XML and Databases
XSLT Stylesheets and Transforms

Kim.Nguyen@nicta.com.au

Lecture 11

Outline

XML: data publication

One source, multiple views

From a single XML file, we want to generate:

- an HTML webpage, to publish the data on the web

XML: data publication

One source, multiple views

From a single XML file, we want to generate:

- an HTML webpage, to publish the data on the web
- a WML webpage, for mobile devices

XML: data publication

One source, multiple views

From a single XML file, we want to generate:

- an HTML webpage, to publish the data on the web
- a WML webpage, for mobile devices
- a CSV file, to easily import into a database/spreadsheet

XML: data publication

One source, multiple views

From a single XML file, we want to generate:

- an HTML webpage, to publish the data on the web
- a WML webpage, for mobile devices
- a CSV file, to easily import into a database/spreadsheet
- a PDF file for printing

XML: data publication

One source, multiple views

From a single XML file, we want to generate:

- an HTML webpage, to publish the data on the web
- a WML webpage, for mobile devices
- a CSV file, to easily import into a database/spreadsheet
- a PDF file for printing
- another XML file, with different tag names or layout

XML: data publication

One source, multiple views

From a single XML file, we want to generate:

- an HTML webpage, to publish the data on the web
- a WML webpage, for mobile devices
- a CSV file, to easily import into a database/spreadsheet
- a PDF file for printing
- another XML file, with different tag names or layout

XML: data publication

One source, multiple views

From a single XML file, we want to generate:

- an HTML webpage, to publish the data on the web
- a WML webpage, for mobile devices
- a CSV file, to easily import into a database/spreadsheet
- a PDF file for printing
- another XML file, with different tag names or layout

One solution: XSLT
XSLT stylesheet: *set of rules* to transform the XML input file

- Version 1.0 introduced in 1999
- Version 2.0 introduced recently

Our goal today

We use the `bib.xml` sample file :

```
<?xml version="1.0" ?>
<bib>
    <book year="1994">
        <title>TCP/IP Illustrated</title>
        <author><last>Stevens</last><first>W.</first></author>
        <publisher>Addison-Wesley</publisher>
        <price>65.95</price>
    </book>
...
```

and publish it as an HTML web page using an XSLT 1.0 stylesheet!

How to apply an XSLT stylesheet

1. use an external tool

```
xsltproc mystyle.xsl myfile.xml > output.html
```

How to apply an XSLT stylesheet

1. use an external tool

```
xsltproc mystyle.xsl myfile.xml > output.html
```

2. Embed the stylesheet and use a web browser (e.g. `firefox`)

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="mystyle.xsl"?>
<bib>
  <book year="1994">
  ...
```

How to apply an XSLT stylesheet

1. use an external tool

   ```
   xsltproc mystyle.xsl myfile.xml > output.html
   ```

2. Embed the stylesheet and use a web browser (e.g. `firefox`)

   ```
   <?xml version="1.0" ?>
   <?xml-stylesheet type="text/xsl" href="mystyle.xsl"?>
   <bib>
     <book year="1994">
     ...
   ```

3. Embed the stylesheet and use an web server (e.g. `tomcat`) to publish the page (*à la* PHP)

How to apply an XSLT stylesheet

1. use an external tool

   ```
   xsltproc mystyle.xsl myfile.xml > output.html
   ```

2. Embed the stylesheet and use a web browser (e.g. `firefox`)

   ```
   <?xml version="1.0" ?>
   <?xml-stylesheet type="text/xsl" href="mystyle.xsl"?>
   <bib>
     <book year="1994">
     ...
   ```

3. Embed the stylesheet and use an web server (e.g. `tomcat`) to publish the page (*à la* PHP)

How to apply an XSLT stylesheet

1. use an external tool

   ```
   xsltproc mystyle.xsl myfile.xml > output.html
   ```

2. Embed the stylesheet and use a web browser (e.g. `firefox`)

   ```
   <?xml version="1.0" ?>
   <?xml-stylesheet type="text/xsl" href="mystyle.xsl"?>
   <bib>
     <book year="1994">
     ...
   ```

3. Embed the stylesheet and use an web server (e.g. `tomcat`) to publish the page (*à la* PHP)

```
<?xml version="1.0" ?>
  <bib>
    <book year="1994">
        <title>TCP/IP Illustrated</title>
        <author><last>Stevens</last><first>W.</first></author>
        <publisher>Addison-Wesley</publisher>
        <price>65.95</price>
    </book>

    <book year="1992">
        <title>Advanced Programming in the Unix environment</title>
        <author><last>Stevens</last><first>W.</first></author>
        <publisher>Addison-Wesley</publisher>
        <price>65.95</price>
    </book>

    <book year="2000">
        <title>Data on the Web</title>
        <author><last>Abiteboul</last><first>Serge</first></author>
        <author><last>Buneman</last><first>Peter</first></author>
        <author><last>Suciu</last><first>Dan</first></author>
        <publisher>Morgan Kaufmann Publishers</publisher>
        <price>39.95</price>
    </book>

    <book year="1999">
        <title>The Economics of Technology and Content for Digital TV</title>
        <editor>
               <last>Gerbarg</last><first>Darcy</first>
                <affiliation>CITI</affiliation>
        </editor>
            <publisher>Kluwer Academic Publishers</publisher>
        <price>129.95</price>
    </book>

