Deliver XHTML applications to mobile devices

Skill Level: Introductory

Naveen Balani
Author

18 Mar 2003

In this tutorial, you'll learn how to deliver XHTML applications to XHTML-enabled mobile devices. We will develop a sample XHTML mobile application that posts requests to a Java servlet and receives responses in XHTML format. We'll deploy and test the application using the Nokia Mobile Internet Toolkit. Developers who want to extend their Internet applications to wireless devices will want to read on and see how it's done.

Section 1. Introduction

About this tutorial

In this tutorial, you'll learn how to deliver XHTML applications to XHTML-enabled mobile devices. We will develop a sample XHTML mobile application that posts requests to a Java servlet and receives responses in XHTML format. We'll deploy and test the application using the Nokia Mobile Internet Toolkit. Developers who want to extend their Internet applications to wireless devices will want to read on and see how it's done.

Prerequisites

Before you start this tutorial, you should have a good working knowledge of HTML and Java servlets. You'll also need the following software packages installed on your system:

- The Nokia Mobile Internet Toolkit 3.1, available from Forum Nokia.
- JSDK 1.4, available from Java.sun.com.
- Tomcat version 4.1 or later, available from the Apache Foundation.
- The sample code that accompanies this article.

We've assumed that you have installed and configured all of the above software in a Windows 2000 environment.

Section 2. Getting started with XHTML

What is XHTML?

XHTML stands for *Extensible Hypertext Markup Language;* it's a cross between HTML and XML. XHTML provides a stricter standard for making Web pages, reducing incompatibilities between browsers. Thus, XHTML can be used on a variety of different devices without changes.

The great thing about XHTML is that it is almost identical to HTML. With XHTML, however, it is much more important to create your code correctly. Badly formed HTML code is not XHTML-compatible.

XHTML is a Web standard that has been approved by the World Wide Web Consortium (W3C; see Resources for a link to the spec). With its backward compatibility, it will replace HTML 4.x in the near future.

XHTML Basic: A mobile version of XHTML

XHTML Basic is a simpler version of XHTML 1.0. It allows for the delivery of Web content to small, non-PC devices, such as mobile phones, PDAs, pagers, and television-based Web browsers.

XHTML takes the tags and syntax from the latest version of HTML (4.1) and adds modularity and enforces strict adherence to language rules. XHTML provides better content presentation abilities for the smaller screens of wireless devices.

WAP 2.0, the next-generation WAP standard, uses XHTML Basic as its markup language, replacing the Wireless Markup Language (WML) used in earlier versions of WAP. The Open Mobile Alliance (formerly the WAP Forum; see Resources for a

link) and the W3C have recommended XHTML as the standard for all Web development for desktops as well as handsets and other mobile devices.

Why use XHTML Basic for delivery of mobile content?

There are a number of advantages to using XHTML, rather than WML or another scripting language, to deliver mobile content:

- **Transformation into alternate forms:** Because XHTML is an application of XML, you can use can use Extensible Stylesheet Language Transformations (XSLT) for automatic parsing and transcoding of content. XSLT can transform content from one form of XML into another -- from XHTML to HTML, for instance -- thereby eliminating the need to develop different formats for the same content.
- **Segregation of content and presentation:** XHTML allows for a clean separation between content and its presentation. This is made possible by Cascading Stylesheets (CSS), which allow you to specify the presentation of applications. You can apply a specific stylesheet for each of your target devices. Layout and presentation can thus be changed without modifying the basic content.
- **Consistent look and feel:** Many types of wireless devices exist, each with its own display, memory, and processing capabilities. By using well-formed documents that adhere to XML's strict syntax rules, you ensure that your content will be rendered consistently on different types of devices.

XHTML's support for CSS and adherence to XML's syntax makes it an ideal format for creating wired and wireless Web content.

Section 3. XHTML Basic tags

Introduction

XHTML takes the tags and syntax from the latest version of HTML (4.1). Thus, it's easy to adapt to XHTML. In this section, we will briefly examine the tags that we will use to develop our application.

