WAP/WM Script

The most common standard of data transfer and presentation for a handheld device involves the combination of Wireless Application Protocol (WAP) with Wireless Markup Language (WML). Although WAP can be used with other forms of presentation, its coders primarily designed it to be used with WML.

WAP

Because of the small size of PCS devices, and because they operate with much less bandwidth and speed, than the rest of the Internet, a special protocol was necessary to redefine how they handle data transmission. This protocol needed to take into consideration that the average user views information on a screen with as little as five lines. When compared to a computer screen, this is a colossal difference. In addition to size, the typical PCS device does not support the same type of navigation that a desktop browser uses. Typically, you perform all PCS navigation with a list of options, or by pushing a button on the PCS device

.

The difference is dramatic. Color, layout, format, and fonts are severely restricted in most PCS devices. This is where WAP becomes important.

When a device connects to the Internet, several actions occur to bring the Web site to the requesting device. The device actually connects through a series of devices that incorporate different parts of the WAP application

stack. The following outlines what happens when you request a Web page using WAP:

1. The device is turned on and accesses the Internet application via the *minibrowser*, a program that simply interprets the downloaded information and enables the user to interact with the presented data.
2. The device searches for and connects to service.
3. A Web site is selected.
4. A request is sent to gateway server using WAP.
5. The gateway server retrieves information as HTML, and converts it to the appropriate language.
6. The converted data is sent to the PCS device.

WAP application stack.

- Wireless Application Environment (WAE)—This part of the stack defines the programming and scripting used for wireless applications. One of the most common of languages is WM Script,

- Wireless Session Protocol (WSP)—This part is responsible for the type of communication established with the PCS device. It defines whether the session is connection-oriented or connectionless. For example, because of the low impact its lost data will have on the resulting communication, a transfer of music would be connectionless. However, for more critical uses, guaranteed two-way communication is required. (This is similar to UDP versus TCP in traditional networking.)

- Wireless Transaction Session Protocol (WTSP)—This part of WAP is used to classify data flow as reliable one-way, reliable two-way, or unreliable one-way.

- Wireless Transport Layer Security (WTLS)—This layer is the security part of WAP. It provides encryption, authentication, data integrity checks, and more.

- Wireless Datagram Protocol (WDP)—This part of WAP is where the data is broken down for the actual carrier. Because of the many different types of data transfer methods, the WDP ensures standardization, so any carrier can be used to transfer wireless data as long as it is compatible with WAP.

- Network carriers—This is the carrier method (also called a *bearer*) responsible for delivering the data to the PCS device. There are numerous carriers, but any will work as long as it can link to the WDP layer.

Once the data maneuvers through this stack, The PCS device processes it and presents it on the screen with a *minibrowser*. This can be as basic as maneuvering through a menu, or it can be as complex as playing an interactive game.

WML

Now that you have a basic understanding of WAP's purpose, let's examine the actual data and how it is presented. As mentioned before, WML is a markup language based on XML. It is not a programming language such as COBOL, Java, or even VBScript. It is only a formatting language that defines text and object placement and appearance. For example, if you wanted to define a word as bold, you would use the following:

To illustrate how WML works with WMLS.

```
_____
_

<?xml version="1.0"?>

<!DOCTYPE wml PUBLIC "-//PHONE.COM//DTD WML 1.3//EN"

"http://www.phone.com/dtd/wml13.dtd">

<!-- WML file created by Openwave SDK -->

<wml>

  <card id="first" >

    <onevent type="onenterforward">

      <refresh>

        <setvar name="firstVal" value=""/>

        <setvar name="secondVal" value=""/>

      </refresh>

    </onevent>

    <p>

      <do type="accept" label="Plus">
```

```
        <go href="#second"/>
      </do>
      Add two numbers...
     First #:
      <input type="text" name="firstVal" format="*N"/>
   </p>
</card>
<card id="second">
   <onevent type="onenterforward">
      <refresh>
       <setvar name="ans" value=""/>
      </refresh>
  </onevent>
   <p>
      <do type="accept" label="Add">
        <go href="addit.wmls#addNum()"/>
      </do>
```

```
        Second number

        <input type="text" name="secondVal" format="*N"/>

       $firstVal + ______ =

     </p>

  </card>



  <card id="answer" title="answer">

    <p>

       $firstVal + $secondVal = $ans

     </p>

  </card>

</wml>
```

---

addIt.wmls

```
extern function addNum(){

  //grab incoming values
```

```
    var fv = WMLBrowser.getVar("firstVal");

    var sv = WMLBrowser.getVar("secondVal");

    var val = WMLBrowser.getVar("ans");



    //convert values to integers

    var fvNum= Lang.parseInt(fv);

    var svNum = Lang.parseInt(sv);



    //add values

    var valNum = fvNum + svNum;



    //set answer and return to answer card in deck

    WMLBrowser.setVar("ans", valNum);

    WMLBrowser.go("#answer");

}
```