Published on **The O'Reilly Network** (http://www.oreillynet.com/)
http://www.oreillynet.com/pub/a/wireless/2004/02/06/mobile_browsing.html
See this if you're having trouble printing code examples

Creating Web Content for Mobile Phone Browsers, Part 1

by Robert Jones
02/06/2004

After a slow start, browsing the Internet from mobile phones is finally taking off. The latest batch of phones have helped the cause considerably. People are buying them for the cameras and other bells and whistles, but the side effect is that mobile Internet access is now in the hands of a huge number of people -- and they are using it. Users in the UK alone downloaded 8 billion pages to their phones in 2003. A huge audience; but bear in mind that most of them are viewing sports scores or downloading custom ring tones, and not a lot else. The opportunity for developers like us to serve up imaginative and useful content to their phones is tremendous.

But the challenges facing someone entering the field are significant; it's not just the issue of learning a new markup language or two. To develop successful content, you need to understand the technical limitations of mobile browsers, the diversity of hardware that is out there, and the practical difficulties and frustrations faced by users trying navigate from a phone keypad.

That diversity is the big issue. If you want to reach a broad audience, then your pages need to work on the tiny monochrome screen of a two-year-old phone as well as the higher-resolution, color screen of the newest models. There are two ways to deal with this. You can limit yourself to the most basic markup so that a page is viewable on any phone, or you can identify the capabilities of each phone as it makes a request and deliver rich content specific to that device.

This series of articles will cover both approaches. In Part 1, I will describe the basic features of the two main markup languages, WML and XHTML MP, and will show how you serve them from a web server. In Part 2, I will show how you can identify the phone that has requested a page and deliver content that is tailored for that device.

WAP, WML, and XHTML MP

Viewing the Web on a mobile phone involves compromises -- a lot of them. The keyboards are numeric and cramped. Connections can be unreliable, with low bandwidth and high latency. Most importantly, screens are small and often monochrome. So the powers that be came up with Wireless Markup Language (WML) as a first crack at the problem.

WML content is served by a regular HTTP server like Apache, but its delivery mechanism involves a gateway device that sits between the phone and the server. This mechanism is referred to as the Wireless Application Protocol (WAP) and specifically, as WAP 1.0. When a mobile phone submits a page request, it gets sent to the WAP gateway that acts like a proxy. The gateway fetches the WML page from the regular web server. It then binary-encodes the content and passes that form of the data on to the phone. This was originally done to make the most of the limited bandwidth available to the phone, as well as to provide the phone company with a way of monitoring usage. The initial release, WML 1.0, only allows for monochrome screens and has very limited page layout features.

Working within the limitations of a technology is a necessary evil, but given the rapid evolution of phones, it does seem short sighted to have built a language and delivery mechanism that is so tied to those limitations. WML has evolved over time, but not as fast as the hardware. So it's not surprising that it has been superceded by something new in the form of XHTML MP (Mobile Profile). XHTML is the extension of HTML that enforces the strict syntax checking of XML. And the Mobile Profile variant is a subset that addresses some of the constraints of mobile browsers.

Bill Kennedy

Table of Contents
Index
Sample Chapter

Read Online--Safari
Search this book on Safari:

Go
Only This Book
Code Fragments only

The good thing about the restricted nature of WML is that any content written in it should be viewable on the majority of mobile browsers. XHTML MP does away with those restrictions, allowing you to write pretty much anything you want. But with freedom comes responsibility: the onus has shifted squarely onto you to make sure it looks good on a mobile browser. With the diversity of phones currently on the market, that's easier said than done.

Along with the new markup language comes WAP 2.0, an evolution of the delivery mechanism. It still involves a WAP gateway, but the XHTML MP content is passed straight through to the phone without binary encoding. The role of the gateway is now reduced to tracking usage of the system by users, which is very important for the phone company but irrelevant to the rest of us.

