Lecture 1: 使用檔名：myXmlXSL_0010.xml & myXSLT_0010.xsl
(Using Extensible Stylesheet Language Transformations; XSLT)

Extensible Stylesheet Language (XSL) is an XML-based language used to create sylesheets.

一、使用 myXmlXSL_0010.xml 檔。

```
<?xml version="1.0" encoding="Big5" ?>
<!--      my first Xml file    myXmlXSL_0010.xml -->

<?xml-stylesheet type="text/xsl" href="myXSLT_0010.xsl"?>

<book>
        <title>通訊原理與應用</title>
        <author sex="M">藍國桐</author>
        <page>288  頁</page>
        <price>NT$ 350</price>
</book>
```

二、再 edit 一個 myXSLT_0010.xsl 檔。

```
<?xml version="1.0" encoding="UTF-16"?>
<!--     my first XSLT file myXSLT_0010.xsl -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

    <xsl:template match="/">
        <html>
            <head>
                <title>This    is    my first xsl pratice!</title>
            </head>
            <body>
                <H2>Hello World!</H2>
                <xsl:value-of select="book/title" /><BR/>
                <xsl:value-of select="book/author" /><BR/>
                <xsl:value-of select="book/price" /><BR/>
            </body>
        </html>

    </xsl:template>
</xsl:stylesheet>
```

執行結果(Demo)：

說明一、Although a CSS allows you to fully specify the formatting of each XML element, the browser merely copies the text in its original XML document order to the displayed output. An XSLT style sheet gives you complete control over the output.

說明二、Specifically, XSLT allows you to precisely select the XML data you want to display, to present that data in any order or arrangement, and to freely modify or add information. XSLT gives you access to almost all XML document components (such as elements, attributes), it lets you easily sort and filter the XML data, it allows you to include loop and conditional structures and use variables as in a programming language, and it provides a set of useful built-in functions you can use to work with the information.

說明三、You can use any of the CSS properties to format the HTML elements that display the XML data.

說明四、A final advantage of XSLT is that an XSLT style sheet is an XML document itself, so its basic syntax will already be familiar to you.

說明五、There are two basic steps for using XSLT style sheet to display an XML document:

1. Create the XML file and link the XSLT style sheet to this XML document.

You can create two separate files: an XML document file (with the .xml extension) and an XSLT style sheet file (with the .xsl extension). You link the XSLT style sheet to the XML document by including the following *xml-stylesheet* processing instruction.

<?xml-stylesheet type="text/xsl" href="http://www.ntou.edu.tw/myXSLT_0010.xsl"?>
(a fully qualified URL)
<?xml-stylesheet type="text/xsl" href="myXSLT_0010.xsl"?>
(relative URL)

Unlike cascading style sheets, if you link more than one XSLT style sheet to an XML document, the browser will use the first one and ignore the others. If you link both a CSS and an XSLT style sheet to an XML document, the browser will use only the XSLT style sheet.
(自行 check: 多個 CSS 時會產生何種效果？多個 XSLT 時？XSLT+CSS 時？)

2. Create the relative XSLT style sheet file.

XSLT is an application of XML. That is, an XSLT style sheet is a well-formed XML document that conforms to the XSLT rules.

說明六、Rather than containing rule, as in a CSS, an XSLT style sheet includes one or more ***templates***. The **templates** tell the browser how to display the XML document by providing instructions for selectively transforming the XML document's elements, attributes, and other components into an HTML page that the browser renders and displays.

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
(XSLT 檔 必含這個 document element)
</xsl:stylesheet>

The ***xsl:stylesheet*** root element of an XSLT style sheet can contain one or more templates, each of which is defined by means of an ***xsl:template*** element.

You use the ***xsl:template*** element's ***match*** attribute to indicate the specific XML document component or set of components that the template is designed to transform.
<xsl:template match="/">
The location path in this example (/) matches the XSLT root node which represents the entire XML document.

(Demo):
(加上一些 HTML 中常常使用的 Tag)：
使用檔名：myXmlXSL_0011.xml & myXSLT_0011.xsl

```
<?xml version="1.0" encoding="UTF-16"?>
<!--      my XSLT file myXSLT_0011.xsl -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

      <xsl:template match="/">
            <html>
                  <head>
                        <title>This    is    my first xsl pratice!</title>
                  </head>
                  <body>
                        <H2>Hello World!</H2>
                        <FONT size="5" color="#0000FF"> Title: </FONT>
                        <xsl:value-of select="book/title" /><BR/>
                        <FONT size="5" color="#0000FF"> Author: </FONT>
                        <xsl:value-of select="book/author" /><BR/>
                        <FONT size="5" color="#0000FF"> Price: </FONT>
                        <xsl:value-of select="book/price" /><BR/>
                  </body>
            </html>
      </xsl:template>

</xsl:stylesheet>
```

(Demo):

(invoke XML 中的 Tag 要小心，大小寫有區別)：
使用檔名：myXmlXSL_0012.xml & myXSLT_0012.xsl

<xsl:value-of select="book/Title" />
　因為 Title 的大寫 以致於產生 錯誤 !!

