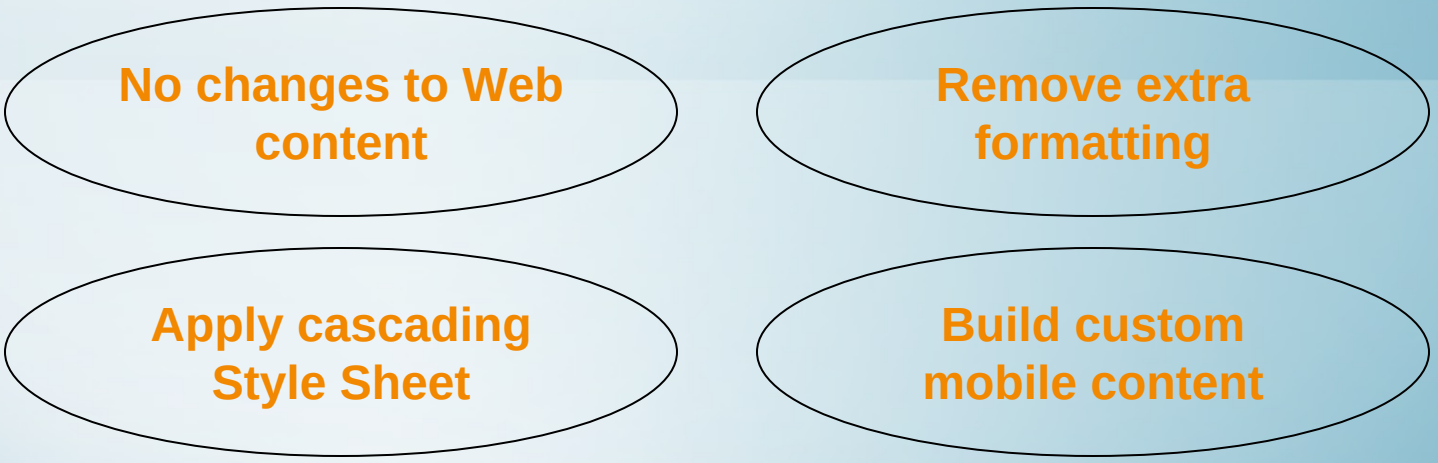
CMER

Centre for Mobile Education and Research

Mobile Web Apps Markup Languages

Delivering Mobile Content to Mobile Devices

- When building mobile Web applications, how content is to be delivered to mobile devices is important
- Four potential approaches



Delivering Mobile Content: *No Changes to Web Content*

- This approach does not require any changes to be made to regular Web content
- A mobile browser (or microbrowser) optimizes Web content to properly be displayed on mobile devices
  - Web content is *rendered* by the mobile browser
  - Small Screen Rendering (SSR) and Medium Screen Rendering (MSR) introduced by Opera Software: render regular Web pages to properly fit on small screens (mobile phones) and medium-sized screens (PDAs)

Delivering Mobile Content: *Remove Extra Formatting*

- This approach does not require any changes to be done to the original design of the content
- Remove any formatting that may consume too much memory on the device (i.e. table formatting, background images, etc.)
  - Faster loading of content
  - Less bandwidth

Apply Cascading Style Sheet (CSS)

- CSS is powerful language for attaching style to Web documents
  - It works with HTML to define how content is presented
- This approach requires a specific CSS for mobile devices
- The style sheet is designed to *optimize* content for mobile devices
  - i.e. optimized formatting

Delivering Mobile Content: *Build Custom Mobile Content*

- This approach requires the development of custom mobile content
- Create a mobile version of the content to be displayed properly on mobile devices
  - ***Requires testing*** **of content on as many mobile devices as possible,**
  - **Maintaining *different content versions* of an application may become time consuming**

Delivering Mobile Content: Summary

Approach	Content is controlled by
No Changes to Web Content	Mobile device (via microbrowser)
Remove Extra Formatting	Mobile device (via microbrowser)
Apply Cascading Style Sheet (CSS)	Content author
Build Custom Mobile Content	Content author

Understanding Wireless Markup Language (WML)

- WML replaced HDML
  - HDML was intended to display Web content for handheld computers, smartphones, etc.
  - HDML was introduced in 1996 by Unwired Planet (now Openwave)
- WML pages have the extension *.WML
- WML pages are called DECKS
- They are constructed as a set of CARDS, related to each other with links
- WML page must have at least one CARD

Understanding the WML (continued)

- When a WML page is accessed from a mobile phone, all the cards in the page are downloaded from the WAP server
- Navigation between the cards is done by the phone computer - inside the phone - without any extra access trips to the server

