Web Application Developer's Guide for the SoundPoint® IP/SoundStation® IP Family

SIP 3.0.2

April, 2008 Edition
1725-17693-300 Rev. A
SIP 3.0.2

POLYCOM®

Trademark Information

Polycom®, the Polycom logo design, SoundPoint® IP, SoundStation®, SoundStation VTX 1000®, ViaVideo®, ViewStation®, and Vortex® are registered trademarks of Polycom, Inc. Conference Composer™, Global Management System™, ImageShare™, Instructor *RP*™, iPower™, MGC™, PathNavigator™, People+Content™, PowerCam™, Pro-Motion™, QSX™, ReadiManager™, Siren™, StereoSurround™, V²IU™, Visual Concert™, VS4000™, VSX™, and the industrial design of SoundStation are trademarks of Polycom, Inc. in the United States and various other countries. All other trademarks are the property of their respective owners.

Patent Information

The accompanying product is protected by one or more U.S. and foreign patents and/or pending patent applications held by Polycom, Inc.

Polycom Inc.
4750 Willow Road
Pleasanton, CA 94588-2708
USA

About This Guide

The Developer's Guide for the SoundPoint IP / SoundStation IP family is for developers of applications which use the Microbrowser on SoundPoint IP / SoundStation IP phones.

The following related documents for SoundPoint IP / SoundStation IP family are available:

- Quick Start Guides, which describe how to assemble the phones
- Quick User Guides, which describe the most basic features available on the phones
- User Guides, which describe the basic and advanced features available on the phones
- Administrator's Guide, which describes how to configure, customize, manage, and troubleshoot SoundPoint IP / SoundStation IP phone systems
- Technical Bulletins, which describe workarounds to existing issues
- Release Notes, which describe the new and changed features and fixed problems in the latest version of the software

For support or service, please go to Polycom Technical Support at http://www.polycom.com/support/voip/.

Polycom recommends that you record the phone model numbers, software (both the bootROM and SIP), and partner platform for future reference.

SoundPoint IP / SoundStation IP models: _____

BootROM version: _____

SIP Application version: _____

Partner Platform: _____

Contents

1

Overview

This chapter provides an overview of the Microbrowser available on SoundPoint IP 330/320, 430, 501, 550, 560, 600, 601, 650, and 670 desktop phones and SoundStation IP 4000, 6000, and 7000 conference phones.

It also provides an introduction to XHTML and guidelines for the application development.

This chapter contains information on:

- What is the Microbrowser
- What is XHTML
- How to Create Applications

To develop an application that can run on the Microbrowser, refer to Application Development on page 2-1. To troubleshoot any problems with your applications, refer to Troubleshooting on page 3-1.

What is the Microbrowser

The Microbrowser is like any Web browser—Microsoft Internet Explorer and Firefox, for example—but supports only a subset of XHTML features. It can connect to Web servers hosted in the Internet or intranet and download XHTML pages. The Microbrowser supports a limited number of XHTML 1.0 features—it does not have full Web browser functionality.

The Microbrowser downloads XHTML content from a Web server into the phone's memory, then parses the content to identify XHTML tags and renders these tags onto the phone's graphic display. The appearance of the rendered page depends on the graphical capabilities and display size of the device on which the browser is running. Complicated pages should be avoided on devices with very small displays.

The Microbrowser does not support scripting (such as JavaScript). All actions on data entered into forms is processed by the server using POST or GET methods.

The XHTML pages displayed on the Microbrowser can contain static or dynamic information.

Static XHTML. These pages are created using XHTML editors and hosted by the Web server. These pages are accessed from the Microbrowser (using HTTP protocol) by entering the URL to access the page. These XHTML pages are called static, because the information that is displayed is already coded into the XHTML pages. These pages do not include information that keep changing or contact other services for update.

Dynamic XHTML. These pages involves dynamic information updates of XHTML pages by an application hosted on the Web server. The application residing on the Web server will get information from an intranet or through the Internet—data service providers like Yahoo, Exchange Server, Call Control Servers and other enterprise servers.

Users can launch the Microbrowser on a SoundPoint IP or SoundStation IP phone by pressing the **Applications** key, or if there isn't one on the phone, it can be accessed through the **Menu** key by selecting ***Features***, and then ***Applications.***

Note

As of SIP 2.2, the **Services** key and menu entry were renamed **Applications**, however the functionality remains the same.