</bib>
```

```
<!ELEMENT bib  (book* )>
<!ELEMENT book  (title,  (author+ | editor+ ), publisher, price )>
<!ATTLIST book  year CDATA  #REQUIRED >
<!ELEMENT author  (last, first )>
<!ELEMENT editor  (last, first, affiliation )>
<!ELEMENT title  (#PCDATA )>
<!ELEMENT last  (#PCDATA )>
<!ELEMENT first  (#PCDATA )>
<!ELEMENT affiliation  (#PCDATA )>
<!ELEMENT publisher  (#PCDATA )>
<!ELEMENT price  (#PCDATA )>
```

All the examples are on the CS4317 webpage.

Outline

XSLT, the programming language

XSLT is a programming language *written in XML*: an XSLT stylesheet is a *valid* XML file.

XSLT, the programming language

XSLT is a programming language *written in XML*: an XSLT stylesheet is a *valid* XML file.

XSLT relies on *templates*: set of rules that can be *applied* on particular parts of the input document.

XSLT, the programming language

XSLT is a programming language *written in XML*: an XSLT stylesheet is a *valid* XML file.

XSLT relies on *templates*: set of rules that can be *applied* on particular parts of the input document.

The *"particular parts"* are selected by XPath queries

XSLT, the programming language

XSLT is a programming language *written in XML*: an XSLT stylesheet is a *valid* XML file.

XSLT relies on *templates*: set of rules that can be *applied* on particular parts of the input document.

The *"particular parts"* are selected by XPath queries

Let's dive in!

Simple stylesheet

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
  <html>
      <body>
        <h1>Bibliography</h1>
      </body>
  </html>
  </xsl:template>
</xsl:stylesheet>
```

Output


file:///home/kim/lecture11/bib.xml
Leave Fullscreen
Bibliography

Two templates

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html>
      <body>
        <h1>Bibliography</h1>
          <xsl:apply-templates select="bib/*" />
      </body>
  </html>
</xsl:template>
<xsl:template match="book" >
  <h2><xsl:value-of select="title/text()"/></h2>
</xsl:template>
</xsl:stylesheet>
```

Output

Bibliography

TCP/IP Illustrated

Advanced Programming in the Unix environment

Data on the Web

The Economics of Technology and Content for Digital TV

Outline

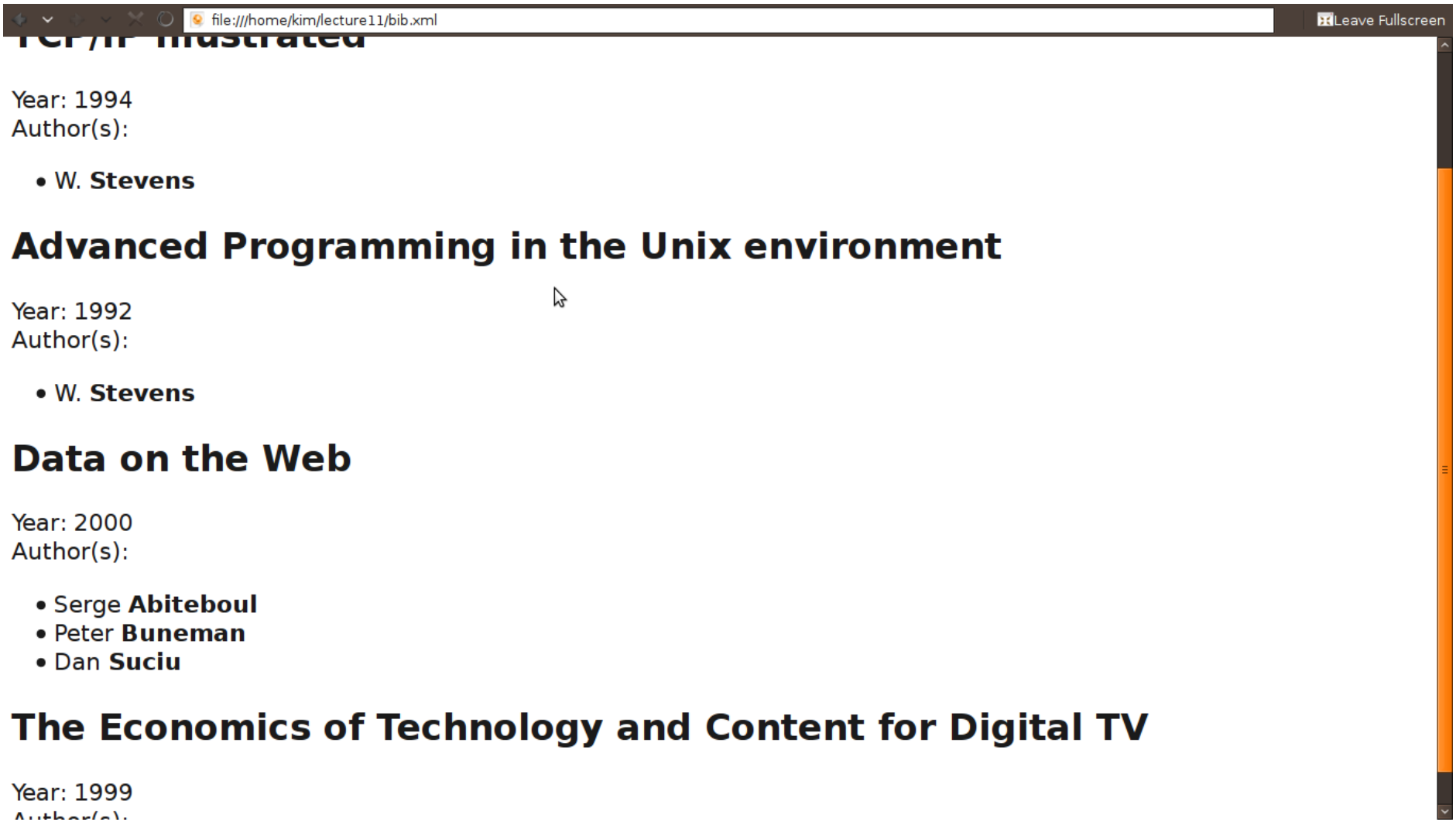
A complex book template

