XSL/XSLT

XML

EXtensible Markup Language.

XML was designed to carry data
- with focus on what data is

HTML was designed to display data
- with focus on how data looks

XML

Good for flexible display
- Compartmentalizes information
- Don’t have to edit html
- Can extend without breaking
- Can filter to display
- Flexible to each user

XML

Can work going into and out of other databases - SQL

XML

Uses:
News, weather
Records for updating
Information transfer
Non - proprietary format

Introduction: XML

- Metadata (data about data) should be stored as attributes
- Data itself should be stored as elements.

- Elements can have Attribute (name/value pair)
- Elements can have text content

Introduction: XML

- XML tags are not predefined like HTML tags are
- Case sensitive, letter or underscore start, no spaces but letters, digits, hyphens, underscores, and periods
- Can extend without breaking
- Attributes or elements -- both work

Introduction: XML

- Attributes cannot contain multiple values (elements can)
- Attributes cannot contain tree structures (elements can)
- Attributes are not easily expandable (for future changes)

- Can use id references on elements
- Can have prefixes (namespaces) more later…

Introduction: XML

- Styling?

CSS?

Can use CSS - but powerful XSLT stylesheet language transforms XML into HTML

- To browser webkit

- More sophisticated - add, remove, test, hide

XML

XML is just information
Browser webkits recognise and can format usefully
Further styling is done by XSL

Introduction: XSL Languages

- It started with XSL and ended up with XSLT, XPath, and XSL-FO.
- XSL stands for E**X**tensible **S**tylesheet **L**anguage.
- The World Wide Web Consortium (W3C) started to develop XSL because there was a need for an XML-based Stylesheet Language.

CSS = HTML Style Sheets

- HTML uses predefined tags and the meaning of the tags are **well understood**.
- The **<table>** element in HTML defines a table - and a browser knows **how to display it**.
- Adding styles to HTML elements is simple. Telling a browser to display an element in a special font or color, is easy with CSS.

XSL = XML Style Sheets

- XML does not use predefined tags (we can use any tag-names we like), and the meaning of these tags are **not well understood**.
- A <table> element could mean an HTML table, a piece of furniture, or something else - and a browser **does not know how to display it**.
- XSL **describes** how the XML document should be displayed!

XSL - More Than a Style Sheet Language

- XSL consists of three parts:
  - XSLT - a language for transforming XML documents
  - XPath - a language for navigating in XML documents
  - XSL-FO - a language for formatting XML documents
- The rest of this lecture is about XSLT - the language for transforming XML documents.

Introduction to XSLT

- XSLT is a language for transforming XML documents into XHTML documents or to other XML documents.
- XPath is a language for navigating in XML documents.
- **What is XSLT?**
  - XSLT stands for XSL Transformations
  - XSLT is the most important part of XSL
  - XSLT transforms an XML document into another XML document
  - XSLT uses XPath to navigate in XML documents
  - XSLT is a W3C Recommendation

XHTML

XHTML is HTML defined as XML

Strict form of HTML

XHTML

Document Structure
XHTML DOCTYPE is mandatory
The xmlns attribute in <html> is mandatory
<html>, <head>, <title>, and <body> are mandatory
XHTML Elements
XHTML elements must be properly nested
XHTML elements must always be closed
XHTML elements must be in lowercase
XHTML documents must have one root element
XHTML Attributes
Attribute names must be in lower case
Attribute values must be quoted

XHTML

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
<title>Title of document</title>
</head>