The Microbrowser is supported on:

- SoundPoint IP 330/320 — screen resolution - 102*22 pixels (3″ by 1″)
- SoundPoint IP 430 — screen resolution – 132*46 pixels (3.5″*1.5″)
- SoundPoint IP 501 — screen resolution – 160*80 pixels (4″ by 2″)
- SoundPoint IP 550/560/601/650 — screen resolution – 320*160 pixels (4″ by 2″)
- SoundPoint IP 670 — screen resolution – 320*160 pixels (4″ by 2″)
- SoundStation IP 4000/6000 — screen resolution – 240*68 pixels (2.4″ by 0.8″)
- SoundStation IP 7000 — screen resolution – 255*60 pixels (3″ by 1.5″)

What is XHTML

XHTML is the abbreviation of eXtensible HyperText Markup Language.

XHTML 1.0 is a transformation of HTML 4.01 into valid XML. The use of the stricter XML syntax makes parsing of XHTML much easier for small clients, but XHTML 1.0 was also the first step towards making HTML easily extensible. Moving to XML allowed the methods used to create XML extensions to apply to HTML as well. Step two occurred with XHTML 1.1,

where XHTML was divided up into 'modules', where any features above and beyond a skeleton set were grouped into individual modules. User agent (UA) developers could then decide which extensions to support. A simple user agent can be considered a fully compliant user agent by supporting only the Basic module, whereas a more powerful browser can support all the official modules, as well as those developed by third parties.

Modularization is also intended to help content creators. As more and more devices become web-enabled, the number of platforms a content creator will be asked to support will become unreasonable. By dividing HTML up into different 'building blocks' content creators can supply a minimal version of their site for user agents that only support the Basic module, a moderate version of their site for user agents who support the additional modules, and a full version of their site for user agents that support the full range of the XHTML specification.

Finally the X in XHTML was intended to help people who wish to extend HTML. The use of XML brought a standard grammar with which they could define their extension, and the modularization meant that their extension would be just another module that a user agent developer or content creator could choose to support. Additionally, since XHTML pages should state what modules are required to accurately render them, the user agent software could dynamically load a 'plug-in' that it could use to render a module that was defined after the user agent had been originally released.

For more information, go to:

- HTML 4.0—http://www.w3.org/TR/1999/REC-html401-19991224
- XHTML™ 1.0—http://www.w3.org/TR/2002/REC-xhtml1-20020801
- XHTML™ Basic—http://www.w3.org/TR/2000/REC-xhtml-basic-20001219/
- XHTML™ 1.1—http://www.w3.org/TR/2001/REC-xhtml11-20010531/
- XHTML Tables Module - XHTML™2.0—http://www.w3.org/TR/2004/WD-xhtml2-20040722/mod-tables.html

For the purposes of this guide, it is assumed that you have experience in HTML and XHTML programming or access to someone who has such experience.

How to Create Applications

You can design the following types of applications:

- Web browser

- Company directory
- Stock ticker

Depending on the type and complexity of the application, you might use one of the following tools for creation:

- Text editor
- XML editor
- Microsoft Word

When designing applications, you might want to consider the following guidelines:

Note These guidelines are for your information only. You are solely responsible for determining the suitability and applicability of this information to your needs.

1. Spend sufficient time designing the application by:
   - Developing a conceptual design
   - Describe all user-application interactions
   - Plan for all user types
2. Create standardized applications to assist in:
   - Lowering design time
   - Speed up debugging
   - Increasing usability
3. Promote consistent output and predictable user input.
4. Create a prototype application to test on sample users.
5. Thoroughly test your application before releasing to:
   - Identify all user interface issues
   - Verify that all error conditions are caught cleanly

For step-by-step instructions on how to develop an XHTML application that can be run on the Microbrowser of all SoundPoint IP and SoundStation IP phones, refer to Application Development on page 2-1.

Note Polycom is not responsible for troubleshooting any programming that you create for the Microbrowser.

2

Application Development

This chapter provides information on supported XHTML elements. It describes HTTP support and the Microbrowser user interface. It also describes the configuration parameters that can be found in **sip.cfg** .

