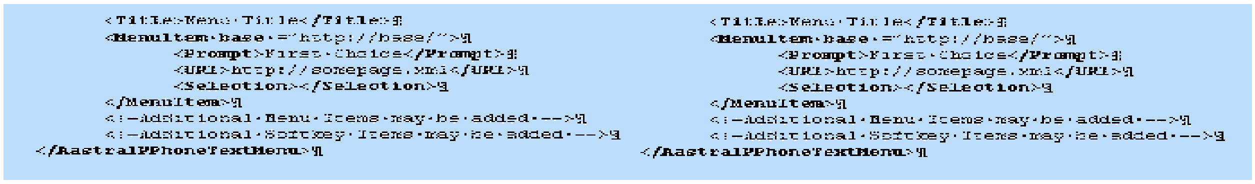
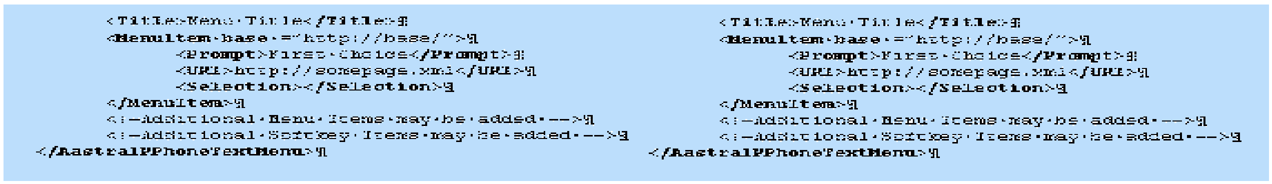


```
<Title>Menu Title</Title>
<MenuItem base ="http://base/">
        <Prompt>First Choice</Prompt>
        <URI>http://somepage.xml</URI>
        <Selection></Selection>
</MenuItem>
<!—Additional Menu Items may be added -->
<!—Additional Softkey Items may be added -->
</AastraIPPhoneTextMenu>
```

```
<Title>Menu Title</Title>
<MenuItem base ="http://base/">
        <Prompt>First Choice</Prompt>
        <URI>http://somepage.xml</URI>
        <Selection></Selection>
</MenuItem>
<!—Additional Menu Items may be added -->
<!—Additional Softkey Items may be added -->
</AastraIPPhoneTextMenu>
```

Development Guide
XML API for Aastra SIP Phones
Firmware 1.4.2

Doc. No.: PA-001004-00-03



```
<Title>Menu Title</Title>
<MenuItem base ="http://base/">
        <Prompt>First Choice</Prompt>
        <URI>http://somepage.xml</URI>
        <Selection></Selection>
</MenuItem>
<!—Additional Menu Items may be added -->
<!—Additional Softkey Items may be added -->
</AastraIPPhoneTextMenu>
```

```
<Title>Menu Title</Title>
<MenuItem base ="http://base/">
        <Prompt>First Choice</Prompt>
        <URI>http://somepage.xml</URI>
        <Selection></Selection>
</MenuItem>
<!—Additional Menu Items may be added -->
<!—Additional Softkey Items may be added -->
</AastraIPPhoneTextMenu>
```

TABLE OF CONTENTS

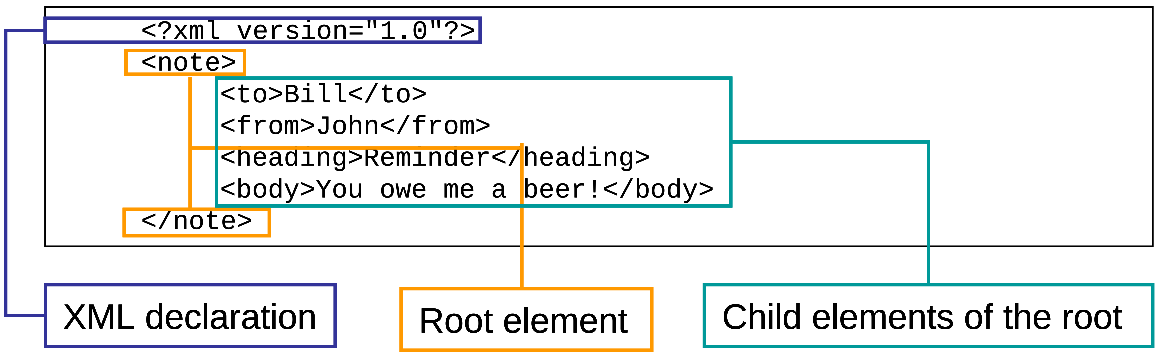
1. What is XML?

XML means **eXtensible Markup Language**. It is a markup language much like HTML. HTML was designed to display data and to focus on how data looks. XML was designed to describe data and to focus on what data is.

The following are characteristics of XML:

- XML tags are not predefined. You must define your own tags
- XML uses a Document Type Definition (DTD) or an XML Schema to describe the data
- XML with a DTD or XML Schema is designed to be self-descriptive
- XML is a W3C Standard Recommendation

A simple XML document



```
      <?xml version="1.0"?>
      <note>
            <to>Bill</to>
            <from>John</from>
            <heading>Reminder</heading>
            <body>You owe me a beer!</body>
      </note>
```

More information available at http://www.xml.com

2. XML and the Aastra IP Phones

2.1 Functionality

The XML browser in Aastra IP phones allows developers to create custom services that they can use via the phone's keypad and display. These services include things like weather and traffic reports, contact information, company info, stock quotes, or custom call scripts.

With firmware release 1.4.1, the Aastra IP phone XML API supports six proprietary objects that allow the creation of menu screens, message screens, input screens and directory screens.

- Text Menu object
- Text Screen object
- UserInput object
- Directory object
- PhoneStatus object
- PhoneExecute object

Some of these objects also support customizable softkeys that are declared as an independent object.

The following sections describe the process of creating XML objects for the Aastra IP phones.

Note: the XML browser is operational only if the phone Web Server is enabled.

2.2 How does it work?

Leveraging on the IP infrastructure, Aastra has decided to develop the browser capability on the phone using the HTTP transport protocol but as a direct support of HTML would not be suitable for the phone horsepower and limited display, the choice has been made to support only XML objects in the browser.

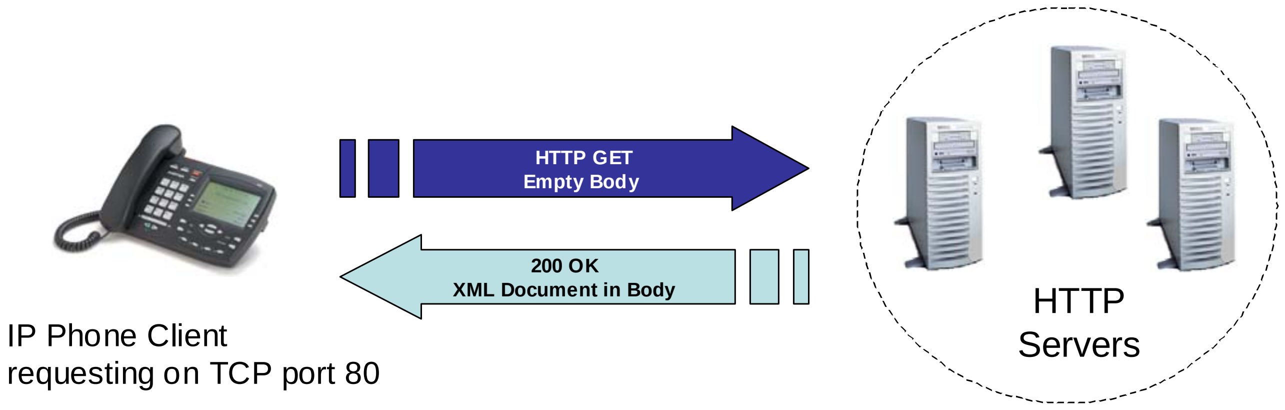
The Aastra IP phones support two types of applications:

- Phone-initiated
- Server-initiated

2.2.1 Phone initiated application

The phone issues an HTTP GET command to the Web server, waits for the answer, decodes and displays this answer as any Web browser such as Microsoft Internet Explorer or Firefox would do as a Web client.

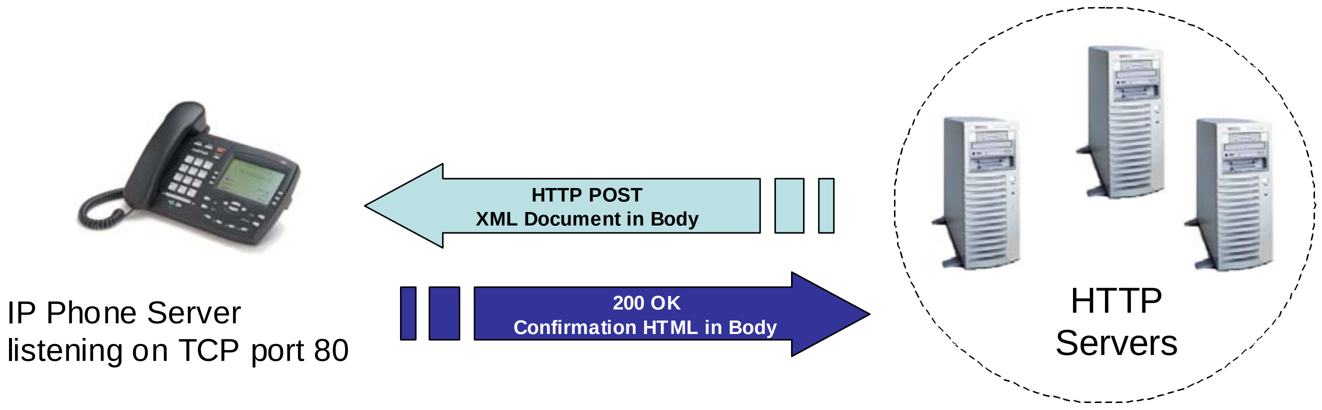
This can be done through a phone custom softkey and from the list of custom features (see chapter 7 for more details).



Aastra IP phone acting as a client

2.2.2 Server initiated application

The other type of application would be more used for alerting as an application is pushing an XML object to the phone. The phone is now acting as a limited Web Server.



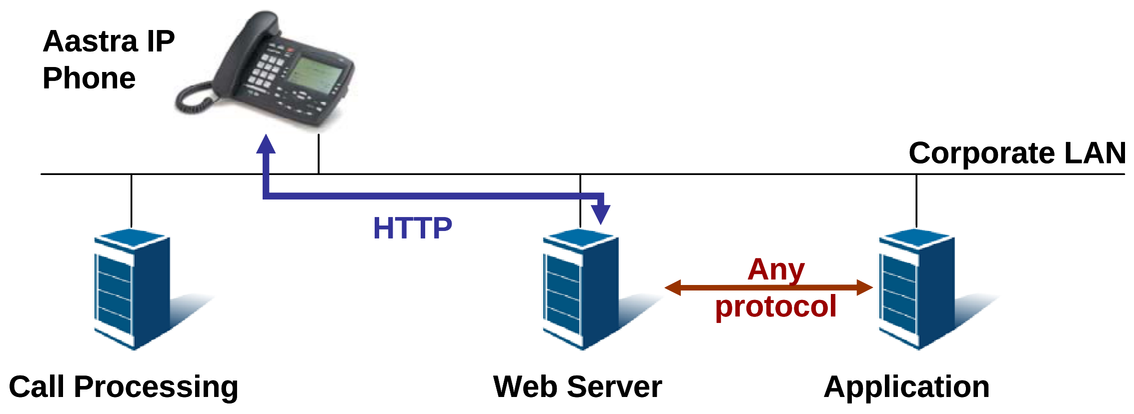
Aastra IP Phone acting as a server

2.3 Architecture

The XML applications are hosted by one or multiple Web servers which will serve as a proxy to either other applications or to Internet Web servers.

2.3.1 Internal application

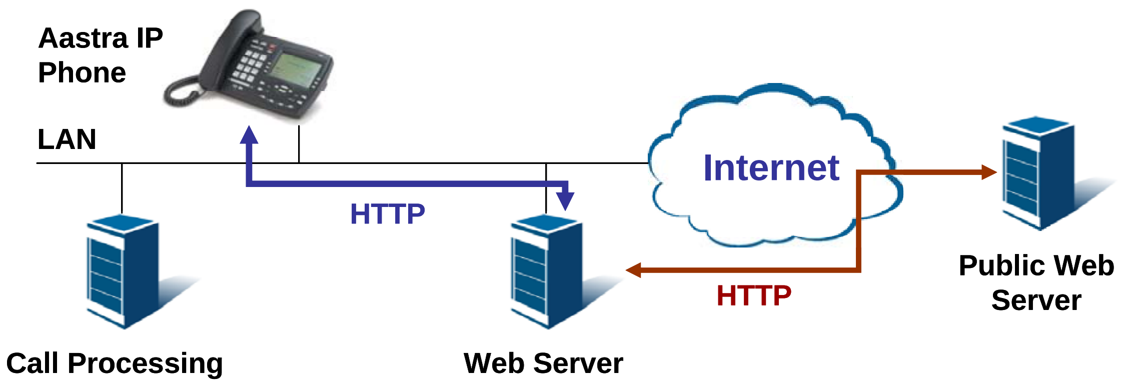
The following figure details the architecture to allow Aastra IP Phones to access an internal application. The application hosted by the Web server translates the phone requests to a protocol specific to the target application and formats the answer as an XML object to be displayed on the phone.



2.3.2 Internet application

The following figure details the architecture of an XML application that would retrieve data from the internet such as a real-time stock-quote service.

Note: for certain Web application that are not real time, the Internet content can be cached on the XML web server for a faster access.



2.4 Development environment

2.4.1 Web server

There is no constraint in the choice of the Web Server software to be used for Aastra XML applications; the choice is more based on the tools needed for the development such as the script language, the corporate policy or even the cost of the platform.

The most common Web server applications supported are:

- Apache (http://www.apache.org) for Microsoft and Linux Operating systems
- Microsoft IIS for Microsoft Operating systems
- Netscape iPlanet

2.4.2 Scripts/Languages

As for the Web Server, there is no specific constraint on the tools to develop the applications. All the languages supported to develop a Web application are supported to develop XML applications. The most common are:

- Compiled languages: C, C++
- Scripting languages: VBscript, Perl, Python, PHP.

2.5 XML format

The text in the Aastra XML objects must be compliant with XML recommendations and special characters must be escape encoded:

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

3 Aastra IP Phone XML Objects

This chapter details all the XML objects supported by Aastra SIP Phones.

Note: The size of an XML object can not exceed 5000 bytes.

3.1 TextMenu Object

The TextMenu object allows developers to create a numerical list of menu items on the IP phones. Go-to line support, arrow indicator, and scroll key support are built into these objects, along with the "**Select"** and "**Done"** softkeys. The TextMenu object allows users to navigate the application, by linking HTTP requests to menu items.