<body>
some content
</body>
```

XSLT = XSL Transformations

- XSLT is the most important part of XSL.
- XSLT is used to transform an XML document into another XML document, or another type of document that is recognized by a browser, like HTML and XHTML. Normally XSLT does this by transforming each XML element into an (X)HTML element.
- With XSLT you can add/remove elements and attributes to or from the output file. You can also rearrange and sort elements, perform tests and make decisions about which elements to hide and display, and a lot more.
- A common way to describe the transformation process is to say that **XSLT transforms an XML source-tree into an XML result-tree**.

XSLT Uses XPath

- XSLT uses XPath to find information in an XML document. XPath is used to navigate through elements and attributes in XML documents.

- In the transformation process, XSLT uses XPath to define parts of the source document that should match one or more predefined templates. When a match is found, XSLT will transform the matching part of the source document into the result document.

XSLT - Transformation

- **Example study: How to transform XML into XHTML using XSLT.**
- **Correct Style Sheet Declaration:**
  - The root element that declares the document to be an XSL style sheet is <xsl:stylesheet> or <xsl:transform>.
  - **Note:** <xsl:stylesheet> and <xsl:transform> are completely synonymous and either can be used!
  - The correct way to declare an XSL style sheet according to the W3C XSLT Recommendation is:

    <xsl:stylesheet version="1.0“ xmlns:xsl=http://www.w3.org/1999/XSL/Transform>

  or:

    <xsl:transform version="1.0" xmlns:xsl=http://www.w3.org/1999/XSL/Transform>

  - To get access to the XSLT elements, attributes and features we must declare the XSLT namespace at the top of the document.
  - xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  - points to the official W3C XSLT namespace. If you use this namespace, you must also include the attribute version="1.0".

XSLT in XML

XSLT is an XML document processing language that uses source code that happens to be written in XML. An XSLT document declares a set of rules for an XSLT processor to use when interpreting the contents of an XML document.

<http://xml.silmaril.ie/style.html>

XSLT to XML/HTML

These rules tell the XSLT processor how to generate a new XML-like data structure and how that data should be emitted — as an XML document, as an HTML document, as plain text, or perhaps in some other format.All current versions of Microsoft Internet Explorer, Firefox, Chrome, Mozilla, Safari, and Opera handle XSLT 1.0 inside the browser. <http://xml.silmaril.ie/style.html>

Namespaces

The XSL namspace is declared at the top of the document and match to the whole template to use the functionality.

Start with a Raw XML Document:

- We want to **transform** the following XML document ("cdcatalog.xml") into XHTML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalog>
    <cd>
            <title>Empire Burlesque</title>
            <artist>Bob Dylan</artist>
            <country>USA</country>
            <company>Columbia</company>
            <price>10.90</price>
            <year>1985</year>
    </cd>
    . . .
</catalog>
```

View cdcatalog.xml

- **Viewing XML Files in Firefox and Internet Explorer:**
  - Open the XML file (typically by clicking on a link) - The XML document will be displayed with color-coded root and child elements. A plus (+) or minus sign (-) to the left of the elements can be clicked to expand or collapse the element structure. To view the raw XML source (without the + and - signs), select "View Page Source" or "View Source" from the browser menu.

Create an XSL Style Sheet

- Then you create an XSL Style Sheet ("cdcatalog.xsl") with a transformation template:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

    <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/">
          <html>
          <body>
            <h2>My CD Collection</h2>
            <table border="1">
              <tr bgcolor="#9acd32">
                <th align="left">Title</th>
                <th align="left">Artist</th>
              </tr>
              <xsl:for-each select="catalog/cd">
              <tr>
                <td><xsl:value-of select="title"/></td>
                <td><xsl:value-of select="artist"/></td>
              </tr>
              </xsl:for-each>
            </table>
          </body>
          </html>
    </xsl:template>
    </xsl:stylesheet>
```

Template Match

The XSL namspace is matched to the whole template to transform XML to XHTML via XSLT

Link the XSL Style Sheet to the XML Document

- Add the XSL style sheet reference to your XML document ("cdcatalog.xml"):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="cdcatalog.xsl"?>
<catalog>
    <cd>
            <title>Empire Burlesque</title>
            <artist>Bob Dylan</artist>
            <country>USA</country>
            <company>Columbia</company>
            <price>10.90</price>
            <year>1985</year>
    </cd> . . .
</catalog>
```

View the result!

XSLT - Templates

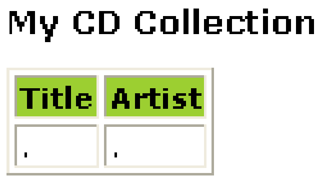
- An XSL style sheet consists of one or more set of rules that are called templates.
- Each template contains rules to apply when a specified node is matched.
- **The <xsl:template> Element:**
  - The <xsl:template> element is used to build templates.
  - The **match** attribute is used to associate a template with an XML element. The match attribute can also be used to define a template for the entire XML document. The value of the match attribute is an XPath expression (i.e. match="/" defines the whole document).