Common WML Elements

- Deck/Card Elements
  - **Wml card template head access meta**
- Event Elements
  - **Do ontimer onenterforward onenterbackward onpick onevent postfield**
- Tasks
  - **Go prev refresh noop**
- Variables
  - **Setvar**
- User input
  - **Input select option optgroup fieldset**
- Anchors, Images, and Timers
  - **a anchor img timer**
- Text Formatting
  - **br p table tr td**

WML Example 1: Overview

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD WML
  2.0//EN"
“http://www.wapforum.org/dtd/wml20.dtd" >

<wml>
<card title=“Example">
<p> This is a paragraph </p>
<p> This is another<br/>with a line break </p>
</card>
</wml>
```

WML Example 2: Timer

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-
  //WAPFORUM//DTD WML 2.0//EN"
“http://www.wapforum.org/dtd/w
  ml20.dtd" >
```

```
<wml>
<card ontimer="test.wml">
    <timer value="30"/>
    <p>Some Message</p>
</card>
</wml>
```

A WML card can be set up to use the timer function of WML. The time unit of the timer is 1/10 of a second.

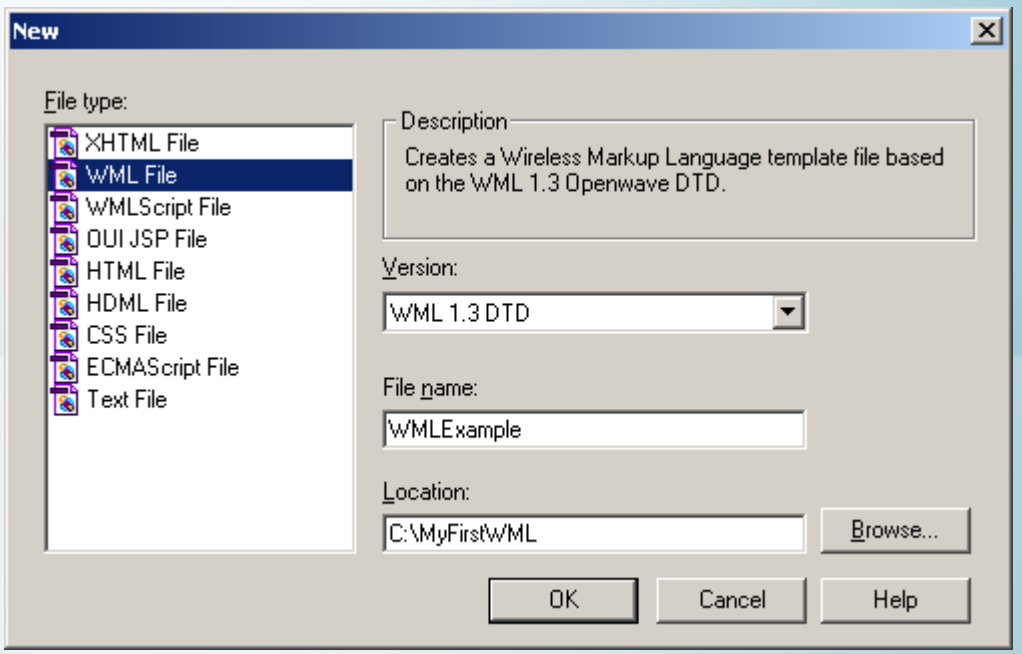
The example below will display a message for 3 seconds, and then take you to the file "test.wml":

Building WML Documents

- There are several tools that can be used to create WML
  - Nokia Mobile Internet Toolkit
  - Openwave Mobile SDK
  - Microsoft Mobile Internet Toolkit with Microsoft .NET Framework
- Building mobile Web applications is not tied to WAP and WML
- New challenging models challenge the future of WAP
  - Example: J2ME, XHTML Basic

Creating WML using Openwave SDK 5.1 Example

- Click on File → New → WML File



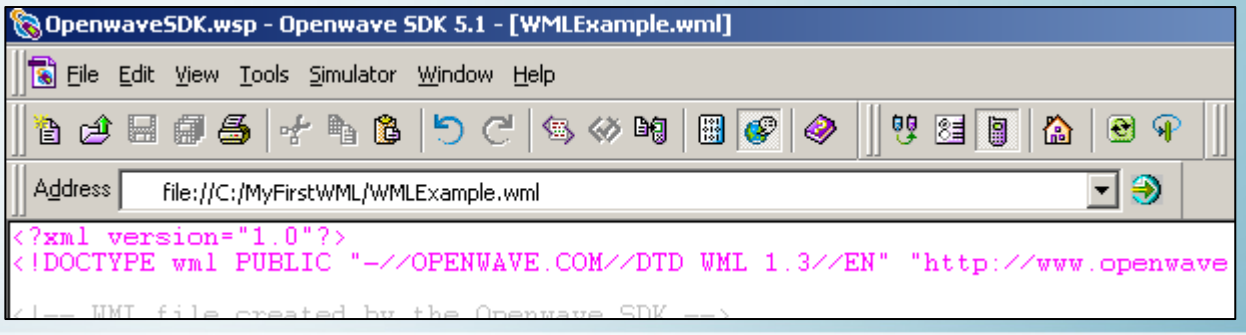
Creating WML using Openwave SDK 5.1 (continued)