Default Softkeys

Position	Label	Description	URIs
1	Select	Executes the content of the URI field assigned to the selected MenuItem;	SoftKey:Select
6	Done	Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

XML Description

```
<AastraIPPhoneTextMenu
    defaultIndex = "some integer"
    destroyOnExit = "yes/no"
    style = "numbered/none/radio"
    Beep = "yes/no"
    >
      <Title>Menu Title</Title>
      <MenuItem base ="http://base/">
            <Prompt>First Choice</Prompt>
            <URI>http://somepage.xml</URI>
            <Selection></Selection>
      </MenuItem>
      <!—Additional Menu Items may be added -->
      <!—Additional Softkey Items may be added -->
</AastraIPPhoneTextMenu>
```

XML Document Objects

Document Object	Position	Type	Comments
`AastraIPPhoneTextMenu`	Root tag	Mandatory	Root object
`defaultIndex`	Root tag	Optional	Position of the cursor when the XML object is open. If not specified the arrow is positioned on the first menu item.
`style`	Root tag	Optional	"numbered/none/radio" indicates the style of the TextMenu. Default is "numbered".
`destroyOnExit`	Root tag	Optional	"yes/no" indicates if the object is kept

Document Object	Position	Type	Comments
			or not in the phone browser after exit. If not specified, the object is kept in the browser.
`Beep`	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
`Title`	Body	Mandatory	Text to be used as title for the object
`MenuItem`	Body	Mandatory	Choice Item (up to 15 instances)
`Base`	MenuItem tag	Optional	The value of this attribute is pre-pended to the value in the URI tags.
`Prompt`	MenuItem body	Mandatory	Label of the item
`URI`	MenuItem body	Mandatory	URI to be used if the user press "Select" on this item
`Selection`	MenuItem body	Optional	This tag can be used in conjunction with custom softkeys. See Section 3.7 for details
`SoftKey`	Body	Optional	See section 3.7 for details

<u>TextMenu styles</u>

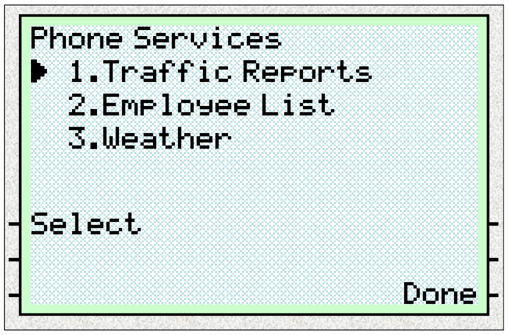


Figure 1: "numbered" style TextMenu

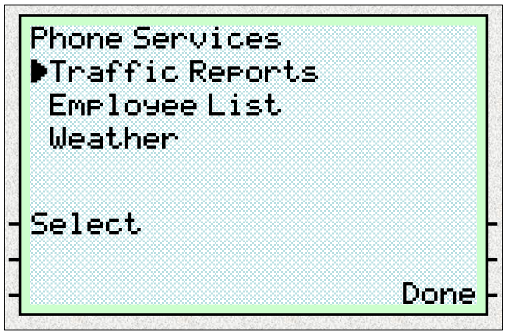


Figure 2: "none" style TextMenu

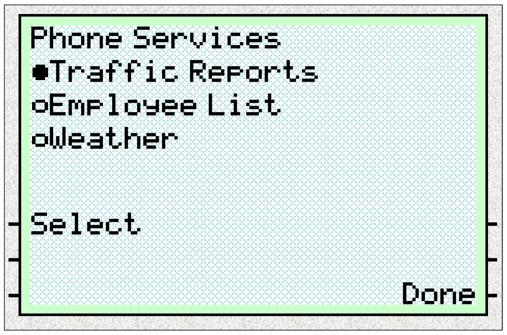


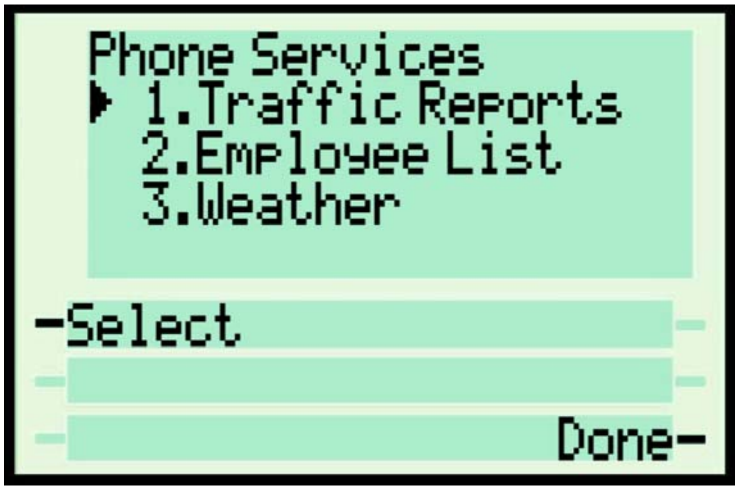
Figure 3: "radio" style TextMenu

<u>XML Example</u>

```
<AastraIPPhoneTextMenu>
```

```
       <Title>Phone Services</Title>
       <MenuItem base = "http://10.50.10.53/">
             <Prompt>Traffic Reports</Prompt>
             <URI> rss_to_xml.pl</URI>
       </MenuItem>
       <MenuItem>
             <Prompt>Employee List</Prompt>
             <URI>employees.xml</URI>
       </MenuItem>
       <MenuItem base ="">
             <Prompt>Weather</Prompt>
             <URI>http://10.50.10.52/weather.pl</URI>
       </MenuItem>
</AastraIPPhoneTextMenu>
```

Resulting Screen



```
Phone Services
▶ 1.Traffic Reports
  2.Employee List
  3.Weather
-Select
                Done-
```

3.2 TextScreen Object

The TextScreen object can be used to display text. The screen word-wraps appropriately and can scroll to display a message longer then three lines.

Default Softkeys

Position	Label	Description	URIs
6	Done	Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

XML Description

```
<AastraIPPhoneTextScreen
    destroyOnExit = "yes/no"
    Beep = "yes/no"
    >
      <Title>Screen Title</Title>
      <Text>The screen text goes here</Text>
<!—Additional Softkey Items may be added -->
</AastraIPPhoneTextScreen>
```

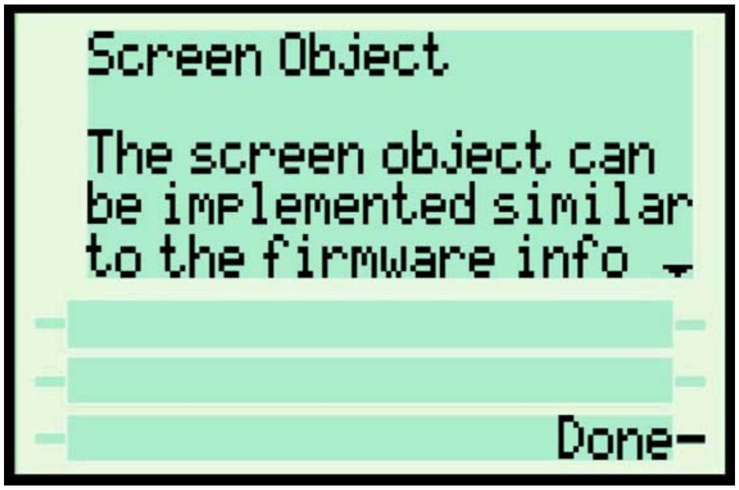
XML Document Objects

Document Object	Position	Type	Comments
`AastraIPPhoneTextScreen`	Root tag	Mandatory	Root object
`destroyOnExit`	Root tag	Optional	"yes/no" indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
`Beep`	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
`Title`	Body	Mandatory	Label to be used as title for the object
`Text`	Body	Mandatory	Text to be displayed.
`SoftKey`	Body	Optional	See section 3.5 for details

XML Example

```
<AastraIPPhoneTextScreen>
      <Title>Screen Object</Title>
      <Text>The screen object can be implemented similar to the
firmware info screen. Note that white space is preserved in XML so
the display should word-wrap appropriately. Only three lines can
display at a time.</Text>
</AastraIPPhoneTextScreen>
```

Resulting Screen


Screen Object

The screen object can
be implemented similar
to the firmware info

3.3 InputScreen Object

The InputScreen object allows developers to create a screen capable of gathering user input. The Aastra IP phones support three input types:

- IP Addresses,

- Numbers (integers),
- Strings.

Each parameter has a URL tag that is used to send information back to the HTTP server. The label in the parameter tag is appended to the address in the URL tag and sent via HTTP GET.

XML Description

```
<AastraIPPhoneInputScreen
     type = ”IP/string/number”
     password = ”yes/no”
     editable = “yes/no”
     destroyOnExit = “yes/no”
     Beep = “yes/no”
>
      <Title>Title string, usually same as menu title</Title>
      <Prompt>Enter IP address or host name</Prompt>
      <URL>Target receiving the input</URL>
      <Parameter>name of the parameter added to URL</Parameter>
      <Default>Default Value</Default>
<!—Additional Softkey Items may be added -->
</AastraIPPhoneInputScreen>
```

XML Document Objects

Document Object	Position	Type	Comments
`AastraIPPhoneInputScreen`	Root tag	Mandatory	Root object
`type`	Root tag	Mandatory	Specifies the type of input, possible values are “IP”, “string” and “number”
`password`	Root tag	Optional	Specifies if the input is masked by “*” characters. Default value is “no”
`editable`	Root tag	Optional	Specifies if the user is allowed to modify the input. Default value is “yes”
`destroyOnExit`	Root tag	Optional	“yes/no” indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
`Beep`	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
`Title`	Body	Mandatory	Text to be used as title for the object
`Prompt`	Body	Mandatory	Text to be displayed as a guidance for the user input.
`URL`	Body	Mandatory	URI called when user completes his input.

Document Object	Position	Type	Comments
`Parameter`	Body	Mandatory	Name of the parameter to be added to the URL after input is complete.
`Default`	Body	Mandatory	Default value to be displayed in the input field.
`SoftKey`	Body	Optional	See section 3.7 for details

3.3.1 Input Type: IP

When the type is set to IP, the user input is restricted to integers only. The phone will validate the user input; if an invalid IP address is entered, nothing will be sent to the server and the user will receive an error message.

Default Softkeys

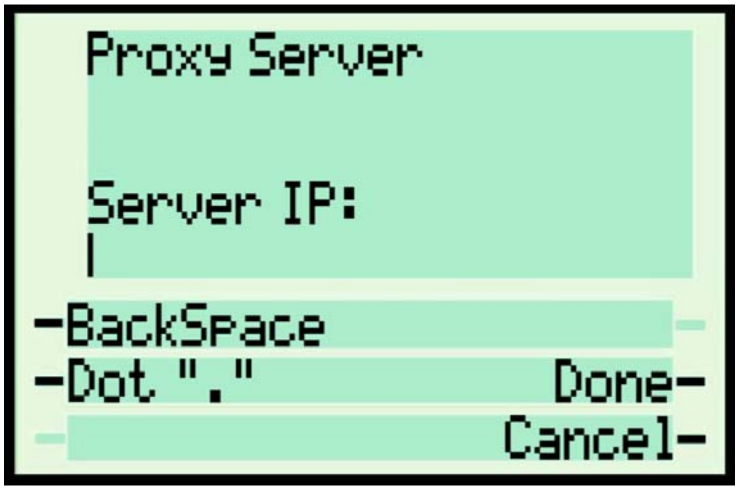
Position	Label	Description	URIs
1	Backspace	Deletes the character before the cursor in the input field.	SoftKey:BackSpace
2	Dot "."	Inserts a "." In the user input at the cursor position	SoftKey:Dot
5	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
6	Cancel	Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type = "IP">
      <Title>Proxy Server</Title>
      <Prompt>Server IP:</Prompt>
      <URL>http://10.50.10.53/script.pl</URL>
      <Parameter>proxy</Parameter>
      <Default></Default>
<AastraIPPhoneInputScreen>
```

Note: In this example, when the user press "Done" on the phone after entering "192.168.0.100", the phone will call the following URL "http://10.50.10.53/script.pl?proxy=192.168.0.100".

Resulting Screen



3.3.2 Input Type: Number

Like an IP screen, a number input screen restricts the user to numbers only. However, no type of validation is performed on the user input.

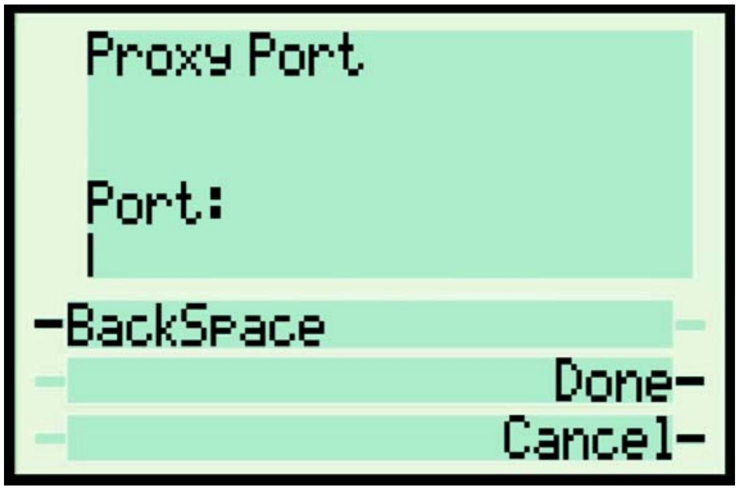
Default Softkeys

Position	Label	Description	URIs
1	Backspace	Deletes the character before the cursor in the input field.	SoftKey:BackSpace
5	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
6	Cancel	Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type = ”number”>
      <Title>Proxy Port</Title>
      <Prompt>Port:</Prompt>
      <URL>http://10.50.10.53/script.pl</URL>
      <Parameter>port</Parameter>
      <Default>5060</Default>
<AastraIPPhoneInputScreen>
```

Resulting Screen


Proxy Port

Port:

Note: In this example, when the user presses "Done" on the phone after entering "5060", the phone will call the following URL "`http://10.50.10.53/script.pl?port=5060`".

3.3.3 Input Type: String

When the input type is set to string, the user can enter uppercase letters, lowercase letters, symbols, and numbers. The input mode can be switched via the Mode Key. The user can scroll through the available input symbols by pressing the 1 key repeatedly.