```
<xsl:template match="book" >
  <h2><xsl:value-of select="title/text()"/></h2>
  <p>
    Year: <xsl:value-of select="@year" /><br/>
    Author(s):
  <ul><xsl:apply-templates select="author" />
  </ul>
</p>
</xsl:template>
```

A complex book template

```
<xsl:template match="book" >
  <h2><xsl:value-of select="title/text()"/></h2>
  <p>
    Year: <xsl:value-of select="@year" /><br/>
    Author(s):
  <ul><xsl:apply-templates select="author" />
  </ul>
</p>
</xsl:template>
<xsl:template match="author" >
  <li><xsl:value-of select="first" />
    <xsl:text> </xsl:text>
    <b><xsl:value-of select="last"/></b>
  </li>
</xsl:template>
```

Output

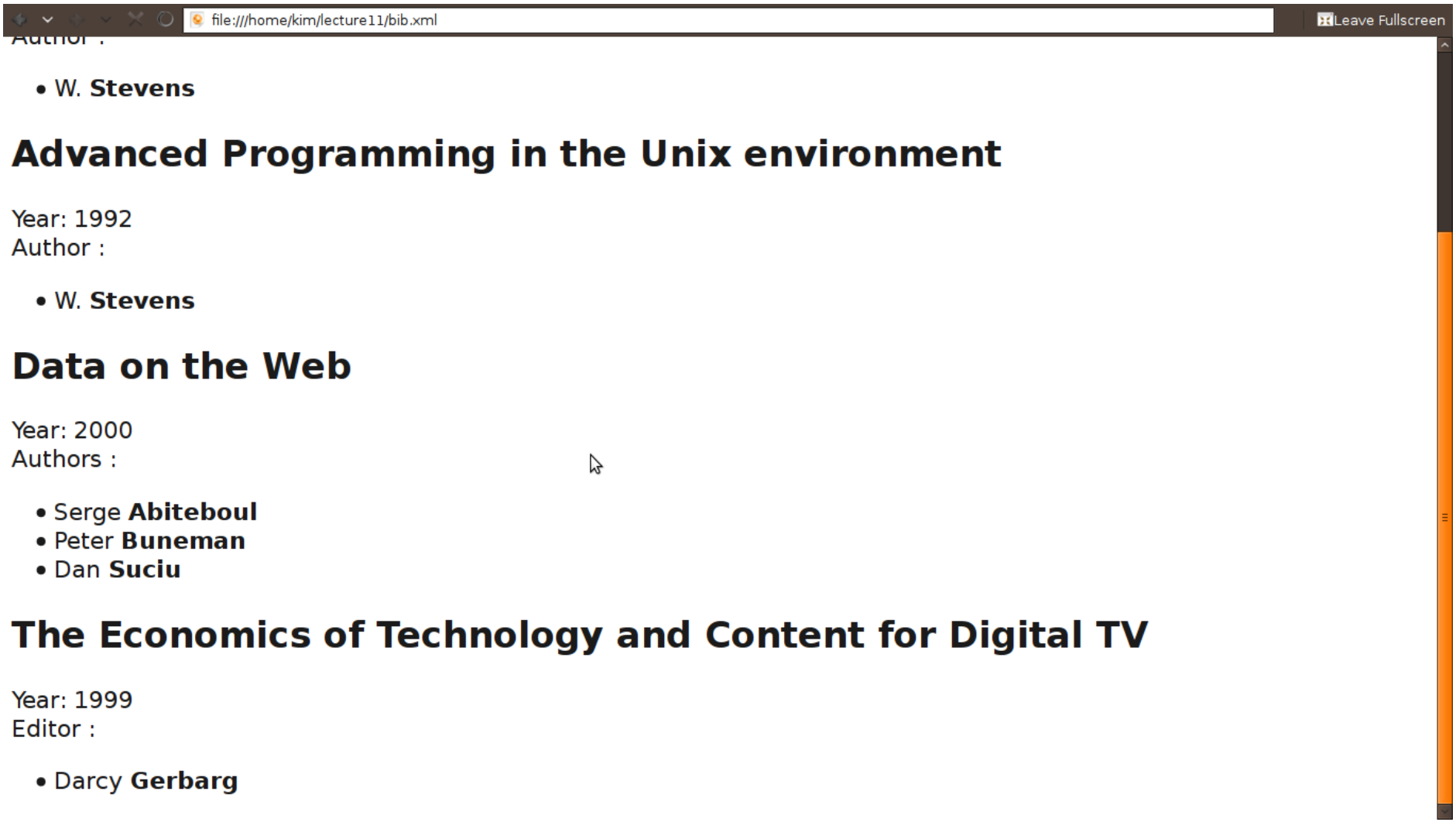

file:///home/kim/lecture11/bib.xml
Leave Fullscreen
TCP/IP Illustrated
Year: 1994
Author(s):
W. Stevens
Advanced Programming in the Unix environment
Year: 1992
Author(s):
W. Stevens
Data on the Web
Year: 2000
Author(s):
Serge Abiteboul
Peter Buneman
Dan Suciu
The Economics of Technology and Content for Digital TV
Year: 1999

Outline

Add the list of editors

