WTP-101
Developing Rich Internet Applications with Java Web Frameworks

Hands on Exercises

© Copyright 2008, eteration a.ş.

© Copyright eteration a.s. 2008

ITU Teknokent ARI-1 No:25

34469 Istanbul Turkey

Except for third party materials and otherwise stated, this course is made available under a Creative Commons Attribution-Share Alike 3.0 License

http://creativecommons.org/licenses/by-sa/3.0

All XML, XSD, CSS, Java source code are made available under the EPL v1.0 license

http://www.eclipse.org/legal/epl-v10.html

© Copyright eteration a.s. 2008

eteration bilisim çözümleri ticaret a.s.

ITU Teknokent ARI-1 No:25

34469 Istanbul Turkey

Except for third party materials and otherwise stated, this course is made available under a Creative Commons Attribution-Share Alike 3.0 License

http://creativecommons.org/licenses/by-sa/3.0

All XML, XSD, CSS, Java source code are made available under the EPL v1.0 license

http://www.eclipse.org/legal/epl-v10.html

Statement of copyright and this permission notice must appear in all copies of this document. If you have any form of access to a copy of this document you will be bound by the same restrictions.

eteration is a registered trademark of eteration a.s in Turkey, Germany, other countries or both. The eteration logo is a registered trademark of eteration a.s, Turkey, Germany, other countries or both.

Java™and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. IBM and Rational are trademarks of International Business Machines Corporation in the United States, other countries, or both. Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both. Other company, product, and service names may be trademarks or service marks of others.

Statement of copyright and this permission notice must appear in all copies of this document.

Information about the Hands-On Sessions

The spirit of this document lends the individual quite a lot of room for creativity. To that end, a very serious attempt has been made to avoid creating questions that simply lead the reader through the construction process. With each question, we hope that the reader is encouraged to think about the solution and the process through which that solution is determined. We hope you agree that this approach makes for a much richer learning experience.

There are often many correct solutions to these problems.

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Eteration.

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

Contents

© Copyright 2008, eteration a.ş.
Made available under a Creative Commons Attribution-Share Alike 3.0 License

Setting up your Environment

Goals
• To setup your development environment • Create a simple Web project and a Page • Start and test your application server • Learn about the TCPIP Monitor

What you should know

Before you complete this set of exercises, you should understanding the following:

- Be able create files and folders
- Change the environment variables of your system

Application Server Installation

Follow the instructions provided by your instructor to install, setup and start your application server. The software is provided on the course CD.

1. Create a folder named c:\wtp-101. This where we will keep all our software and work.
2. Use the JDK provided to you on the course CD. Use the setup tool to install it to the folder

   `c:\wtp-101\jdk1.6.0`

3. Make sure that your JDK is on the PATH. To verify that you correctly installed the JDK, open a command window and enter the following command:

   java -version

4. Create a folder named c:\wtp-101 and unzip the Application Server software to a subdirectory in this folder

   c:\wtp-101\apache-tomcat-..

5. Verify that Tomcat is installed correctly before proceeding. Tomcat includes commands for starting and stopping the server. Do the following:

   a. The startup command requires that the environment variable JAVA_HOME be set to the installation directory of the JDK. If the JDK installation process did not set JAVA_HOME correctly, set it now to the directory where you installed the JDK.

   b. To start Tomcat, open a command window and change the current directory to the bin subdirectory of the Tomcat installation directory. Then invoke this command: **startup**

   c. A second command window opens and launches Tomcat on port 8080 by default.

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

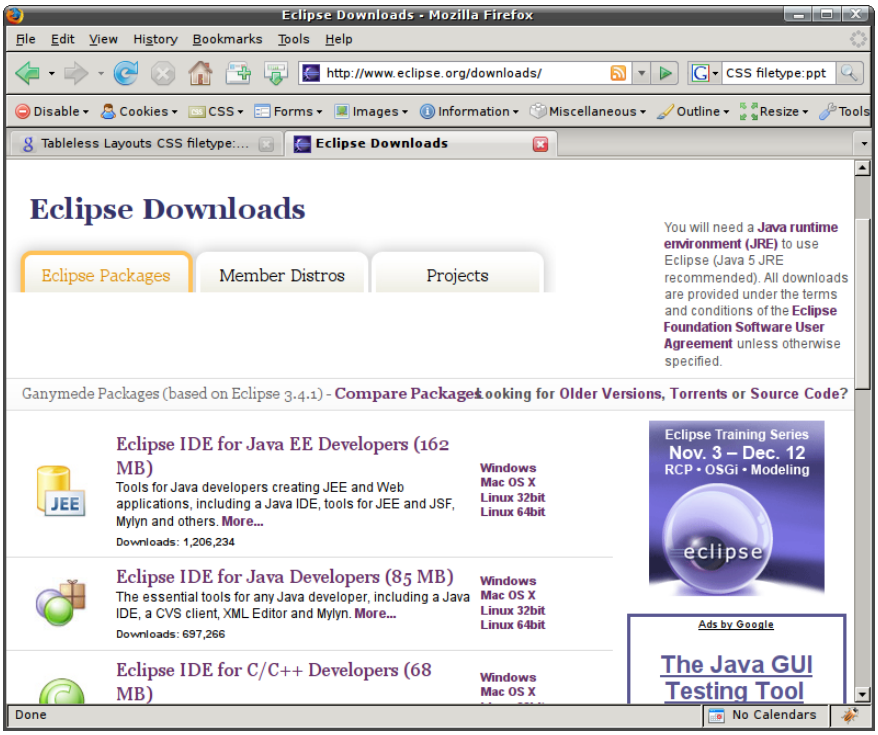
d. After the startup process completes, verify that Tomcat is running by opening this URL in a Web browser: **http://localhost:8080/**

e. You should see the Tomcat home page displaying.

f. To stop Tomcat, enter: **shutdown**

Eclipse Web Tools Platform (WTP) Installation

You will need a copy of eclipse for your platform (windows, mac, linux) with WTP to do these exercises. The software is provided at http://www.eclipse.org/downloads/. The easiest way to install is to choose an Eclipse Package with WTP in it. To do this download the "Eclipse IDE for Java EE Developers" packaging, and unarchive it in a convenient directory. This archive includes WTP and all of its Eclipse prerequisites, including Eclipse itself.



© Copyright 2008, eteration a.ş.