Default Softkeys

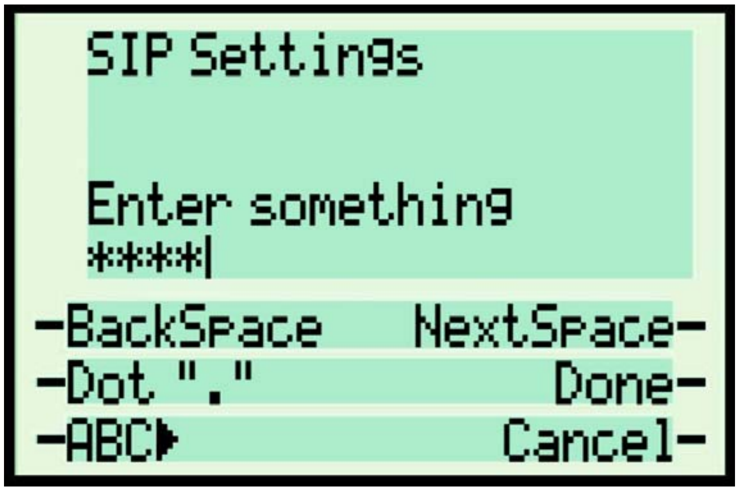
Position	Label	Description	URIs
1	Backspace	Deletes the character before the cursor in the input field.	SoftKey:BackSpace
2	Dot "."	Inserts a "." in the user input at the cursor position	SoftKey:Dot
3	ABC>	Toggle between input modes, "ABC", "123", "abc".	SoftKey:ChangeMode
4	NextSpace	Inserts a space in the user input at the cursor position	SoftKey:NextSpace
5	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
6	Cancel	Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen
  type = ”string”
  password = “yes”>
      <Title>SIP Settings</Title>
```

```
        <Prompt>Enter something</Prompt>
        <URL>http://10.50.10.53/script.pl</URL>
        <Parameter>passwd</Parameter>
        <Default></Default>
<AastraIPPhoneInputScreen>
```

Resulting Screen


SIP Settings

Enter something
****|

Note: In this example, when the user press "Done" on the phone after entering "12345", the phone will call the following URL "http://10.50.10.53/script.pl?passwd=12345".

3.4 Directory Object (480i/480iCT only)

The Directory object allows the user to browse an online directory in real time. It displays an automatically numbered list of contacts. By selecting a contact with the cursor, the contact can be dialed directly by pressing the "Dial" softkey, picking up the handset, or pressing a line button. The Directory object has the optional softkeys of "Previous" and "Next" which can be linked to other XML objects.

Note: The Directory object has been kept in R1.3.1 for compatibility reasons with R1.3.0 but thanks to the customizable softkeys, it can be replaced by a TextMenu object with a "Dial" custom softkey.

Default Softkeys

Position	Label	Description	URIs
1	Dial	Launches a call on the first available line using the item URI as a phone number.	SoftKey:Dial
2	Previous	Optional, if pressed the phone will call the "Previous" URI as defined in the object.	Not defined
5	Next	Optional, if pressed the phone will call the "Next" URI as defined in the object.	Not defined
6	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Exit

XML Description

```
<AastraIPPhoneDirectory
  next=”uri”
  previous=”uri”
  destroyOnExit = ”yes/no”
  Beep = “yes/no”
>
      <Title>Directory Title</Title>
      <MenuItem>
            <Prompt>Contact Name</Prompt>
            <URI>number</URI>
      </MenuItem>
      <!—Additional Menu Items may be added -->
      <!—Additional Softkey Items may be added -->
</AastraIPPhoneDirectory>
```

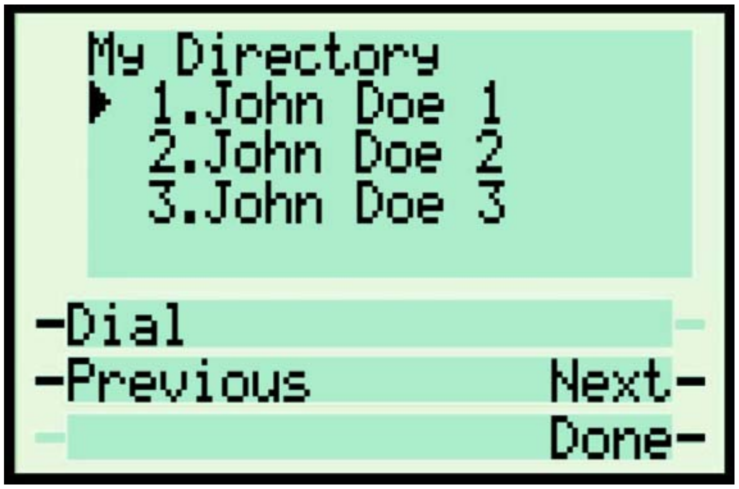
XML Document Objects

Document Object	Position	Type	Comments
`AastraIPPhoneDirectory`	Root tag	Mandatory	Root object
`next`	Root tag	Optional	Specifies the URI to be call for the “Next” softkey
`previous`	Root tag	Optional	Specifies the URI to be call for the “Previous” softkey
`previous`	Root tag	Optional	Specifies the URI to be call for the “Previous” softkey
`destroyOnExit`	Root tag	Optional	“yes/no” indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
`Beep`	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
`Title`	Body	Mandatory	Text to be used as title for the object
`MenuItem`		Mandatory	Choice Item (up to 15 instances)
`Prompt`		Mandatory	Label of the item
`URI`		Mandatory	Phone number or IP address to be dialed
`SoftKey`	Body	Optional	See section 3.5 for details

XML Example

```
<AastraIPPhoneDirectory
  next=”http://myserver.com/more.php”
  previous=http://myserver.com/back.xml
>
      <Title>My Directory</Title>
      <MenuItem>
            <Prompt>480i – John Doe 1</Prompt>
            <URI> 10.50.10.49</URI>
      </MenuItem>
      <MenuItem>
            <Prompt>9133i – John Doe 2</Prompt>
            <URI>4326</URI>
      </MenuItem>
      <MenuItem>
            <Prompt>480iCT – John Doe 3</Prompt>
            <URI>9982691234</URI>
      </MenuItem>
</AastraIPPhoneDirectory
```

Resulting Screen


```
My Directory
▶ 1.John Doe 1
  2.John Doe 2
  3.John Doe 3
-Dial
-Previous        Next-
                 Done-
```

3.5 PhoneStatus Object

The `AastraIPPhoneStatus` object provides the ability to display a status message on a single designated line on the phone's idle screen when XML information is pushed from the servers.

The 480i/480i CT phones display messages on the second line in the phone window (where "No Service" would display if there was no service. If there is no service on the phone, the "No Service" message overrides the XML object message). The 9112i/9133i phones display messages on the first line (overriding the `DisplayName`). Long messages that are wider then the phone screen get truncated.

If the phone receives multiple messages, the first message received displays first and the remaining messages scroll consecutively one at a time.

The AastraIPPhoneStatus object supports 2 modes for the messages to be displayed:

- Normal mode: messages remain displayed until they are removed (by the server) or the phone reboots.
- Alert mode: Messages are displayed on top of the existing messages only for a limited time (3 seconds by default)

The `AastraIPPhoneStatus` object feature is always enabled.

Note: You can set the amount of time, in seconds, that a message displays to the phone before scrolling to the next message (See *Scroll delay* below).

Default Softkeys

Not Applicable, this object has no predefined softkey as it impacts the idle screen of the Aastra SIP Phone

XML Description

```
<AastraIPPhoneStatus Beep= "yes/no">
      <Session>Session ID</Session>
      <Message
       index="index"
       type="alert"
       Timeout="timeout"
       >Message</Message>
      <!—Additional Message Items may be added -->
</AastraIPPhoneStatus>
```

Note: The *Session ID* must be unique to the application sending the XML object to the phone. It is up to the application to generate that session ID, which does not have to be limited to just numbers. It could be a combination of letters and numbers. There could only be one Session tag per PhoneStatusMsg object. If the `Session` tag is not provided, the phone assumes a default value (0) for it; this can be used if you don't have multiple applications displaying messages on the idle screen.

Note: The `type="alert"` tag indicates the alert mode if not specified the message is displayed in the normal mode.

Scroll Delay

The Scroll delay can be configured via the configuration files and the Aastra Web UI using the following parameters:

- `xml status scroll delay` (via configuration files)
- Status Scroll Delay (in seconds) (via the Aastra Web UI see chapter 7.6)

XML Document Objects

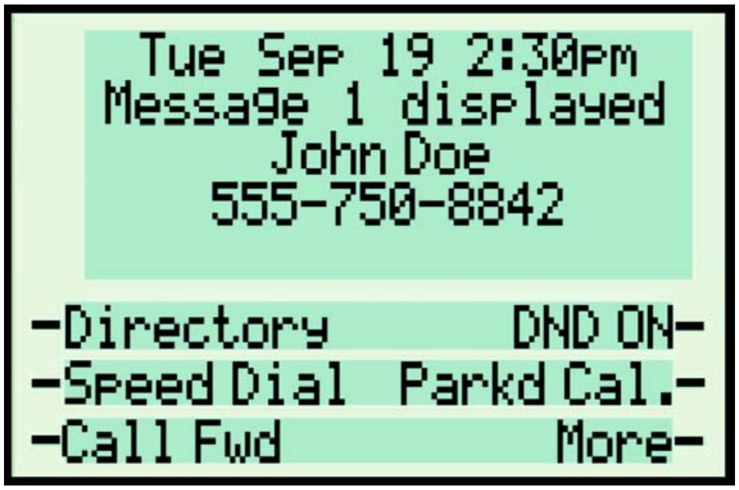
Document Object	**Position**	**Type**	**Comments**
`AastraIPPhoneStatus`	Root tag	Mandatory	Root object
`Beep`	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
`Session`	Body	Optional	Session ID used to identify the application displaying message. It allows message change and message reset.
`Message`	Body	Mandatory	Message to be displayed or empty to reset the message.
`index`	Message tag	Mandatory	Index of the message for the session, the index starts at 0.
`type`	Message tag	Optional	Type of message, only "alert" is supported. If not specified the message stays on the screen until it is reset by an empty message or after a phone reboot.

Document Object	Position	Type	Comments
`Timeout`	Message tag	Optional	Timeout of the "alert" message, overrides the 3s default value.
`SoftKey`	Body	Optional	See section 3.7 for details

XML Example

```
<AastraIPPhoneStatus Beep="yes">
      <Session>abc12345</Session>
      <Message index="0">Message 1 displayed</Message>
      <Message index="1" type="alert" Timeout="5">Alert
displayed</Message>
</AastraIPPhoneStatus>
```

Resulting Screen


Tue Sep 19 2:30pm
Message 1 displayed
John Doe
555-750-8842

Note: The PhoneStatusMsg object can also be used to remove status messages from the display using the same SessionID and the same index. This can be accomplished by setting an empty tag for the Message tag. For example, here is the XML object to remove the previous messages (as the second message was in alert mode it does mot remain on the display so no need to remove it).

```
<AastraIPPhoneStatus>
<Session>abc12345</Session>
<Message index="0"/>
<AastraIPPhoneStatus/>
```

3.6 PhoneExecute Object

The `PhoneExecute` object allows an external application to ask the phone to execute a sequence of local actions using a URI. The actions can be:

- Any HTTP URL
- Phone Reset (URI="Command: Reset")
- Do nothing (URI="")

More actions will be implemented in future firmware versions.

Default Softkeys

Not Applicable, this object has no predefined softkey.

XML Description

```
<AastraIPPhoneExecute Beep = "yes/no">
      <ExecuteItem URI="URI">
      <!—Additional ExecuteItems may be added -->
</AastraIPPhoneExecute>
```

XML Document Objects

Document Object	Position	Type	Comments
`AastraIPPhoneExecute`	Root tag	Mandatory	Root object
`Beep`	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
`ExecuteItem`	Body	Optional	Tag for the action to be executed
`URI`	ExecuteItem tag	Optional	URI describing the action to be executed.

XML Example

```
<AastraIPPhoneExecute>
      <ExecuteItem URI="http://myserver.com/myscript.php">
      <ExecuteItem URI="Command: Reset">
</AastraIPPhoneExecute>
```

This example will make the phone execute 2 actions:

- Do an HTTP GET to myserver.com/myscript.php
- Reset the phone

Note: the "do nothing" can be used when an application needs to display nothing as an answer to a HTTP GET.

3.7 Customizable Softkeys (480i/480i CT only)

The Softkey object can be used to override the default softkeys in each of the above objects. It allows developers to link arbitrary URIs to keys in the XML screens and invoke softkey behavior native to each XML screen type.

Note: when you use custom softkeys, the default softkeys of the XML object are not displayed anymore. This means they have to be recreated as softkeys.

XML Description

```
<SoftKey index = "1-6">
      <Label>Text</Label>
      <URI>http://someserver/somepage.pl                        OR
SoftKey:someaction</URI>
</Softkey>
```

You can define up to six softkeys before the closing tag of any object on the 480i and 480i CT. The following example illustrates the use of the softkey XML element used with the Text Menu object, and the resulting screen output.

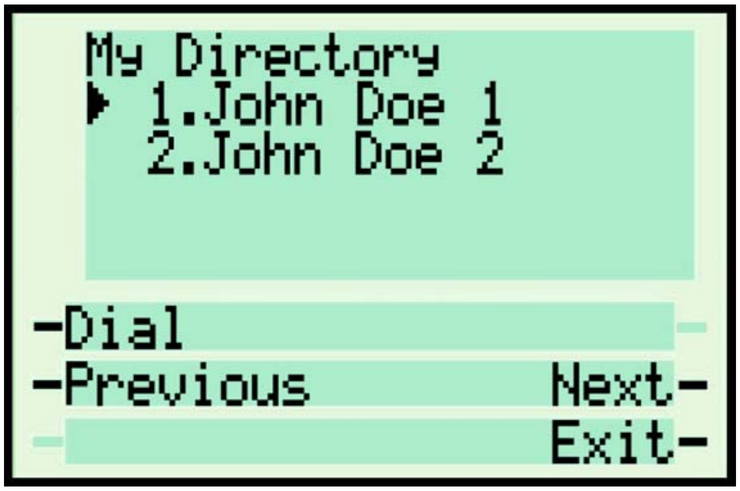
XML Document Objects

Document Object	Position	Type	Comments
`SoftKey`	Body	Mandatory	Root object (up to 6)
`index`	SoftKey Root tag	Mandatory	Indicates the softkey number
`Label`	Body	Mandatory	Label of the softkey
`URI`	Body	Mandatory	URI called if the softkey is pressed

XML Example