Lecture 2: 使用檔名：myXmlXSL_0020.xml & myXSLT_0020.xsl

一、先修改 myXmlXSL_0010.xml 成爲 myXmlXSL_0020.xml 檔。

(增加 一 個 <book> 元件)

```
<?xml version="1.0" encoding="Big5" ?>
<!--       my first Xml file    myXml_0010.xml -->

<?xml-stylesheet type="text/xsl" href="myXSLT_0020.xsl"?>

<inventory>
    <book>
        <title>通訊原理與應用 </title>
        <author sex="M">藍國桐</author>
        <page>288 頁</page>
        <price>NT$ 350</price>
     </book>
    <book>
          <title>樂路程式設計</title>
          <author sex="M">王 李 吉</author>
          <page>550 頁</page>
          <price>NT$ 450</price>
    </book>
</inventory>
```

二、再修改 myXSLT_0010.xsl 成爲 myXSLT_0020.xsl 檔。

```
<?xml version="1.0" encoding="UTF-16"?>
<!--      my first XSLT file myXSLT_0010.xsl -->

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
    <xsl:template match="/">
          <html>
               <head>
                    <title>This    is    my first xsl pratice!</title>
               </head>
               <body>
                    <H2>Hello World!</H2>
                    <xsl:value-of select="inventory/book/author"/><BR/>
                    <xsl:value-of select="inventory/book/title" /><BR/>
                    <xsl:value-of select="inventory/book/price" /><BR/>
               </body>
          </html>
    </xsl:template>
</xsl:stylesheet>
```

執行結果(Demo)：

只出現一組 <book> 的資料？

說明一、How the Internet Explore applied XSLT templates.

It is permissible for a style sheet to omit the template matching the XSLT root node, to have several templates, or even to have no templates at all. In all cases, the browser begins by "applying a template" to the XSLT root node. The following are the steps that it takes when it applies a template to the root node or to any other node:

(1) It looks for a template defined in the style sheet that matches the node.
(2) If it finds a matched template. It turns over the transformation of the node to that template.
(3) If it doesn't find a matching template, it uses the appropriate built-in template.

A built-in template is one whose behavior is defined by the XSLT specification, rather than in an ***xsl:template*** element that you include in your style sheet. The particular built-in template the browser uses depends upon the type of the node that is being transformed, as follows:

- The built-in template for the XSLT root node applies a template to each child node of the root node- that is, for each child node it performs steps (1) through (3).
- The built-in template for an element node applies a template to each child node of the element node- that is, it performs steps (1) through (3) for each of these nodes.

The possible child nodes of an element node include a text node as well as nested element nodes, comment nodes, and processing instruction nodes.

- The built-in template for a text node displays the text- that is, it outputs the character data associated with the text node's parent element node.
- The built-in template for an attribute node also display the associated text (i.e. the attribute's value).

(However, if an element has an attribute, the attribute node is not considered to be a child of the element node (attribute node 和 element node 不被視為同樣的node 形態) ; therefore, the built-in template for an element node does not apply a template to the attribute node. The browser applies a template to an attribute node only if one of the templates that you write explicitly selects the attribute node within an ***xsl:apply-templates*** elements.)

- The built-in template for a comment or processing instruction node does nothing- that is, it doesn't display the node's text content.

只出現一 組 <book> 的資料？ 因為 : the style sheet contained template matching the <book> element node, that would never be used (unless the root node template contained an ***xsl: apply-templates*** element that selected the <book> element nodes).

說明二、XSL 的功用不只限於 作 style sheet 而已，利用 parser 核心程式及 xslt ，我們可以將 xml 的資料檔作適當的 選取(只傳部分資料)及包裝(加上 tag)，將客戶所需的資料傳出去即可。

作法：

(1) 以 msxls.exe 作範例：

(Demo) 用 XMXSL.exe 呼叫 myXmlXSL_0021.xml 及 myXSLT_0021.xsl 並將其轉成 html 檔案。

(2) 以 XMLSPY 套裝軟體作範例：

(Demo) 用 XMLSPY 呼叫 myXmlXSL_0021.xml 並將其作 XSL Transformation，可以轉成 html 檔案。

(3) 以 VB 呼叫 MS xml parser core 作範例：

Lecture 3: 使用檔名：myXmlDSO_0110.HTML & myXmlCSS_0092.xml

Lecture 4: 使用檔名：myXmlDSO_0120.HTML & myXmlCSS_0092.xml

(直接由 RecordSet 物件及 XML 的 Tag name 存取 其 Tag element 的資料內容)

一、修改 myXmlDSO_0110.HTML 成爲 myXmlDSO_0120.HTML 檔。

(增加 一 個 Java Script 的函數到 <JavaScript> 中)

(Demo: myXmlDSO_0120.html)