This chapter presents step-by-step instructions on how to develop an XHTML application that can be run on the Microbrowser of certain SoundPoint IP and SoundStation IP phones.

This chapter contains information on:

- Supported XHTML Elements
- HTTP Support
- Microbrowser User Interface
- Developing an XHTML Application

To troubleshoot any problems with your applications, refer to Troubleshooting on page 3-1.

Note Polycom is not responsible for troubleshooting any programming that you create for the Microbrowser.

Supported XHTML Elements

The Microbrowser supports a subset of XHTML elements. Most are derived from HTML 4.01.

The supported elements and attributes are:

- Basic Tags
- Link Tags
- Input Tags
- Image Tags

- Table Tags
- Meta Information Tags

Unsupported elements and attributes are described in Unsupported XHTML Elements on page A-1.

Basic Tags

The following basic tags are supported:

- <!DOCTYPE>—Defines the document type
- <!--...-->—Defines a comment

<!DOCTYPE>

The <!DOCTYPE> declaration is the very first thing in your document, before the <html> tag. This tag tells the browser which XHTML specification the document uses. XHTML 1.0 specifies three XML document types: Strict, Transitional, and Frameset.

- XHTML Strict
  - Use this DTD when you want clean markup, free of presentational clutter.

    For example,
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
- XHTML Transitional
  - Use this DTD when you need to use XHTML's presentational features.

    For example,
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
- XHTML Frameset
  - Use this DTD when you want to use frames.

    For example,
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

This tag does not have any attributes.

<!--...-->

The comment tag is used to insert a comment in the source code. A comment will be ignored by the browser. You can use comments to explain your code, which can help you when you edit the source code at a later date.

This tag does not have any attributes.

Link Tags

The following link tag is supported:

- <a> — Defines an anchor

Note

The Microbrowser supports both `http://` and `tel://` URL schemes. When a tel:// URL is selected, the phone switches to the telephony application and dials the number specified in the URL. Currently the number is dialed as-is, however, full support for tel:// URL parsing as specified in RFC 2806 will be available in a future release.

`sip://` URLs are not supported at this time.

<a>

The <a> tag defines an anchor. An anchor can be used to create a link to another document by using the href attribute.

The following attributes are supported:

Attribute	Value/s	Description
href	URL (Ex:" www.polycom.com")	The target URL of the link
name	section_name	Names an anchor. Use this attribute to create a bookmark in a document. In future versions of XHTML the name attribute will be replaced by the id attribute. ***Note**: This attribute is parsed, but not used.*

Input Tags

The following input tags are supported:

- <form> — Defines a form
- <input> — Defines an input field

Note The Microbrowser supports both the GET and POST methods for submitting forms. Nesting forms within tables is supported. However, nesting of one form tag within another is not supported and may lead to unexpected results.

<form>

The form element creates a form for user input. A form can contain text fields, check boxes, radio buttons and more. Forms are used to pass user data to a specified URL.

The following attributes are supported:

Attribute	Value/s	Description
action	URL Ex: http://www.google.com	A URL that defines where to send the data when the submit button is pushed
method	get post	The HTTP method for sending data to the action URL. Default is get. **method="get":** This method sends the form contents in the URL: URL?name=value&name=value. **Note:** If the form values contains non-ASCII characters or exceeds 100 characters you MUST use method="post". **method="post":** This method sends the form contents in the body of the request. **Note:** Most browsers are unable to bookmark post requests.
name	form_name	Defines a unique name for the form

<input>

The <input> tag defines the start of an input field where the user can enter data. In XHTML the <input> tag must be properly closed.