```
<AastraIPPhoneTextMenu>
      <Title> New Directory </Title>
      <MenuItem>
            <Prompt>John Doe 1</Prompt>
            <URI>10.50.10.49</URI>
      </MenuItem>
      <MenuItem>
            <Prompt>John Doe 2</Prompt>
            <URI>200</URI>
      </MenuItem>
      <SoftKey index = “1”>
            <Label>Dial</Label>
            <URI>SoftKey:Dial</URI>
      </Softkey>
      <SoftKey index = “2”>
            <Label>Previous</Label>
            <URI>http://someserver/somepage.xml</URI>
      </Softkey>
      <SoftKey index = “5”>
            <Label>Next</Label>
            <URI>http://someserver/somepage.xml</URI>
      </Softkey>
      <SoftKey index = “6”>
            <Label>Exit</Label>
            <URI>SoftKey:Exit</URI>
      </Softkey>
</AastraIPPhoneTextMenu>
```

Resulting Screen


My Directory
▶ 1.John Doe 1
2.John Doe 2
-Dial
-Previous Next-
Exit-

Available object commands

The following softkey functionality is available to the developer for the purpose of reordering or preserving the default functionality of the XML screens. The "Dial" function is available to screens that allow input. The dial string for the "Dial" function is taken from the menu items URI on the Menu Screen, and from the editor field input on the Input Screen.

SoftKey	TextScreen	TextMenu	InputScreen	Directory
Select		X		
Exit	X	X	X	X
Dial		X	X	X
Submit			X	
BackSpace			X	
NextSpace			X	
Dot			X	
ChangeMode			X	

3.8 Beep Option

You can enable or disable a BEEP option in all the Aastra XML objects via the Beep attribute in the root tag. When the phone receives an XML object, the BEEP notifies the user that an object is being displayed.

This attribute is optional. If the BEEP attribute is set to "yes" (i.e. Beep="yes") then it is an indication to the phone to sound a beep when it receives the object. If the Beep attribute is set to "no" (i.e. Beep="no") or not present, then the default behavior is no beep is heard when the object arrives to the phone.

The BEEP option can also be enabled or disabled via the configuration files and the Aastra Web UI using the following parameters:

- `xml beep notification` (via configuration files)

- XML Beep Support (via the Aastra Web UI see chapter 7.5)

The value set in the configuration files and Aastra Web UI override the attribute you specify in the `AastraIPPhoneStatus` object. For example, if the `AastraIPPhoneStatus` object has the attribute of Beep="yes", and you uncheck (disable) the "XML Beep Support" in the Aastra Web UI, the phone does not beep when it receives an `AastraIPPhoneStatus` object.

3.9 TextMenu user selection

It is also possible to send information related to a user's choice in a Text Menu.

"Select" and "Dial" system keys use the URI passed in the `MenuItem` attribute, a custom key might need to know the user selection in the TextMenu, and this is the purpose of the **`Selection`** attribute.

When a user accesses an arbitrary URI via a custom softkey, the Text Menu will send along a bit of information regarding the user's currently selected option using an optional XML attribute **`Selection`**.

XML Example

```
<AastraIPPhoneTextMenu destroyOnExit = "yes">
      <Title>Parameter Tester</Title>
      <MenuItem>
            <Prompt>First Selection</Prompt>
            <URI>http://someserver/somepage.xml</URI>
            <Selection>dataToAppend</Selection>
      </MenuItem>
      <SoftKey index = "1">
            <Label>Custom Key</Label>
            <URI>http://someotherserver/someotherpage.xml</URI>
      </SoftKey>
</AastraIPPhoneTextMenu>
```

When the user selects item 1 and presses softkey 1, the URI requested is
http://someotherserver/someotherpage.xml?selection=dataToAppend

Note: If a "?" already exists in the URI, then a "&" is used to separate the parameters. Note the parameter name "selection" is automatic. If the `Selection` attribute is omitted, then nothing extra is appended to the URI.

Note: the `Selection` attribute can be more than one parameter, for instance, the following `Selection` attribute is valid `<Selection>200&action=set</Selection>` (don't forget to escape encode the attribute for the "&").

3.10 "Select" and "Dial" in the same TextMenu object

The only way to have a "Select" and "Dial" keys in the same text menu, the only option is to use "Dial" as a system command and mimic "Select" with a custom key.

```
<AastraIPPhoneTextMenu destroyOnExit = "yes">
      <Title>Dial and Select</Title>
      <MenuItem>
            <Prompt>First Selection</Prompt>
            <URI>200</URI>
```

```
                <Selection>200</Selection>
        </MenuItem>
        <MenuItem>
                <Prompt>Second Selection</Prompt>
                <URI>201</URI>
                <Selection>201</Selection>
        </MenuItem>
        <SoftKey index = “1”>
                <Label>Dial</Label>
                <URI>SoftKey:Dial</URI>
        </SoftKey>
        <SoftKey index = “2”>
                <Label>Select</Label>
                <URI>http://myserver.com/script.php</URI>
        </SoftKey>
</AastraIPPhoneTextMenu>
```

When the user selects item 1 and presses

- Softkey 1, the phone dials 200
- Softkey 2, the requested URI requested is http://myserver.com/script.php?selection=200

3.11 Refresh of an XML page

All XML screen objects have the ability to be refreshed. This is accomplished by adding a Refresh setting to the HTTP headers. This setting comprises two parameters:

- a time in seconds
- a URL.

The Refresh setting is set by the XML application and it is up to the application to decide which objects they want to refresh. So, it is an optional setting.

If the setting appears, then both parameters must be set for the Refresh to work.

The format of the HTTP header is:

```
        Refresh: timeout; url=page to load
```

Example

```
        Refresh: 3; url=http://10.50.10.140/update.xml
```

All of the parameters are mandatory.

With php you will have to use the `header` command in your script.

```
        header("Refresh: 1; url= http://10.50.10.140/update.xml”);
```

3.12 HTTP header format for a phone HTTP GET

When the Aastra SIP performs a GET to an http server, it sends a HTTP header for the User Agent that includes:

- The type of phone (9112i, 9133i, 480i or 480iCT)
- The MAC address of the phone
- The firmware version of the phone

Usually the HTTP server is able to retrieve the User Agent parameters. The Aastra SIP phones send the following string for the User Agent.

```
Phone_Model[space]MAC:-XX-XX-XX-XX-XX-XX[space]V:Version
```

This allows the application to adapt to the type of phone it is dealing with but also allows to have a single application to provide XML (for the phones) and HTML (for a web browser).

The following source code is a php example that can be used to decode the HTTP header.

```
function Aastra_decode_HTTP_header()
{
$user_agent=$_SERVER["HTTP_USER_AGENT"];
if(stristr($user_agent,"Aastra"))
      {
      $value=preg_split("/ MAC:/",$user_agent);
      $end=preg_split("/ /",$value[1]);
      $value[1]=preg_replace("/\-/","",$end[0]);
      $value[2]=preg_replace("/V:/","",$end[1]);
      }
else
      {
      $value[0]="MSIE";
      $value[1]="NA";
      $value[2]="NA";
      }
return($value);
}
```

This function returns an array with

- Array[0] is the phone model ("Aastra9112i", "Aastra9133i", "Aastra480i" and "Aastra480i Cordless")
- Array[1] is the MAC address
- Array[2] is the firmware version

Example of execution

- Array[0]=> Aastra480i
- Array[1]=> 00085D031C81
- Array[2]=> 1.4.1.1224-SIP

3.13 Some development guidelines

There are some simple rules that you should follow when you develop XML applications for the Aastra SIP phones.

- Don't forget the "Exit" key when you use custom softkeys
- Place custom softkeys as they are for the standard objects, also it is better to use the same labels. The Exit key should preferably be placed in position 6.
- Setting **`destroyOnExit`** attribute to "yes" is preferable when you write complex applications as it is the only way to properly control the pages that are stacked in the phone.
- Avoid using the Directory Object; TextMenu object with custom keys is much more flexible.
- If you want to access data from the internet, it is preferable to use a RSS feed than Web scraping as Web sites frequently change their interface.
- For XML applications using data from the Internet, it they are not real time you might consider to "cache" the data on the server and update them on a regular basis (a weekly update is necessary for Movie schedules but a 10 minutes update might be necessary for a weather application).
- As the XML objects are limited to 5000 bytes, you might sometime hit this limit for a complex `TextMenu` with a lot of custom keys. Don't forget to use the base attribute in the `MenuItem` object as it is a good way to reduce the size of the object.

4 URL Format and Variables

4.1 URL format

To access to an XML application the phone performs a HTPP GET on a URL. The supported syntax is:

http://host[:port]/dir/file

where:

- host is the hostname or IP address of the Web server supporting the XML application
- port is the port number the phones is using for his HTTP request.

Note: If the port is not specified, the phone uses port 80.

4.2 URL Variables

The phones support system variables in the HTTP URL to pass dynamic data to the XML applications.

The variables are in the form $$VARIABLENAME$$ and the following variables are supported:

• $$SIPUSERNAME$$	line user name
• $$DISPLAYNAME$$	the display name of the focused line
• $$SIPAUTHNAME$$	the SIP auth name of the focused line
• $$PROXYURL$$	the SIP proxy of the focused line
• $$INCOMINGNAME$$	returns the Caller-ID of the incoming call
• $$REMOTENUMBER$$	returns the number of the remote party (incoming or outgoing)

At the time of the HTTP call the variables are replaced with the value of the appropriate variable.

The variables are available in all the HTTP URL for programmable/soft keys as well as for the action URI. They can also be used in the XML objects themselves (TextMenu, InputScreen…).

Example

For example, if the administrator specifies an XML softkey with the value:

```
http://10.50.10.140/script.pl?name=$$SIPUSERNAME$$
```

This softkey executes a GET on:

```
http://10.50.10.140/script.pl?name=42512
```

assuming that the SIP username of the specific line is 42512.

The variables can be used, for instance, in a context where the phone has multiple SIP registrations and the XML application need to know which SIP registration the phone is currently using.

Note: the value of a variable depends on the status of the phone, if the variable has no meaning in the current status; the phone sends an empty string. For example if the URL is "http://myserver.com/script.pl?number=$$REMOTENUMBER$$&key=5", the phone will call "http://myserver.com/script.pl?number="&key=5" if the phone is idle.

The following table details when the variables are meaningful with an action URI.

	startup	registered	onhook	offhook	incoming	outgoing
$$SIPUSERNAME$$	X	X	X	X	X	X
$$DISPLAYNAME$$	X	X	X	X	X	X
$$SIPAUTHNAME$$	X	X	X	X	X	X
$$PROXYURL$$	X	X	X	X	X	X
$$INCOMINGNAME$$					X	
$$REMOTENUMBER$$					X	X

The following table details when the variables are meaningful for a programmable/soft key depending on the phone state.

	idle	connected	incoming	outgoing
$$SIPUSERNAME$$	X	X	X	X
$$DISPLAYNAME$$	X	X	X	X
$$SIPAUTHNAME$$	X	X	X	X
$$PROXYURL$$	X	X	X	X
$$INCOMINGNAME$$		X*	X	
$$REMOTENUMBER$$		X	X	X

* only if you have answered the current call.

5 XML Objects Pushed to the Phone

The phone can request an XML object via HTTP GET, or an object can be pushed to the phone via a POST. The phone parses this object immediately upon receipt and displays the information to the screen.

The HTTP POST packet must contain an "xml=" line in the message body. The string to parse is located after the equals sign in the message. HTML forms that post objects to the phone must use a field named "xml" to send their data. Any applications that construct HTTP packets on the fly must also specify this line.

Note: the content of the HTTP POST is not url-encoded.

Below is a sample php source code that sends an XML object to an Aastra phone.

```php
<?php
#
function push2phone($server,$phone,$data)
{
$xml = "xml=".$data;

$post = "POST / HTTP/1.1\r\n";
$post .= "Host: $phone\r\n";
$post .= "Referer: $server\r\n";
$post .= "Connection: Keep-Alive\r\n";
$post .= "Content-Type: text/xml\r\n";
$post .= "Content-Length: ".strlen($xml)."\r\n\r\n";

$fp = @fsockopen ( $phone, 80, $errno, $errstr, 5);
if($fp)
      {
      fputs($fp, $post.$xml);
      flush();
      fclose($fp);
      }
}

##############################
$xml = "<AastraIPPhoneTextScreen>\n";
$xml .= "<Title>Push test</Title>\n";
$xml .= "<Text>This is a test for pushing a screen to a phone. It
is a way to demonstrate that we can push XML objects to an Aastra
Phone.</Text>\n";
$xml .= "</AastraIPPhoneTextScreen>\n";
push2phone("192.168.0.112","192.168.0.150",$xml);
?>
```

Note: To accept a pushed page the "XML Push Server List" phone parameter must be configured with the list of the HTTP servers allowed pushing a page.

Note: When a page is pushed to the phone the MWI lamp blinks to indicate a new message to the phone.

6 Action URIs

With firmware 1.4.1 new configuration parameters have been introduced to configure the phone to make an HTTP GET based on events, these events are:

• End of the boot sequence	`action uri startup`
• Successful registration	`action uri registered`
• On-hook	`action uri onhook`
• Off-hook	`action uri offhook`
• Incoming call	`action uri incoming`
• Outgoing call	`action uri outgoing`

Note: The only supported use of offhook action URI is to perform a GET that returns a NoOp Execute XML object. Although the phone will correctly display all other returned XML objects, the interaction with other phone features, such as speed dial, redial, etc., is undefined.

6.1 Configuration

A URI can be configured for each type of events; the configuration can be made using:

- The configuration files
- The Web UI.

See chapter 7.4 for more details on the Web UI Configuration.

Configuration description

These URIs will be configurable via the configuration files using the following parameters.

```
action uri startup: <URI to GET on startup>
action uri registered: <URI to GET on successful registration>
action uri incoming: <URI to GET on an incoming call>
action uri outgoing: <URI to GET on outgoing call>
action uri offhook: <URI to GET on an off-hook event>
action uri onhook: <URI to get on an on-hook event>
```

As described in chapter 4.2 the action URI support variables in their configuration.

Example