```
<xsl:template match="book" >
  <h2><xsl:value-of select="title/text()"/></h2>
  <p>
    Year: <xsl:value-of select="@year" /><br/>
    <xsl:if test="author">Author</xsl:if>
    <xsl:if test="editor">Editor</xsl:if>
    <xsl:if test="(author|editor)[2]">s</xsl:if>
    :
  <ul><xsl:apply-templates select="author|editor" />
  </ul>
</p>
</xsl:template>
<xsl:template match="author|editor">
  <li><xsl:value-of select="first" />
    <xsl:text> </xsl:text>
    <b><xsl:value-of select="last"/></b>
  </li>
</xsl:template>
```

Output


file:///home/kim/lecture11/bib.xml
Leave Fullscreen
Author :
• W. Stevens
Advanced Programming in the Unix environment
Year: 1992
Author :
• W. Stevens
Data on the Web
Year: 2000
Authors :
• Serge Abiteboul
• Peter Buneman
• Dan Suciu
The Economics of Technology and Content for Digital TV
Year: 1999
Editor :
• Darcy Gerbarg

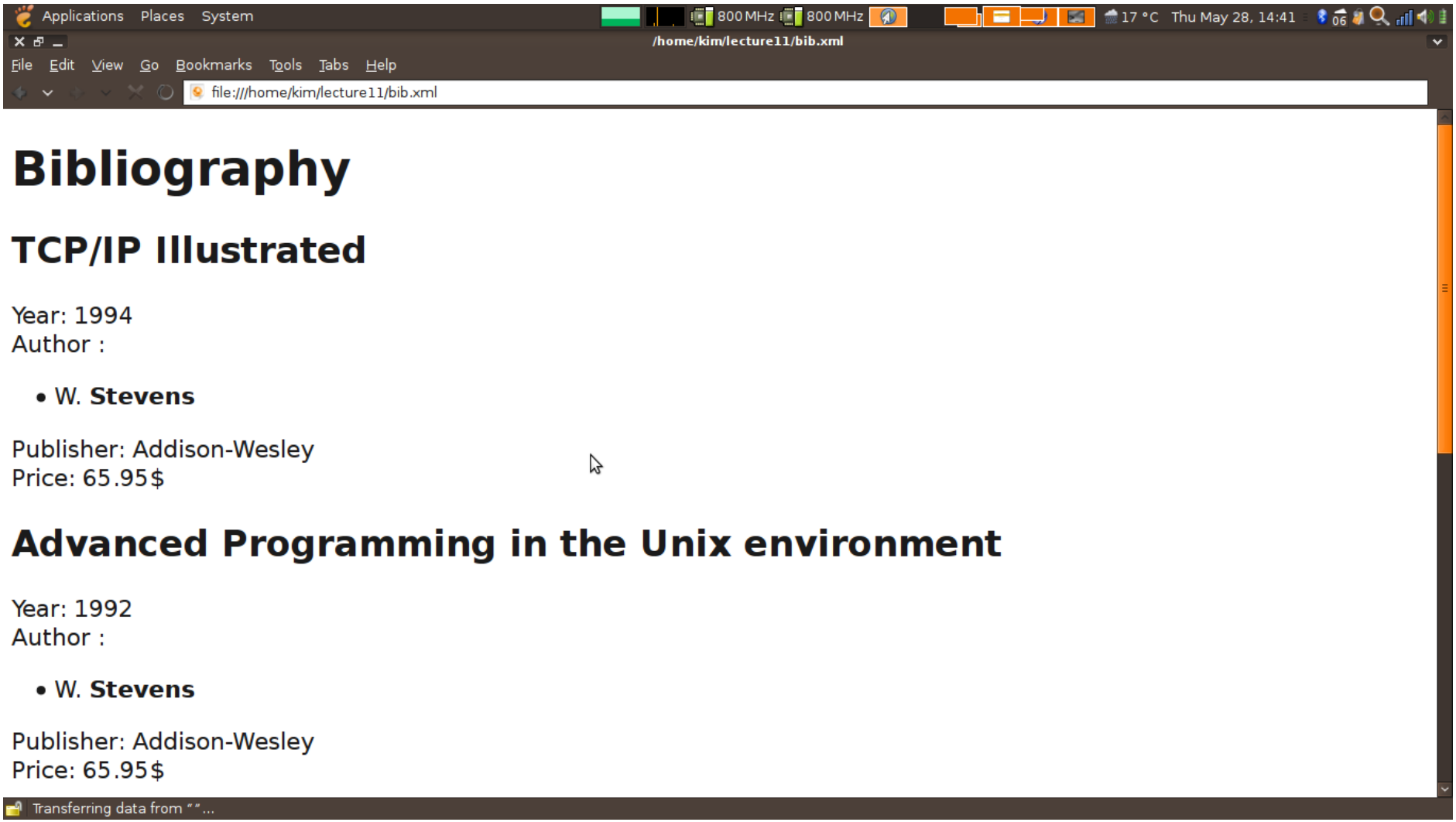
Another form of conditional