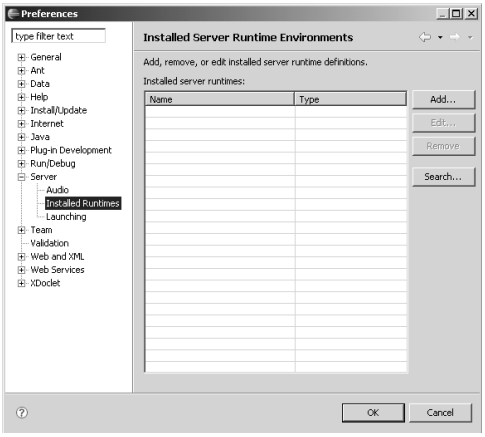
Made available under a Creative Commons Attribution-Share Alike 3.0 License

A Simple Web Application

The purpose of this exercise is to create and execute the simplest possible Web application, namely a Web application that contains a single Web page that displays the message, “Hello, world.” You’ll perform the following development tasks in this iteration:

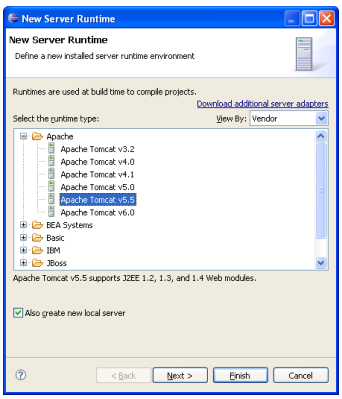
1. Add a Server Runtime Environment.
2. Create a Dynamic Web Application Project.
3. Create and Edit a JSP.
4. Run the JSP on the Server.

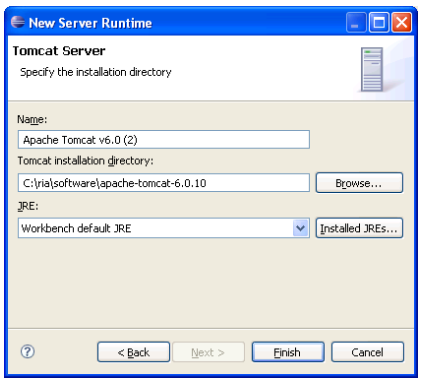
1. Do the following to extend Eclipse with the Tomcat server runtime environment. Launch Eclipse and invoke the command
   - Window > Preference from the menu bar to open the Preference dialog. Expand the Server preferences category and select the Installed Runtimes page. Note that initially there are no server runtime environment definitions.



2. Click the Add button to open the New Server Runtime wizard. Expand the Apache category and select Apache Tomcat v6.0. Click the Next button to display the Tomcat Server page. Click the Browse button to select the directory where you installed Tomcat for example,

   c:\wtp-101\apache-tomcat-..





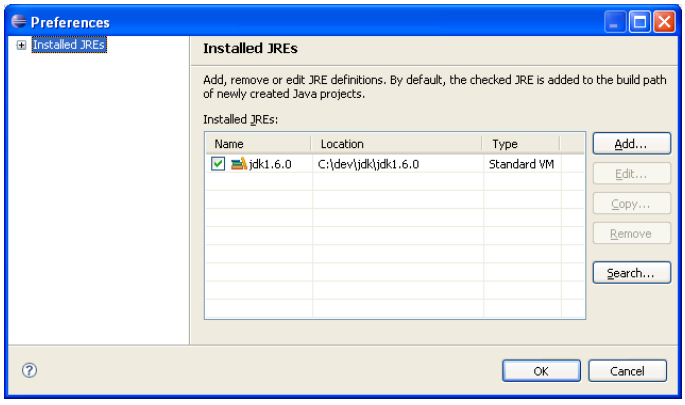
© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

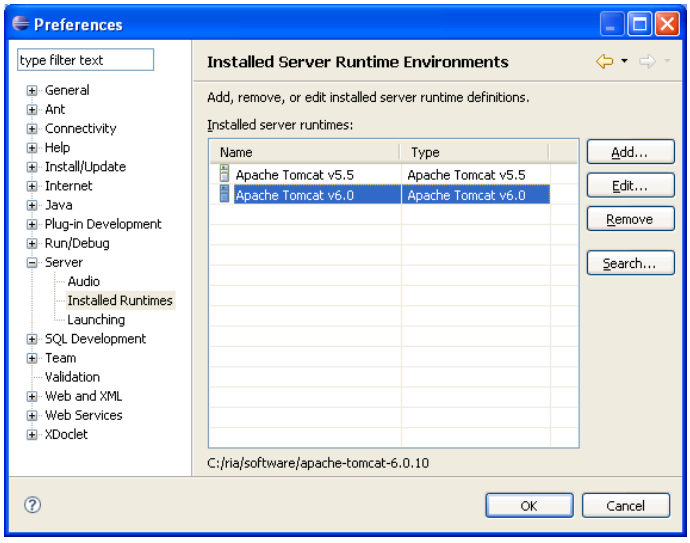
3. The JRE field is initialized to the Workbench default JRE, which is the JVM you used to launch Eclipse. Click the Installed JREs button to open the Installed JREs wizard. This wizard lets you define additional JREs to Eclipse. Click the Add button and select the directory where you installed the JDK for example,

   `c:\wtp-101\jdk1.6.0`

   The JDK is added to the Installed JREs page. Select its checkbox to make it the default JRE, and click the OK button to return to the Tomcat Server page The Tomcat installation directory field should now show the selected directory



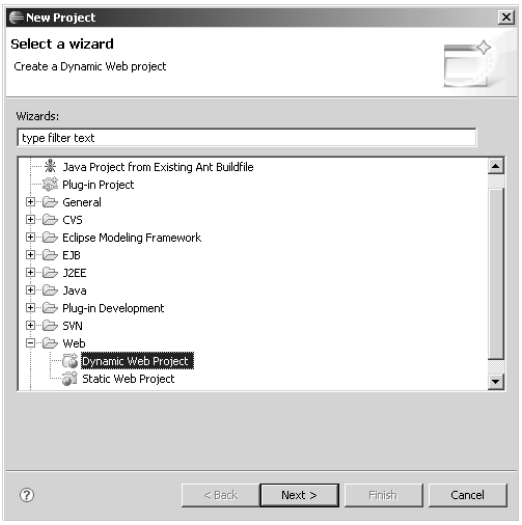
4. Tomcat is now listed on the Installed Server Runtime Environments. You have now extended Eclipse with a J2EE servlet container and are ready to create your first Java Web application development project



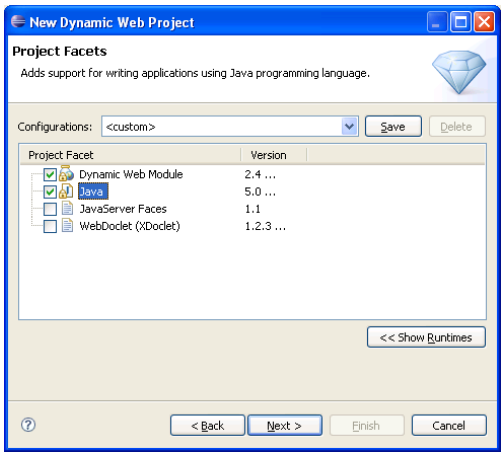
© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

5. To begin development, create a new Dynamic Web project by invoking the
    - **File > New > Project** menu command to open the New Project wizard.
    - Select **Web > Dynamic Web Project** as the project type

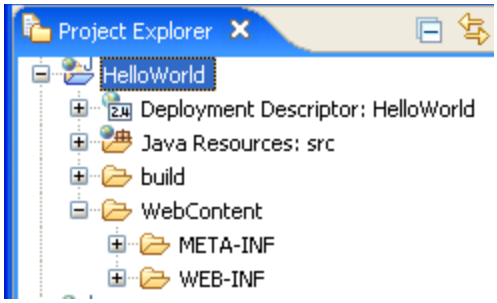


6. Click the Next button. The Dynamic Web Project page will be displayed Enter **HelloWorld** in the Project Name field. Note that **Apache Tomcat** is selected as the Target Runtime since it is the only server runtime environment you have defined.
    - Click the Next button to advance to the Project Facets page simply accept the selections – Make sure Web Module 2.4 and Java 5 are selected as facets.
    - Click the Next button to advance to the Web Module page Accept the defaults for now.
    - Click Finish.