Header tags

XML declaration: XHTML Basic is an XML language, so every document must begin with the following XML declaration:

```
<?xml
version="1.0"?>
```

DOCTYPE element: This element follows the XML declaration; it specifies the Document Type Definition (DTD) to be used in processing the document. The existence of this element implies that the document conforms to the rules contained in the specified DTD and is thus a strictly conforming document. Such a document can only use those elements and attributes specified in the DTD, and must only use them in the manner specified. Here's an example:

```
<!DOCTYPE html PUBLIC "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
```

Within the DOCTYPE element, the first quoted string is the public name; the second is a URL specifying the location of the DTD.

Body and form elements

An XHTML document *must* contain one <body> element. All text in XHTML Basic must be enclosed by <p> elements or other XHTML tags such as headers, lists, tables, etc.

In addition, there are a number of important form elements that we'll be using in our application:

Anchor tags: <anchor> or <a> elements define a hypertext link to another resource (e.g, a document or an image) or to another location in the same document (e.g., a fragment anchor). On mobile devices, you can also use these tags to place a phone call or create an e-mail message. To do so, specify a value of an `href` attribute that invokes a corresponding action or links to a record in some application.

Anchored links are widely used on the Web and are equally efficient for mobile devices. They represent the best way to control service navigation.

Forms: Forms are used to handle the input given to `input` or `textarea` elements and also to handle selections made in lists.

XHTML forms are posted to a server using a Submit button, unlike WML, which uses anchors containing `postfield` elements to perform this task. You can use CSS in

forms to position elements via padding and margin properties.

The `<form>` element defines the action and method used to send the form data to the Web server. The `action` attribute specifies the URL on the server where the form data should be sent; the `method` attribute can have a value of either `get` or `post`, indicating how the data is sent.

Input elements: Use these elements to provide the user with a means to enter short text, check boxes to select multiple values, or choose a radio button to select a single value.

Submit button: Use the `<input type="submit"/>` element to create a button that submits the selections or entries made in a form.

For a description of all the XHTML tags available to you, see the links in Resources. Now, let's deploy our sample XHTML application!

Section 4. Building and deploying an XHTML application

Building the sample XHTML application

Download the zip file containing this tutorial's sample code from Resources and extract it into a directory. For our discussion, we'll call this directory `C:\xhtmlbasic`.

The file `feeback.xhtml` in `C:\xhtmlbasic\xhtml` is the XHTML file for our sample application. You'll want to open this code in a text editor and keep it handy for reference as we run the application so that you can better understand what's going on under the covers. It represents a typical feedback form, which takes feedback information from the user and sends it to a server. In our case, the feedback information is posted to a Java servlet, which parses the request information and gives back a confirmation response to the mobile browser. The file `XHTMLServlet.java`, located in `C:\xhtmlbasic\src\test`, is the source code for our servlet.

Deploying the servlet in Tomcat

Copy the `xhtmlwebapp` folder from `C:\xhtmlbasic` to `<jakarta-installation-directory>\webapps`. This folder represents the Web application that we will deploy in Tomcat.

Before running the application, you must configure the Tomcat server to accept the MIME type `application/xhtml+xml` and the file extension `xhtml`. To do so, add the following entry to the `web.xml` file located in the `<jakarta-installation-directory>\conf` directory:

```
<mime-mapping>
  <extension>xhtml</extension>
    <mime-type> application/xhtml+xml </mime-type>
</mime-mapping>
```

Next, start the Tomcat server by running the `startup.bat` file in `<jakarta-installation-directory>\bib`.

By default, the Tomcat server listens on port 8080 for HTTP connections, and our sample XHTML application assumes such a default configuration. You may have to make changes to match your particular system's configuration.