```
Year: <xsl:value-of select="@year" /><br/>
<xsl:choose>
 <xsl:when test="author">Author</xsl:when>
 <xsl:otherwise>Editor</xsl:otherwise>
</xsl:choose>
<xsl:if test="(author|editor)[2]">s</xsl:if>
```

Another form of conditional

```
<xsl:template match="book" >
  <h2><xsl:value-of select="title/text()"/></h2>
  <p>
    Year: <xsl:value-of select="@year" /><br/>
    <xsl:choose>
      <xsl:when test="author">Author</xsl:when>
      <xsl:otherwise>Editor</xsl:otherwise>
    </xsl:choose>
    <xsl:if test="(author|editor)[2]">s</xsl:if>
    :
  <ul><xsl:apply-templates select="author|editor" />
  </ul>
Publisher: <xsl:value-of select="publisher"/><br/>
  Price: <xsl:value-of select="price"/>
</p>
</xsl:template>
```

Output


Applications Places System
800 MHz 800 MHz
17 °C Thu May 28, 14:41
/home/kim/lecture11/bib.xml
File Edit View Go Bookmarks Tools Tabs Help
file:///home/kim/lecture11/bib.xml
Bibliography
TCP/IP Illustrated
Year: 1994
Author :
W. Stevens
Publisher: Addison-Wesley
Price: 65.95$
Advanced Programming in the Unix environment
Year: 1992
Author :
W. Stevens
Publisher: Addison-Wesley
Price: 65.95$
Transferring data from "" ...

Outline

for-each statement

```
<h1>List of co-authors</h1>
<xsl:for-each
```

for-each statement

```
<h1>List of co-authors</h1>
<xsl:for-each
       select="//author[(. != ../preceding-sibling::*/author)
       or (. = ../preceding-sibling::*/author)]">
```

for-each statement

```
<h1>List of co-authors</h1>
<xsl:for-each
       select="//author[(. != ../preceding-sibling::*/author)
       or (. = ../preceding-sibling::*/author)]">

    <xsl:variable name="currentauthor" select="string(self::*)"/>
    <xsl:call-template name="displayname">
<xsl:with-param name="myelement" select="self::*"/>
```

for-each statement

```
  <h1>List of co-authors</h1>
  <xsl:for-each
         select="//author[(. != ../preceding-sibling::*/author)
         or (. = ../preceding-sibling::*/author)]">

      <xsl:variable name="currentauthor" select="string(self::*)"/>
      <xsl:call-template name="displayname">
  <xsl:with-param name="myelement" select="self::*"/>
  </xsl:call-template> :
<xsl:for-each
         select="//author[ . = $currentauthor ]/preceding-sibling::a

<xsl:call-template name="displayname">
  <xsl:with-param name="myelement" select="self::*"/>
</xsl:call-template><xsl:text> </xsl:text>
</xsl:for-each><br/>
    </xsl:for-each>
```

Named templates