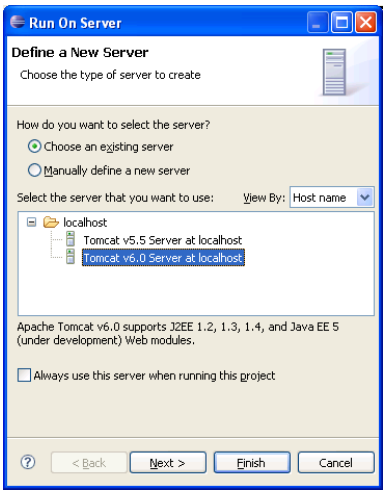


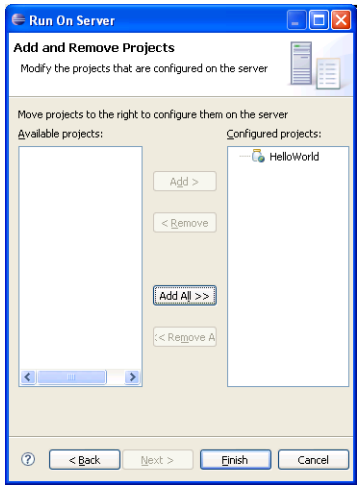
7. You now have a Dynamic Web Project like the following:

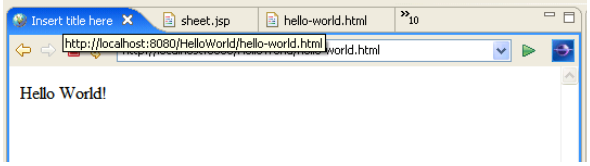
© Copyright 2008, eteration a.ş.
Made available under a Creative Commons Attribution-Share Alike 3.0 License



8. Create and Edit an HTML file as follows:
    - Add a new HTML file to the **WebContent** folder of HelloWorld as follows. Project Explorer view, expand HelloWorld, right click on **WebContent**, and invoke the **New > HTML** command to open the New HTML Page wizard. Give the new file the name hello-world.html and click the Finish button.
    - Open the file and type Hello World into the body.
9. Select hello-world.html and invoke the Run As > Run on Server command from the context menu. Since this is the first time you have tried to run any artifact from the HelloWorld project, WTP will prompt you to define a new server. WTP defaults the server runtime environment to Apache Tomcat, which you previously associated with the project. Click the Next button to advance to the Add and Remove Projects page. Click the Finish button to confirm that you want WTP to add the HelloWorld module to the server configuration. WTP then starts the server and opens a Web browser with the Uniform Resource Locator (URL) of the HTML file



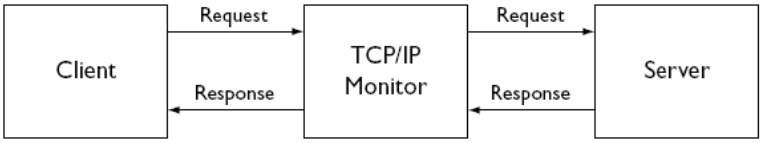




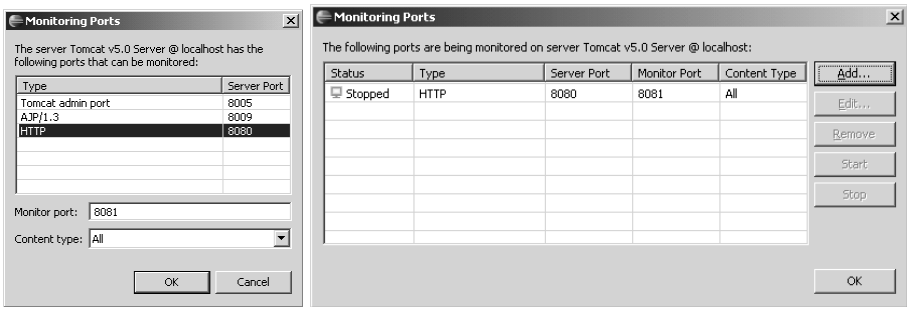
© Copyright 2008, eteration a.ş.
Made available under a Creative Commons Attribution-Share Alike 3.0 License

TCP/IP Monitor

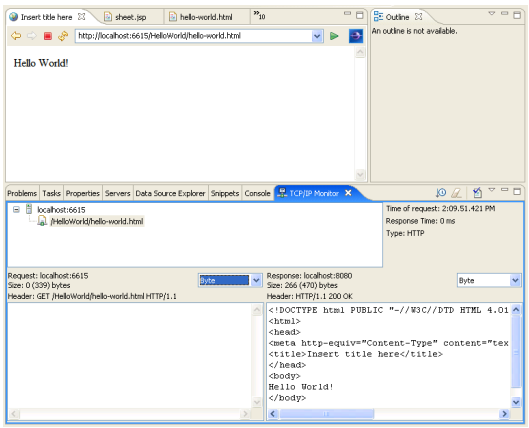
WTP contains a very useful tool called the TCP/IP Monitor that lets you peek into the HTTP traffic and see what's going on. The TCP/IP monitor sits between a client and a server, playing the role of a "man-in-the-middle." The TCP/IP monitor accepts requests from clients, forwards those requests to a server, receives the responses, and sends those responses back to the clients. The TCP/IP monitor records the messages and allows you to view, modify, resend, and save them.



1. In the Servers view, select the Tomcat server, right click, and select the **Monitoring > Properties** menu item. The Monitoring Ports dialog opens. Select the HTTP port. Accept the entries for the Monitor Port (8081) and the Content type filter (All). The monitor port is the port that the monitor listens to. It forwards requests to Tomcat and relays replies back the client. In the process, it records the requests and responses so you can view them. The content type filter controls the type of content that gets recorded. Click the OK button. The monitor is created, and the Monitoring Ports dialog is redisplayed with the new monitor added.



Select the newly created monitor and click the Start button. The monitor is now listening to port 8081 and will forward requests received there on to port 8080. Enter the following URL http://localhost:8081/HelloWorld/hello-world.html The TCP/IP Monitor view opens and displays three entries



© Copyright 2008, eteration a.ş.
Made available under a Creative Commons Attribution-Share Alike 3.0 License

XHTML and Layouts with Tables

Goals
• To create a static HTML page. • To gain some experience using HTML to assemble web pages. • Be able to save and view HTML.