- Enter the following WML code

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN"
"http://www.openwave.com/dtd/wml13.dtd">

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

<wml>
    <card id="MyFirstWMLExample" title="MyWML">
        <p>Welcome to my First WML Example</p>
    </card>
</wml>
```

← Notation for writing comments

- In the Address field, enter the location of the WML file
  - **Example: C:\MyFirstWML\WMLExample.wml**



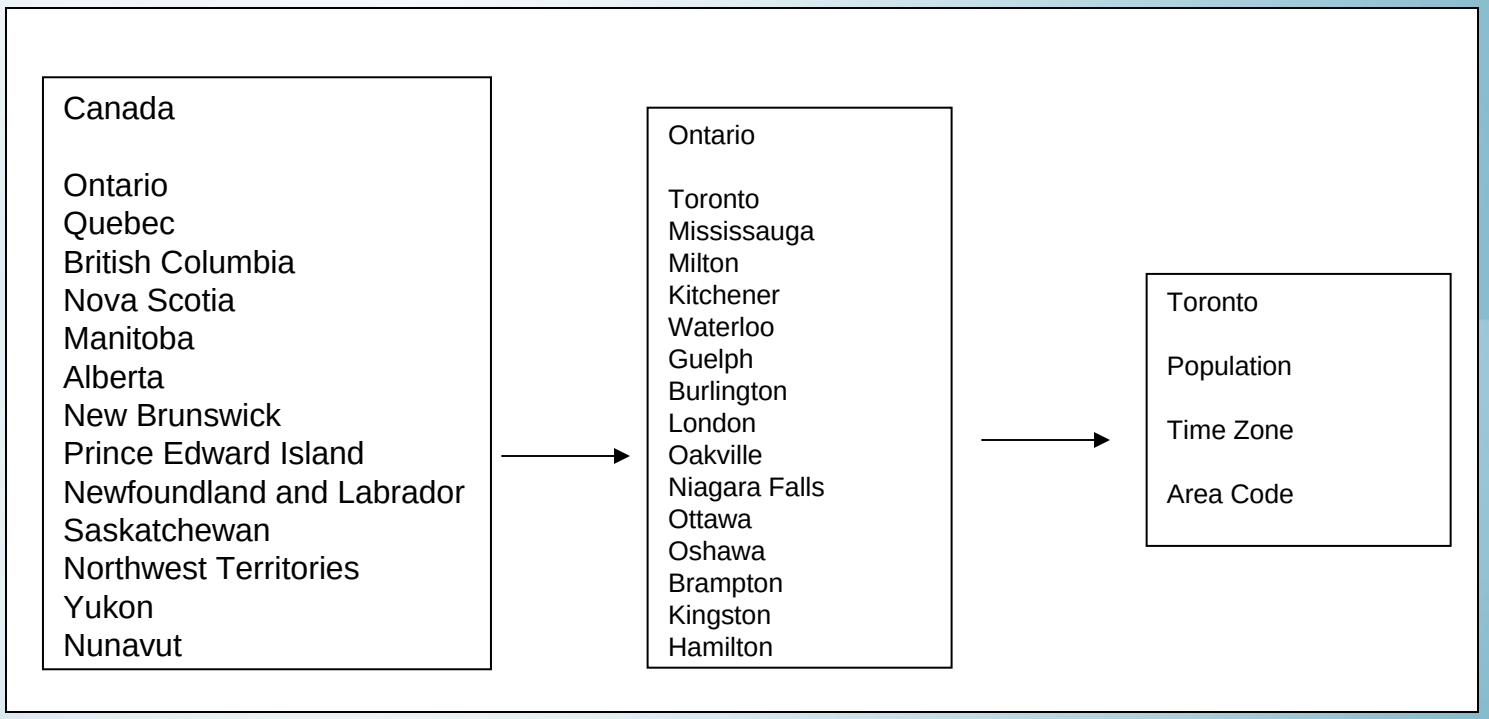
Creating WML using Openwave SDK 5.1 (continued)

- Openwave Emulator used to display Example 1

  You may change the emulator device by clicking on Simulator → Select Device → (i.e. Siemens S45 5.0.1)

WML Cards Example