Section 5. Running the XHTML application

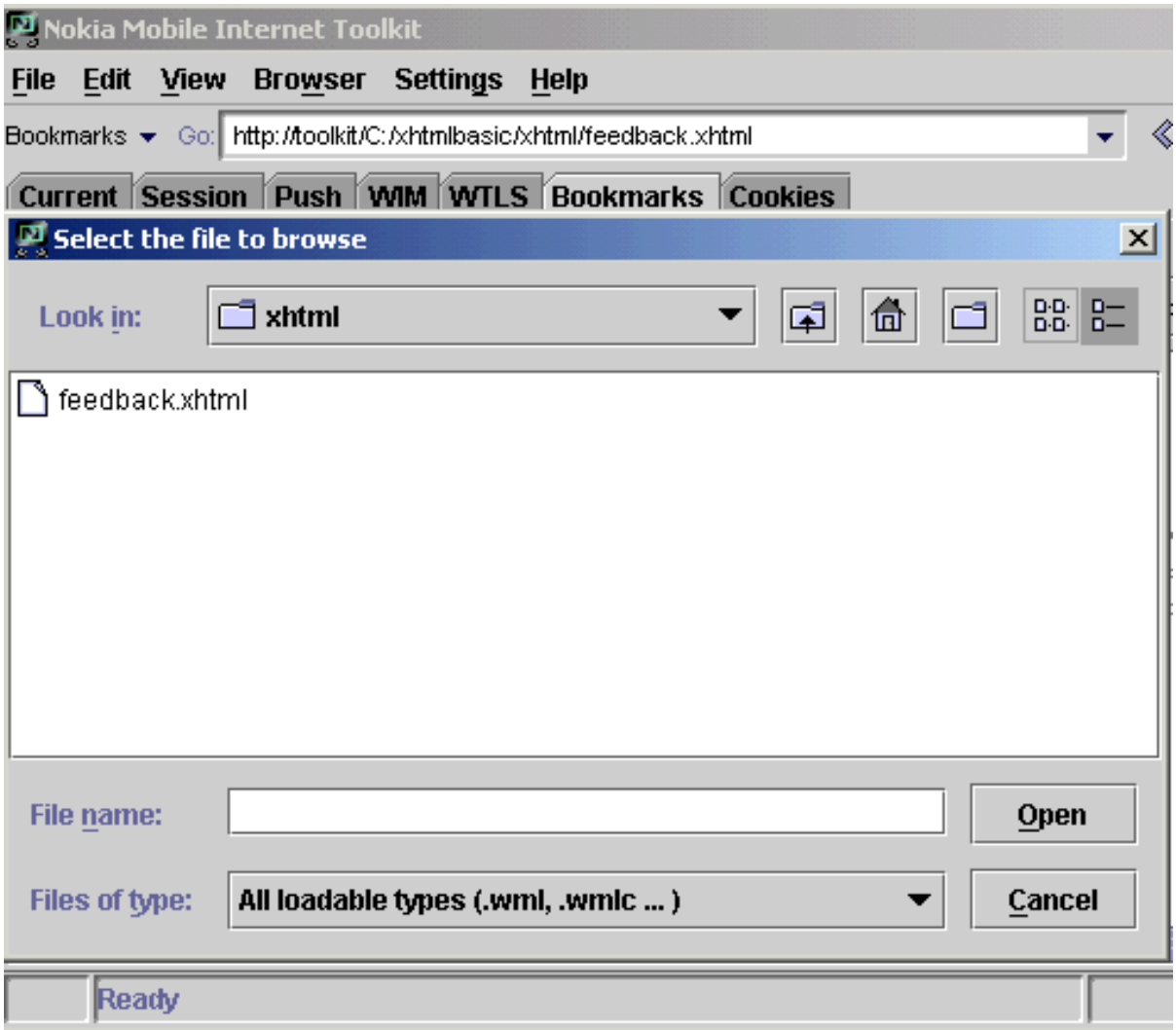
The Nokia Mobile Internet Toolkit

The Nokia Mobile Internet Toolkit is a suite of tools that facilitate the development of applications for the mobile Internet. They include a set of editors for creating content of the various types supported by mobile browsers, as well as the debugging, logging, and testing facilities required for true application development.

The toolkit provides one emulator, which provides a graphical interface that simulates the display screen, keys, and controls of a mobile device.