What you should know

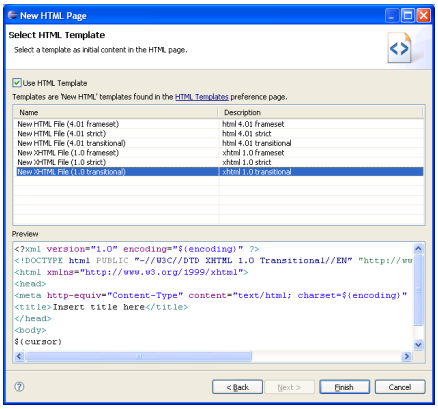
Before you complete this set of exercises, you should understanding the following:

- Basic HTML
- How to use a text editor
- How to construct an HTML file and store it on the Web Module's document root.

ObjectShop Web Page

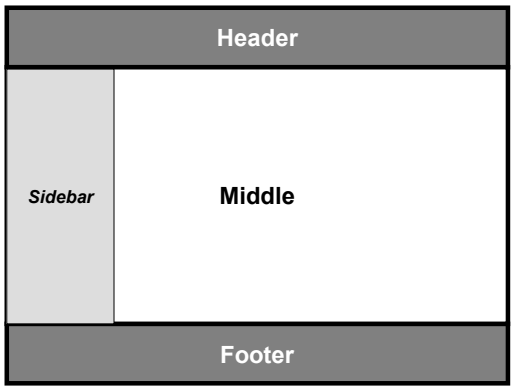
1. Create a a new Dynamic Web Project called **ObjectShop**. Create a new web HTML page.

    a. Select the ObjectShop project using the project explorer. Using the context menu, choose File>New >HTML. Use XHTML 1.0 Transitional as the Web Standard. Save it as "index.html".



This web page will be the gate to online shopping at the ObjectShop, it will have links to different parts of the shop, and give information products sold in the shop.

The document consists of three main blocks: **header**, **middle** and **footer**. **header** at the top and **footer** at the bottom of the view. The middle region will contain a subregion for the sidebar.

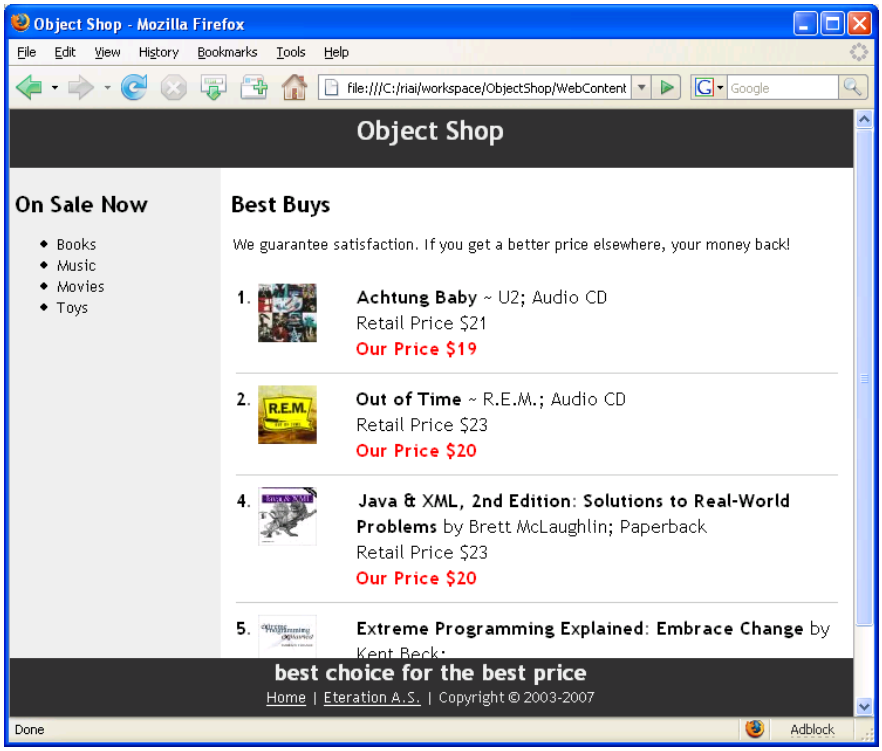


© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

Tables are a useful tool for building a web page that features columns as shown below. In the below example, the layout of the page and the sample catalog for the Object Shop are displayed with tables.

There are three rows (header, middle, and footer). And the middle row has two columns (sidebar and content), so tope and footer must the whole row. Side bar has an unordered list, and the content row holds another table for the catalog of items. Create a page that looks like the following



HTML provides very extensive support for tables; it is possible to create tables with cells of various sizes and shapes. Two adjacent cells, for example, can be joined to form one large cell. For the purposes of this session, and future sessions, only simple tables need be constructed.

2. Enter the code for the index.html body as the following:

```
<table width="100%">
        <tr bgcolor="gray">
                <td colspan="2" align="center"><h2>Object Shop</h2></td>
        </tr>
        <tr bgcolor="silver">
                <td>
                <h2>On Sale Now</h2>
                <ul>
                        <li>Books</li>
                        <li>Music</li>
                        <li>Movies</li>
                        <li>Toys</li>
                </ul>
                </td>
                <td>
                ... Catalog goes Here...
                </td>
        </tr>
        <tr bgcolor="gray">
                <td colspan="2" align="center">
                 <h2>best buys for the best prices</h2></td>
        </tr>
</table>
```

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

A table is constructed using a number of different tags. The <table> and </table> tags surround the table's definition. Rows can be added to the table using the <tr> and </tr> tags. Columns can be added to a row using the <td> and </td> tags. The <table>, <tr> and <td> tags all have parameters to customize their appearance. In the absence of parameters, the columns will automatically line up across rows.

The following HTML defines the structure of the table pictured above:

```
<table>
  <tr>
    <td> ... contents of row 1, column 1... </td>
    <td> ... contents of row 1, column 2... </td>
  </tr>
  <tr>
    <td> ... contents of row 2, column 1... </td>
    <td> ... contents of row 2, column 2... </td>
  </tr>
</table>
```

Note that, in the absence of explicit parameters, a row will be as tall as it needs to be to accommodate the text displayed in each column.

While declaring a cell in a table with <td></td> tags, the column-wise width of the cell can be specified as ;

```
  </tr>
    <td colspan=3> ... contents of row x, column x-x+3... </td>
    ..
  </tr>
```

Row 1 Col 1	Row 1 Col 2	Row 1 Col 3
Row 2 Col 1	Row 2 Col 2	Row 2 Col 3
Row 3 Col 1 to col 3 combined		

3. Add the table catalog page as shown above. You can find the HTML for the tables and the images in your exercises CD
   a. `hands-on/02/images`
   b. `hands-on/02/catalog-snippet.txt`

Save your page as "index.html" and try it.

BONUS

You have created an example of an XHTML document that contains a number of tags. To validate that your HTML document is a valid XHTML 1.0 Transitional document.

Validate your page using the validator available at: http://validator.w3.org/

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

Using XML and XML Schemas for the Web

Goals
• To create an XML schema. • Create valid XML documents. • Validate XML documents using XML Schemas.