- **In this example, three cards are used**



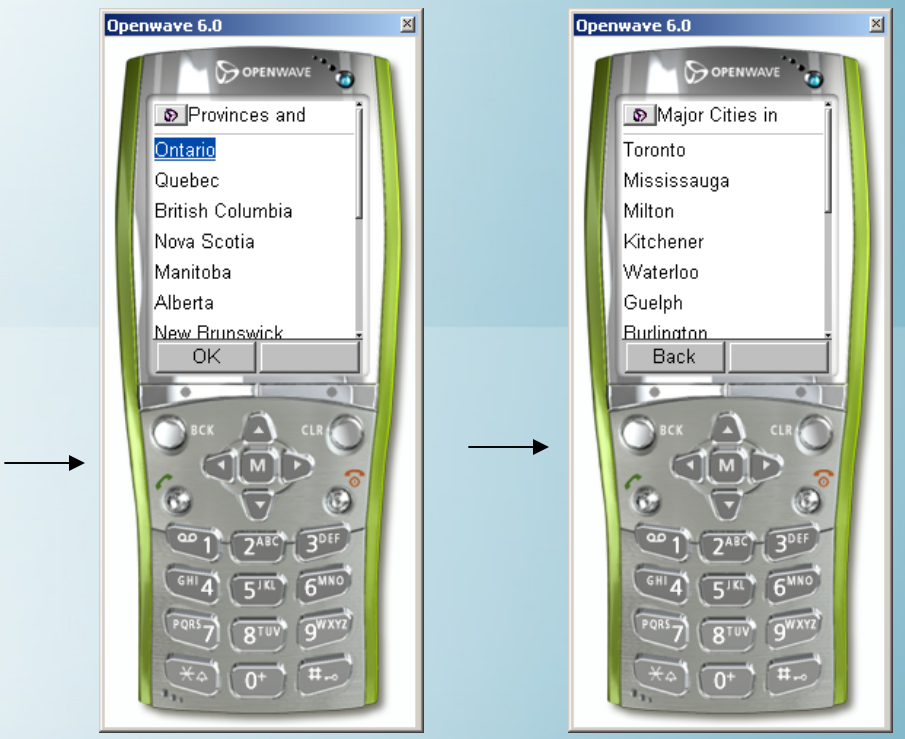
WML Cards Example (continued)

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN"
"http://www.openwave.com/dtd/wml13.dtd">

<!-- Canadian Provinces and Territories Main Screen -->
<wml>
<card id="main" title="Provinces and Territories" -->
    <p>
        <a href="#cardon">Ontario</a>
    </p>
    <p>Quebec</p>
    <p>British Columbia</p>
    <p>Nova Scotia</p>
    <p>Manitoba</p>
    <p>Alberta</p>
    <p>New Brunswick</p>
    <p>Prince Edward Island</p>
    <p>Newfoundland and Labrador</p>
    <p>Saskatchewan</p>
    <p>Northwest Territories</p>
    <p>Yukon</p>
    <p>Nunavut</p>
</card>
</wml>
```