The following attributes are supported:

Attribute	Value/s	Description
checked	checked	Indicates that the input element should be checked when it first loads. **Note:** Used with type="checkbox" and type="radio"
name	field_name	Defines a unique name for the input element. **Note:** This attribute is required with type="button", type="checkbox", type="file", type="hidden", type="image", type="password", type="text", and type="radio"
type	checkbox hidden password radio reset submit text	Indicates the type of the input element. The default value is "text" **Note:** This is not a required attribute, but we think you should include it. If omitted, IE 5.5 will still display a text field, but Netscape 4.7 will not.
value	value	For buttons, reset buttons and submit buttons: Defines the text on the button. For image buttons: Defines the symbolic result of the field passed to a script. For checkboxes and radio buttons: Defines the result of the input element when clicked. The result is sent to the form's action URL. For hidden, password, and text fields: Defines the default value of the element. **Note:** Cannot be used with type="file" **Note:** This attribute is required with type="checkbox" and type="radio"

Image Tags

The following image tag is supported:

- <img> — Defines an image

The Microbrowser supports images stored in uncompressed **.bmp** format. While all BMP bit depths will be displayed to the best of the phone's ability, it is recommended that the image format most suitable for the target platform be chosen. For example:

- The SoundPoint IP 601 LCD supports four levels of grey, so a 16-color BMP format would be most appropriate.
- The SoundPoint IP 670 LCD supports 12-bit color.

Images can be scrolled up and down, however images that are too wide will be truncated.

Various platforms have differing limits due to memory. There are also differing pixel limits for devices of differing pixel depth. A 1 bit per pixel image 160x80 requires only 1600 bytes. For a 24 bit picture, the memory requirement is 38400 bytes.

There are several limits depending on the source data (this involves the cache limits in configuration) and the display converted data, which is dependant on available RAM (and is limited in the code depending on platform).

The img element defines an image.

Note The "align", "border", "hspace", and "vspace" attributes of the image element are not supported in XHTML 1.0 Strict DTD.

The following attributes are supported:

Attribute	Value/s	Description
src	URL (Ex: "http://www.topxml.com/images/topxml_site.gif" or "c:\images\img1.jpg")	The URL of the image to display
height	Pixels (number, EX: "30") %	Specifies the height of the image in pixel or percent.
width	Pixels (number, EX: "30") %	Specifies the width of the image in pixel or percent.

Table Tags

The following table tags are supported:

- <table>—Defines a table
- <tr>—Defines a table row
- <td>—Defines a table cell
- <tbody>—Defines a table body

Note XHTML tables must be properly formatted (should include <tbody> and </tbody> tags).

<table>

The <table> tag defines a table. Inside a <table> tag you can put table headers, table rows, table cells, and other tables.

The following attributes are supported:

Attribute	Value/s	Description
align	Left center right	Aligns the table. Deprecated. Use styles instead.
border	Pixels (number, EX: “30”)	Specifies the border width. **Tip:** Set border="0" to display tables with no borders!
cellpadding	Pixels (number, EX: “30”) %	Specifies the space between the cell walls and contents
cellspacing	Pixels (number, EX: “30”) %	Specifies the space between cells.
width	% Pixels (number, EX: “30”)	Specifies the width of the table

<tr>

This tag defines a row in a table.

The following attributes are supported:

Attribute	Value/s	Description
align	right left center justify char	Defines the text alignment in cells.

<td>

This tag defines a cell in a table.

The following attributes are supported:

Attribute	Value/s	Description
align	left right center justify char	Specifies the horizontal alignment of cell content
colspan	number	Indicates the number of columns this cell should span.
rowspan	number	Indicates the number of rows this cell should span.

<tbody>

This tag defines a table body. The thead, tfoot and tbody elements enable you to group rows in a table.

The following attributes are supported:

Attribute	Value/s	Description
align	right left center	Defines the text alignment in cells.

Meta Information Tags

The following meta information tags are supported:

- <head> – Defines information about the document

Note Due to space constraints, there isn't a static title bar at the top of the Microbrowser window, as there is in most other browsers. The title is displayed in large bold text in the first line of the page, and is scrolled off the screen as the focus is moved down the page.

<head>

The head element can contain information about the document. The browser does not display the "head information" to the user. The following tag can be in the head section: <title>.

No attributes are supported.

HTTP Support

The Microbrowser is a fully compliant HTTP/1.1 user agent:

- It supports:
  - Cookies

Note Cookies are stored in RAM, therefore they are not are not preserved when the phone reboots or is reconfigured. Cookies are not shared between the idle display Microbrowser and the main Microbrowser.