What you should know

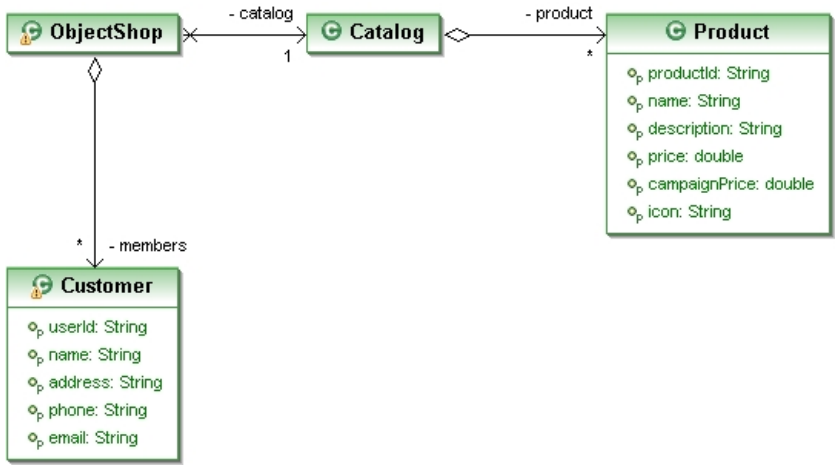
Before you complete this set of exercises, you should understanding the following:

- Basic XML Concepts

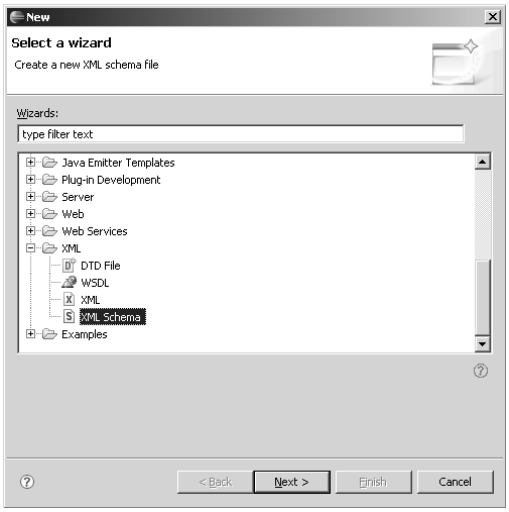
ObjectShop XML Schema

XML Schema Description (XSD) is the W3C Recommendation for describing the format or schema of XML documents, and is the preferred schema description language for use with Web services. XSD is far more expressive than its predecessor, DTD, and, like many specifications produced by industrial collaborations, is extremely feature rich.

The following Object model describes the Catalog and Products for ObjectShop.

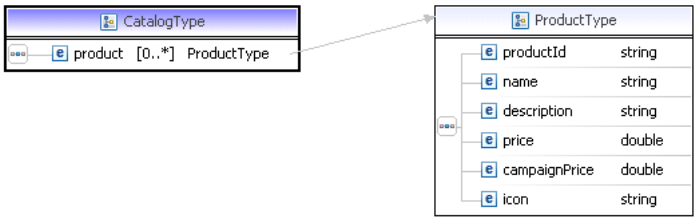


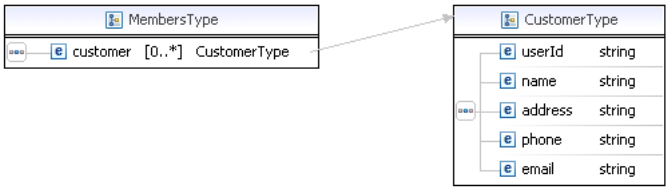
1. Create a new XML Schema file named objectshop.xsd in ObjectShop.

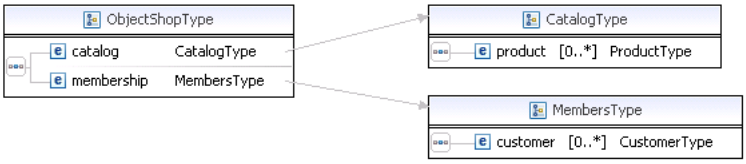


© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

2. In general, there are many equivalent ways to describe a given format using XSD. It's a good practice to describe formats in a way that works well with XML data binding toolkits. That means we will define complex types for the content model of each element in our object model. The XSD editor lets you edit in the source tab, the graphical tab, the outline view, and the property view.
3. In the XSD editor use the graphical view to add a new complex types for the following:
    a. **ProductType** that contains a sequence of elements
    ```
    string productId
    string name
    string description
    double price
    double campaignPrice
    string icon
    ```
    b. **CustomerType** that contains a sequence of elements
    ```
    string userId
    string name
    string address
    string phone
    string email
    ```
    c. **CatalogType** that contains a sequence of elements
    ```
    ProductType product    minimum=0 maximum=unbounded
    ```
    

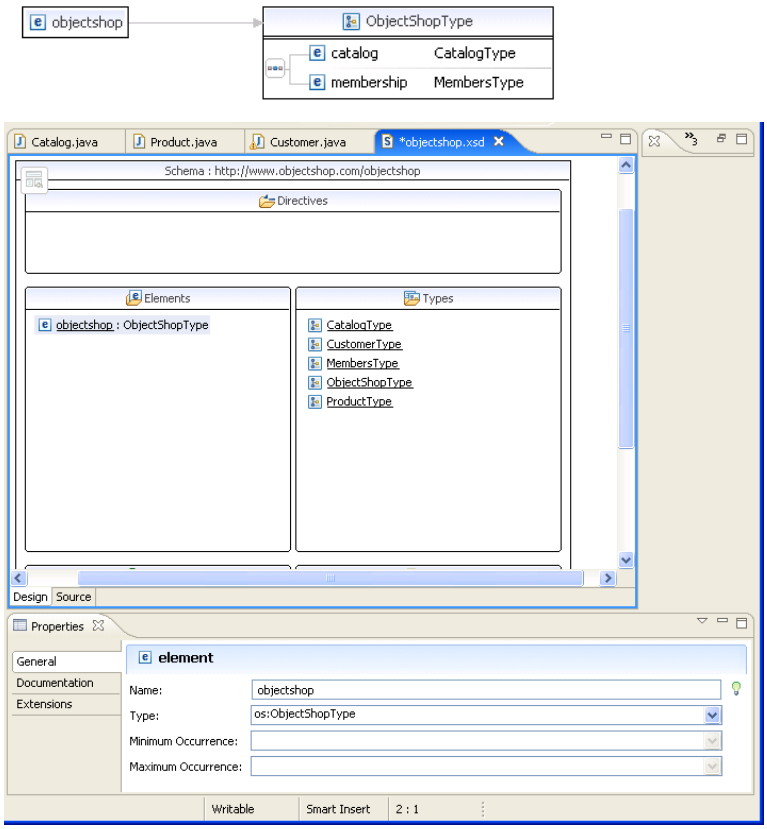
    d. **MembersType** that contains a sequence of elements
    ```
    CustomerType customer    minimum=0 maximum=unbounded
    ```
    

    e. **ObjectShopType** that contains a sequence of elements
    ```
    CatalogType catalog    minimum=0 maximum=unbounded
    MembersType membership    minimum=0 maximum=unbounded
    ```
    