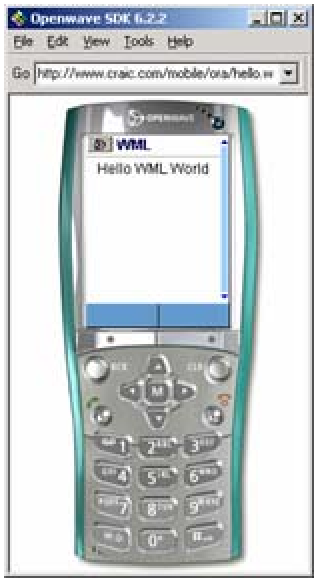
It would be great if we could ditch WML and move directly to XHTML MP. But life is never that simple. There are still a lot of phones out there that can only handle WML. Because of that, most of the major mobile web sites are still using WML. That, in turn, requires that new phones need to handle both WML and XHTML MP. This cycle of dependency will be broken eventually, but for now we need to consider providing mobile content in both markup languages.

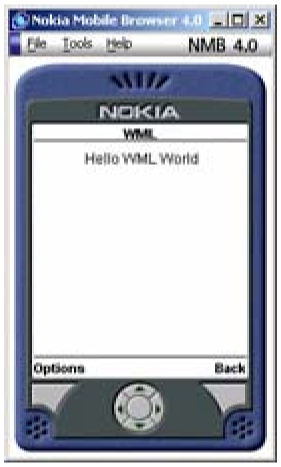
I'll illustrate them both with an example web site for a fictitious restaurant, which should serve as a starting point for your own experiments. But before we get into that, I want make highlight two things that will make your life a lot easier.

First of all, both WML and XHTML MP are XML-based languages that have significant implications for those of us coming from the sloppy world of HTML. You must use double quotes around each attribute in a tag (such as `align="center"`) and you must match each tag with a closing tag (such as `<p>...</p>`). That is, except for the tags that don't follow that rule, most notably `<br/>` and `<img.../>`. You have to write perfectly clean code or it will not be displayed on your phone. Trust me, this gets very frustrating very quickly.

Secondly, you should install a WAP browser simulator on your PC. Simulators are essential tools for anyone developing content for phones. Running on your PC over a regular Internet connection, they load pages much faster than your phone, they avoid the frustrations of using the phone keypad, and they can provide information that helps debug problems. In fact, they are so useful that I recommend you install two of them!

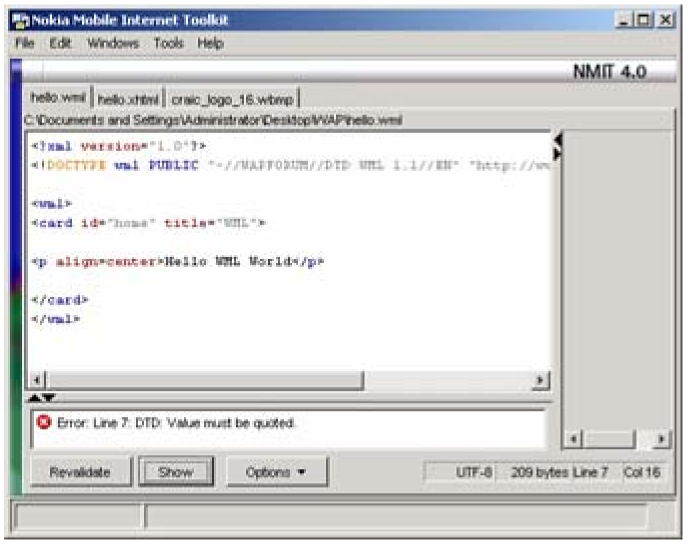
The first, and easiest to install, is from Openwave and can be found at developer.openwave.com. It is available for Windows only and is free of charge, but you do need to register with Openwave. The equivalent software from Nokia is available from forum.nokia.com. Drill down through Tools and SDKs until you find the page for Nokia Mobile Internet Toolkit 4.0. Again, the software is free, but you will need to register. You should download the entire toolkit, which includes a Mobile Browser Simulator (NMB), a WAP Gateway Simulator, and the toolkit itself (NMIT), which is an application containing editors for the markup languages, images, and multimedia messages. The simulators look like this :


Openwave SDK 6.2.2
File Edit View Tools Help
Go http://www.craic.com/mobile/ora/hello.w
WML
Hello WML World



The Openwave simulator is very simple to use; just type in a URL and it retrieves the page directly. It simulates the entire phone, allowing you to access all of the keys available to a user. The Nokia simulator requires that you also run their WAP gateway simulator, but that is easy enough. Although it does not include the full phone keypad, it does allow you flexibility in setting the screen size, and the display mode can be set to color, grayscale, or monochrome.