```
action uri startup: http://myserver.com/startup.php
action uri incoming:
http://myserver.com/incoming.php?number=$$REMOTENUMBER$$
```

6.2 Applications

Action URIs are very powerful as they allow an external application to take control of the display when an event occurs.

Here are some examples of potential applications.

Self-configuration

Using the startup URI, it is possible to develop self-provisioning on the phone. If a new phone boots and gets its configuration server IP address (through DHCP option 66 for example), it can download the `aastra.cfg` file with a startup URI set to an XML application, as it is a new phone the `<MAC>.cfg` config files does not exist. At the end of the boot, the phone will go to this XML application which can identify the phone and then generate the `<MAC>.cfg` file "on the fly" and ask the phone to reboot using the `PhoneExecute` object.

Screen pop

Using the incoming URI, it is possible to display extra information on the phone for an incoming call. For instance, the XML application that is called when there is an incoming call can do a database lookup (Microsoft Exchange or any database) and display information on the caller. Basically it is like having a screen pop application directly on the phone.

Call Center

As for the Screen pop, the incoming URI can be used in a call center environment to display CRM or queue information on the caller. The on-hook URI can also be used to collect the wrap-up code at the end of a call.

And many more…

7 XML Configuration

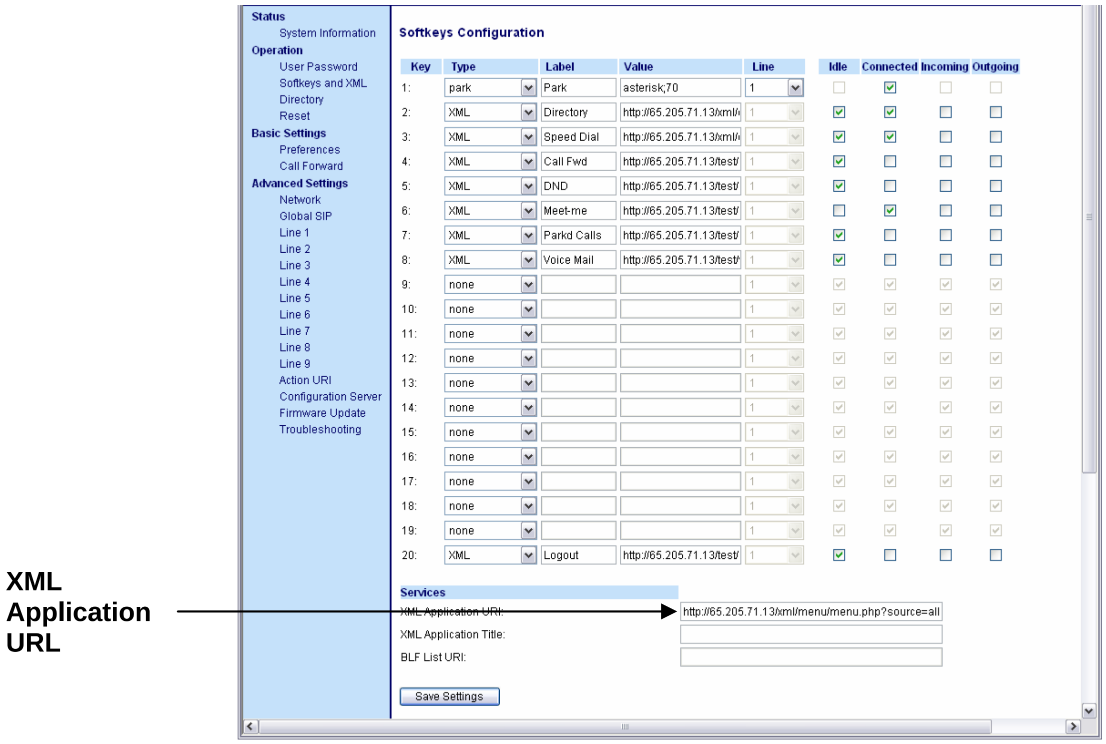
After creating an XML application to use on the IP phone, the application can be accessed as a Service or a Key.

7.1 Configuring a Custom Service from the Web UI (480i/480i CT only)

To load a new custom XML application to the IP phone, enter an HTTP address for the application at the “**Softkeys and XML**” screen in the Aastra IP Phone Web interface.

1. Select Operation=>Softkeys and XML.
2. Enter the HTTP address in the “**XML Application URL**” field.
3. If desired, give the XML application a custom title in the “**XML Application Title**” field.

The following illustration shows an XML application named “*http://65.205.71.13/xml/menu/menu.php?source=all*” in the applicable field. After clicking Save Settings in the Softkeys Configuration screen, the XML application is dynamically applied to the IP phone you are configuring. The application URI can be accessed by pressing the “Services” button on your IP phone and selecting option 4.



Softkeys Configuration

Key	Type	Label	Value	Line	Idle	Connected	Incoming	Outgoing
1:	park	Park	asterisk;70	1	☐	✓	☐	☐
2:	XML	Directory	http://65.205.71.13/xml/	1	✓	✓	☐	☐
3:	XML	Speed Dial	http://65.205.71.13/xml/	1	✓	✓	☐	☐
4:	XML	Call Fwd	http://65.205.71.13/test/	1	✓	☐	☐	☐
5:	XML	DND	http://65.205.71.13/test/	1	✓	☐	☐	☐
6:	XML	Meet-me	http://65.205.71.13/test/	1	☐	✓	☐	☐
7:	XML	Parkd Calls	http://65.205.71.13/test/	1	✓	☐	☐	☐
8:	XML	Voice Mail	http://65.205.71.13/test/	1	✓	☐	☐	☐
9:	none			1	✓	✓	✓	✓
10:	none			1	✓	✓	✓	✓
11:	none			1	✓	✓	✓	✓
12:	none			1	✓	✓	✓	✓
13:	none			1	✓	✓	✓	✓
14:	none			1	✓	✓	✓	✓
15:	none			1	✓	✓	✓	✓
16:	none			1	✓	✓	✓	✓
17:	none			1	✓	✓	✓	✓
18:	none			1	✓	✓	✓	✓
19:	none			1	✓	✓	✓	✓
20:	XML	Logout	http://65.205.71.13/test/	1	✓	☐	☐	☐

Services

XML Application URI:	http://65.205.71.13/xml/menu/menu.php?source=all
XML Application Title:	
BLF List URI:	

7.2 Configuring a Soft or Programmable Key from the Web UI

In addition to linking an XML application to a custom service, an application can be also be linked to a softkey (480i/ 480iCT) or programmable key (91xxi).

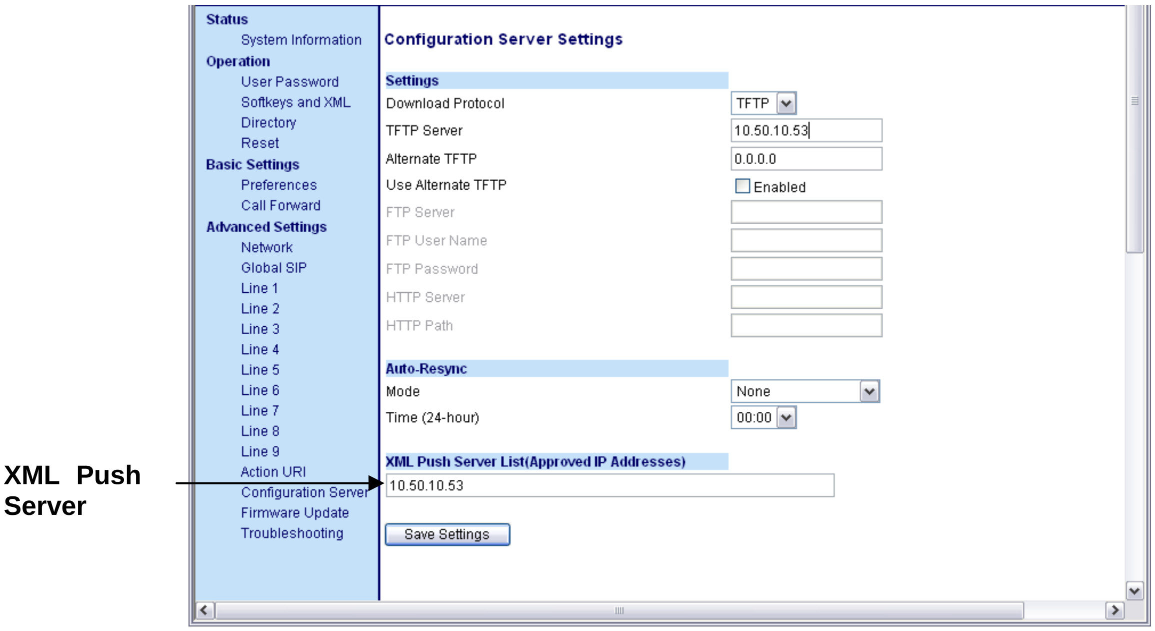
1. Select Operation=>Softkeys and XML or Programmable Key (91xx) .
2. Choose type “XML” for the desired key.

3. Enter the URI in the value field.
4. Click Save Settings and reboot the phone.

7.3 Configuring the XML Push Server List from the Web UI

The IP phone will only accept HTTP POSTs from the IP addresses set in the **XML Push Server List**.

1. Select Advanced Settings=>Configuration Server.
2. Enter a comma-separated list of IP addresses or domain names in the **XML Push Server List** field.
3. Click Save Settings and reboot the phone.



Configuration Server Settings

Settings	
Download Protocol	TFTP
TFTP Server	10.50.10.53
Alternate TFTP	0.0.0.0
Use Alternate TFTP	☐ Enabled
FTP Server	
FTP User Name	
FTP Password	
HTTP Server	
HTTP Path	
Auto-Resync	
Mode	None
Time (24-hour)	00:00

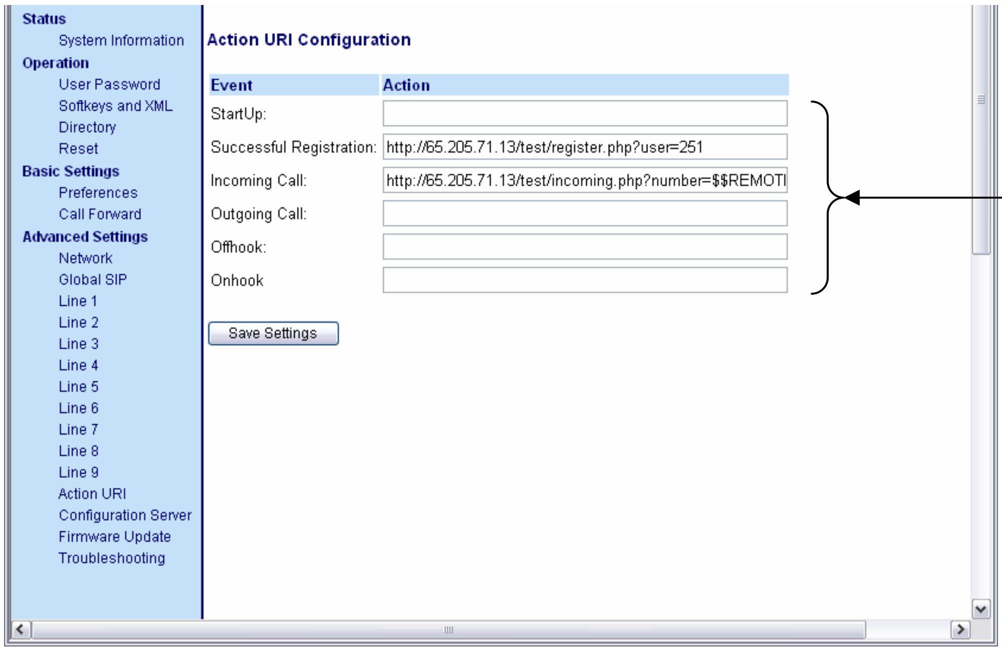
XML Push Server List(Approved IP Addresses)

10.50.10.53

7.4 Configuring the Action URIs from the Web UI

You can configure the URI to be called for each type of event supported by the phone from the Web UI..

1. Select Advanced Settings=>Action URI.
2. Enter the URI for each type of event.
3. Click Save Settings and reboot the phone.



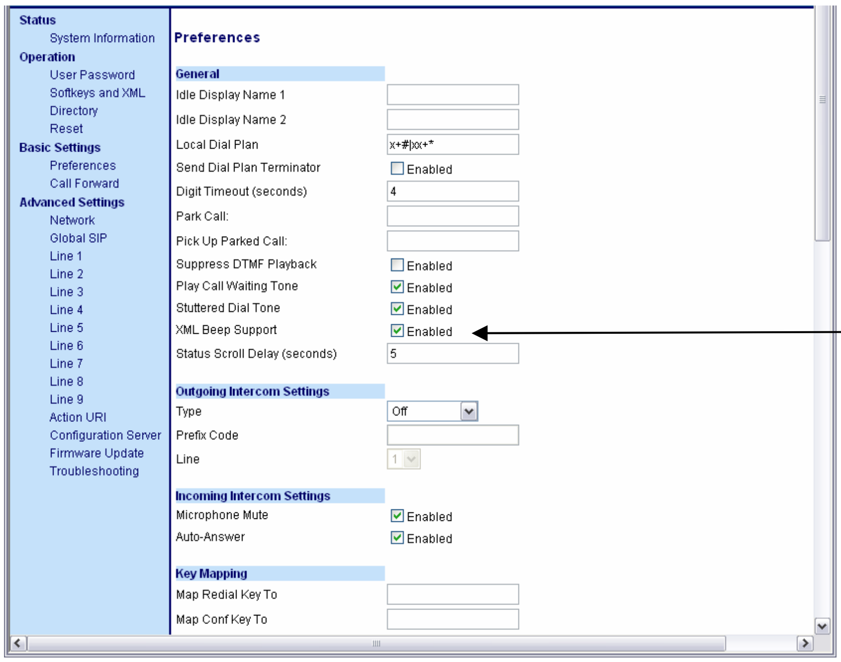
Action URI Configuration

Event	Action
StartUp:	
Successful Registration:	http://65.205.71.13/test/register.php?user=251
Incoming Call:	http://65.205.71.13/test/incoming.php?number=$$REMOTI
Outgoing Call:	
Offhook:	
Onhook	

7.5 Configuring the XML Beep Support from the Web UI

You can configure the **XML Beep Support** (enabled or disabled) from the Web UI. It impacts the behavior of the `AastraIPPhoneStatus` object regarding the phone notification.

1. Select Basic Settings=>Preferences.
2. Enable or disable the parameter.
3. Click Save Settings.



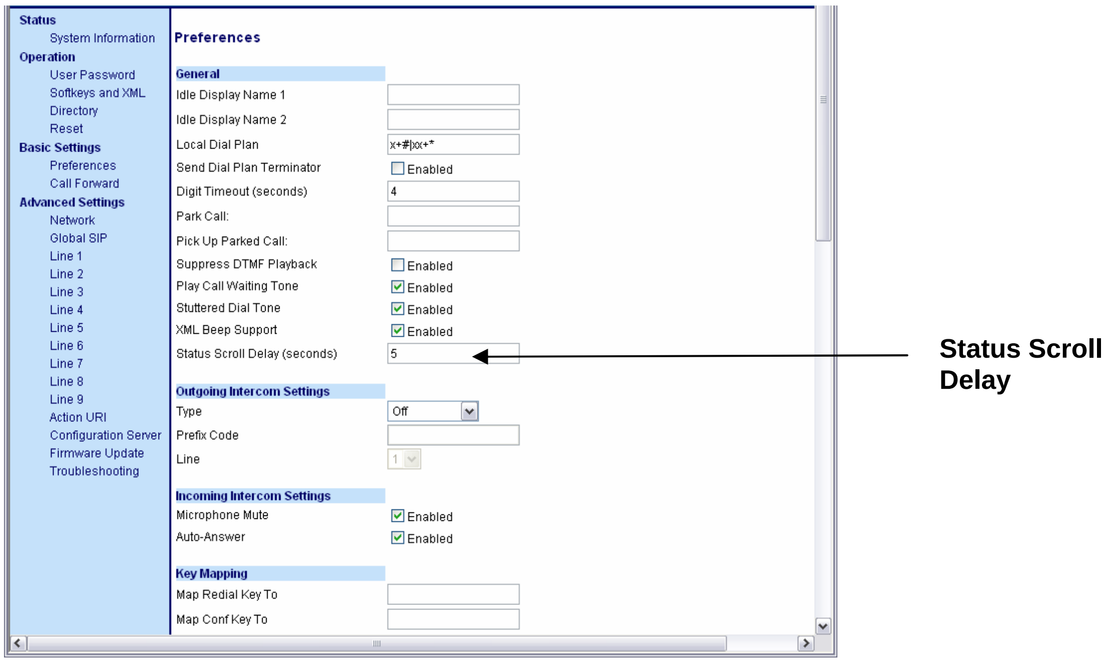
Preferences

General	
Idle Display Name 1	
Idle Display Name 2	
Local Dial Plan	x+#|xx+*
Send Dial Plan Terminator	☐ Enabled
Digit Timeout (seconds)	4
Park Call:	
Pick Up Parked Call:	
Suppress DTMF Playback	☐ Enabled
Play Call Waiting Tone	✓ Enabled
Stuttered Dial Tone	✓ Enabled
XML Beep Support	✓ Enabled
Status Scroll Delay (seconds)	5
Outgoing Intercom Settings	
Type	Off
Prefix Code	
Line	1
Incoming Intercom Settings	
Microphone Mute	✓ Enabled
Auto-Answer	✓ Enabled
Key Mapping	
Map Redial Key To	
Map Conf Key To	

7.6 Configuring the Status Scroll Delay from the Web UI

You can configure the **Status Scroll Delay** (delay in seconds, default 5) from the Web UI. It impacts the behavior of the `AastraIPPhoneStatus` object defining the delay between each message.

1. Select Basic Settings=>Preferences.
2. Enter the value in seconds.
3. Click Save Settings.



Preferences

General	
Idle Display Name 1	
Idle Display Name 2	
Local Dial Plan	x+#|xx+*
Send Dial Plan Terminator	☐ Enabled
Digit Timeout (seconds)	4
Park Call:	
Pick Up Parked Call:	
Suppress DTMF Playback	☐ Enabled
Play Call Waiting Tone	✓ Enabled
Stuttered Dial Tone	✓ Enabled
XML Beep Support	✓ Enabled
Status Scroll Delay (seconds)	5
Outgoing Intercom Settings	
Type	Off
Prefix Code	
Line	1
Incoming Intercom Settings	
Microphone Mute	✓ Enabled
Auto-Answer	✓ Enabled
Key Mapping	
Map Redial Key To	
Map Conf Key To	

7.7 XML Configuration using the Configuration Files

The *aastra.cfg* and *<mac>.cfg* Configuration File contains all the configuration parameters for the phone. Please refer to the phone administration guide for more details.

7.7.1 General XML parameters

You can enter an XML application in the *aastra.cfg* or *<mac>.cfg* file using the following parameters:

```
xml application URI
xml application title
xml application post list
xml beep notification
xml status scroll delay
```

Parameter – `xml application URI`	Configuration Files aastra.cfg, <mac>.cfg
Description	This is the XML application you are loading into the IP phone configuration.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	`xml application URI: http://172.16.96.63/aastra/internet.php`

Parameter – `xml application title`	Configuration Files aastra.cfg, <mac>.cfg
Description	This parameter allows you to rename the XML application in the IP phone UI (Services->4. Custom Feature). By default, when you load an XML application to the IP phone, the XML application title is called "Custom Feature". The "xml application title" parameter allows you to change that title. For example, if you are loading a traffic report XML application, you could change this parameter title to "Traffic Reports", and that title will display in the IP phone UI as Services->4. Traffic Reports.
Format	Alphanumeric characters
Default Value	Not Applicable
Range	Not Applicable
Example	`xml application title: Traffic Reports`

Parameter – `xml application post list`	Configuration Files aastra.cfg, <mac>.cfg
Description	The HTTP server that is pushing XML applications to the IP phone.
Format	IP address in dotted decimal format and/or Domain name address
Default Value	Not Applicable
Range	Not Applicable
Example	`xml application post list: 10.50.10.53, dhcp10-53.ana.aastra.com`

Parameter – `xml beep notification`	Configuration Files aastra.cfg, <mac>.cfg
Description	Enables or disables a BEEP notification on the phone when an AastraIPPhoneStatus object containing a “beep” attribute arrives to the phone.
Format	Boolean
Default Value	Value 1 (ON)
Range	0 (OFF) No beep is audible even if the beep attribute is present in the XML object. 1 (ON) The phone beeps when an XML object with the “beep” attribute arrives to the phone.
Example	`xml beep notification: 0`

Parameter – `xml status scroll delay`	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the length of time, in seconds, that each XML status message displays on the phone.
Format	Integer
Default Value	5
Range	1 to 25
Example	`xml status scroll delay: 3`

7.7.2 Programmable and Soft keys

You can configure keys (softkeys on the 480i/480iCT, hard keys on the 9112i/9133i) calling an XML application using the following parameters.

```
softkeyX/prgkeyX type: xml
softkeyX/prgkeyX value: http://someapp.xml
```

As described in chapter 4.2, system variables can be used in the URI to be called by pressing a key.

7.7.3 Examples

Example (9112i/9133i)

```
# XML configuration
xml application URI: http://172.16.96.63/aastra/internet.php
xml application post list: 10.10.50.53, xmlserver.aastra.com
xml beep notification: 1
xml status scroll delay: 5

# Softkey 1
softkey1 type: xml
softkey1 label: My XML
softkey1 value: http://172.16.96.63/aastra/internet.php

# Softkey 2
Softkey2 type: xml
Softkey2 label: Login
Softkey2 value: http://myserver.com/login.php?user=$$SIPUSERNAME$$
```

Example (480i/480i CT)

```
# XML configuration
xml application URI: http://172.16.96.63/aastra/internet.php
xml application post list: 10.10.50.53, xmlserver.aastra.com
xml beep notification: 1
xml status scroll delay: 5
```

```
# Key 1
prgkey1 type: xml
prgkey1 value: http://172.16.96.63/aastra/internet.php

# Key 2
prgkey2 type: xml
prgkey2 value: http://myserver.com/login.php?user=$$SIPUSERNAME$$
```

8 Why XML Applications for an IP Phone?

8.1 Telephony applications

This chapter details potential XML telephony applications which could be developed to enhance integration of an IP phone with the other telecom applications.

8.1.1 Directory

The first obvious applications that can be developed are the directory application, it includes:

- PBX directory
- Corporate directory (Global list from a Microsoft Exchange server)
- Personal contacts (My contacts in Outlook)
- Any LDAP directory (public or private)

8.1.2 Call Processing

XML applications can also be used to develop interactions between the call processing and the phone:

- DND
- Call Forward
- Parked calls
- Call pickup
- PBX configuration

8.1.3 Voice-Mail

- Voice mail messages management (play, skip, delete, …)
- Display message envelopes
- Presence management

8.1.4 Conference bridge

- Conference booking
- Conference reminder
- Audio console

8.1.5 Contact Center

- Agent Login/Logout
- Access to call center reports
- Account codes
- Wrap codes

8.2 Vertical applications

This chapter details potential applications that can be developed as an XML application for the Aastra IP phones. The list is far from being exhaustive.

8.2.1 Human Resources

- Available Vacation Days
- Available Personal Days
- 401K balance
- Clock-In/Clock-Out

8.2.2 Travel/Hotel

- Current Balance
- In-Room Dining Ordering
- Delivery Dining Options
- Extend Stay
- Schedule Airport Shuttle
- Request Housekeeping/Engineering
- Leave Feedback
- Wake-Up Call
- Book Corporate Travel
- Do Not Disturb

8.2.3 General Mobile Phone

- Weather Alerts
- Stock Alerts
- Stock Prices

- Worldwide Time/Temperature
- Server Alarms and Notifications
- Server Status
- Account Balances
- Current Gas Prices
- Local Movie Times
- Upcoming Concerts-By Category
- Order Flowers-by Category
- Send Order to Starbucks
- Send SMS Messages

8.2.4 Health Care

- Test Results
- Manage Appointment
- Appointment Reminders
- Take-Your-Medicine Reminders
- Order Hospital Meals
- Check Pharmacy Inventory
- Schedule Blood Donation

8.2.5 Education

- Attendance
- Request Substitute Teacher (used by primary teacher)
- Review Open Requests for Substitute Teacher (used by potential subs)
- Schedule Classes
- Request Dorm Room Change
- School Closing Notification/Status
- Parent Contact Info

8.2.6 General

- Track Fedex Package (or Airborne, UPS, etc.)

- Calling Card Minutes Remaining
- Reserve Meeting Rooms
- Contact center Metrics (Calls Waiting, Longest Hold, Performance against Service Level, etc.)
- Pro Sports Scores, Vegas Betting Lines
- Multitude of Banking apps: Balances, Transfers, etc.
- Music/Marketing On Hold Options
- Language Translation
- Daily Horoscope
- Broadcast Joke Of The Day/Inspirational Quote of the Day

8.2.7 Law Enforcement

- Amber Alerts
- Traffic Ticket Plead By Phone
- Fugitive Alerts

9 XSL Model for Aastra SIP Phones

```
< <?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="AastraIPPhoneTextScreen">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Title" type="xs:string" />
   <xs:element name="Text">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:minLength value="1" />
      <xs:maxLength value="1000" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element  name="SoftKey" type="softKeyType" minOccurs="0"
maxOccurs="6"/>
  </xs:sequence>
  <xs:attribute name="destroyOnExit" type="textAttributeType"
default="no" />
  <xs:attribute name="Beep"          type="textAttributeType"
default="no" />
 </xs:complexType>
</xs:element>

<xs:element name="AastraIPPhoneTextMenu">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Title" type="xs:string" />
   <xs:element name="MenuItem" minOccurs="1" maxOccurs="15">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="Prompt" type="xs:string" />
      <xs:element name="URI" type="xs:string" />
      <xs:element name="Selection" type="xs:string" minOccurs="0"
maxOccurs="1" />
     </xs:sequence>
     <xs:attribute name="base" type="xs:string" />
    </xs:complexType>
   </xs:element>
   <xs:element name="SoftKey" type="softKeyType" minOccurs="0"
maxOccurs="6"/>
  </xs:sequence>
  <xs:attribute name="destroyOnExit" type="textAttributeType"
default="no" />
  <xs:attribute name="Beep"          type="textAttributeType"
default="no" />
  <xs:attribute name="style"         type="xs:string"
default="numbered" />
  <xs:attribute name="defaultIndex"  type="integerAttributeType"
default="1" />
 </xs:complexType>
```

```
</xs:element>

<xs:element name="AastraIPPhoneInputScreen">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Title" />
   <xs:element name="Prompt" />
   <xs:element name="URL" />
   <xs:element name="Parameter" />
   <xs:element name="Default" />
   <xs:element name="SoftKey" type="softKeyType" minOccurs="0"
maxOccurs="6"/>
  </xs:sequence>
  <xs:attribute name="type" use="required">
   <xs:simpleType>
    <xs:restriction base="xs:string">
     <xs:pattern value="IP|string|number" />
    </xs:restriction>
   </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="password"      type="textAttributeType"
default="no"  />
  <xs:attribute name="destroyOnExit" type="textAttributeType"
default="no"  />
  <xs:attribute name="editable"      type="textAttributeType"
default="yes" />
  <xs:attribute name="Beep"          type="textAttributeType"
default="no"  />
 </xs:complexType>
</xs:element>

<xs:element name="AastraIPPhoneDirectory">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Title" type="xs:string" />
   <xs:element name="MenuItem" minOccurs="1" maxOccurs="15">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="Prompt" type="xs:string" />
      <xs:element name="URI" type="xs:string" />
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="SoftKey" type="softKeyType" minOccurs="0"
maxOccurs="6"/>
  </xs:sequence>
  <xs:attribute name="destroyOnExit" type="textAttributeType"
default="no" />
  <xs:attribute name="Beep"          type="textAttributeType"
default="no" />
  <xs:attribute name="next"          type="xs:string" />
  <xs:attribute name="previous"      type="xs:string" />
 </xs:complexType>
</xs:element>

<xs:element name="AastraIPPhoneExecute">
 <xs:complexType>
```

```
  <xs:sequence>
   <xs:element name="ExecuteItem" minOccurs="0"
maxOccurs="unbounded">
    <xs:complexType>
     <xs:attribute name="URI" type="xs:string" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="Beep" type="textAttributeType" default="no"
/>
 </xs:complexType>
</xs:element>

<xs:element name="AastraIPPhoneStatus">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Session" type="xs:string" minOccurs="0" />
   <xs:element name="Message">
    <xs:complexType mixed="true">
     <xs:attribute name="index"   type="integerAttributeType"
use="required" />
     <xs:attribute name="type">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:pattern value="alert" />
       </xs:restriction>
      </xs:simpleType>
     </xs:attribute>
     <xs:attribute name="Timeout" type="integerAttributeType"
default="3" />
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="Beep" type="textAttributeType" default="no"
/>
 </xs:complexType>
</xs:element>

<xs:simpleType name="textAttributeType">
 <xs:restriction base="xs:string">
  <xs:pattern value="yes|no" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="integerAttributeType">
 <xs:restriction base="xs:integer" />
</xs:simpleType>

<xs:complexType name="softKeyType">
 <xs:sequence>
  <xs:element name="Label" type="xs:string" />
  <xs:element name="URI" type="xs:string" />
 </xs:sequence>
 <xs:attribute name="index" use="required">
  <xs:simpleType>
   <xs:restriction base="xs:integer">
    <xs:minInclusive value="1" />
    <xs:maxInclusive value="6" />
   </xs:restriction>
  </xs:simpleType>
 </xs:attribute>
</xs:complexType>

</xs:schema>
```

10 Object Oriented PHP Classes

Aastra Telecom provides an object oriented API to develop XML applications. The API is available on the Aastra Web site http://www.aastratelecom.com.

Note: The PHP objects are taking care of the XML escape encoding.

10.1 AastraIPPhoneTextMenu()

This class allows you to create a XML Directory object.

Include

AastraIPPhoneTextMenu.class.php

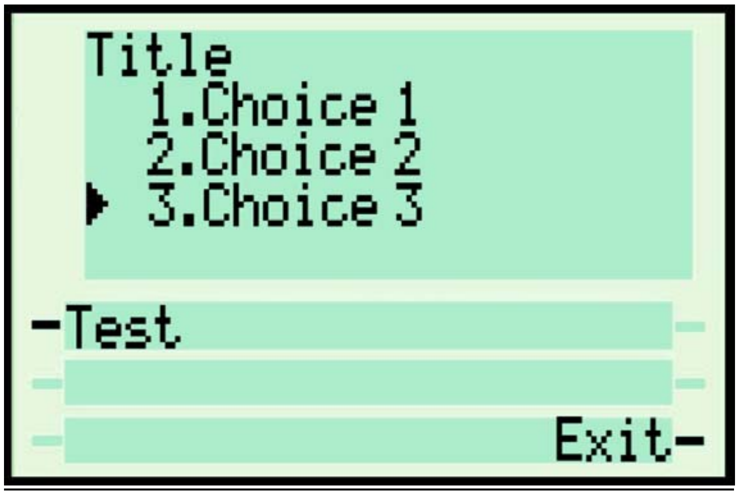
Methods

- setTitle(Title) to setup the title of an object
- setDestroyOnExit() to set DestroyOnExit parameter to "yes", optional
- setStyle(style) to set the style of the list numbered/none/radio (optional)
- setBeep() to enable a notification beep with the object (optional)
- addSoftkey(index,label,uri) to add custom softkeys to the object (optional)
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
- output() to display the object
- setDestroyOnExit() to set DestroyOnExit parameter to "yes", optional
- addEntry(name,url,selection) to add an element in the list to be displayed, at least one is needed.
- natsortbyname() to order the list (optional)

Example