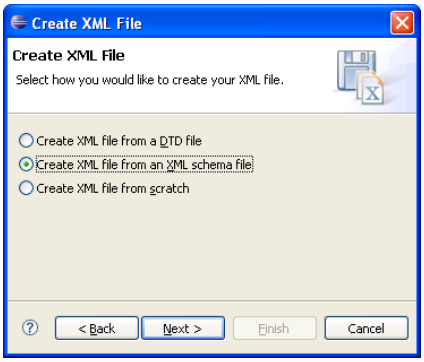
© Copyright 2008, eteration a.ş.
Made available under a Creative Commons Attribution-Share Alike 3.0 License

4. In the XSD editor define the document root element as:

ObjectShopType objectshop



5. You are done. Create an XML file based on this XSD. Go to File > New > XML. Choose Create XML file from an XML Schema file, name the file **objectshop.xml**. Click Next and choose the objectshop.xsd.



Your XML file should look like the one below. You can also import the **objectshop.xsd**, and the **objectshop.xml** from the CD

c. `hands-on/03/objectshop.xsd`

d. `hands-on/02/objectshop.xml`

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

```
<?xml version="1.0" encoding="UTF-8"?>
<os:objectshop xmlns:os="http://www.objectshop.com/objectshop"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.objectshop.com/objectshop objectshop.xsd ">
  <os:catalog>
    <os:product>
      <os:productId>1</os:productId>
      <os:name>Achtung Baby ~ U2</os:name>
      <os:description>Audio CD</os:description>
      <os:price>21.0</os:price>
      <os:campaignPrice>19.0</os:campaignPrice>
      <os:icon>images/u2AchtungBaby.jpg</os:icon>
    </os:product>
    <os:product>
      <os:productId>2</os:productId>
      <os:name>Out of Time ~ R.E.M.</os:name>
      <os:description>Audio CD</os:description>
      <os:price>23.0</os:price>
      <os:campaignPrice>20.0</os:campaignPrice>
      <os:icon>images/remoutoftime.jpg</os:icon>
    </os:product>
    <os:product>
      <os:productId>3</os:productId>
      <os:name>Java &amp;amp; XML, 2nd Edition: Solutions to
          Real-World Problems by Brett McLaughlin</os:name>
      <os:description>Book</os:description>
      <os:price>23.0</os:price>
      <os:campaignPrice>20.0</os:campaignPrice>
      <os:icon>images/javaxml.jpg</os:icon>
    </os:product>
    <os:product>
      <os:productId>4</os:productId>
      <os:name>Extreme Programming Explained:
              Embrace Change by Kent Beck</os:name>
      <os:description>Book</os:description>
      <os:price>36.0</os:price>
      <os:campaignPrice>31.0</os:campaignPrice>
      <os:icon>images/xp.jpg</os:icon>
    </os:product>
  </os:catalog>
  <os:membership>
    <os:customer>
      <os:userId>eteration</os:userId>
      <os:name>Esma Meral</os:name>
      <os:address>ITU Teknokent ARI-1 25 Maslak Istanbul</os:address>
      <os:phone>+90 (212) 329 0825</os:phone>
      <os:email>esma@eteration.com</os:email>
    </os:customer>
  </os:membership>
</os:objectshop>
```

6. Try deleting or changing some of the tags. XML file validator should give errors of the validation checks.

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

CSS – Separating Content and Presentation

Goals
• To create CSS file • Use CSS for Layouts.

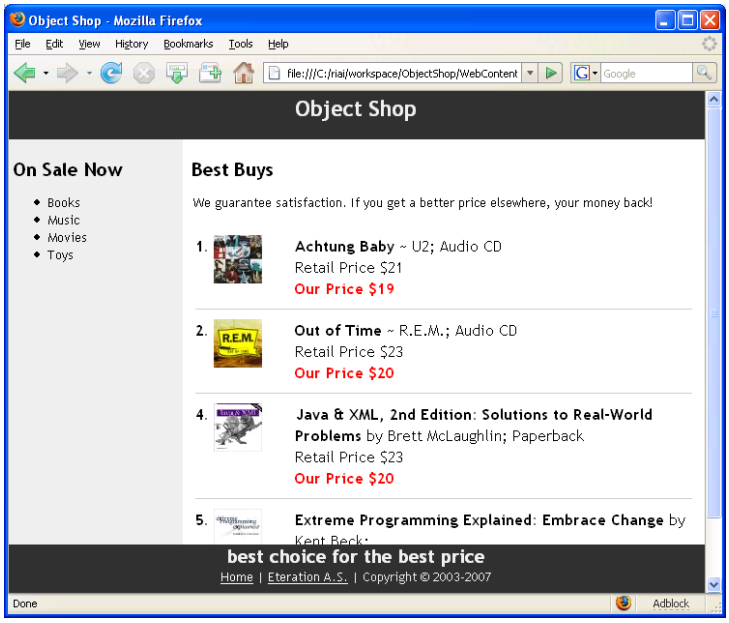
What you should know

Before you complete this set of exercises, you should understanding the following:

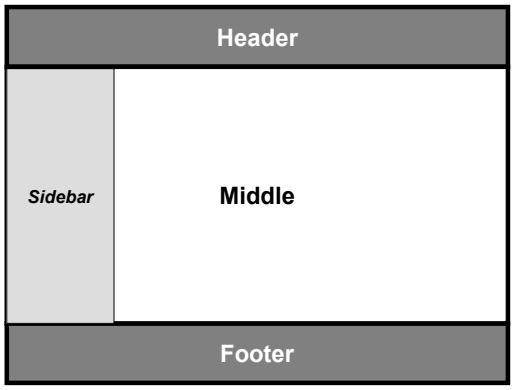
- Basic HTML Concepts

ObjectShop CSS

In this exercise you will create a file named objectshop.css to manage the layout of the ObjectWeb web pages and its look and feel. The page will look like the following when it is finished. The catalog list will scroll without moving the footer and the header.