WML Cards Example (continued)

- Using <a> to link the first two cards together

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN"
"http://www.openwave.com/dtd/wml13.dtd">
<!-- Canadian Provinces and Territories Main Screen -->
<wml>
<card id="main" title="Provinces and Territories" -->
    <p>
        <a href="#cardon">Ontario</a>
    </p>
    <p>Quebec</p>
    <p>British Columbia</p>
    <p>Nova Scotia</p>
    <p>Manitoba</p>
    <p>Alberta</p>
    <p>New Brunswick</p>
    <p>Prince Edward Island</p>
    <p>Newfoundland and Labrador</p>
    <p>Saskatchewan</p>
    <p>Northwest Territories</p>
    <p>Yukon</p>
    <p>Nunavut</p>
</card>
<!-- Major Cities in Ontario Screen -->
<card id="cardon" title="Major Cities in Ontario" -->
    <p>Toronto</p>
    <p>Mississauga</p>
    <p>Milton</p>
    <p>Kitchener</p>
    <p>Waterloo</p>
    <p>Guelph</p>
    <p>Burlington</p>
    <p>London</p>
    <p>Oakville</p>
    <p>Niagara Falls</p>
    <p>Ottawa</p>
    <p>Oshawa</p>
    <p>Brampton</p>
    <p>Kingston</p>
    <p>Hamilton</p>
</card>
</wml>
```

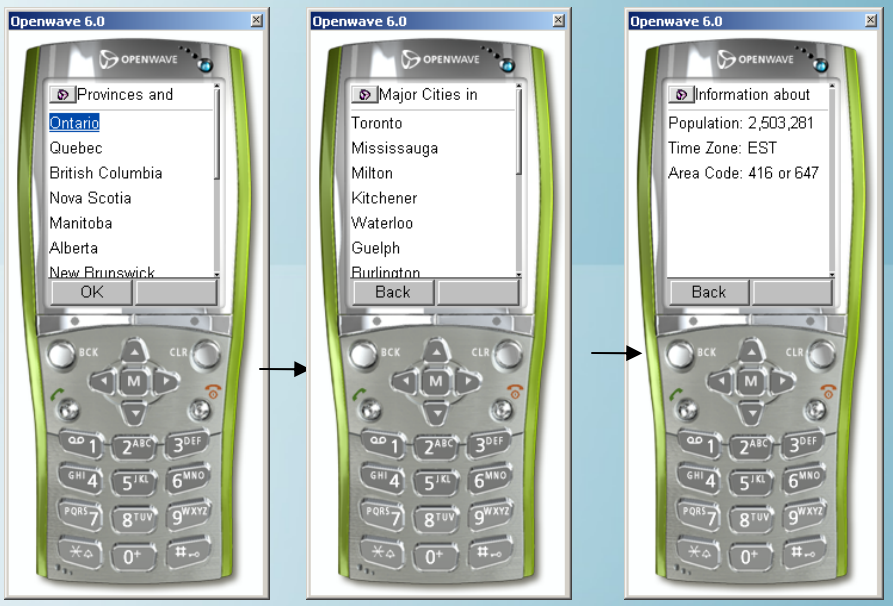


WML Cards Example (continued)

- Using <a> to link the second and third cards together

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN"
"http://www.openwave.com/dtd/wml13.dtd">
<!-- Canadian Provinces and Territories Main Screen -->
<wml>
<card id="main" title="Provinces and Territories" -->



</card>
```

```
<!-- Major Cities in Ontario Screen -->
<card id="cardon" title="Major Cities in Ontario" -->
    <p>
        <a href="#cardtoronto">Toronto</a>
    </p>
    <p>Mississauga</p>
    <p>Milton</p>
    <p>Kitchener</p>
    <p>Waterloo</p>
    <p>Guelph</p>
    <p>Burlington</p>
    <p>London</p>
    <p>Oakville</p>
    <p>Niagara Falls</p>
    <p>Ottawa</p>
    <p>Oshawa</p>
    <p>Brampton</p>
    <p>Kingston</p>
    <p>Hamilton</p>
</card>
<!-- Information about Toronto Screen -->
<card id="cardtoronto" title="Information about Toronto" -->
    <p>Population: 2,503,281</p>
    <p>Time Zone: EST</p>
    <p>Area Code: 416 or 647</p>
</card>


</wml>
```



WML Tables Example

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN"
"http://www.openwave.com/dtd/wml13.dtd">
<wml>
<card title="CD Table">
    <p>
        <table columns="3">
        <tr>
            <td>Title</td>
            <td>Artist</td>
            <td>Price</td>
        </tr>
        </table>
    </p>
</card>
</wml>
```

WML Input Fields Example

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN"
"http://www.openwave.com/dtd/wml13.dtd">
<wml>
<card title="Personal Info">
    <p>
        <table columns="3">
        <tr>
            <td>First Name</td>
            <td><input name="fname" size="5"/></td>
        </tr>
        <tr>
            <td>Last Name</td>
            <td><input name="fname" size="5"/></td>
        </tr>
        <tr>
            <td>Age</td>
            <td><input name="age" size="5"/></td>
        </tr>
        </table>
    </p>
</card>
</wml>
```

WML Variables

- WML does not support user input
  - Developers must use other scripting languages (i.e. WMLScript) to manipulate data (i.e. user input)
- When a user switches from card to card in a deck, we need to store data in variables. WML variables are case sensitive.
- Specify a Variable with the setvar Command
- The following example will create a variable named i with a value of 500:

```
<setvar name="i" value="500"/>
```

