



Published: August 2006

Author: Infogile Inc.

In our vision, the Internet will go mobile, just as voice communication has done. The key enabler of the Mobile Internet is the Wireless Application Protocol (WAP). WAP enables Internet content to be distributed to and displayed in standard mobile phones and other mobile information devices. The current WAP architecture meets the requirements of current low bandwidth networks and mobile devices with limited capabilities. As more advanced mobile networks are adopted, the existing specifications can continue to be used. In other words, the current WAP standard is mature and successful.

Introduction to WAP and WML

Wireless Application Protocol (WAP) is the de facto worldwide standard for providing Internet communications and advanced telephony services on digital mobile phones, pagers, personal digital assistants and other wireless terminals. It is an open, global specification that empowers mobile users with wireless devices to easily access and interact with information and services instantly. The WAP standard developed by the WAP Forum, a group founded by Nokia, Ericsson, OpenWave.com (formerly Phone.com), and Motorola.

The WAP Forum released its first specification - WAP 1.0 - in 1998. And the WAP 2.0 specification was released early this month (August 2001) for public review. WAP 2.0 allows the Wireless Application Protocol to further integrate with the Internet.

Wireless Markup Language (WML) is a markup language based on XML. WML is developed and maintained by the WAP Forum. WML is designed for specifying user interface behavior (data input and forms) and displaying content (text and image

White Paper Overview

Abstract

Wireless business enablement for enterprises in 2004 is hitting a wave of momentum and is demonstrating an important value proposition through cost savings, increased productivity, improved business processes, and realized return on investment (ROI). Mobile and wireless fit into a variety of useful business scenarios, many of which will evolve and expand over time as companies come to better understand the benefits of the instant access and dissemination of knowledge. Building Wireless web applications is easy and very similar to traditional browser based applications development, as far as server-side programming is concerned. Instead of generating HTML, we need to generate WML (Wireless Markup Language). In this paper, we'll talk about building a small calendaring wireless Web application that can be accessed using cellular phones. This paper assumes you have knowledge of ASP/Database programming and familiarity with XML.

Document Audience

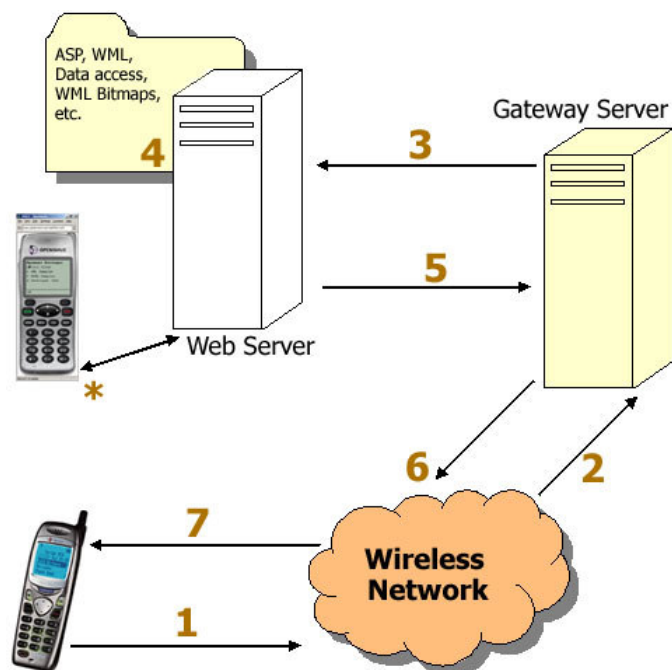
This document is primarily intended for Marketing, Sales, Product Support, Internet Services Group, Project Engineering and anyone who is interested in WAP Technology.

presentation) on wireless devices such as phones, pagers, and PDAs. Just as HTML is broken into small units called Web pages, WML is segmented into units called "cards" and grouped into "decks." The WML deck is also similar to HTML pages in that it is invoked by reference to a URL.

Remember that since WML is based on XML, it is case-sensitive.

How does it Work?

The Web request from the phone is first served by the Gateway Server, which generally is provided as part of services from mobile operator's network. However, it is possible to set up a private gateway. The gateway server translates mobile phone requests (WAP) into HTTP requests and sends them to Web server. The Web server processes the request, and sends WML to gateway server, which in turn sends the WML to phone in the binary compressed WML format.



1. Mobile phone sends a Web request
 2. Wireless Network provider forwards the request to gateway server.
 3. Gateway server sends request to Web server in HTTP format.
 4. Web server accesses data, runs ASP/JSP/CGI Script/... pages and generates WML.
 5. Web server sends WML back to gateway server as HTTP Response.
 6. Gateway server binary compresses WML and sends that to cellular phone as WAP response.
 7. Cell phone displays the content.
- *. The WAP Emulator directly talks to the Web server.

How does WML look like?

