Department of Foundation and Pathways

Swinburne University of Technology

COS10024 – Web Development **Week 4 - Tutorial Activity 2**

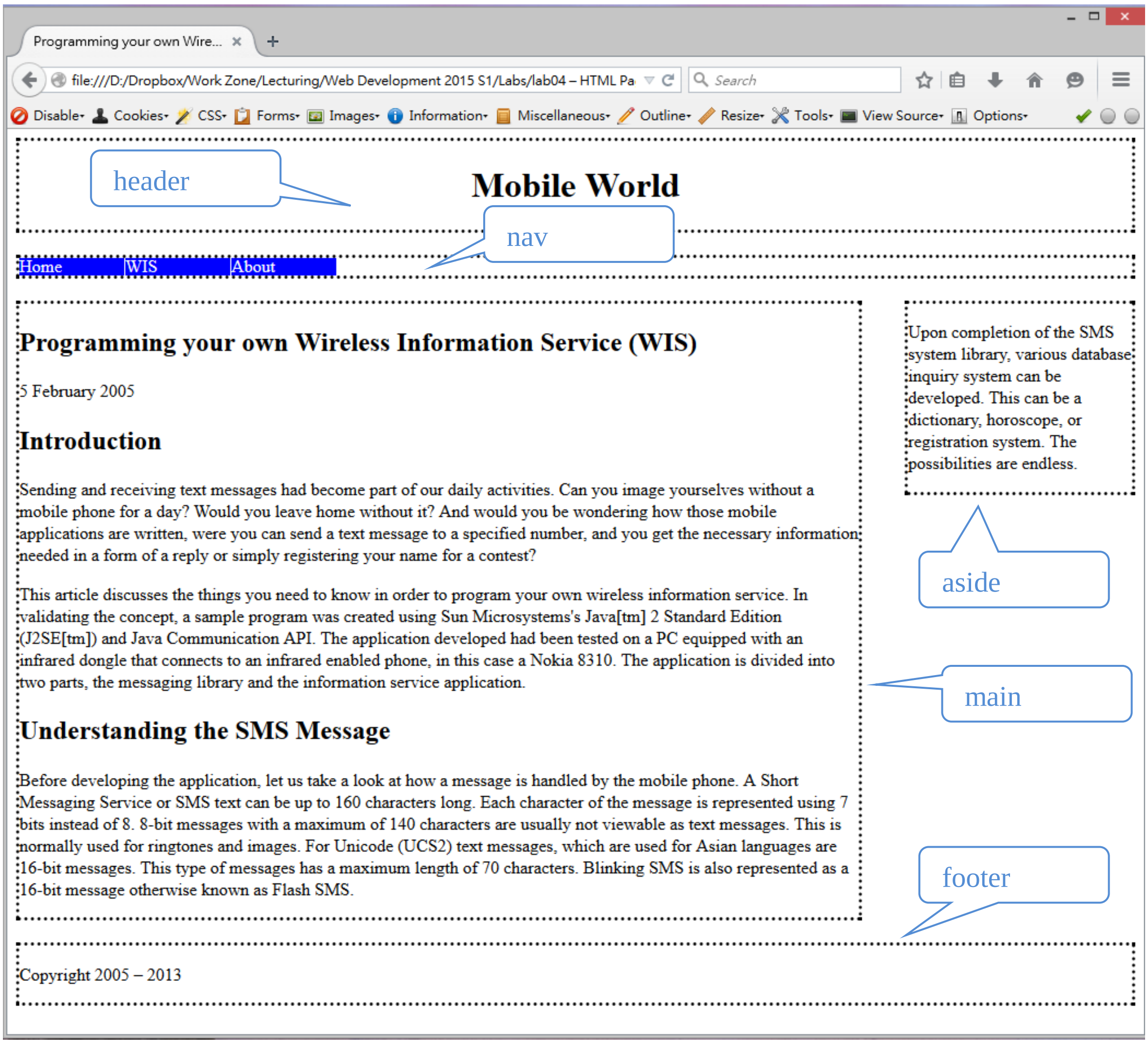
Activity 2 – HTML Page Structure and CSS Presentation

Aims:

- Understand and use HTML structural elements
- Present a web document using CSS
- Gain skills and knowledge to complete Assignment 1.