XSLT - Templates

- let's look at a simplified version of the XSL file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
    <xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/">
    <html>
    <body>
      <h2>My CD Collection</h2>
      <table border="1">
        <tr bgcolor="#9acd32">
          <th>Title</th>
          <th>Artist</th>
        </tr>
        <tr>
          <td>.</td>
          <td>.</td>
        </tr>
      </table>
    </body>
    </html>
    </xsl:template>
    </xsl:stylesheet>
```

- Since an XSL style sheet is an XML document itself, it always begins with the XML declaration: **<?xml version="1.0" encoding="ISO-8859-1"?>**.
- The next element, **<xsl:stylesheet>**, defines that this document is an XSLT style sheet document (along with the version number and XSLT namespace attributes).
- The **<xsl:template>** element defines a template. The **match="/"** attribute associates the template with the root of the XML source document.
- The content inside the <xsl:template> element defines some HTML to write to the output.
- The last two lines defines the end of the template and the end of the style sheet.
- The result of the transformation above will look like this:



My CD Collection

Title	Artist
.	.

Template HTML

The HTML in the template identifies XML elements to output

XSLT - Templates

- View the XML file, View the XSL file, and View the result
- The result from this example is a little disappointing, because no data is copied from the XML document to the output.
- Now, we will look at how to use the **<xsl:value-of>** element to select values from the XML elements.

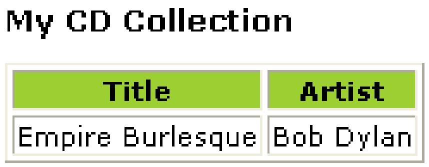
XSLT - The <xsl:value-of> Element

- The <xsl:value-of> element can be used to extract the value of an XML element and add it to the output stream of the transformation:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:template match="/">
        <html>
        <body>
           <h2>My CD Collection</h2>
           <table border="1">
                <tr bgcolor="#9acd32">
                   <th>Title</th>
                   <th>Artist</th>
                </tr>
                <tr>
                   <td> <xsl:value-of select="catalog/cd/title"/> </td>
                   <td> <xsl:value-of select="catalog/cd/artist"/> </td>
                </tr>
           </table>
        </body>
        </html>
   </xsl:template>
   </xsl:stylesheet>
```

XSLT - The <xsl:value-of> Element

- **Note:** The value of the **select** attribute is an XPath expression. An XPath expression works like navigating a file system; where a forward slash (/) selects subdirectories.
- The result of the transformation above will look like this:



My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan

- View the XML file, View the XSL file, and View the result
- The result from this example is also a little disappointing, because only one line of data is copied from the XML document to the output.
- We will now look at how to use the **<xsl:for-each>** element to loop through the XML elements, and display all of the records.

XSL Namespace / XPath

The XSL namespace for the template allows extraction of values from XML - (<xsl:value-of>)

XPath - The value uses an Xpath expression to navigate the XML - ("catalog/cd/title")