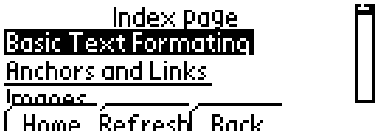
  - Refresh headers
  - HTTP proxies

  - HTTP over SSL/TLS
  - Self-signed or custom CA certificates
- There are the following exceptions:
  - There is no sophisticated caching. The HTML cache refresh META tag is not supported.
  - Any images in the body of a document with the same URL are assumed to be the same image. The image is loaded from the Microbrowser's memory instead of making another request to the server.
  - When a new page is requested, the Microbrowser's internal memory is cleared and all components of the new page are downloaded from the server.

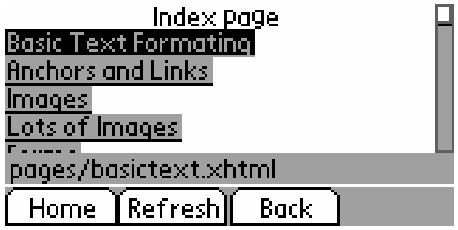
Microbrowser User Interface

Two instances of the Microbrowser may run concurrently:

- An instance with standard interactive user interface

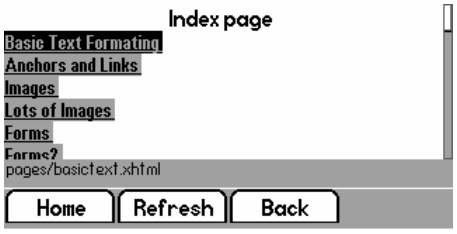


SoundPoint IP 430



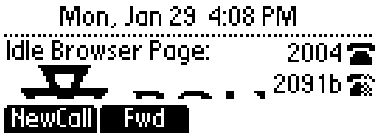
SoundPoint IP 650

SoundPoint IP 501

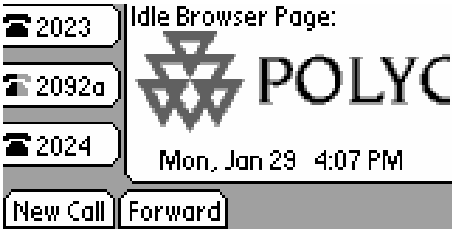


SoundStation IP 4000

- An instance that does not support user input, but appears in a window on the idle display

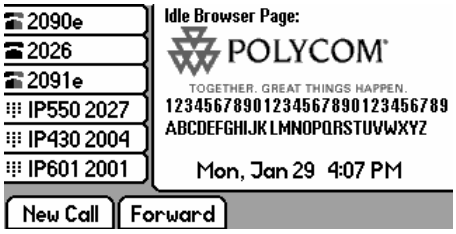


SoundPoint IP 430

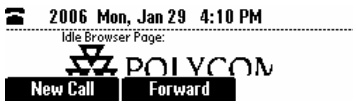


SoundPoint IP 650

SoundPoint IP 501



SoundStation IP 4000



Launching the Microbrowser

The first time the **Applications** key is pressed, the main Microbrowser loads the home page specified in the `mb.main.home` configuration parameter. Subsequent presses of the **Applications** key simply toggle between the Microbrowser and SIP telephony applications. The active page remains loaded in memory when you toggle.

Whenever there is an event in the telephony application that requires the user's attention, the telephony application is brought to the foreground automatically.

The Microbrowser can be displayed again by simply pushing the **Applications** key. While the Microbrowser application is not displayed, it is still active and pending transactions will complete in the background and be immediately visible when the browser is brought to the foreground.

Navigation and Form Editing

The user navigates through pages by moving the focus among the focusable items with the up and down arrow keys. Focusable items include links, form elements, and buttons. The focus moves between all focusable items on a page in the order that they appear in the XHTML source, including tables. For newly displayed pages, the focus will automatically move to the first focusable item visible on the current page.

When the user has focused on a link that they would like to follow, or a form element they would like to toggle, they press the **Select** key. This will either generate a request for the linked page or toggle the selection of an element in the form. When the focus moves to fields which are editable, the user may simply enter text at will, then move the focus to the next selectable item when complete using the up and down arrow keys. If there is a large area of the page without a focusable element, the page is only scrolled by one screen for each push of the arrow key.

To submit form data, navigate to and select a submit button on the page or press the **Submit** soft key when available.

The **Back** soft key takes the user to the previous page viewed. The left arrow key performs a similar function unless the user is editing a text field. The **Refresh** and **Home** soft keys behave in the expected manner, reloading the current page and returning to the user's home page respectively.