Task 1: Structure and Present a Webpage (2 Marks)

In this lab, you will structure a webpage named **wis.html** and style it as presented in Figure 1:


Programming your own Wire...
file:///D:/Dropbox/Work Zone/Lecturing/Web Development 2015 S1/Labs/lab04 – HTML Pa
Search
Disable▾ Cookies▾ CSS▾ Forms▾ Images▾ Information▾ Miscellaneous▾ Outline▾ Resize▾ Tools▾ View Source▾ Options▾
header
nav
aside
main
footer

Mobile World

Home WIS About

Programming your own Wireless Information Service (WIS)

5 February 2005

Introduction

Sending and receiving text messages had become part of our daily activities. Can you image yourselves without a mobile phone for a day? Would you leave home without it? And would you be wondering how those mobile applications are written, were you can send a text message to a specified number, and you get the necessary information needed in a form of a reply or simply registering your name for a contest?

This article discusses the things you need to know in order to program your own wireless information service. In validating the concept, a sample program was created using Sun Microsystems's Java[tm] 2 Standard Edition (J2SE[tm]) and Java Communication API. The application developed had been tested on a PC equipped with an infrared dongle that connects to an infrared enabled phone, in this case a Nokia 8310. The application is divided into two parts, the messaging library and the information service application.

Understanding the SMS Message

Before developing the application, let us take a look at how a message is handled by the mobile phone. A Short Messaging Service or SMS text can be up to 160 characters long. Each character of the message is represented using 7 bits instead of 8. 8-bit messages with a maximum of 140 characters are usually not viewable as text messages. This is normally used for ringtones and images. For Unicode (UCS2) text messages, which are used for Asian languages are 16-bit messages. This type of messages has a maximum length of 70 characters. Blinking SMS is also represented as a 16-bit message otherwise known as Flash SMS.

Upon completion of the SMS system library, various database inquiry system can be developed. This can be a dictionary, horoscope, or registration system. The possibilities are endless.

Copyright 2005 – 2013

Figure 1. Styled webpage wis.html

Step 1:

1.1 Create a lab04 folder on the server and upload and test your work from there.

1.2 Download file wis.html from Canvas. Use wis.html as a template for today's lab work.

Step 2:

The text enclosed in <body>…</body> tags represents a partial article that contains two sections about creating a Wireless Information Service (WIS) using a mobile phone and a PC.

2.1 Using NotePad++ (or Sublime Text for Mac users), structure wis.html using appropriate HTML5 structural elements, including:

- <header>…</header>, <nav>…</nav>, <article>…</article>, <section>…</section>, <aside>…</aside>, <footer>…</footer>
- Other structural elements may also be used. such as <summary>…</summary>, <time>…</time>, <div>…</div>, and <span>…</span>

2.2 Use other HTML elements as needed:

- Heading <h#>…</h#>, paragraph <p>…</p>, horizontal rule <hr />
- List <ol>…</ol> or <ul>…</ul>, table <table>…</table>, Image <img … /> and anchor <a … >…</a>
- Special characters

An example web page structure is presented in Figure 1 for your reference.

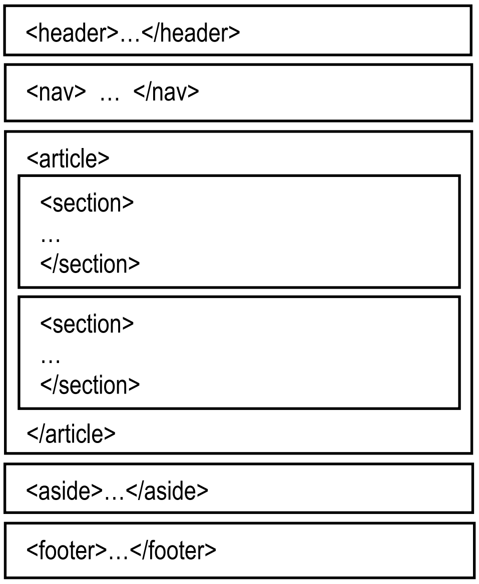


Figure 1. Webpage Structure

Step 3:

3.1 Add the following content into the correct position within the webpage.

- Title in <header>…</header> of the webpage

  Mobile World

- A menu list in <nav>…</nav> as navigator of the webpage. Do not forget to use <a>…</a> to create hyperlinks that direct users to other webpages on your web site. The navigator should have at least the following hyperlinks.