The real benefit of installing the Nokia software is the NMIT, which provides "language-aware" editors for WML and XHTML MP. This is extremely useful in troubleshooting pages that do not display on your phone. Chances are the problem is XML-related, such as a failure to properly quote an attribute. This screenshot shows an example of that:



C:\Documents and Settings\Administrator\Desktop\WAP\hello.wml

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" "http://ww

<wml>
<card id="home" title="WML">

<p align=center>Hello WML World</p>

</card>
</wml>
```

```
Error: Line 7: DTD: Value must be quoted.
```

UTF-8 209 bytes Line 7 Col 16

This capability has been a lifesaver. Writing perfectly valid XML by hand does not come easily to me; I've been writing sloppy, but perfectly functional, HTML for far too long. I suspect there are many others out there like me. The problem is that when I attempt to view a page with bad XML, my phone beeps and gives me the message "Services: Reply Unknown." This is not very helpful. The simulators should be a solution, but they are more tolerant of certain types of XML errors than my phone, so they may display the page just fine -- also not helpful. By loading the page into the Nokia editor, I can see immediately where the problem lies. My advice is to install both of these tools and use them routinely. Hand-coding pages in Emacs and hoping they work is a recipe for frustration.

An Example of a WML Web Site

Let's look at a real code in the form of a simple web site for a restaurant. It will introduce multiple pages, links between them, and images. Pretty basic stuff, but with some issues that are unique to mobile browsers. XHTML MP is straightforward, but WML has its own way of doing things, so we'll spend a bit more time on that.

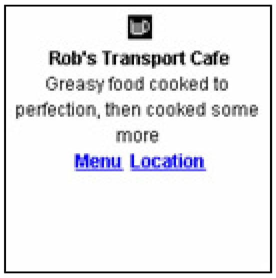
cafe.wml is a WML page consisting of three *cards*: the first represents the home page, the second is the current menu, and the third is the location of the restaurant. The concept of multiple cards on a single page is unique to WML. The idea was to conserve bandwidth by combining a group of related pages into a deck that can be downloaded as a single block, thereby avoiding multiple requests and responses. It works quite well, but it makes for ugly code. We link between cards using anchor (`<a>`) tags, where the target is the ID of the destination card preceded by a hash character.

cafe.wml

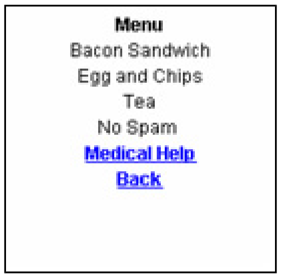
```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
           "http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>

<card id="card1" title="Cafe">
<do type="options" label="Menu">
   <go href="#card2" />
</do>

<do type="accept" label="Location">
   <go href="#card3" />
</do>

<p align="center">
   <img src="cafe.wbmp" alt="cafe"/><br/>
   <strong>Rob's Transport Cafe</strong>
</p>

<p align="center">
   Greasy food cooked to perfection, then cooked some more
</p>

<p align="center">
   <a href="#card2">Menu</a>
   <a href="#card3">Location</a>
</p>

</card>


<card id="card2" title="Menu">
<do type="accept" label="Back">
   <go href="#card1" />
</do>

<p align="center"><strong>Menu</strong></p>

<p align="center">
Bacon Sandwich<br/>
Egg and Chips<br/>
Tea<br/>
No Spam
</p>

<p align="center">
   <a href="http://www.google.com/wml?q=indigestion">Medical Help</a>
</p>