Text is entered into text boxes using the dial pad through the same entry method used elsewhere on the phone. When editing text, a soft key allows the user to cycle through uppercase letter, lowercase letter or numeric entry modes. A **Cancel** soft key is available to undo the current edits.

Idle Display Microbrowser

The idle display Microbrowser is independent of the main Microbrowser, but is capable of rendering the same content. Its home page is configured via the `mb.idleDisplay.home` configuration parameter. The idle display Microbrowser does not accept any user input and will only appear when the user has no phone calls in progress and the phone is in the idle user interface state. The idle display Microbrowser can update its content based on a configurable refresh timer or by honoring the value of the Refresh header.

Developing an XHTML Application

Changing Configuration Parameters

Create a new configuration file in the style of **sip.cfg** so that users will connect to your application when they press the **Application** key (or select the **Application** feature item).

Note For more information on why to create another configuration file, refer to the "Configuration File Management on SoundPoint IP Phones" whitepaper at www.polycom.com/support/voice/ .

To allow an application to be run from the Microbrowser:

1. Open a new configuration file in an XML editor.
2. Add the Microbrowser <mb> parameter.
3. Set `mb.proxy` to the address of the desired HTTP proxy to be used by the Microbrowser.

   For example, `mb.proxy=10.11.32.103:8080`

   where 10.11.32.103 is proxy server IP address and 8080 is the port number.
4. Set `mb.idleDisplay.home` to the URL used for Microbrowser idle display home page.

   For example,
   `mb.idleDisplay.home=http://10.11.32.128:8080/sampleapps/idle`
5. Set `mb.idleDisplay.refresh` to the period in seconds between refreshes of the idle display Microbrowser's content.

   For example, `mb.idleDisplay.refresh=10`
6. Set `mb.main.home` to the URL used for Microbrowser home page.

   For example,
   `mb.main.home=http://10.11.32.128:8080/sampleapps/login`
7. Set `mb.limits.nodes` to the maximum number of tags that the XML parser will handle.

   For example, `mb.limits.nodes= 256`
8. Set `mb.limits.cache` to the maximum total size of objects downloaded for each page (both XHTML and images).

   For example, `mb.limits.cache= 200`
9. Save your changes and close the XML editor.
10. Add the new file to the master configuration file's CONFIG_FILES list in the appropriate order.

    Since the files are processed left to right, any parameter which appears in first file will override the same parameter in later files.

For more information on configuration parameters, refer to the *Administrator's Guide for the SoundPoint IP / SoundStation IP Family* at http://www.polycom.com/support/voicedocumentation/.

Sample Applications

This section presents two sample applications that you can use as a starting point for writing your own application.

To develop a static XHTML application:

1. Create a `Sample.xhtml` page with static information to be displayed.

   In this case, the static information will be "Hello World!".

```
<html>
<head>
<title>Sample Application</title>
</head>
<body>
<p>HelloWorld!</p>
</body>
</html>
```

2. Configure the Web server to serve the above XHTML file.

   For example, if you are using Apache Tomcat to try this example, then put this file into the `webapps\PLCM` folder of Tomcat.

3. Configure SoundPoint IP and SoundStation IP phones to point to the XHTML file in the **sip.cfg** configuration file.

   For this example, change `mb.main.home` to `http://<WEBSERVER_ADDRESS:PORT>/PLCM/Sample.xhtml` .

4. Reboot the phones.

5. On a SoundPoint IP phone, press the **Applications** (or **Services**) key.

   The text "Hello World!" appears on the graphic display.