```
<xsl:template name="displayname">
  <xsl:param name="myelement" select="." />
  <xsl:value-of select="$myelement/first" />
  <xsl:text> </xsl:text>
  <b><xsl:value-of select="$myelement/last"/></b>
</xsl:template>
```

Output


file:///home/kim/lecture11/bib.xml
Leave Fullscreen
Year: 2000
Authors :
• Serge Abiteboul
• Peter Buneman
• Dan Suciu
Publisher: Morgan Kaufmann Publishers
Price: 39.95$
The Economics of Technology and Content for Digital TV
Year: 1999
Editor :
• Darcy Gerbarg
Publisher: Kluwer Academic Publishers
Price: 129.95$
List of co-authors
W. Stevens :
Serge Abiteboul : Peter Buneman Dan Suciu
Peter Buneman : Serge Abiteboul Dan Suciu
Dan Suciu : Serge Abiteboul Peter Buneman

Sorting alphabetically

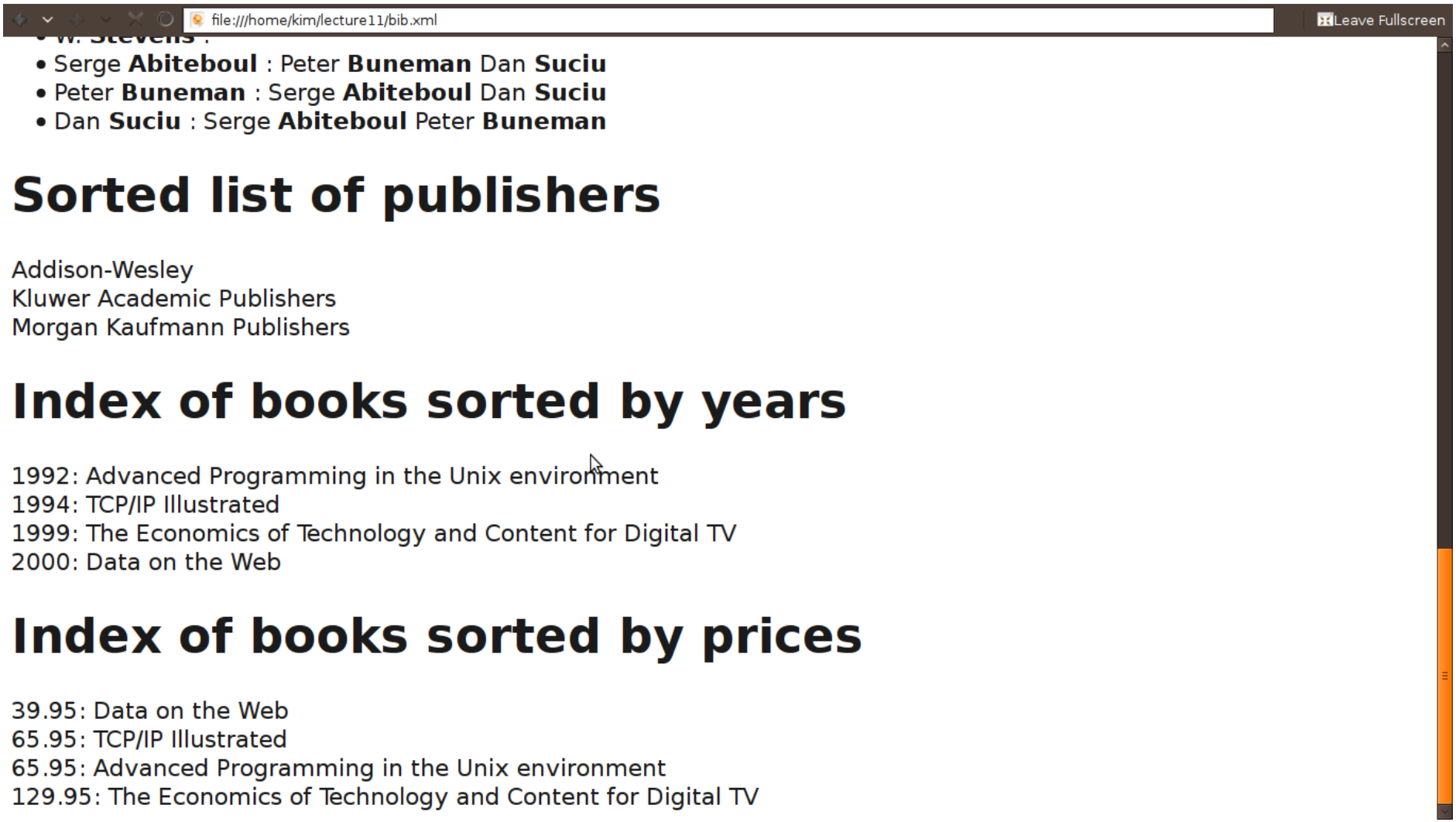
```
<h1>Sorted list of publishers</h1>
<xsl:for-each
          select="//publisher[(. != ../preceding-sibling::*/publi
 or (. = ../preceding-sibling::*/publisher)]">
  <xsl:sort select="."/>
  <xsl:value-of select="."/> <br/>
</xsl:for-each>
```

Sorting numerically

```
<h1>Index of books sorted by years</h1>
<xsl:for-each select="//book">
  <xsl:sort select="@year"/>
  <xsl:value-of select="@year"/><xsl:text>: </xsl:text>
<xsl:value-of select="title" />
<br/>
</xsl:for-each>
```

```
<h1>Index of books sorted by prices</h1>
<xsl:for-each select="//book">
  <xsl:sort select="price" data-type="number"/>
  <xsl:value-of select="price"/><xsl:text>: </xsl:text>
<xsl:value-of select="title" />
<br/>
</xsl:for-each>
```

Output


file:///home/kim/lecture11/bib.xml
Leave Fullscreen
• Serge Abiteboul : Peter Buneman Dan Suciu
• Peter Buneman : Serge Abiteboul Dan Suciu
• Dan Suciu : Serge Abiteboul Peter Buneman
Sorted list of publishers
Addison-Wesley
Kluwer Academic Publishers
Morgan Kaufmann Publishers
Index of books sorted by years
1992: Advanced Programming in the Unix environment
1994: TCP/IP Illustrated
1999: The Economics of Technology and Content for Digital TV
2000: Data on the Web
Index of books sorted by prices
39.95: Data on the Web
65.95: TCP/IP Illustrated
65.95: Advanced Programming in the Unix environment
129.95: The Economics of Technology and Content for Digital TV

Outline

Default templates

What will be the output of:

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

</xsl:stylesheet>
```

Output (surprise)

Output (surprise)

TCP/IP Illustrated StevensW. Addison-Wesley 65.95 Advanced Programming in the Unix environment StevensW. Addison-Wesley 65.95 Data on the Web AbiteboulSerge BunemanPeter SuciuDan Morgan Kaufmann Publishers 39.95 The Economics of Technology and Content for Digital TV GerbargDarcy CITI Kluwer Academic Publishers 129.95

Priorites

```
<xsl:template match="/">
  <html>
  <body>
    <xsl:apply-templates select="//*" />
  </body>
  </html>
</xsl:template>

<xsl:template match="book" >
  Book template for <xsl:value-of select="name()"/> <br/>
</xsl:template>
<xsl:template match="*" >
  Star template for <xsl:value-of select="name()"/> <br/>
</xsl:template>
```