<p align="center"><a href="#card1">Back</a></p>
```

```
</card>


<card id="card3" title="Location">
<do type="accept" label="Back">
   <go href="#card1" />
</do>

<p align="center"><strong>Location</strong></p>

<p align="center">911 East Pike St.<br/>Seattle<br/>WA 98122</p>

<p align="center">
   <a href="mailto:bogus@craic.com">Email Us</a>
</p>

<p align="center">Phone:
   <a href="wtai://wp/mc;2065551212">206-555-1212</a><br/>
   <a href="wtai://wp/ap;2065551212;Rob's_Cafe">Add to Your Contacts</a>
</p>

<p align="center"><a href="#card1">Back</a></p>
</card>

</wml>
```

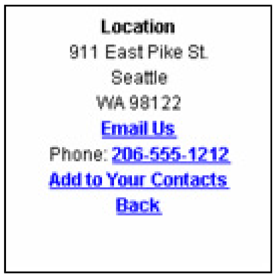
You can find this deck at www.craic.com/mobile/ora/cafe.wml. The three pages should look something like this:



Rob's Transport Cafe
Greasy food cooked to perfection, then cooked some more
Menu Location



Menu
Bacon Sandwich
Egg and Chips
Tea
No Spam
Medical Help
Back



Location
911 East Pike St.
Seattle
WA 98122
Email Us
Phone: 206-555-1212
Add to Your Contacts
Back

Our WML page also includes an image. It should not come as too much of a surprise to learn that images in WML use a special format, called WBMP. This is a very simple format was originally intended as a way to display images on early monochrome displays while minimizing bandwidth consumption. They are really only useful for icons or your company logo. The Nokia Mobile Internet Toolkit includes a useful WBMP editor, and some graphics software, such as Macromedia Fireworks, will generate the format.

Let's walk through this example. The `<xml>` and `<!DOCTYPE>` lines are required XML headers. The `<wml>` ... `</wml>` tags enclose the rest of the page. Within this block there are one or more `<card>` ... `</card>` blocks -- our example has three. Card blocks are a concept unique to WML and let us hope it stays that way. The idea is that each WML file contains a "deck" of related cards, with each one the equivalent of a single HTML page. Packing them into a single file means you can download a set of related pages in a single WAP transaction, thereby conserving the limited bandwidth available. That made a lot of sense in the early days of WAP, but much less so these days.

Each card has an `id`, which we will use for navigation, and a `title`. Within each card, you have the basic formatting tags available to HTML: `<p>`, `<a>`, `<i>`, etc. Think really simple HTML. Early phone browsers had monochrome screens with lousy resolution, making creative page design irrelevant. You should treat WML page design as a way to get your information across without any fancy formatting. If design is important, then you want to work in XHTML MP.

Navigation is a critical design issue for mobile browsing. Keying in a long URL on a phone keypad is a real pain. It is essential that you include convenient navigation links in each of your pages. The safe choice is to use the familiar `<a>` ... `</a>` tag pair, using card `id`s as targets, preceded by a `#` character. The problem is that links take up a lot of valuable screen real estate.

The `<do>...</do>` blocks provide an alternative way to navigate. All phones have two programmable keys immediately below the screen. The left one is called the Options key and the right one is the Accept key. Within a `<do>` block, the `type` attribute determines which key we want to program. The `label` attribute defines the text that we want to appear on the screen directly above that key. The `<go>` tag within the block defines the URL or card that you jump to when that key is pressed.

This mechanism is a great idea -- in principle. One thing you learn very quickly in this field is that not all phones implement all of the features available in the markup languages. For example, `<do>` blocks do not work as expected on the Nokia 3650. The keys are not reprogrammed, but you can access the functions by clicking Options, then Service options, and selecting the appropriate function. But that series of key clicks defeats the whole purpose of the `<do>` block! On the Openwave simulator, the keys are programmed but they are not evident until you click through all of the other `<a>` links on that card. Again, this defeats the purpose of the shortcut.

The bottom line is that you cannot assume that any "advanced" feature of WML will work on any given phone. As a result, you **must** include regular `<a>` links on every card.

Enough of these frustrations (at least for a moment) -- let's do something unique to mobile phone browsers. Look at the two `<a>` tag pairs in the last card with the `wtai` targets. These are two calls to the Wireless Telephony Application Interface (WTAI), a library of functions that link web pages to the primary functions of your phone. You can initiate a call to the number specified in the tag, you can send a series of DTMF tones down the line (to access your voicemail, for example), and you can add the phone number to your list of contacts.

In our example page, the first tag, with the target `wtai//wp/mc;2065551212`, will initiate a call to that phone number when clicked. This is extremely useful and, importantly, it works on all the phones that I have tried. The second tag pair, with the target `wtai//wp/ap;2065551212;Rob's Cafe`, will add the specified phone number and name to your phone's contact list. This would be much more useful if it could download a vCard record, but it's still pretty handy.

Built in functions that tie the Web directly to the capabilities of your phone will be of great importance in this market. Indeed, there is a tremendous amount of work going on in the field of multimedia messaging, but that seems to be at the expense of simple, useful functions. Being able to access a sophisticated hardware feature simply by embedding a tag in your web page is a very powerful feature.

WML has several other important features that you might want to look at, such as WMLScript and WAP CSS. But we are going to leave WML here and jump to the next level in the evolution of this field.

The XHTML MP Version of the Site

In XHTML MP, we need three separate pages instead of the deck of WML cards. The page source for *cafe.xhtml*, *cafe_menu.xhtml* and *cafe_location.xhtml* looks pretty much like regular HTML.

cafe.xhtml

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
        "http://www.wapforum.org/DTD/xhtml-mobile10.dtd" >

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
   <title>Cafe</title>
</head>

<body>

<p align="center" style="color: red;">
   <img src="cafe.gif" alt="cafe"/><br/>
   <b>Rob's Transport Cafe</b>
</p>
```

```
<p align="center">
   Greasy food cooked to perfection, then cooked some more
</p>

<p align="center">
   <a href="cafe_menu.xhtml" accesskey="1" title="Menu">Menu</a>

   <a href="cafe_location.xhtml" accesskey="2" title="Location">Location</a>
</p>

</body>
</html>
```