To develop a dynamic XHTML application:

1. Create a `AddStock.xhtml` page.

   This XHTML page is designed for getting a stock symbol as input from the SoundPoint IP or SoundStation IP phone, then retrieve the information for this stock symbol.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<!- - HEADER START - ->
<head>
<title>Stocks</title>
</head>
<!- - HEADER END - ->
<!- - BODY START - ->
<body>
<!- - ADD STOCK FORM START - ->
<form method="POST" action="GetQuote.jsp">
<p>Symbol<input type="text" name="stockname"/>
<input type="submit" value="Get Quote"/></p>
</form>
<!- - ADD STOCK FORM END - ->
</body>
```

```
<!- - BODY END - ->
</html>
```

2. Configure the Web server to serve the above XHTML file.

   For example, if you are using Apache Tomcat to try this example, put this file into the `webapps\PLCM` folder of Tomcat.

3. Write an application that is going to retrieve the stock information from a data service provider.

   For this example, this application will be retrieving stock information from Yahoo and will send it to the Microbrowser. This application is written using a Java Server Page (JSP).

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<%@page
import="java.io.File,java.io.IOException,java.net.URL,java.awt.image.B
ufferedImage,javax.imageio.ImageIO"%>
<html>
<head>
<title>Stock Quote</title>
</head>
<body>
<%
// GETTING THE PATH WHERE BMP FILE HAS TO BE SAVED
String bmpFilePath = application.getRealPath(File.separator) +
"quote.bmp";
// DEFINE URL FROM WHERE CONTENT TO BE RETRIEVED
String stockUrl = "http://ichart.yahoo.com/t?s=";
// RETRIEVE THE STOCK SYMBOL FROM REQUEST
String stockSymbol = "PLCM"; // DEFAULT TO PLCM
if ( request.getParameter("stockname") != null ) {
stockSymbol = request.getParameter("stockname");
}
readAndConvertContentToBmp(stockUrl + stockSymbol, bmpFilePath,
stockSymbol);
%>
<%!
// READ THE CONTENT FROM GIVEN URL AND THEN CONVERT THE CONTENT TO A
BMP FILE
private void readAndConvertContentToBmp(String a_stockUrl, String
a_filePath, String a_name) throws IOException {
try {
BufferedImage stockImage = ImageIO.read(new URL(a_stockUrl));
ImageIO.write(stockImage, "bmp", new File(a_filePath));
}
catch (IOException ex) {   throw ex;}
}
%>
```

```
<!-- START DISPLAY BMP FILE -->
<img src="quote.bmp"/>
<!-- END DISPLAY BMP FILE -->
</body>
</html>
```

4. Configure the Web server to deploy the above JSP file.

   For example, if you are using Apache Tomcat to try this example, put this file into the `webapps\PLCM` folder of Tomcat.

5. Configure SoundPoint IP and SoundStation IP phones to point to the XHTML file in the **sip.cfg** configuration file.

   For this example, change `mb.main.home` to `http://<WEBSERVER_ADDRESS:PORT>/PLCM/AddStock.xhtml` .

6. Reboot the phones.

7. On a SoundPoint IP phone, press the **Applications** (or **Services**) key.

   The `AddStock.xhtml` appears on the graphic display.

8. Enter a stock symbol, then select the **Get Quote** soft key.

   The stock quote for the entered stock symbol appears on the graphic display.

Note

Static and dynamic XHTML applications can be developed using any Web server. Even though Tomcat is used in the example, the developer is free to use any Web server.

Dynamic XHTML applications can be developed using any Web technologies—for example, ASP.net, Java Servlets, Java Server Pages, CGI-PERL, and PHP.

3

Troubleshooting

This chapter presents problems, likely causes, and corrective actions. Problems are grouped as follows:

If you still need assistance, contact your system administrator.

XML Errors

Symptom	Problem	Corrective Action
Improperly formatted tables could cause the phone to stop and restart or display the error “XML Error (17,75) mismatched tag”.	A table tag was improperly formatted.	Correct the improperly formatted table.

A

Appendix

This appendix provides information on unsupported XHTML elements.

Unsupported XHTML Elements

The unsupported elements and attributes are:

Tag Type	Tag Description
Basic Tags	<html>—Defines HTML document.
	<body>—Defines documents' body.
	<h1> to <h6>—Defines header 1 to header 6.
	<p>—Defines a paragraph.
	 —Inserts a single line break.
	<hr>—Defines a horizontal rule.
Character Format Tags	<b>—Defines bold text.
	<font>—Deprecated. Defines text font, size, and color.
	<i>—Defines italic text..
	—Defines emphasized text.
	<big>—Defines big text.
	—Defines strong text.
	<small>—Defines small text.
	<sup>—Defines superscripted text.
	<sub>—Defines subscripted text.
	<bdo>—Defines the direction of text display.
	<u>—Deprecated. Defines underlined text.