Attribute - Select: <xsl:value-of select="catalog/cd/title"/>

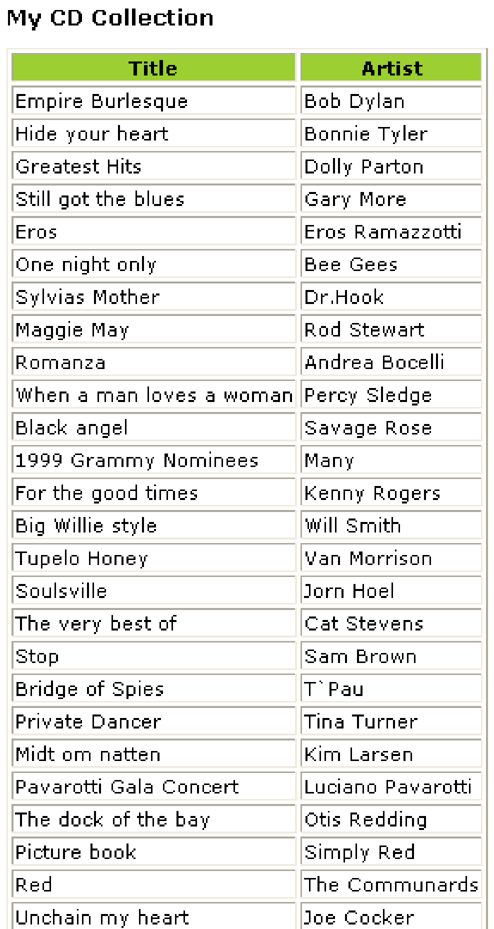
XSLT - The <xsl:for-each> Element

- The <xsl:for-each> element allows you to do looping in XSLT.
- The The XSL <xsl:for-each> element can be used to select every XML element of a specified node-set:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
        <html>
        <body>
            <h2>My CD Collection</h2>
            <table border="1">
                    <tr bgcolor="#9acd32">
                        <th>Title</th>
                        <th>Artist</th>
                    </tr>
                    <xsl:for-each select="catalog/cd">
                    <tr>
                        <td> <xsl:value-of select="title"/> </td>
                        <td> <xsl:value-of select="artist"/> </td>
                    </tr>
                    </xsl:for-each>
            </table>
        </body>
        </html>
</xsl:template>
</xsl:stylesheet>
```

XSLT - The <xsl:for-each> Element

- **Note:** The value of the **select** attribute is an XPath expression. An XPath expression works like navigating a file system; where a forward slash (/) selects subdirectories.
- The result of the transformation above will look like this:



My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary More
Eros	Eros Ramazzotti
One night only	Bee Gees
Sylvias Mother	Dr.Hook
Maggie May	Rod Stewart
Romanza	Andrea Bocelli
When a man loves a woman	Percy Sledge
Black angel	Savage Rose
1999 Grammy Nominees	Many
For the good times	Kenny Rogers
Big Willie style	Will Smith
Tupelo Honey	Van Morrison
Soulsville	Jorn Hoel
The very best of	Cat Stevens
Stop	Sam Brown
Bridge of Spies	T`Pau
Private Dancer	Tina Turner
Midt om natten	Kim Larsen
Pavarotti Gala Concert	Luciano Pavarotti
The dock of the bay	Otis Redding
Picture book	Simply Red
Red	The Communards
Unchain my heart	Joe Cocker

View the XML file, View the XSL file, and View the result

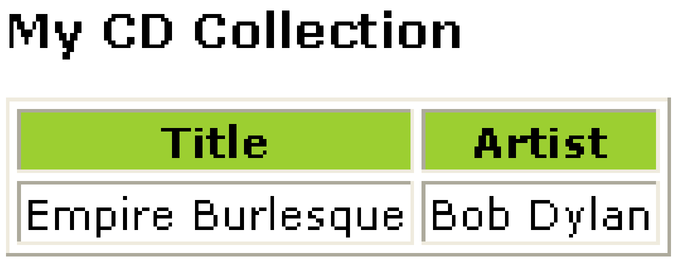
Filtering the Output

- We can also filter the output from the XML file by adding a criterion to the select attribute in the <xsl:for-each> element
- **<xsl:for-each select="catalog/cd[artist='Bob Dylan']">**
- Legal filter operators are:
  - = (equal)
  - != (not equal)
  - &lt; less than
  - &gt; greater than
- Take a look at the adjusted XSL style sheet:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
        <html>
        <body>
           <h2>My CD Collection</h2>
           <table border="1">
                       <tr bgcolor="#9acd32">
                          <th>Title</th>
                          <th>Artist</th>
                       </tr>
                       <xsl:for-each select="catalog/cd[artist='Bob Dylan']">
                       <tr>
                          <td> <xsl:value-of select="title"/> </td>
                          <td> <xsl:value-of select="artist"/> </td>
                       </tr>
                       </xsl:for-each>
           </table>
        </body>
        </html>
</xsl:template>
</xsl:stylesheet>
```

Filtering the Output

- The result of the transformation above will look like this:



My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan

- View the XML file, View the XSL file, and View the result

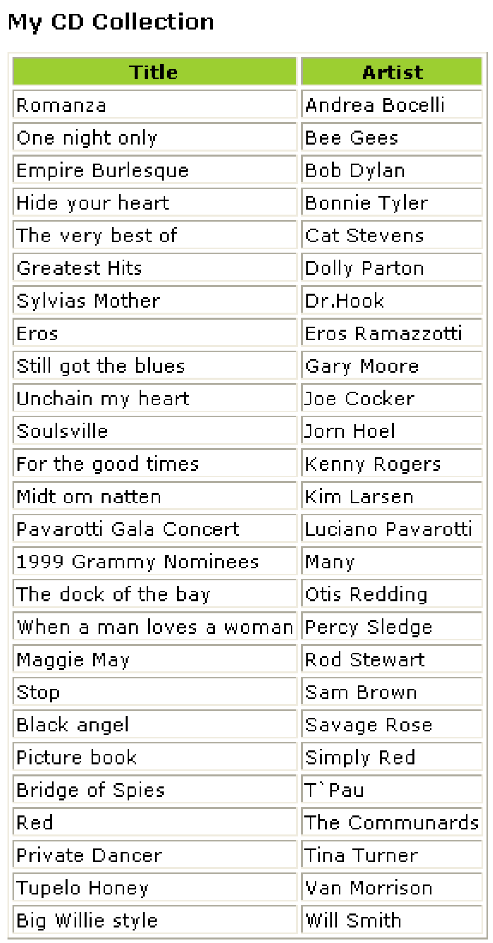
XSLT - The <xsl:sort> Element

- The <xsl:sort> element is used to sort the output.
- To sort the output, simply add an <xsl:sort> element inside the <xsl:for-each> element in the XSL file:

```
<?xml version="1.0" encoding="ISO-8859-1"?> <xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
        <html>
        <body>
            <h2>My CD Collection</h2>
            <table border="1">
                        <tr bgcolor="#9acd32">
                            <th>Title</th>
                            <th>Artist</th>
                        </tr>
                        <xsl:for-each select="catalog/cd">
                        <xsl:sort select="artist"/>
                        <tr>
                            <td> <xsl:value-of select="title"/> </td>
                            <td> <xsl:value-of select="artist"/> </td>
                        </tr>
                        </xsl:for-each>
            </table>
        </body>
        </html>
</xsl:template>
</xsl:stylesheet>
```

XSLT - The <xsl:sort> Element

- **Note:** The **select** attribute indicates what XML element to sort on.
- The result of the transformation above will look like this:



My CD Collection

Title	Artist
Romanza	Andrea Bocelli
One night only	Bee Gees
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
The very best of	Cat Stevens
Greatest Hits	Dolly Parton
Sylvias Mother	Dr.Hook
Eros	Eros Ramazzotti
Still got the blues	Gary Moore
Unchain my heart	Joe Cocker
Soulsville	Jorn Hoel
For the good times	Kenny Rogers
Midt om natten	Kim Larsen
Pavarotti Gala Concert	Luciano Pavarotti
1999 Grammy Nominees	Many
The dock of the bay	Otis Redding
When a man loves a woman	Percy Sledge
Maggie May	Rod Stewart
Stop	Sam Brown
Black angel	Savage Rose
Picture book	Simply Red
Bridge of Spies	T`Pau
Red	The Communards
Private Dancer	Tina Turner
Tupelo Honey	Van Morrison
Big Willie style	Will Smith

View the XML file, View the XSL file, and View the result

XSLT - The <xsl:if> Element

- The <xsl:if> element is used to put a conditional test against the content of the XML file.
- Syntax:

```
<xsl:if test="expression">
      ...
      ...some output if the expression is true...
      ...
</xsl:if>
```

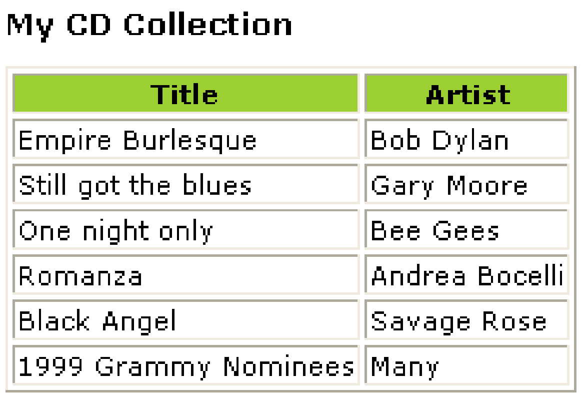
- To add a conditional test, add the <xsl:if> element inside the <xsl:for-each> element in the XSL file.

XSLT - The <xsl:if> Element

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/">
          <html>
          <body>
             <h2>My CD Collection</h2>
             <table border="1">
                      <tr bgcolor="#9acd32">
                         <th>Title</th>
                         <th>Artist</th>
                      </tr>
                      <xsl:for-each select="catalog/cd">
                      <xsl:if test="price &gt; 10">
                      <tr>
                         <td> <xsl:value-of select="title"/> </td>
                         <td> <xsl:value-of select="artist"/> </td>
                      </tr>
                      </xsl:if>
                      </xsl:for-each>
             </table>
          </body>
          </html>
    </xsl:template>
    </xsl:stylesheet>
```

XSLT - The <xsl:if> Element

- **Note:** The value of the required **test** attribute contains the expression to be evaluated.
- The code above will only output the title and artist elements of the CDs that has a price that is higher than 10.
- The result of the transformation above will look like this:



My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Still got the blues	Gary Moore
One night only	Bee Gees
Romanza	Andrea Bocelli
Black Angel	Savage Rose
1999 Grammy Nominees	Many

View the XML file, View the XSL file, and View the result

XSLT - The <xsl:choose> Element

- To express a multiple conditional test against the content of the XML file, add an <xsl:choose> element to the XSL document.
- Syntax:

```
<xsl:choose>
        <xsl:when test="expression">
                ... some output ...
        </xsl:when>
        <xsl:otherwise>
                ... some output ....
        </xsl:otherwise>
</xsl:choose>
```

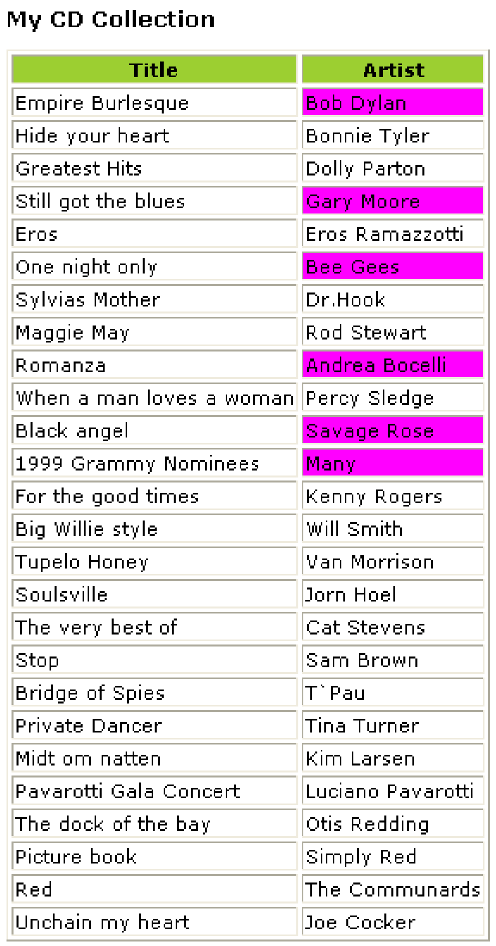
- To insert a conditional choose test against the content of the XML file, add the <xsl:choose>, <xsl:when>, and <xsl:otherwise> elements to the XSL file

XSLT - The <xsl:choose> Element

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/">
          <html>
          <body>
             <h2>My CD Collection</h2>
             <table border="1">
                      <tr bgcolor="#9acd32">
                         <th>Title</th>
                         <th>Artist</th>
                      </tr>
                      <xsl:for-each select="catalog/cd">
                      <tr>
                         <td> <xsl:value-of select="title"/> </td>
                         <xsl:choose>
                         <xsl:when test="price &gt; 10">
                           <td bgcolor="#ff00ff">
                           <xsl:value-of select="artist"/>
                           </td>
                         </xsl:when>
                         <xsl:otherwise>
                           <td><xsl:value-of select="artist"/></td>
                           </xsl:otherwise>
                         </xsl:choose>
                      </tr>
                      </xsl:for-each>
             </table>
          </body>
          </html>
    </xsl:template>
    </xsl:stylesheet>
```

XSLT - The <xsl:choose> Element

- The code above will add a pink background-color to the "Artist" column WHEN the price of the CD is higher than 10.
- The result of the transformation above will look like this:



My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore
Eros	Eros Ramazzotti
One night only	Bee Gees
Sylvias Mother	Dr.Hook
Maggie May	Rod Stewart
Romanza	Andrea Bocelli
When a man loves a woman	Percy Sledge
Black angel	Savage Rose
1999 Grammy Nominees	Many
For the good times	Kenny Rogers
Big Willie style	Will Smith
Tupelo Honey	Van Morrison
Soulsville	Jorn Hoel
The very best of	Cat Stevens
Stop	Sam Brown
Bridge of Spies	T`Pau
Private Dancer	Tina Turner
Midt om natten	Kim Larsen
Pavarotti Gala Concert	Luciano Pavarotti
The dock of the bay	Otis Redding
Picture book	Simply Red
Red	The Communards
Unchain my heart	Joe Cocker