WML Variables Example

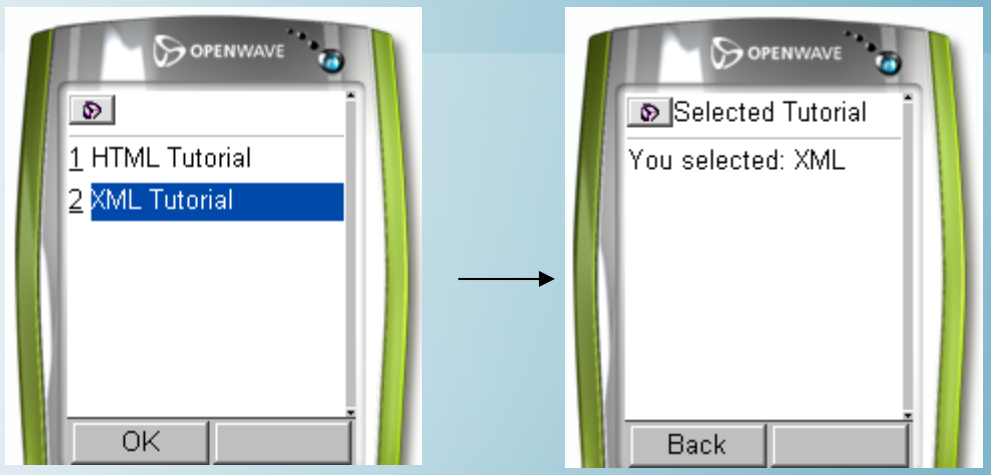
```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN
"http://www.openwave.com/dtd/wml13.dtd">
<wml>
<card id="card1">
    <p>
    <select title="Tutorial Name" name="tutorial">
        <option value="HTML">HTML Tutorial</option>
        <option value="XML">XML Tutorial</option>
    </select>
    <do type="accept" label="Answer">
        <go href="#card2"/>
    </do>
    </p>
</card>

<card id="card2" title="Selected Tutorial">
    <p>
        You selected: $(tutorial)
    </p>
</card>
</wml>
```

WML Variables Example (continued)

The <do> element:

<do> ... </do>

The first card has a <do> element that defines an event to be triggered.
The **type="accept"** attribute of the <do> element causes the
label="Answer" to be displayed in the lower left corner of the display.



WML Options Group

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN"
"http://www.openwave.com/dtd/wml13.dtd">
<wml>
<card id="card1" title="Online Tutorials">
    <p>
    <select name="tutorial" multiple="true">
        <optgroup title="Tutorials">
            <option value="HTML">HTML Tutorial</option>
            <option value="XML">XML Tutorial</option>
        </optgroup>
    </select>
    <do type="accept" label="Select">
        <go href="#tutorials" />
    </do>
</p>
</card>

<card id="tutorials">
    <p>Selected Tutorial <br/>$(tutorial)</p>
</card>
</wml>
```

XHTML Basic

- XHTML Basic is a subset of XHTML
  - designed for mobile devices
  - introduced by the W3C
- XHTML Basic versus WML
  - Using XHTML Basic, one does not have to create multiple version of a mobile Web application
  - Mobile Web application developed using XHTML can be viewed by regular browsers (i.e. Internet Explorer) or microbrowsers
  - Reduces amount of time needed to build a mobile Web application (only have to maintain a single version)

Supported HTML Elements by XHTML Basic

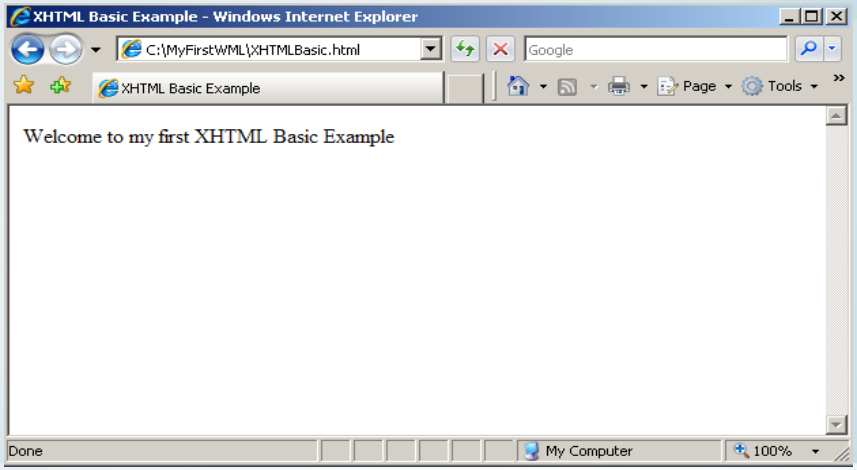
Structure	**body, head, html, title**
Text	**abbr,acronym,address,blockquote,br,cite,code, dfn,div,em,h1,h2,h3,h4, h5,h6,kbd, pn,pre,q,samp,span,string,var**
Hypertext	**a**
List	**dl,dt,dd,ul,li**
Forms	**form,input,label,select,option,textarea**
Tables	**caption,table,td,tr,th**
Image	**img**
Object	**object,param**
Meta Info	**meta**
Link	**link**
Base	**base**

Valid XHTML Basic Syntax

- XHTML documents must
  - Contain valid DOCTYPE definition (DTD)
  - Have a root element must be html
  - Have a root element must contain XHTML default namespace

XHTML Basic Example

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- XHTML 1.0 Strict file created by the Openwave SDK -->

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
    <head><title>XHTML Basic Example</title></head>
    <body>
        <p>Welcome to my first XHTML Basic Example</p>
    </body>
</html>
```