Tag Type	Tag Description
Output Tags	<pre>—Defines preformatted text.
	<code>—Defines computer code text.
	<tt>—Defines teletype text.
	<kbd>—Defines keyboard text.
	<var>—Defines a variable.
	<dfn>—Defines a definition term.
	<samp>—Defines sample computer code.
	<xmp>—Deprecated. Defines preformatted text.
Block Tags	<acronym>—Defines an acronym.
	<abbr>—Defines an abbreviation.
	<address>—Defines an address element.
	<blockquote>—Defines a long quotation.
	—Deprecated. Defines centered text.
	<q>—Defines a short quotation.
	<cite>—Defines a citation.
	<ins>—Defines inserted text.
	<del>—Defines deleted text.
	<s>—Deprecated. Defines strikethrough text.
	<strike>—Deprecated. Defines strikethrough text.
Link Tags	<a>—Defines an anchor. The following attributes are not supported: charset, coords, hreflang, rel, rev, shape, target, type, id, class, title, style, dir, lang, xml:lang, tabindex, and accesskey.
	<link>—Defines a resource reference.
Frame Tags	<frame>—Defines a sub window (frame).
	<frameset>—Defines a set of frames.
	<noframes>—Defines a noframe section.
	<iframe>—Defines an inline sub window (frame).

Tag Type	Tag Description
Input Tags	<form>—Defines a form. The following attributes are not supported: accept, accept charset, enctype, target, class, id, style, title, dir, lang, and accesskey.
	<input>—Defines an input field. The following attributes are not supported: accept, align, alt, disabled, maxlength, readonly, size, arc, type:button, type:file, type:image, class, is, style, title, dir, lang, accesskey.
	<textarea>—Defines a text area.
	<button>—Defines a push button.
	<select>—Defines a selectable list.
	<optgroup>—Defines an option group.
	<option>—Defines an item in a list box.
	<label>—Defines a label for a form control.
	<fieldset>—Defines a fieldset.
	<legend>—Defines a title in a fieldset.
	<isindex>—Deprecated. Defines a single-line input field.
List Tags	—Defines an unordered list.
	—Defines an ordered list.
	—Defines a list item.
	<dir>—Deprecated. Defines a directory list.
	<dl>—Defines a definition list.
	<dt>—Defines a definition term.
	<dd>—Defines a definition description.
	<menu>—Deprecated. Defines a menu list.
Image Tags	<img>-Defines an image. The following attributes are not supported: alt, align, border, hspace, ismap, longdesc, usemap, vspace, id, class, title, style, xml:lang, and lang
	<map>—Defines an image map.
	<area>—Defines an area inside an image map.

Tag Type	Tag Description
Table Tags	<table>—Defines a table. The following atrributes are not supported: bgcolor, frame, rules, summary, id, class, title, style, dir, lang, and xml:lang.
	<caption>—Defines a table caption.
	<col>—Defines attributes for table columns.
	<th>—Defines a table header.
	<tr>—Defines a table row. The following attributes are not supported: bgcolor, cahr, charoff, valign, id, class, title, style, dir, lang, and xml:lang.
	<td>—Defines a table cell. The following attributes are not supported: abbr, axis, bgcolor, char, charoff, headers, height, nowrap, scope, valign, width, id, class, title, style, dir, lang, and xml:lang.
	<thead>—Defines a table header.
	<tbody>—Defines a table body. The following attributes are not supported: align:justify, align:char, char, charoff, valign, id, class, title, style, dir, lang, and xml:lang.
	<tfoot>—Defines a table footer.
	<colgroup>—Defines groups of table columns.
Style Tags	<style>—Defines a style definition.
	—Defines a section in a document.
	—Defines a section in a document.
Meta Information Tags	<head>—Defines information about the document. No attributes are supported.
	<title>—Defines the document title.
	<meta>—Defines meta information
	<base>—Defines a base URL for all the links in a page
	<basefont>—Deprecated. Defines a base font

Tag Type	Tag Description
Programming Tags	<script>—Defines a script
	<noscript>—Defines a noscript section
	<applet>—Deprecated. Defines an applet
	<object>—Defines an embedded object
	<param>—Defines a parameter for an object

Index

X