  Home
  WIS
  About

- Copyright notice in <footer>…</footer> as footer of the webpage.

  Copyright 2009 – 2014

- Extra notes in <aside>…</aside> on the webpage

Step 4:

Create and save a new file named ***style.css***. In style.css, specify the CSS rules that will style wis.html as shown in Figure 2.

4.1 Using Notepad++ (or Sublime Text for Mac users), add the following CSS code into the *style.css* file

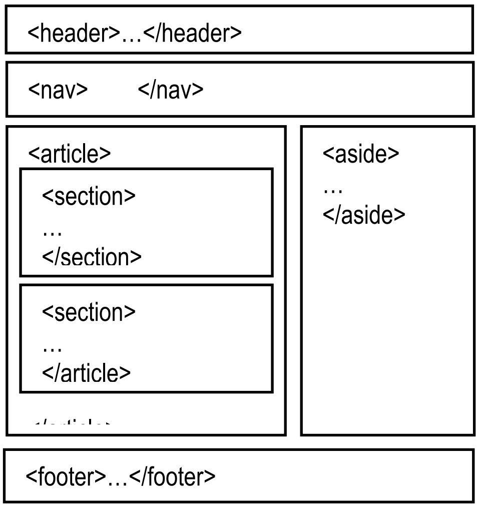


Figure 2. Web Page Structure

```
main {
overflow:auto;
}

nav {
height:1em;
line-height:1em;
margin:20px 0px;
}

header, nav {
	border-style:dotted;
}

article {
	float:left;
	width:75%;
	border-style:dotted;
}

aside{
	float:right;
	width:20%;
	border-style:dotted;
}

footer {
	clear:both;
	border-style:dotted;
	margin-top: 20px;
}
```

Note: You can modify the percentages used in those CSS rules and see the differences made to the web page structure.

4.2 To link the CSS to the wis.html, add the following code into the <head> … </head> section of the web page

```
<link href="style.css" rel="stylesheet" type="text/css" />
```

4.3 View the web page in a browser and see if the presentation of the web page has been changed.

Step 5:

5.1 Using Notepad++ (or Sublime Text for Mac users), open style.css and add the CSS rules presented below that will present the menu list as a horizontal navigation bar, similar to the one in Figure 3.

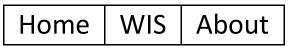


Figure 3. Horizontal Menu

```
ul {
	float:left;
	width:100%;
	padding:0;
	margin:0;
	list-style-type:none;
}
```

```
a {
      float:left;
      width:6em;
      text-decoration:none;
      color:white;
      background-color:blue;
      border-right: 1px solid white;
}

a:hover {
      background-color: #2288ff;
}

header h1 {
text-align:center;
}
```

Note: You can modify the above CSS rules and see what those changes will do to the presentation of the web page. This way, you can find out what each CSS rule is used for.

Step 6: Test and view web pages:

6.1 Using WinSCP (or Filezilla for Mac users), upload your files, including wis.html and style.css, onto Mercury.
6.2 To view the pages through http, use any Web browser and type in the following address,

http://mercury.swin.edu.au/<your unit code>/***s**<your Swinburne ID>*/<folder>/<filename>

Please refer to the following examples to identify the URLs of your web pages.

Folder on Mercury Web Server	**URL**
~/cos10005/www/htdocs/index.html	http://mercury.swin.edu.au/cos10024/s1234567/index.html
~/cos60002/www/htdocs/lab04/wis.htm	http://mercury.swin.edu.au/cos10024/s1234567/lab04/wis.html

Note: You can copy the URLs in the table but remember to replace the unit codes and student id in the above examples with yours to obtain the URLs of your web pages on Mercury.

[IMPORTANT] When the browser authorization request dialog pops up, use your SIMS username *and* password to confirm access, NOT your mercury username and password.

Step 7: HTML and CSS Validation

To validate the HTML file, either use 'Tools'/ 'Validate Local HTML' from the Web Developer toolbar or use the online validator at http://validator.w3.org.

To validate the CSS file, either use the Web Developer toolbar and use 'Tools'/ 'Validate Local CSS' or use the CSS validator at http://jigsaw.w3.org/css-validator/.