View the XML file, View the XSL file, and View the result

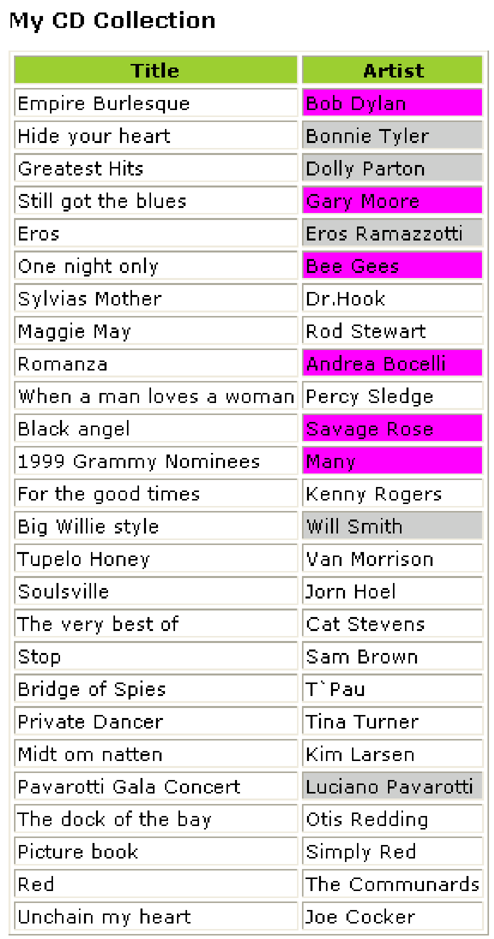
XSLT - The <xsl:choose> Element

- It is also possible to have more than one <xsl:when> element:

```
<xsl:choose>
        <xsl:when test="price &gt; 10">
                        <td bgcolor="#ff00ff">
                        <xsl:value-of select="artist"/></td>
        </xsl:when>
        <xsl:when test="price &gt; 9">
                        <td bgcolor="#cccccc">
                        <xsl:value-of select="artist"/></td>
        </xsl:when>
        <xsl:otherwise>
                        <td><xsl:value-of select="artist"/></td>
        </xsl:otherwise>
</xsl:choose>
```

XSLT - The <xsl:choose> Element

- The code above will add a pink background color to the "Artist" column WHEN the price of the CD is higher than 10, and a grey background-color WHEN the price of the CD is higher than 9 and lower or equal to 10.
- The result of the transformation will look like this:



My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore
Eros	Eros Ramazzotti
One night only	Bee Gees
Sylvias Mother	Dr.Hook
Maggie May	Rod Stewart
Romanza	Andrea Bocelli
When a man loves a woman	Percy Sledge
Black angel	Savage Rose
1999 Grammy Nominees	Many
For the good times	Kenny Rogers
Big Willie style	Will Smith
Tupelo Honey	Van Morrison
Soulsville	Jorn Hoel
The very best of	Cat Stevens
Stop	Sam Brown
Bridge of Spies	T`Pau
Private Dancer	Tina Turner
Midt om natten	Kim Larsen
Pavarotti Gala Concert	Luciano Pavarotti
The dock of the bay	Otis Redding
Picture book	Simply Red
Red	The Communards
Unchain my heart	Joe Cocker

View the XML file, View the XSL file, and View the result

XSLT - attributes

```
<xsl:attribute name="attributename”>
  <!-- Content:template -->
</xsl:attribute>
```

Eg: add an image:

```
<img>
  <xsl:attribute name=”src">
     <xsl:value-of select="images/name" />
  </xsl:attribute>
</img>
```

XSLT - The <xsl:apply-templates> Element

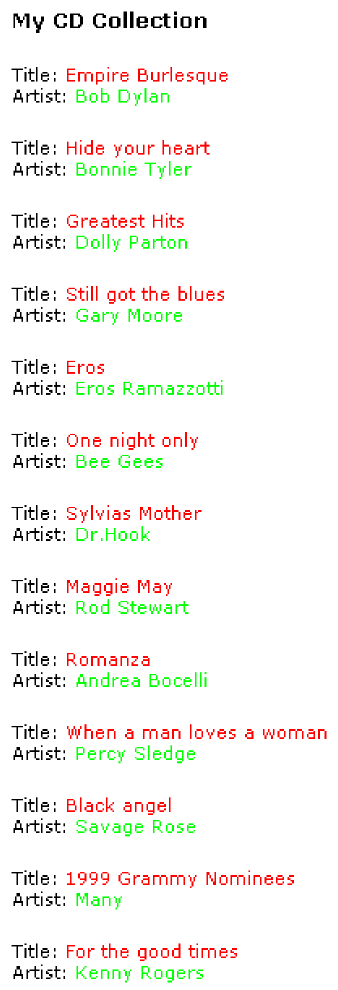
- The <xsl:apply-templates> element applies a template to the current element or to the current element's child nodes.
- If we add a select attribute to the <xsl:apply-templates> element it will process only the child element that matches the value of the attribute. We can use the select attribute to specify the order in which the child nodes are processed.