WML is an XML vocabulary that consists of a deck, and a deck contains multiple cards. Here is how a "Hello World" WML document will look like:

```
<?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="Greeting">
    <p align="center">
      Hello World!
    </p>
    <p align="center">
      From PerfectXML
    </p>
  </card>
</wml>
```

The document starts with the standard XML declaration. The next line specifies that the document adheres to WML version 1.1 deck. The deck has <wml> element, which can have one or more <card> element, and each <card> element can have one or more <p> elements. See References section for more details on WML syntax. If you open this WML file in WAP emulators, here is what you should see:



WML file in OpenWave UP.Simulator



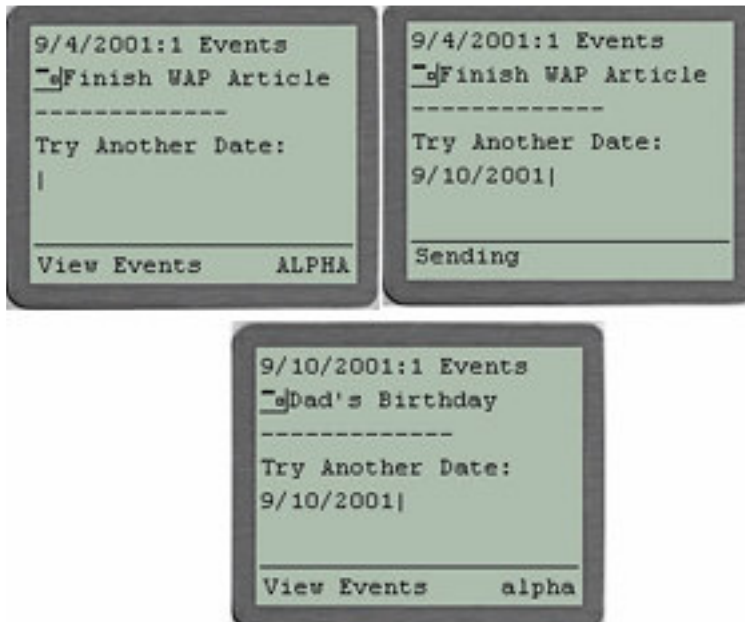
WML file in Nokia Blueprint Emulator (part of Nokia WAP Toolkit)

With this brief introduction to WAP and WML, let's now get started and write our first cool WAP Application!

WAP Calendar

In this article, we'll write a tiny calendaring application that can be accessed using cellular phone, using WAP and WML.

- We'll save all our calendar events into a Microsoft Access database.
- Then we'll write an ASP page that connects to the database and generates WML.
- The ASP page by default returns today's events. If a date parameter is passed, events for that date are returned in WML format.
- We'll then try out this application using WAP Emulators and finally try this application from a real wireless web enabled cellular phone.



The Microsoft Access Calendar database has just one table "tblCalendar", which has three fields ID, fldDate, and fldEvent. Let's look at the ASP Page:

The ASP page starts with standard database access code:

```

<%
    Response.ContentType = "text/vnd.wap.wml"
    Dim objConn, objRS
    Set objConn = Server.CreateObject("ADODB.Connection")
    Set ObjRS = Server.CreateObject("ADODB.Recordset")

    objConn.Provider = "Microsoft.Jet.OLEDB.4.0"
    strDBFile = Server.MapPath(".") & "\calendar.mdb"
    objConn.Open strDBFile
    Set objRS.ActiveConnection = objConn
    objRS.CursorLocation = 3

    strParamDate = Request.Form("calDate")
    if strParamDate = "" then
        strParamDate = Date()
    end if

    objRS.Open "SELECT * from tblCalendar where fldDate = #" &
strParamDate & "#"
%>

```

The above code first sets the response content type. It then connects to the database and retrieves events for a particular date. As mentioned earlier, by default the ASP page retrieves records for current date, however a date parameter can be posted to retrieve the events for that date.

Next, the ASP Page spits out the standard XML and DOCTYPE declarations, followed by the <wml> and <card> tags. We display the date and count of events next, followed by each event. Finally, the ASP page closes the recordset, connection, and frees the objects.

```

<?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="calCard" title="Calendar">
        <p>
            <%=strParamDate%>:<%=objRS.RecordCount%>
Events
        </p>
        <%
            While not objRS.EOF
                <p>
                    
                    <%=objRS.Fields("fldEvent").Value%>
                </p>
                <%
                    objRS.MoveNext()
            Wend

            objRS.Close
            objConn.Close

```



```
Set objRS = Nothing
Set objConn = Nothing
%>
```

Note how we display the image before each event in the while loop.

Let's now look at the last lines in the ASP page:

```
<p>
-----
</p>
<p>
    Try Another Date:
    <input type="text" name="calDate"
maxlength="10"/>
    <do type="accept" label="View Events">
        <go method="post"
href="http://www.PerfectXML.com/wml/C.asp">
        <postfield name="calDate"
value="$calDate" />
    </go>
    </do>
</p>
</card>
</wml>
```

The above lines of code add a form and an input text field where in you can type the date. When you post this form, the same ASP Page (C.asp) is called, which returns the events for the specified date (instead of today's date) in WML format.

Testing the Application

In this section, we'll look at tools available that help develop WAP applications.

OpenWave™ UP.SDK.4.1

You can download the OpenWave™ SDK from <http://developer.openwave.com/download/index.html>. It includes a nice simulator that you can use to try out your WAP applications.

Nokia WAP Toolkit

You can download Nokia WAP Toolkit from <http://forum.nokia.com/>. It includes a nice WML editor and few emulators.

The above calendar application is available at <http://www.PerfectXML.com/wml/c.asp>. So, download the simulator(s), install it, and try out your first WAP application!



References

- [Dynamic WAP Application Development](#), Manning Publications, Inc.
- [WAP Forum](#)
- [Top 10 Usability Guidelines for WAP Applications](#)
- [WML Version 2.0](#)

Summary

In this paper, we learned that WAP application development mostly involves learning WML syntax. We created a simple WAP application that retrieves the calendar event information from the Access database and sends that back from an ASP page in WML format.

For more information about Infogile products and services, contact the Infogile Sales Information Center at business@infogile.com. To access information on internet, go to www.infogile.com.

© 2006 Infogile Corporation. All rights reserved.

This case study is for informational purposes only. INFOGILE MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY. Infogile and the Infogile logo are either registered trademarks or trademarks of Infogile Corporation in the United States and/or other countries. The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