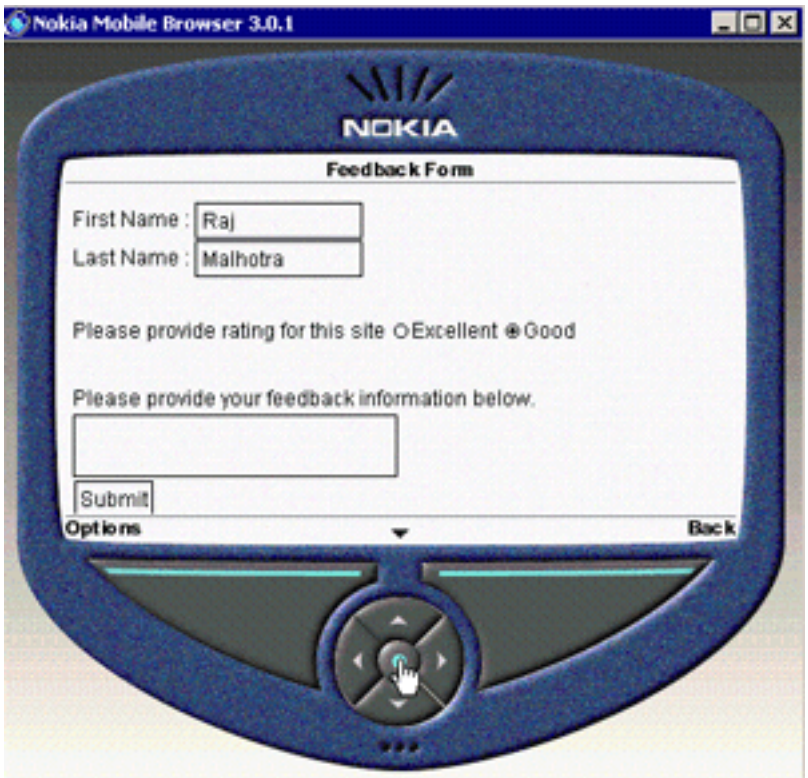
Launching the application

To run our sample XHTML application, we'll need to load our XHTML file into the Nokia Browser. To do this, open the Mobile Internet Toolkit 3.1 application, select Browser => Load File from the menu, and select `feeback.xhtml`, as shown in the figure below.



The application in action

In the Nokia browser, you should now see our XHTML feedback application, as shown in the figure below:



Enter the appropriate data. You can navigate through the form by using the emulated device's arrow keys. Click on the Submit button to submit the form.

Processing the input

Once you click Submit, the application posts the information to `XHTMLServlet`, which parses the request and displays the user-entered information back to the mobile browser, as shown in the figure below:

Section 6. Wrap up

Summary

In this tutorial, we have successfully delivered a sample XHTML application to an XHTML-compatible mobile device. This should offer you some direction on how you can deliver any complex application to mobile devices. For instance, Web-based applications developed for a PC environment, such as banking applications, can be easily ported to mobile devices using XHTML.

However, there's one important thing you need to keep in mind when building such applications: Mobile users are not browsers. They are often paying per-minute charges, and they want instant and easy access to specific information.

Downloads

Description	Name	Size	Download method
Code sample	wi-xhtml-mob.zip	6KB	HTTP

Information about download methods

Resources

Learn

- Visit Java.sun.com to find the latest version of the Java development kit and J2ME for all platforms.
- The W3C's XHTML site includes the latest information on XHTML for mobile applications.
- Check out the latest WAP specs at the Open Mobile Alliance (formerly the WAP Forum).
- Stay current with developerWorks technical events and Webcasts.

Get products and technologies

- Build your next development project with IBM trial software, available for download directly from developerWorks.

Discuss

- Participate in developerWorks blogs and get involved in the developerWorks community.

About the author

Naveen Balani
Naveen Balani spends most of his time designing and developing J2EE-based products. He has written various articles and tutorials for IBM developerWorks in the past, covering such topics as Web services, MQSeries, WebSphere Studio, Java wireless devices, DB2 Everyplace for Palm, Java on EPOC, and wireless data synchronization. You can reach him at naveenbalani@rediffmail.com.