cafe_menu.xhtml

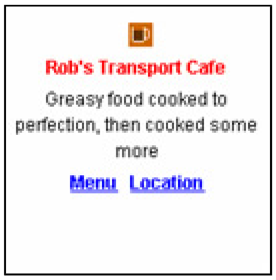
```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
        "http://www.wapforum.org/DTD/xhtml-mobile10.dtd" >

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
   <title>Cafe - Menu</title>
</head>

<body>

<p align="center"><b>Menu</b></p>

<p align="center">
Bacon Sandwich<br/>
Egg and Chips<br/>
Tea<br/>
No Spam
</p>

<p align="center">
   <a href="http://www.google.com/wml?q=indigestion">Medical Help</a>
</p>

<p align="center">
   <a href="cafe.xhtml">Back</a>
</p>

</body>
</html>
```

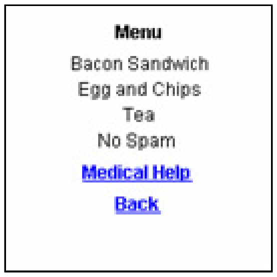
cafe_location.xhtml

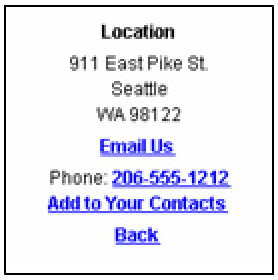
```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
        "http://www.wapforum.org/DTD/xhtml-mobile10.dtd" >

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
   <title>Cafe - Location</title>
</head>

<body>

<p align="center"><b>Location</b></p>
```

```
<p align="center">911 East Pike St.<br/>Seattle<br/>WA 98122</p>

<p align="center">
   <a href="mailto:bogus@craic.com">Email Us</a>
</p>

<p align="center">Phone:
   <a href="wtai://wp/mc;2065551212">206-555-1212</a><br/>
   <a href="wtai://wp/ap;2065551212;Rob's_Cafe">Add to Your Contacts</a>
</p>

<p align="center">
   <a href="cafe.xhtml">Back</a>
</p>

</body>

</html>
```