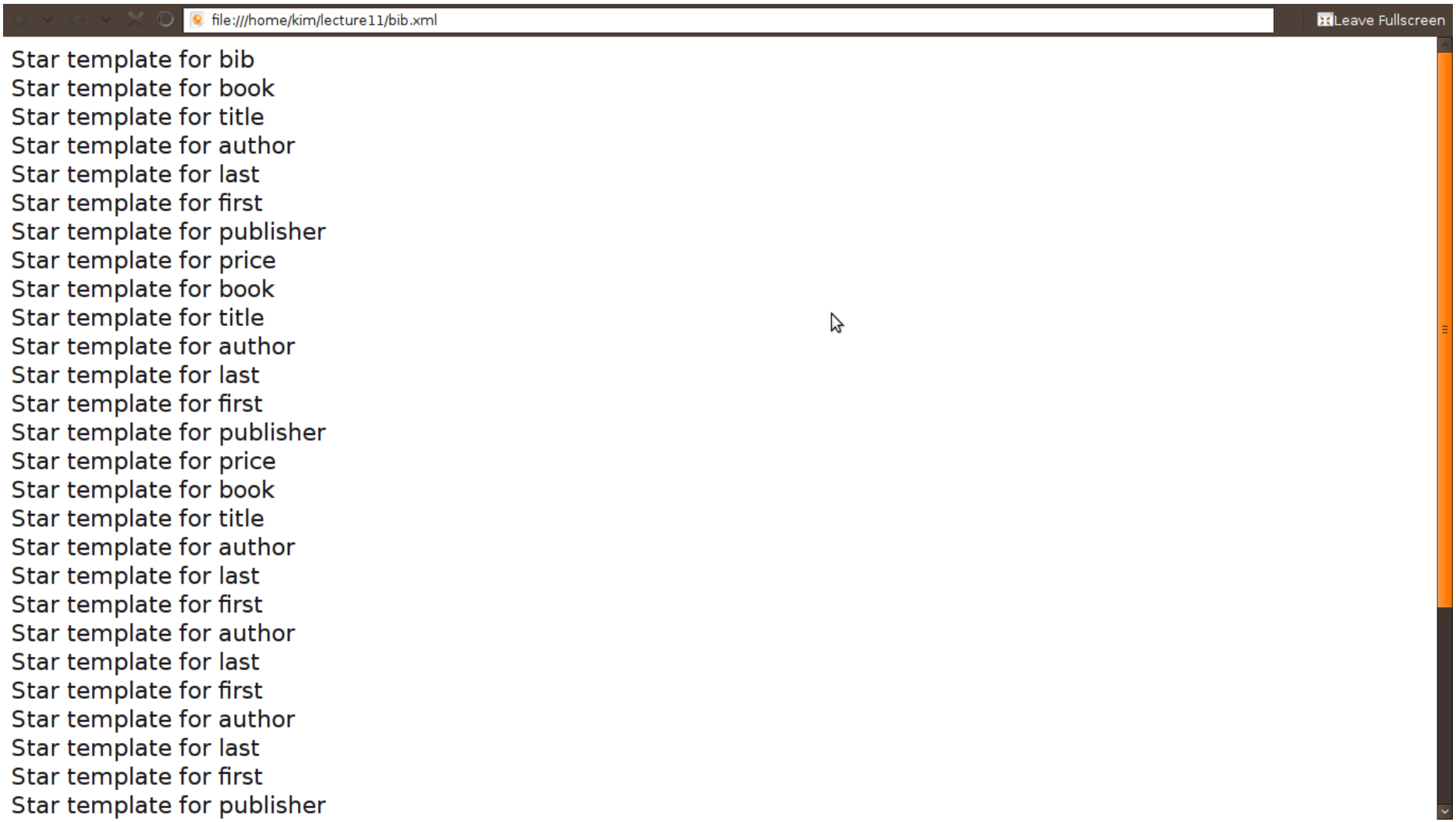
Output


file:///home/kim/lecture11/bib.xml
Leave Fullscreen
Star template for bib
Book template for book
Star template for title
Star template for author
Star template for last
Star template for first
Star template for publisher
Star template for price
Book template for book
Star template for title
Star template for author
Star template for last
Star template for first
Star template for publisher
Star template for price
Book template for book
Star template for title
Star template for author
Star template for last
Star template for first
Star template for author
Star template for last
Star template for first
Star template for author
Star template for last
Star template for first
Star template for publisher

```
<xsl:template match="/">
  <html>
  <body>
    <xsl:apply-templates select="//*" />
  </body>
  </html>
</xsl:template>

<xsl:template match="book" priority="-1">
  Book template for <xsl:value-of select="name()"/> <br/>
</xsl:template>
<xsl:template match="*" >
  Star template for <xsl:value-of select="name()"/> <br/>
</xsl:template>
```