```
require_once('AastraIPPhoneTextMenu.class.php');
$menu = new AastraIPPhoneTextMenu();
$menu->setTitle('Title');
$menu->setDestroyOnExit();
$menu->setDeFaultIndex('3');
$menu->addEntry('Choice 2',
'http://myserver.com/script.php?choice=2','Value=2');
$menu->addEntry('Choice 1',
'http://myserver.com/script.php?choice=1','Value=1');
$menu->addEntry('Choice 3',
'http://myserver.com/script.php?choice=3','Value=3');
$menu->natsortByName();
$menu->addSoftkey('1', 'Label',
'http://myserver.com/script.php?action=1');
$menu->addSoftkey('6', 'Exit', 'SoftKey:Exit');
$menu->output();
```

Output


Title
1.Choice 1
2.Choice 2
3.Choice 3
Test
Exit

10.2 AastraIPhoneTextScreen()

This class allows you to create a XML TextScreen object.

Include

AastraIPPhoneTextScreen.class.php'

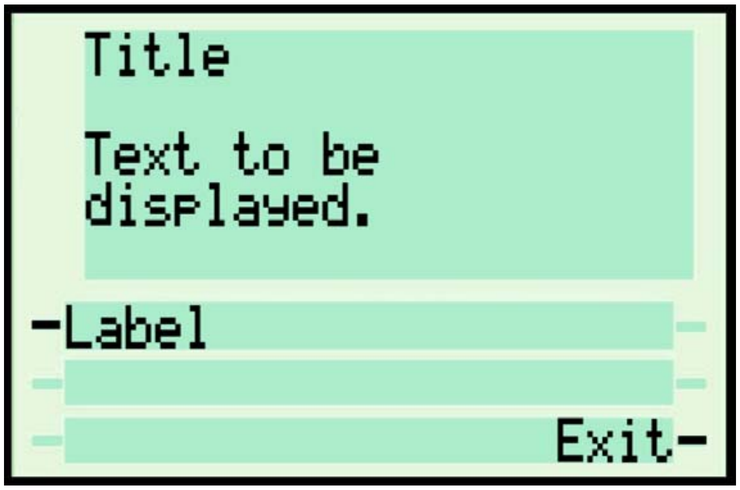
Methods

- setTitle(Title) to setup the title of an object
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- addSoftkey(index, label, uri) to add custom softkeys to the object (optional)
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
- output() to display the object
- setText(text) to set the text to be displayed.

Example

```
require_once('AastraIPPhoneTextScreen.class.php');
$text = new AastraIPPhoneTextScreen();
$text->setTitle('Title');
$text->setText('Text to be displayed.');
$text->setDestroyOnExit();
$text->addSoftkey('1','Label
1','http://myserver.com/script.php?action=1');
$text->addSoftkey('6','Exit','SoftKey:Exit');
$text->output();
```

Output


Title

Text to be
displayed.

-Label
Exit-

10.3 AastraIPPhoneInputScreen()

This class allows you to create a XML InputScreen object.

Include

AastraIPPhoneInputScreen.class.php'

Methods

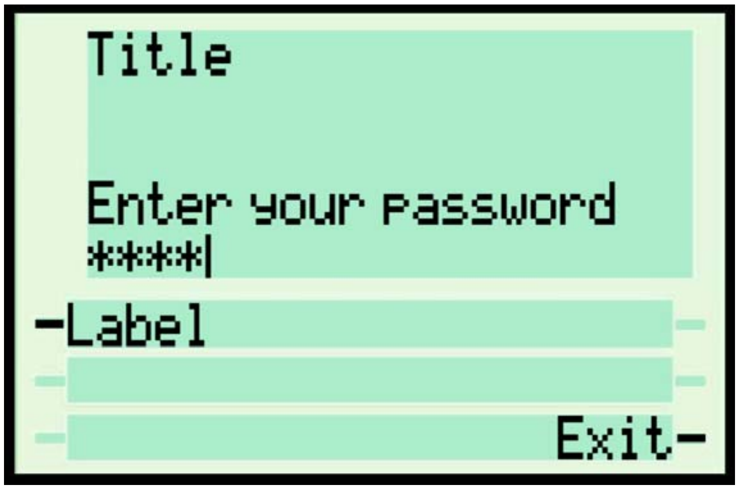
- setTitle(Title) to setup the title of an object
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- addSoftkey(index,label,uri) to add custom softkeys to the object (optional)
- output() to display the object
- setURL() to set the URL to called after the input
- setType(type) to set type of input ('IP', 'string'or 'number'), 'string'by default
- setDefault(default) to set default value for the input (optional)
- setParameter(param) to set the parameter name to be parsed after the input
- setPassword() to set the Password parameter to 'yes', 'no' by default (optional)
- setNotEditable() to set the editable parameter to 'no', 'yes' by default (optional)
- *setEditable()* is now replaced by setNotEditable but kept for compatibility reasons (optional)
- setPrompt(prompt) to set the prompt to be displayed for the input.

Example

```
require_once('AastraIPPhoneInputScreen.class.php');
$input = new AastraIPPhoneInputScreen();
$input->setTitle('Title');
$input->setPrompt('Enter your password');
$input->setParameter('param');
$input->setType('string');
```

```
$input->setURL('http://myserver.com/script.php');
$input->setPassword();
$input->setDestroyOnExit();
$input->addSoftkey('1',                                    'Label',
'http://myserver.com/script.php?action=1');
$input->addSoftkey('6', 'Exit', 'SoftKey:Exit');
$input->output();
```

Output



10.4 AastraIPPhoneDirectory()

This class allows you to create a XML Directory object.

Include

AastraIPPhoneDirectory.class.php'

Methods

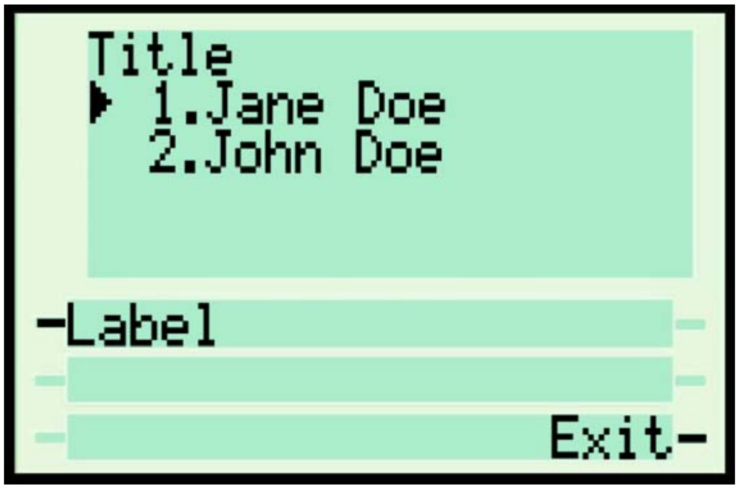
- setTitle(Title) to setup the title of an object
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- addSoftkey(index,label,uri) to add custom softkeys to the object (optional)
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
- output() to display the object
- setNext(next) to set URI of the next page, optional
- setPrevious(previous) to set URI of the previous page, optional
- addEntry(name,phone) to add an element in the list to be displayed, at least one is needed.
- natsortbyname() to order the list

Example

```
require_once('AastraIPPhoneDirectory.class.php');
```

```
$directory = new AastraIPPhoneDirectory();
$directory->setTitle('Title');
$directory->setNext('http://myserver.com/script.php?page=2');
$directory->setPrevious('http://myserver.com/script.php?page=0');
$directory->setDestroyOnExit();
$directory->addEntry('John Doe', '200');
$directory->addEntry('Jane Doe', '201');
$directory->natsortByName();
$directory->addSoftkey('1', 'Label',
'http://myserver.com/script.php?action=1');
$directory->addSoftkey('6', 'Exit', 'SoftKey:Exit');
$directory->output();
```

Output



```
Title
▶ 1.Jane Doe
  2.John Doe
-Label
                      Exit-
```

10.5 AastraIPPhoneStatus()

This class allows you to create a XML PhoneStatus object.

Include

AastraIPPhoneStatus.class.php'

Methods

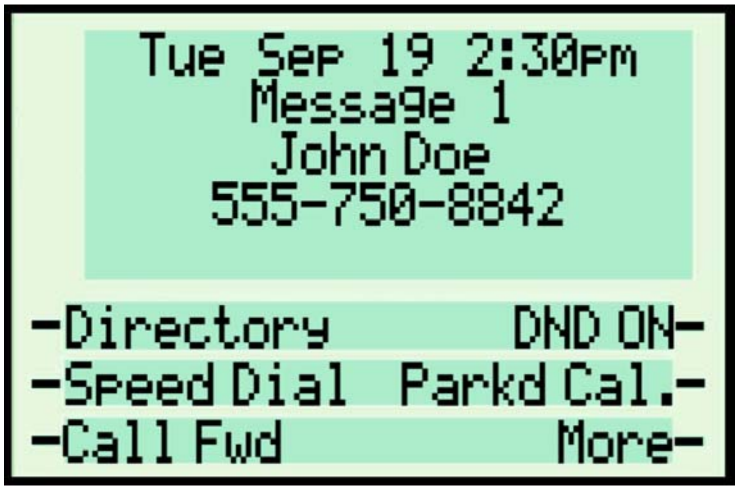
- output() to display the object
- setBeep() to enable a notification beep with the object (optional)
- setSession(session) to setup the session ID
- addEntry(index,message,type,timeout) to add a message to be displayed on the idle screen. Type and timeout are optional and type=”alert” is supported.

Example

```
require_once('AastraIPPhoneStatus.class.php');
$status = new AastraIPPhoneStatus();
$status->setSession('Session');
$status->setBeep();
$status->addEntry('1','Message 1');
```

```
$status->addEntry('2','Alert',’alert’,5);
$status->output();
```

Output


```
Tue Sep 19 2:30pm
Message 1
John Doe
555-750-8842

-Directory          DND ON-
-Speed Dial  Parkd Cal.-
-Call Fwd            More-
```

10.6 AastraIPPhoneExecute()

This class allows you to create a XML PhoneExecute object.

Include

AastraIPPhoneExecute.class.php'

Methods

- output() to display the object
- setBeep() to enable a notification beep with the object (optional)
- addEntry(url) to add an action to be executed.

Example

```
require_once('AastraIPPhoneExecute.class.php');
$execute = new AastraIPPhoneExecute();
$execute->addEntry('http://myserver.com/script.php?choice=2');
$execute->addEntry('Command: Reset');
$execute->output();
```

11 Sample Applications for Asterisk / Digium Open PBX

These examples are available on the Aastra Web site http://www.aastratelecom.com.

11.1 Directory

The following example show how to create a simple directory for the Asterisk users configured on your Asterisk server.

```
<?
#####################################################################
# Asterisk Directory for Aastra SIP Phones R1.4.1 or better
#
# php source code
# Provided by Aastra Telecom Ltd 2006
#
# Parses
#    - SIP users
#    - IAX users
#
# Warning
#    Location of Asterisk config files is set to "/etc/asterisk"
#####################################################################

# Location of asterisk config files
$location = "/etc/asterisk/";

# Global Variables
$Server = "http://$SERVER_ADDR".$_SERVER['SCRIPT_NAME'];

# Init number of records
$index=0;

# Parse sip.conf
$sip_array=parse_ini_file($location."sip_additional.conf", true);
while ($v=current($sip_array))
       {
       if((isset($v['callerid'])) and (key($sip_array)!="register"))
              {
              $temp=$v['callerid'];
              $len=strlen($temp);
              $callerid=substr($temp,0,$len-(strlen(strrchr($temp,  '<')))-
1);
              $directory[]                 =                 "<Prompt>".
$callerid."</Prompt>\n"."<URI>".key($sip_array)."</URI>\n"."<Selection>".
key($sip_array)."&amp;name=".$callerid."</Selection>\n";
              $index++;
              }
       next($sip_array);
       }

# Parse iax.conf
$iax_array=parse_ini_file($location."iax.conf", true);
while($v=current($iax_array))
       {
       if(isset($v['name']))
              {
```

```
       $directory[]="<Prompt>".$v['name']."</Prompt>\n"."<URI>".key($iax_a
rray)."</URI>\n";
              $index++;
              }
       next($iax_array);
       }

# Sort Directory
sort($directory);

# Retrieve last page
$last=intval($index/15);
if(($index-$last*15) != 0) $last++;

# Retrieve current page
$page=$_GET['page'];
if (empty($page)) $page=1;

# Display Page
$output ="<AastraIPPhoneTextMenu destroyOnExit=\"yes\">";
$output .= "<Title>Directory ($page/$last)</Title>\n";
$index=1;
foreach ($directory as $v)
       {
       if(($index>=(($page-1)*15+1)) and ($index<=$page*15))
              {
              $output .= "<MenuItem>\n";
              $output .= $v;
              $output .= "</MenuItem>\n";
              }
       $index++;
       }

# Dial button
$output .= "<SoftKey index=\"1\">\n";
$output .= "<Label>Dial</Label>\n";
$output .= "<URI>SoftKey:Dial</URI>\n";
$output .= "</SoftKey>\n";

# Next button
if($page!=$last)
       {
       $next=$page+1;
       $output .= "<SoftKey index=\"5\">\n";
       $output .= "<Label>Next</Label>\n";
       $output .= "<URI>$Server?page=$next</URI>\n";
       $output .= "</SoftKey>\n";
       }

# Previous button
if($page!=1)
       {
       $previous=$page-1;
       $output .= "<SoftKey index=\"2\">\n";
       $output .= "<Label>Previous</Label>\n";
       $output .= "<URI>$Server?page=$previous</URI>\n";
       $output .= "</SoftKey>\n";
       }

# Exit Button
$output .= "<SoftKey index=\"6\">\n";
$output .= "<Label>Exit</Label>\n";
```

```
$output .= "<URI>SoftKey:Exit</URI>\n";
$output .= "</SoftKey>\n";

# End of the object
$output .= "</AastraIPPhoneTextMenu>\n";

# HTTP header and output
header("Content-Type: text/xml");
header("Content-Length: ".strlen($output));
echo $output;
?>
```

11.2 DND

This example requires the phpagi includes.

The directory number must be passed as an argument to the script.

`script.php?user=XXXX` where XXXX is the directory number of the user.