XHTML Basic Supported Features

- XHTML Basic does not support WMLScript
  - One has to manually call a WML file that contains a WMLScript
- XHTML supports the use of Cascading Style Sheets (CSS)
  - CSS are used to apply common formatting

XHTML Mobile Profile

- eXtensible HyperText Markup Language Mobile Profile (XHTML MP) is
  - a subset of XHTML
  - simply XHTML Basic plus additional elements and attributes from the XHTML full version
- The current version of XHTML MP is 1.2

Understanding XHTML MP Document Structure

- Each markup document contains a DTD
  - A Document Type Definition specifies the syntax (grammatical structure) of a markup document
- XHTML MP is XML-based, which means
  - DOCTYPE must be defined,
  - Documents must be well formed
  - All tags are case sensitive,
  - All tags must be properly closed,
  - Elements must be properly nested,
  - Elements and attributes must be in lowercase,
  - Attributes must be enclosed in quotes,
  - All attributes must have values,

Supported HTML Elements in XHTML MP

- Most of the common HTML elements are supported by XHTML MP
- However, XHTML MP does not support many features in WML
  - For example:
    - The concept of decks and cards is not present
    - Unlike WML, XHTML-MP does not support client-side variables
    - Using XHTML-MP, one must submit a form using a submit button. Using WML, it is possible to submit a form using a link.
    - Unlike WML, XHTML-MP does not support timers

Some Common XHTML MP Syntax Errors

- **Nested elements**
  - **Incorrect**

    ** Welcome to XHTML MP good. **
  - **Correct**

    ** Welcome to XHTML MP good. **

- **Attributes must be enclosed in quotes**
  - **Incorrect**

    ** Welcome to XHTML MP **
  - **Correct**

    ** Welcome to XHTML MP **

Some Common XHTML MP Syntax Errors (continued)

- Do not minimize attributes
  - Incorrect

```
<select>
  <option value="0" selected>No</option>
  <option value="1">Yes</option>
<select>
```

  - Correct

```
<select>
  <option value="0" selected="selected">No</option>
  <option value="1">Yes</option>
<select>
```

Supported HTML Elements by XHTML MP

Structure	**body, head, html, title**
Text	**abbr, acronym, address, blockquote, br, cite, code, dfn, div, em,h1, h2, h3, h4, h5, h6, kbd, p, pre, q, samp, span, strong, var**
Presentation	**b, big, hr, i, small**
Style Sheet	**style element and style attribute**
Hypertext	**a**
List	**dl, dt, dd, ol, ul, li**
Basic Forms	**form, input, label, select, option, textarea, fieldset, optgroup**
Basic Tables	**caption, table, td, th, tr**
Image	**img**
Object	**object, param**
Meta Info.	**meta**
Link	**link**
Base	**base**

Be Prepared

- Have an Internet Information Services (IIS) installed and ready to use
- Have Adobe ColdFusion MX or 8 installed and ready to use
  - IIS should support ColdFusion (i.e. CFM pages)
- Microsoft Visual Studio (2005 or 2008) installed

Prepare BlackBerry Environment

- Download
  - BlackBerry® Device Simulators v4.6.0.150 (9000)
    - **Download file into a local folder (i.e. C:\BlackBerry\Simulators\9000)**
- Download
  - BlackBerry® Email and MDS Services Simulator Package v4.1.2 (Bundle 17)
    - **Download file into a local folder (i.e. C:\BlackBerry\MDS**