The following layout design will be used for the page



1. To start. Go to the ObjectShop project, and create a new XHTML 1.0 transitional page called the main.html.

© Copyright 2008, eteration a.ş.
Made available under a Creative Commons Attribution-Share Alike 3.0 License

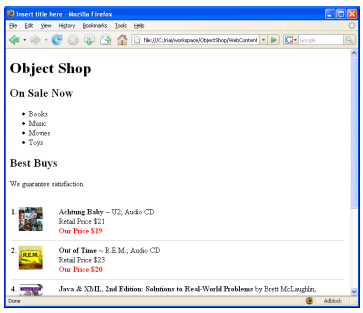
2. First of all, we create the basic HTML structure.

```
<body>
<div id="headerregion"></div>
<div id="middleregion">
       <div id="sidebar"></div>
       <div id="content"></div>
</div>
<div id="footerregion"></div>
</body>
```

3. Put some content into these regions as follows. You can use the same table snippet from the XHTMl handson session for the product data.

```
<body>

<div id="headerregion">
       <div id="header">
              <h1>Object Shop</h1>
       </div>
</div>

<div id="middleregion">
<div id="middle">
       <div id="sidebar">
              <h2>On Sale Now</h2>
              <ul>
                     <li>Books</li>
                     <li>Music</li>
                     <li>Movies</li>
                     <li>Toys</li>
              </ul>
       </div>

       <div id="content">
              <h2>Best Buys</h2>
              <p>We guarantee satisfaction.</p>
              <br/>
              …..   Catalog Table Snippet Goes Here ….
       </div>
</div>
</div>

<div id="footerregion">
   <div id="footer">
         <h2>best choice for the  best price</h2>
         <div id="footerlinks">
         <p><a href="/">Home</a> | <a href="/">Eteration A.S.</a> |
                                    Copyright&#169; 2003-2007</p>
       </div>
   </div>
</div>
</body>
```

4. Try you page. You will notice that it does not look anything like we wanted to have.



5. Create a CSS file named objectshop.css in the folder WebContents/styles

© Copyright 2008, eteration a.ş.
Made available under a Creative Commons Attribution-Share Alike 3.0 License

6. Add a link from the main.html file to the objectshop.css

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

7. Adjust the body and html elements. To make the content reach the edges of the browser window, we set the margin and padding of the body and html elements to zero. We also specify colors for text and background.

```
body,html{
        margin:0;
        padding:0;
}
```

8. Give the content area a width and center it horizontally. We do that by specifying the width and margins of the main container. We also give it a background colour to make it show up on the page. The method we use to center the content is based on the fact that when an element's left and right margins are set to auto, they will share whatever is left when the element's width has been subtracted from that of its container. In this case the width of #main will be subtracted from the width of the browser window. To avoid problems that can occur in some browsers when the window is narrower than #main We set the <body> element's min.

```
body {
        min-width: 720px;
        background: #99c;
}
```

9. Give the different sections of the document different background colours to make them show up.

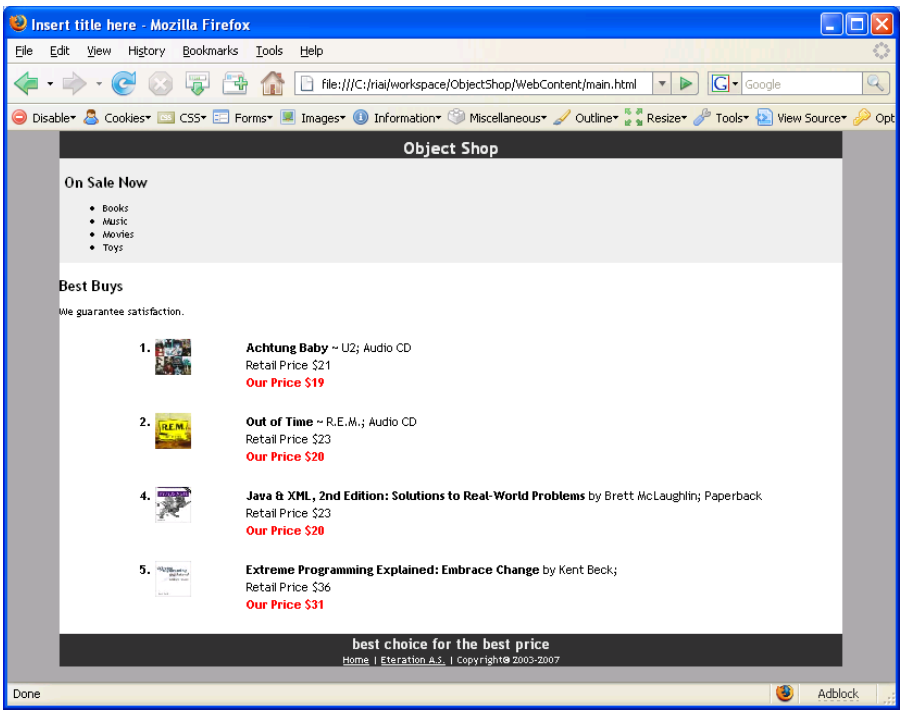
```
body,html {
        font-family: "Trebuchet MS", Georgia, Verdana, serif;
        color: #000;
        background: #aaa;
}

div#header,div#footer {
        background: #333;
        color: #eee;
}

div#middle,div#sidebar {
        background: #eee;
}

div#content {
        background: #fff;
}

h1,h2 {
        padding: 0;
        margin: 0;
}

div#sidebar h2 {
        padding-left: 5px;
}

div#footer h2 {
        text-align: center;
        padding: 0;
        margin: 0;
}
```

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

```
div #footer p {
	margin: 0;
	padding: 0;
	text-align: center;
}

div#footer a {
	color: #fff;
}

h1 {
	font-size: 1.4em;
	text-align: center;
	padding-top: 5px;
}

h2 {
	font-size: 1.2em;
	padding-top: 1em;
	margin-top: 0;
}

p {
	margin-bottom: 0;
	font-size: 0.8em;
	line-height: 1.4em;
}

pre {
	font-size: 0.9em;
	line-height: 1.4em;
}

ul {
	font-size: 0.8em;
	line-height: 1.4em;
}
```

You can copy the CSS code from the file available on the CD. Your page will look like the following now. Next we will put the sections where they belong.



© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

10. Place the sidebar and content columns side by side. To make the two columns /#sidebar and #content) display side by side we **float** them, one to the left and the other to the right. We also specify the widths of the columns.

```
/* set left margin for modern browsers */
body>div#middleregion {
	margin-left: 0;
}

div#middle {
	width: 720px;
	margin: 0 auto;
}

div#sidebar {
	width: 180px;
	float: left;
}

div#content {
	padding: 0 10px;
	margin-left: 180px;
}
```

11. This will make `#sidebar` appear to the left of `#content`, but the footer is not where it should be. It scrolls of the screen with the table. We will have a fixed position for the footer so that it always remains on the screen.

```
/* position:absolute for all browsers - the whole page scrolls
*/
div#headerregion {
	position: absolute;
	width: 100%;
	top: 0;
	left: 0;
	height: 50px;
}

/* position:fixed for modern browsers - header and footer do
not scroll */
body>div#headerregion {
	position: fixed;
}

div#header {
	height: 50px;
	width: 720px;
	margin: 0 auto;
}

/* set a left margin to compensate for IE/Win always making
room for a scrollbar */
div#middleregion {
	padding: 50px 0 0 0;
	margin-left: 0;
	padding-bottom: 50px;
}
div#footerregion {
	width: 100%;
	position: absolute;
	bottom: 0;
```

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

```
        left: 0;
        height: 50px;
}
body>div#footerregion {
        position: fixed;
}

div#footer {
        height: 50px;
        width: 720px;
        margin: 0 auto;
    }
```

12. Try the page by resizing the window. Header and footer should always be visible on the page. The page should work as expected now.