```
<?
#####################################################################
# Asterisk DND for Aastra SIP Phones R1.4.1 or better
#
# php source code
# Provided by Aastra Telecom Ltd 2006
#####################################################################

include (dirname(__FILE__)."/phpagi/misc.php");
include (dirname(__FILE__)."/phpagi/phpagi-asmanager.php");

#####################################################################
# Aastra_decode_HTTP_header
#
# Returns an array
#   0 Phone Type
#   1 Phone MAC Address
#   2 Phone firmware version
#####################################################################

function Aastra_decode_HTTP_header()
{
$user_agent=$_SERVER["HTTP_USER_AGENT"];
if(stristr($user_agent,"Aastra"))
       {
       $value=preg_split("/ MAC:/",$user_agent);
       $fin=preg_split("/ /",$value[1]);
       $value[1]=preg_replace("/\-/","",$fin[0]);
       $value[2]=preg_replace("/V:/","",$fin[1]);
       }
else
       {
       $value[0]="MSIE";
       $value[1]="NA";
       $value[2]="NA";
       }

return($value);
}
```

```
######################################################################
# Global parameters
######################################################################
$Server = "http://$SERVER_ADDR".$_SERVER['SCRIPT_NAME'];
$dnd=0;

# Retrieve parameters
$user=$_GET['user'];
$action=$_GET['action'];
$status=$_GET['status'];

# Force default action
if($action=="") $action="change";

# Get header info
$header=Aastra_decode_HTTP_header();

# Get current value only if action is change or update
if(($action=="change") || ($action=="update"))
       {
       # Connect to AGI
       $as = new AGI_AsteriskManager();
       $res = $as->connect();

       #DND GET
       $res = $as->Command('database get DND '.$user);
       $line=split("\n", $res['data']);
       $value=split(" ", $line[0]);
       if($value[1]=="YES") $dnd=1;
       if($dnd==0)
              {
              $value=split(" ", $line[1]);
              if($value[1]=="YES") $dnd=1;
              }

       # CHANGE CURRENT VALUE
       if($action=="change")
              {
              # change DND status
              if($dnd==0)
                     {
                     $res = $as->Command('database put DND '.$user.' YES');
                     $dnd=1;
                     }
              else
                     {
                     $res = $as->Command('database del DND '.$user);
                     $dnd=0;
                     }
              }

       # Disconnect properly
       $as->disconnect();
       }

# Setup header type
header("Content-Type: text/xml");

# Update action
if($action=="update")
       {
       $output = "<AastraIPPhoneExecute>\n";
       $output .= "<ExecuteItem
URI=\"".$Server."?action=msg&amp;status=".$status."\"/>\n";
       $output .= "</AastraIPPhoneExecute>\n";
       }

# Update message
if($action=="msg")
       {
       $output = "<AastraIPPhoneStatus Beep=\"yes\">\n";
       $output .= "<Session>CFDND12345</Session>\n";
       if ($status==1) $output .= "<Message index=\"0\">DND
activated</Message>\n";
       else $output .= "<Message index=\"0\"></Message>\n";
       $output .= "</AastraIPPhoneStatus>\n";
       }

# Action=change
if($action=="change")
       {
       switch($header[0])
              {
              case "Aastra9112i":
              case "Aastra9133i":
                     $output = "<AastraIPPhoneStatus Beep=\"yes\">\n";
                     $output .= "<Session>CFDND12345</Session>\n";
                     if ($dnd==1) $output .= "<Message index=\"0\">DND
activated</Message>\n";
                     else $output .= "<Message index=\"0\"></Message>\n";
                     $output .= "</AastraIPPhoneStatus>\n";
                     break;

              default:
                     $output = "<AastraIPPhoneTextScreen
destroyOnExit=\"yes\">\n";
                     $output .= "<Title></Title>\n";
                     if ($dnd==0) $output .= "<Text>DND
deactivated.</Text>\n";
                     else $output .= "<Text>DND activated.</Text>\n";
                     $output .= "<SoftKey index=\"6\">\n";
                     $output .= "<Label>Done</Label>\n";
                     $output .=
"<URI>".$Server."?action=update&amp;status=".$dnd."</URI>\n";
                     $output .= "</SoftKey>\n";
                     $output .= "</AastraIPPhoneTextScreen>\n";
                     break;
              }
       }

header("Content-Length: ".strlen($output));
echo $output;
?>
```

11.3 Call Forward

This example requires the phpagi includes.

The directory number must be passed as an argument to the script.

`script.php?user=XXXX` where XXXX is the directory number of the user.

```
<?
######################################################################
# Asterisk Call Forward for Aastra SIP Phones R1.4.1 or better
#
# php source code
# Provided by Aastra Telecom Ltd 2006
######################################################################

include (dirname(__FILE__)."/phpagi/misc.php");
include (dirname(__FILE__)."/phpagi/phpagi-asmanager.php");

######################################################################
# Aastra_decode_HTTP_header
#
# Returns an array
#   0 Phone Type
#   1 Phone MAC Address
#   2 Phone firmware version
######################################################################

function Aastra_decode_HTTP_header()
{
$user_agent=$_SERVER["HTTP_USER_AGENT"];
if(stristr($user_agent,"Aastra"))
       {
       $value=preg_split("/ MAC:/",$user_agent);
       $fin=preg_split("/ /",$value[1]);
       $value[1]=preg_replace("/\-/","",$fin[0]);
       $value[2]=preg_replace("/V:/","",$fin[1]);
       }
else
       {
       $value[0]="MSIE";
       $value[1]="NA";
       $value[2]="NA";
       }

return($value);
}

######################################################################
# Global parameters
$Server = "http://$SERVER_ADDR".$_SERVER['SCRIPT_NAME'];
$cf="";

# Retrieve parameters
$user=$_GET['user'];
$action=$_GET['action'];
$value=$_GET['value'];

# Get header info
$header=Aastra_decode_HTTP_header();

# Connect to AGI
$as = new AGI_AsteriskManager();
$res = $as->connect();

#GET CFWD
$res = $as->Command('database get CF '.$user);
$line=split("\n", $res['data']);
$data=split(" ", $line[0]);
if($data[0]=="Value:") $cf=$data[1];
if($cf=="")
```

```
       {
       $data=split(" ", $line[1]);
       if($data[0]=="Value:") $cf=$data[1];
       }

# change CF status
if(isset($action))
       {
       if($action=="cancel")
              {
              $res = $as->Command('database del CF '.$user);
              $output = "<AastraIPPhoneExecute>\n";
              $output .= "<ExecuteItem
URI=\"".$Server."?action=msg&amp;user=".$user."\"/>\n";
              switch($header[0])
                     {
                     case "Aastra9112i":
                     case "Aastra9133i":
                            break;

                     default:
                            $output .= "<ExecuteItem
URI=\"".$Server."?user=".$user."\"/>\n";
                            break;
                     }
              $output .= "</AastraIPPhoneExecute>\n";
              }

       if($action=="set")
              {
              $res = $as->Command('database put CF '.$user.' '.$value);
              $output = "<AastraIPPhoneExecute>\n";
              $output .= "<ExecuteItem
URI=\"".$Server."?action=msg&amp;user=".$user."\"/>\n";
              switch($header[0])
                     {
                     case "Aastra9112i":
                     case "Aastra9133i":
                            break;

                     default:
                            $output .= "<ExecuteItem
URI=\"".$Server."?user=".$user."\"/>\n";
                            break;
                     }
              $output .= "</AastraIPPhoneExecute>\n";
              }

       if($action=="change")
              {
              $output = "<AastraIPPhoneInputScreen type=\"number\"
destroyOnExit=\"yes\">\n";
              $output .= "<Title>Call Forward</Title>\n";
              $output .= "<Prompt>Enter destination</Prompt>\n";
              $output .=
"<URL>".$Server."?user=$user&amp;action=set</URL>\n";
              $output .= "<Parameter>value</Parameter>\n";
              $output .= "<Default></Default>\n";
              $output .= "</AastraIPPhoneInputScreen>\n";
              }

       if($action=="msg")
              {
```

```
              $output = "<AastraIPPhoneStatus Beep=\"yes\">\n";
              $output .= "<Session>CFDND</Session>\n";
              if ($cf=="") $output .= "<Message index=\"1\"></Message>\n";
              else $output .= "<Message index=\"1\">CFWD
activated</Message>\n";
              $output .= "</AastraIPPhoneStatus>\n";
              }

       # Disconnect properly
       $as->disconnect();

       header("Content-Type: text/xml");
       header("Content-Length: ".strlen($output));
       echo $output;
       exit;
       }

# Disconnect properly
$as->disconnect();

# Setup header type
header("Content-Type: text/xml");

switch($header[0])
       {
       case "Aastra9112i":
       case "Aastra9133i":
              $output = "<AastraIPPhoneTextMenu destroyOnExit=\"yes\">\n";
              if($cf=="") $output .= "<Title>CFWD deactivated</Title>\n";
              else $output .= "<Title>CFWD set (".$cf.")</Title>\n";
              if($cf!="")
                     {
                     $output .= "<MenuItem>\n";
                     $output .= "<Prompt>Cancel</Prompt>\n";
                     $output .=
"<URI>$Server/cfwd.php?action=cancel&amp;user=$user</URI>\n";
                     $output .= "</MenuItem>\n";
                     }
              $output .= "<MenuItem>\n";
              $output .= "<Prompt>Change</Prompt>\n";
              $output .=
"<URI>$Server/cfwd.php?action=change&amp;user=$user</URI>\n";
              $output .= "</MenuItem>\n";
              $output .= "</AastraIPPhoneTextMenu>\n";
              break;

       default:
              $output = "<AastraIPPhoneTextScreen
destroyOnExit=\"yes\">\n";
              $output .= "<Title>Call Forward for $user</Title>\n";
              if ($cf=="") $output .= "<Text>Call Forward is currently
deactivated.</Text>\n";
              else $output .= "<Text>Call Forward is currently set to
$cf.</Text>\n";
              $output .= "<SoftKey index=\"1\">\n";
              $output .= "<Label>Change</Label>\n";
              $output .=
"<URI>$Server/cfwd.php?action=change&amp;user=$user</URI>\n";
              $output .= "</SoftKey>\n";
              if($cf!="")
                     {
                     $output .= "<SoftKey index=\"2\">\n";
                     $output .= "<Label>Cancel</Label>\n";
```

```
                    $output .= 
"<URI>$Server/cfwd.php?action=cancel&amp;user=$user</URI>\n";
                    $output .= "</SoftKey>\n";
                    }
             $output .= "<SoftKey index=\"6\">\n";
             $output .= "<Label>Done</Label>\n";
             $output .= "<URI>SoftKey:Exit</URI>\n";
             $output .= "</SoftKey>\n";
             $output .= "</AastraIPPhoneTextScreen>\n";
             break;
       }

header("Content-Length: ".strlen($output));
echo $output;
?>
```

12 Sample XML applications

Aastra Telecom has made available, **for demonstration purpose only**, a list of XML applications on the Internet.

The applications currently available are:

- Area code
- Ask Google
- CNN News
- ESPN News
- Horoscope
- Movies
- Stock
- Today…
- Weather
- World Clock

Note: Aastra Telecom does not guarantee the availability of these applications. The applications can change any time without notice.

Note: These applications should not be used commercially; any abusive use of these applications will be detected and the phone will be automatically banned from the applications.

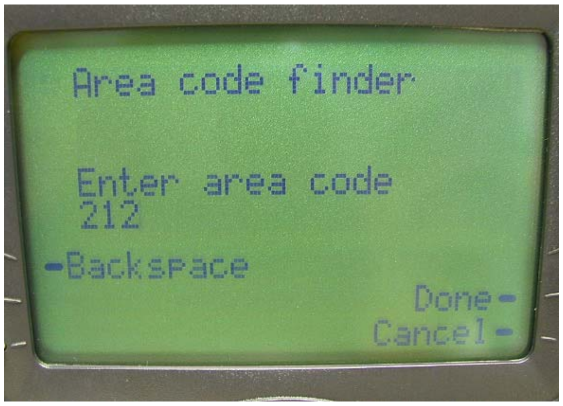
Note: your Aastra SIP phone must have Internet access to use these applications.

These XML applications can be configured individually or as a global menu.

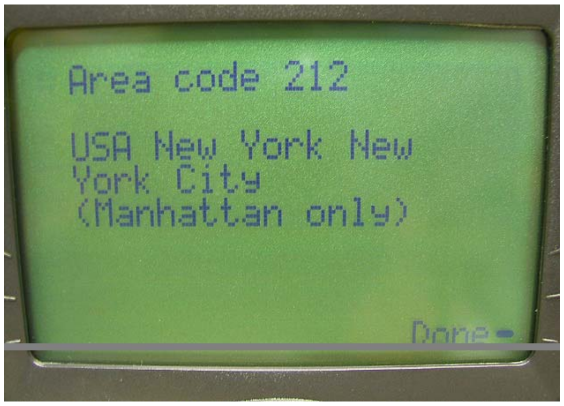
12.1 Area code

This application allows the user to lookup for the State/Cities of any given area code.

```
uri="http://65.205.71.13/xml/area/area.php"
```


```
Area code finder

Enter area code
212
-Backspace
                Done-
              Cancel-
```


```
Area code 212

USA New York New
York City
(Manhattan only)

                Done-
```

12.2 Ask Google

This application allows the user to ask a general question to Google.com. Please refer to http://www.google.com/sms for more details.

Questions

- 1 USD in Euro
- Define TCP
- Translate bread in French
- …

```
uri="http://65.205.71.13/xml/google/google.php"
```

12.3 CNN News

RSS feed from CNN.com including Top Stories, World, Politics…

```
uri="http://65.205.71.13/xml/rss/rss.php?feed=cnn"
```

12.4 ESPN News

RSS feed from ESPN.com bringing news for the most popular sports in North America.

```
uri="http://65.205.71.13/xml/rss/rss.php?feed=espn"
```

12.5 Horoscope

RSS feed from dailyhoroscopes.com.

```
uri="http://65.205.71.13/xml/horoscope/horoscope.php"
```

12.6 Movies

RSS feed from movies.com including the following topics:

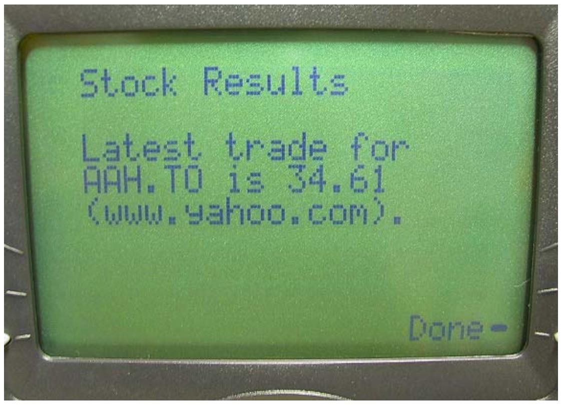
- In theaters
- Upcoming movies
- Now on DVD

```
uri="http://65.205.71.13/xml/rss/rss.php?feed=movies"
```

12.7 Stock

This application uses www.yahoo.com to get the value of any given stock. Please refer to yahoo.com for the syntax of the stock ticker.

```
uri="http://65.205.71.13/xml/stock/stock.php"
```


Stock Results

Latest trade for AAH.TO is 34.61 (www.yahoo.com).

12.8 Today…

RSS feed from answers.com including the following topics:

- Word of the Day
- Birthdays today
- This day in History
- Quote of the Day

```
uri="http://65.205.71.13/xml/rss/rss.php?feed=day"
```

12.9 Weather

RSS feed from rssweather.com providing weather forecast for the following cities:

- Boston, MA, USA
- Toronto, Canada
- Billerica, MA, USA
- Montreal, Canada
- Frisco, TX, USA
- Berlin, Germany
- Paris, France
- Zurich, Switzerland

```
uri="http://65.205.71.13/xml/rss/rss.php?feed=weather
```

12.10 World Clock

Date and time from around the world using www.timeanddate.com

```
uri="http://65.205.71.13/xml/clock/clock.php"
```

12.11 Global menu

The following URI displays a menu with all the above applications. It can be used as the “Custom feature” menu on a SIP Phone.

```
uri=”http://65.205.71.13/xml/menu/menu.php?source=all”
```