Output

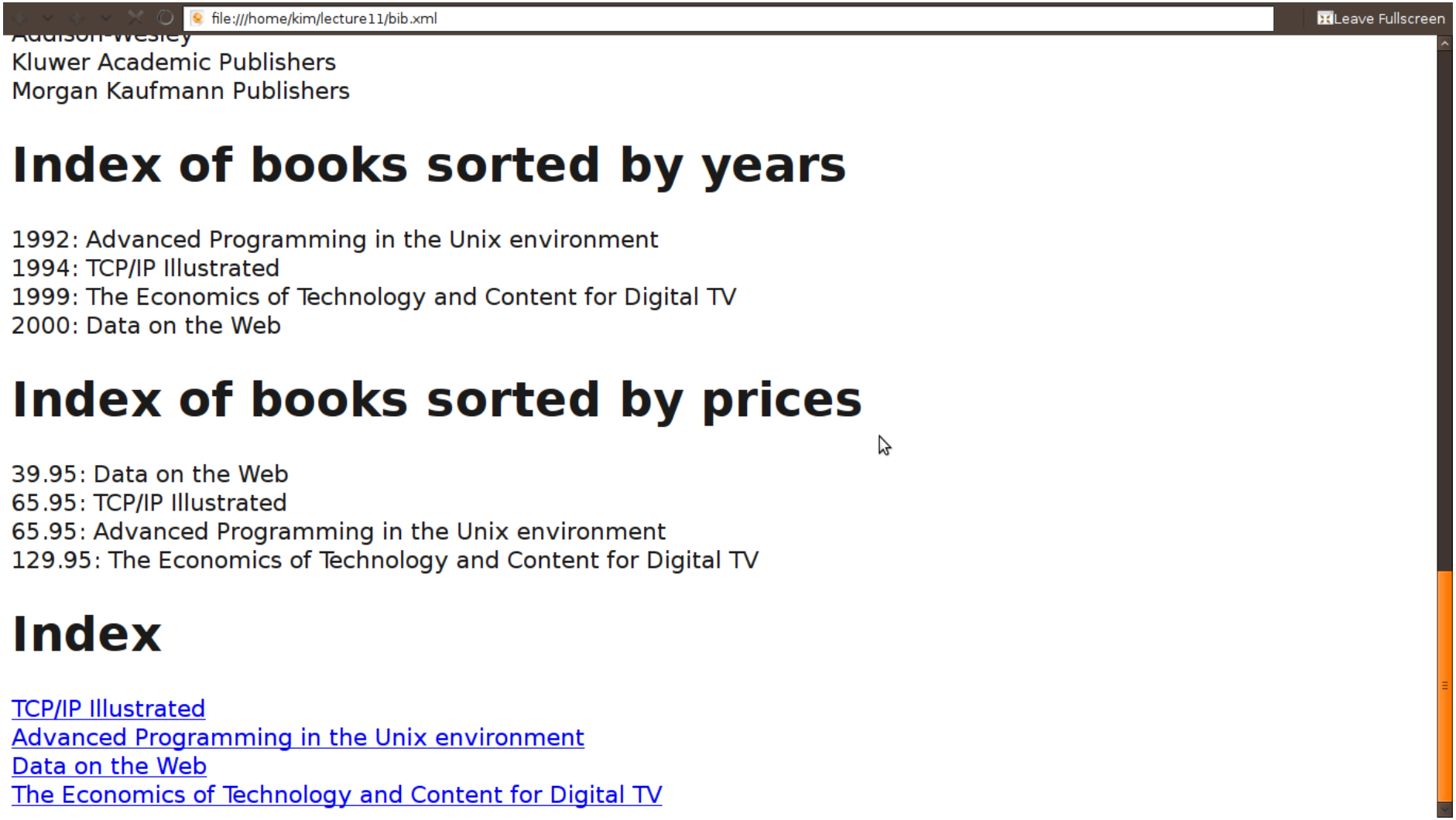

file:///home/kim/lecture11/bib.xml
Leave Fullscreen
Star template for bib
Star template for book
Star template for title
Star template for author
Star template for last
Star template for first
Star template for publisher
Star template for price
Star template for book
Star template for title
Star template for author
Star template for last
Star template for first
Star template for publisher
Star template for price
Star template for book
Star template for title
Star template for author
Star template for last
Star template for first
Star template for author
Star template for last
Star template for first
Star template for author
Star template for last
Star template for first
Star template for publisher

Modes

```
    <h1>Index</h1>
    <xsl:apply-templates select="//book" mode="lastone" />
...
<xsl:template match="book" mode="lastone">
  <a href="concat('#id_',generate-id(.))">
  <xsl:value-of select="title"/></a><br/>
</xsl:template>

<xsl:template match="book" >
  <h2><a name="concat('id_',generate-id(.))">
  <xsl:value-of select="title"/></a></h2>
  <p>
    Year: <xsl:value-of select="@year" /><br/>
    <xsl:choose>
...
```

Output


file:///home/kim/lecture11/bib.xml
Leave Fullscreen
Addison-Wesley
Kluwer Academic Publishers
Morgan Kaufmann Publishers
Index of books sorted by years
1992: Advanced Programming in the Unix environment
1994: TCP/IP Illustrated
1999: The Economics of Technology and Content for Digital TV
2000: Data on the Web
Index of books sorted by prices
39.95: Data on the Web
65.95: TCP/IP Illustrated
65.95: Advanced Programming in the Unix environment
129.95: The Economics of Technology and Content for Digital TV
Index
TCP/IP Illustrated
Advanced Programming in the Unix environment
Data on the Web
The Economics of Technology and Content for Digital TV