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

XSL Transformations

Goals
• To create XSL file • Generate XHTML from XML using XSL • Use CSS for Presentation

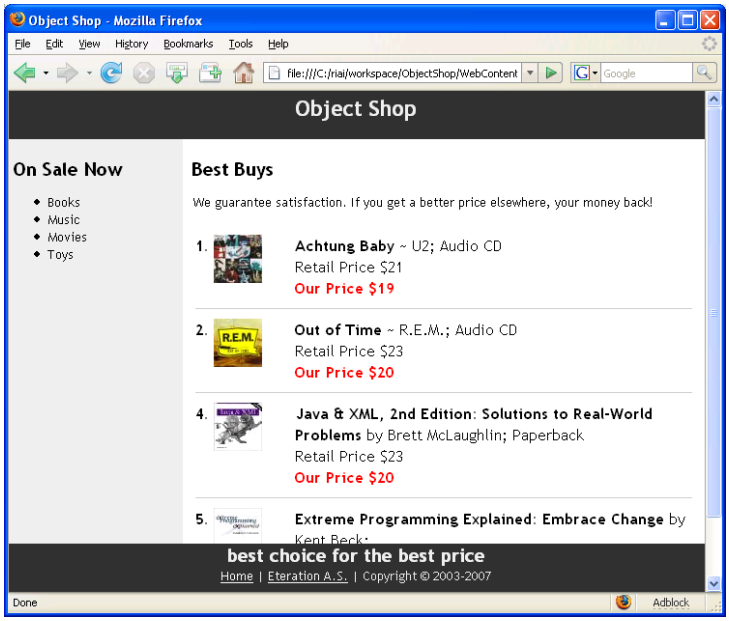
What you should know

Before you complete this set of exercises, you should understand the following:

- Basic HTML Concepts
- Basic XML Concepts
- Basic CSS
- Basic XSLT

ObjectShop XSLT

In this exercise you will create a file named objectshop.xsl to to present the data provided in objectshop.xml using and XSLT transformation and use a CSS to provide the look-and-feel. The page will look like the following when it is finished.



1. To start. Go to the ObjectShop project, and create a new XSL file named objectshop.xsl in ObjectShop/WebContent folder.
2. Import the objectshop.xml and objectshop.css files from the CD. They are provided in the handson-04-xslt folder.

© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License

3. Open the XSL file. Create a structure like the following. This divides the transformation into two modules: Main page is generated in the root template and the Table that lists product in the catalog is generated in a subtemplate. Make sure you add the namespace for the objectshop.xsd to the xsl file. Create two templates. One that matches `match="/"` and the other one that matches `match="/os:objectshop/os:catalog"`.

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

  <xsl:template match="/">
         <html>
                <body>
                <div id="headerregion">
                </div>

                <div id="middleregion">
                <div id="middle">
                <div id="sidebar">
                  ...
                </div>
                <div id="content">
                    ....
                <xsl:apply-templates
                select="/os:objectshop/os:catalog" />
                </div>
                </div>
                </div>
                <div id="footerregion">
                ..
                </div>
                </body>
                </html>
</xsl:template>


<xsl:template match="/os:objectshop/os:catalog">
        <table align="center">
                ...
        </table>
</xsl:template>

</xsl:stylesheet>
```

Copy the HTMl code from the main.html we created in the previos exercise in the main template. Delete the HTML that generated table and replace it with a call to the Catalog template

```
        <xsl:apply-templates select="/os:objectshop/os:catalog" />
```

4. Add the transform to generate the table in the subtemplate. Note the image tag. The attributes must be set from the XML so they will erquire special attention. Add an xsl:foreach to loop over the catalog.

```
<xsl:template match="/os:objectshop/os:catalog">
  <table align="center">
   <xsl:for-each select="//os:product">
    <xsl:sort select="os:productId" />
    <tr valign="top"><td valign="top"><b>
       <xsl:value-of select="os:productId" />
      </b></td><td>
      <img>
       <xsl:attribute name="height">50</xsl:attribute>
       <xsl:attribute name="width">50</xsl:attribute>
       <xsl:attribute name="align">top</xsl:attribute>
```

© Copyright 2008, eteration a.ş.

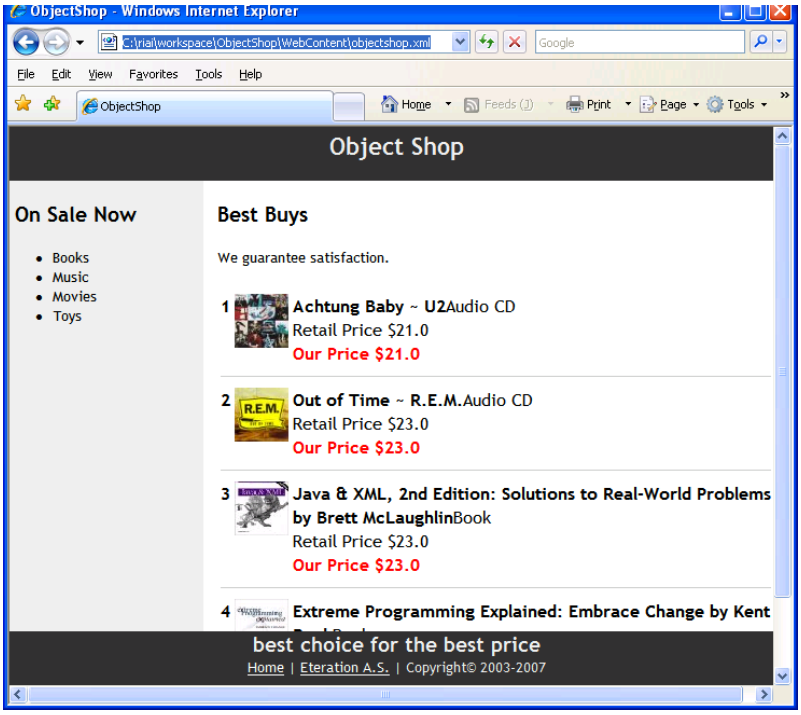
Made available under a Creative Commons Attribution-Share Alike 3.0 License

```
        <xsl:attribute name="border">0</xsl:attribute>
        <xsl:attribute name="src">
         <xsl:value-of select="os:icon" />
        </xsl:attribute>
       </img>
      </td>
      <td><b><xsl:value-of select="os:name" /></b>
       <xsl:value-of select="os:description" />
       <br />Retail Price $<xsl:value-of select="os:price" />
       <br /><font color="RED">
       <b>Our Price $<xsl:value-of select="os:price" /></b>
       </font></td></tr><tr>
      <td colspan="3">
       <img height="1" src="images/grey-pixel.gif" width="100%"
        align="top" vspace="6" />
      </td>
     </tr>
    </xsl:for-each>
   </table>
  </xsl:template>
```

5. Add link tag to the XML file to associate the XSL transformation with the XML file. Open the objectshop.xml file in an editor and the following line:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="objectshop.xsl"?>
<os:objectshop ...
```

6. Open the objectshop.xml using a browser. You should see:



© Copyright 2008, eteration a.ş.

Made available under a Creative Commons Attribution-Share Alike 3.0 License