XSLT - The <xsl:apply-templates> Element

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0“
xmlns:xsl="http://www.w3.org/1999/XSL/Tran
    sform">

<xsl:template match="/">
<html>
<body>
<h2>My CD Collection</h2>
<xsl:apply-templates/>
</body>
</html>

</xsl:template>
<xsl:template match="cd">
<p>
<xsl:apply-templates select="title"/>
<xsl:apply-templates select="artist"/>
</p>
</xsl:template>

<xsl:template match="title">
Title: <span style="color:#ff0000">
<xsl:value-of select="."/></span>
<br />
</xsl:template>

<xsl:template match="artist">
 Artist: <span style="color:#00ff00">
 <xsl:value-of select="."/></span>
 <br />
 </xsl:template>

</xsl:stylesheet>
```

XSLT - The <xsl:choose> Element

- The result of the transformation above will look like this:



My CD Collection

Title: Empire Burlesque
Artist: Bob Dylan

Title: Hide your heart
Artist: Bonnie Tyler

Title: Greatest Hits
Artist: Dolly Parton

Title: Still got the blues
Artist: Gary Moore

Title: Eros
Artist: Eros Ramazzotti

Title: One night only
Artist: Bee Gees

Title: Sylvias Mother
Artist: Dr.Hook

Title: Maggie May
Artist: Rod Stewart

Title: Romanza
Artist: Andrea Bocelli

Title: When a man loves a woman
Artist: Percy Sledge

Title: Black angel
Artist: Savage Rose

Title: 1999 Grammy Nominees
Artist: Many

Title: For the good times
Artist: Kenny Rogers

View the XML file, View the XSL file, and View the result

XSLT - On the Client

- If your browser supports it, XSLT can be used to transform the document to XHTML in your browser.
- **A JavaScript Solution:**
- In the previous slides we have looked at how XSLT can be used to transform a document from XML to XHTML. We did this by adding an XSL style sheet to the XML file and let the browser do the transformation.
- Even if this works fine, it is not always desirable to include a style sheet reference in an XML file (e.g. it will not work in a non XSLT aware browser.)
- A more versatile solution would be to use a JavaScript to do the transformation.
- By using a JavaScript, we can:
  - do browser-specific testing
  - use different style sheets according to browser and user needs
- That is the beauty of XSLT! One of the design goals for XSLT was to make it possible to transform data from one format to another, supporting different browsers and different user needs.
- XSLT transformation on the client side is bound to be a major part of the browsers work tasks in the future, as we will see a growth in the specialized browser market (Braille, aural browsers, Web printers, handheld devices, etc.)

XSLT - On the Client

- Look at the XML document that you have seen in the previous slides:

```
<?xml version="1.0" encoding="ISO-
8859-1"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia </company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  . . .
</catalog>
```

View the XML file

- And the accompanying XSL style sheet:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform
">
<xsl:template match="/">
  <html>
  <body>
    <h2>My CD Collection</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th align="left">Title</th>
        <th align="left">Artist</th>
      </tr>
      <xsl:for-each select="catalog/cd">
      <tr>
        <td><xsl:value-of select="title" /></td>
        <td><xsl:value-of select="artist" /></td>
      </tr>
      </xsl:for-each>
    </table>
  </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

View the XSL file

XSLT - On the Client

Transforming XML to XHTML in the Browser:

- Here is the source code needed to transform the XML file to XHTML on the client:

```
<html>
<body>
<script type="text/javascript">

// Load XML
var xml = new ActiveXObject("Microsoft.XMLDOM")
xml.async = false
xml.load("cdcatalog.xml")

// Load XSL
var xsl = new ActiveXObject("Microsoft.XMLDOM")
xsl.async = false
xsl.load("cdcatalog.xsl")

// Transform
document.write(xml.transformNode(xsl))
</script>
</body>
</html>
```

- The first block of code creates an instance of the Microsoft XML parser (XMLDOM), and loads the XML file into memory.
- The second block of code creates another instance of the parser and loads the XSL file into memory.
- The last line of code transforms the XML document using the XSL document, and displays the result as XHTML in your browser. Nice!

XSLT on the Server

- Using PHP

```
<?php
$xslDoc = new DOMDocument();
$xslDoc->load("cdcatalog.xsl");

$xmlDoc = new DOMDocument();
$xmlDoc->load("cdcatalog.xml");

$proc = new XSLTProcessor();
$proc->importStylesheet($xslDoc);

echo $proc->transformToXML($xmlDoc);
?>
```

More on XSLT

- See W3Schools for XSLT Elements, Functions and more:

  http://www.w3schools.com/xsl/default.asp

CW1

Due 17th March 10:00 AM on Blackboard

London Attractions - XML, XSL, HTML, JQuery Mobile