You can find these pages at www.craic.com/mobile/ora/cafe.xhtml and they should look something like this :


Rob's Transport Cafe
Greasy food cooked to perfection, then cooked some more
Menu Location


Menu
Bacon Sandwich
Egg and Chips
Tea
No Spam
Medical Help
Back


Location
911 East Pike St.
Seattle
WA 98122
Email Us
Phone: 206-555-1212
Add to Your Contacts
Back

XHTML MP includes pretty much every feature that we are used to in HTML. The temptation, therefore, is to start using tables for page layout, transparent images for spacing, and stylesheets to control text formatting. You must resist this urge!

If a browser cannot properly render the tags in your page, it will look terrible. Some features, and in particular tables, require significant calculation in their display, and the wimpy processors in most phones may take a long time to display the pages. Do not, under any circumstances, use images for page layout. The way most browsers operate is to download the page first, display it with placeholders, and then replace those as they download the real images. In general, use images only where they really add to the content.

In our example, you will see two things that are not found in regular HTML. First is the required XHTML MP `<!DOCTYPE...>` line at the top of the page. Second is the `accesskey` attribute in some of the `<a>` tags. These are link navigation shortcuts that allow you to press the specified key and direct the browser to the specified target. This seems like a good idea, and it is actually implemented on my Nokia. But, as you might be able to anticipate by now, there is a problem. How do I know what key gives me the shortcut? To use this feature, I need to take up valuable screen real estate with hints to the user, such as "Press 1" or even something cryptic like "(1)" next to the link. Even in their shortest forms, these hints take up too much space to be worth the effort.

Providing decent navigation links while conserving screen space is an ever-present challenge in creating WAP content. There are no easy answers.

Web Server Configuration

We have our pages, and so the next step is to serve them from a web site. For this discussion I am assuming you have a public web site on a Linux system running Apache. Copy the pages into a suitable location in your path, such as */var/www/html*.

We tell Apache what to do with files like these by specifying a set of MIME types that associate a filename suffix with a specific file type. There are several types that are relevant to mobile web content and we'll just set all of them without going into the details. There are two ways to do this. You can explicitly add these to the *httpd.conf* file using the following block of code at the end of the file:

```
AddType text/vnd.wap.wml wml
AddType text/vnd.wap.wmlscript wmls
AddType application/vnd.wap.wmlc wmlc
AddType application/vnd.wap.wmlscriptc wmlsc
AddType image/vnd.wap.wbmp wbmp
AddType application/xhtml+xml xhtml
```

But chances are most or all of these are already in your */etc/mime.types* file. Typical Apache configurations will automatically import these types from the file. Look for these lines and add any that are not present. In my installation, I only needed to add the line for the .xhtml suffix.

```
text/vnd.wap.wml                  wml
text/vnd.wap.wmlscript            wmls
application/vnd.wap.wmlc          wmlc
application/vnd.wap.wmlscriptc    wmlsc
image/vnd.wap.wbmp                wbmp
application/xhtml+xml             xhtml
```

For these to take effect, you need to restart Apache. You can copy the source for the example site in both markup languages from my site www.craic.com/mobile/ora and try serving them from your own site. Use a desktop browser to get the source code.

That's it for Part 1. Next time, I will show how we can detect the capabilities of a phone making a request to our site, and then deliver content suitable for the device. This involves more work for us on the server end of things, but lets us take advantage of the features available on newer browsers.

Robert Jones runs Craic Computing, a small bioinformatics company in Seattle, which provides advanced software and data analysis services to the biotechnology industry. He was a bench molecular biologist for many years before programming got the better of him.

Return to the Wireless DevCenter