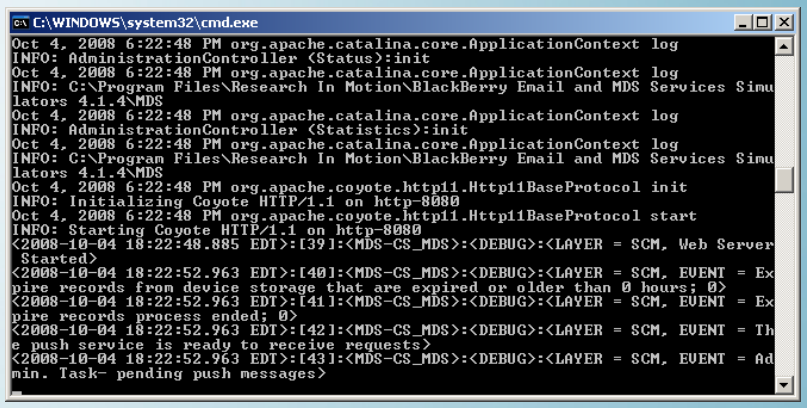
Steps for running Mobile Web Applications on BlackBerry Simulators

- **Before running any mobile Web application, it is important to prepare the proper BlackBerry environment**
  - **Run MDS Services**
    - **Click on Start → Research In Motion → BlackBerry Email and MDS Services Simulators 4.1.2 → MDS**
  - **Run the BlackBerry Device Simulator v4.6.0.150**
    - **Go to C:\BlackBerry\Simulators\9000\ and run 9000.bat**

1. **It is important to setup the Java environment on your PC to properly launch and run MDS Simulator.**
2. **Certain MDS and Email Simulators may require different versions of Java**

Steps for running Mobile Web Applications on BlackBerry Simulators

- After running MDS Services, the following command prompt should open



- After running Device Simulator v4.6.0.150, the following window opens

BlackBerry Device Characteristics

BlackBerry Browser

- **BlackBerry Browser: included with every BlackBerry device**

First generation

- Supports simple Web page presentations and wireless-specific content,
- Provides limited JavaScript, and CSS support

Second generation

- Introduced a new rendering agent and JavaScript engine for greater support of Web standards,
- Supports HTML 4.01, CSS 2.1, and DOM Level 2
- Ability to render most existing Web content

Trackball

- Most BlackBerry devices have a trackball
- Trackballs are used to control the navigation (i.e. similar to a mouse)
- BlackBerry devices with a trackball also include a Menu key to the left of the trackball
- Older BlackBerry devices use a Trackwheel for control the navigation

Touch Screen

- **BlackBerry devices with SurePress touch screen, users are able to use their fingers to interact with applications on the device**

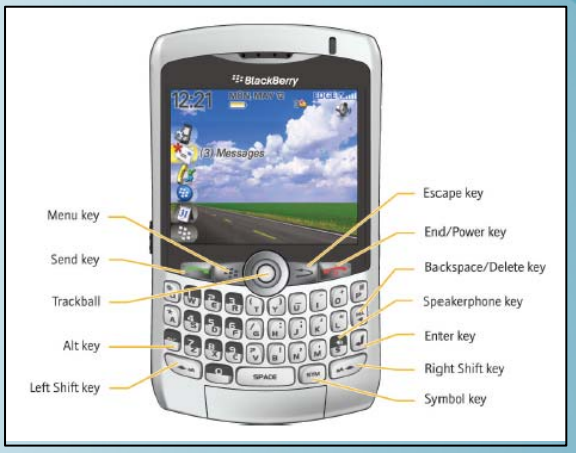
Picture is a courtesy of GSM MAG

Keyboard

- Keyboards are used to type text
- BlackBerry users can use keyboards to also move around a screen (i.e. map)
- Two types of keyboards: QWERTY or SureType

Keyboard (continued)

- QWERTY keyboard: similar to PC keyboards
- SureType: Integrates a traditional phone keypad and a familiar QWERTY-style keyboard
  - SureType technology has the ability to predict words as users type



MDS Connection Service

- Designed to provide users with secure access to their organization's intranets and access to the Internet
- It is a component of the BlackBerry Enterprise Server (BES) that exists on the organization's network behind a firewall
- It acts as a proxy for the BlackBerry browser and makes requests on behalf of the browser.
- It also optimizes the response to enhance network efficiency and display on smaller screens

MDS Connection Service Security

- To access a Web page, MDS Connection Service opens a connection to the Internet from within an organization
- MDS Connection Service can use SSL and TLS protocols to encrypt communication over the Internet between BES and Web servers
  - Supports Triple DES

Internet Service Browsing

- Wireless service providers can use the BlackBerry Internet Service Browsing to offer BlackBerry device users access to the Internet without using BES
- It acts as a proxy for the browser and makes requests on behalf of the BlackBerry browser
- It optimizes content in the response for improving the efficiency and the display on smaller screens

Internet Service Browsing Security

- It does not support Triple DES
- It is not designed to access intranets that are protected by firewalls
- It allows users to access secure sites using HTTPS
- Supports SSL and TLS