Developing J2EE Web Applications

Using Servlets and JavaServer Pages

Exercise book

Version 1.4

01-Servlet Basics - Exercises

Exercise 1 – Hello World Servlet

Exercise description:

Write a simple "Hello World" Servlet. The Servlet should be based on a **GenericServlet** class and print "Hello world" when called.

Exercise directory:

<exercises>\servlets basics\

Files to use:

\exercise\HelloWorldServlet.java

Run:

http://localhost:8080/01-servlets_basics/hello

Exercise 2 – Life Cycle Servlet

Exercise description:

Write a Servlet which prints a message to the standard output (not to the client) on every lifecycle phase (Instantiation, Initialization, Service and Destruction). Complete the needed methods with the proper messages.

Exercise directory:

<exercises>\servlets basics\

Files to use:

\exercise\LifeCycleServlet.java

Run:

http://localhost:8080/01-servlets_basics/lifecycle

Exercise 3 – HTTP Request Snooping

Exercise description:

Write a Servlet which prints the details of the HTTP request that was sent from the browser. The Servlet should read the following request's parts (Request-Line, Request Headers) and send them back to the client as an HTML document.

Exercise directory:

<exercises>\servlets api\

Files to use:

\exercise\SnoopRequestServlet.java

Run:

http://localhost:8080/02-servlets_api/snoopRequest

Exercise 4 – Simple Form Processing

Exercise description:

Write a Servlet that consists of a simple HTML form, which includes the following fields: First name, Last name, ID and Email. When the user submits the form, a POST request is sent to the Servlet, the Servlet reads the parameters from the request and sends them back to the client as an HTML table.

Exercise directory:

<exercises>\servlets api\

Files to use:

\exercise\FormServlet.java

Run:

http://localhost:8080/02-servlets_api/form

Exercise 5 – Numbers Guessing

Exercise description:

Write a simple number-guessing game using a Servlet. While initializing, the Servlet creates a random number between 1 and a defined maximum value. The user is asked to guess the number. If the guess is too high or too low, a proper message is displayed. If the guess is correct, a "You are a winner" message is displayed.

NOTE: The user can make only one guess, after which the result is printed out. For each request a new random number is generated.

The Servlet should read the defined maximum value from the ServletConfig object, which is passed as a parameter to the init method. Add the needed element to the **web.xml** file.

Exercise directory:

<exercises>\servlets api\

Files to use:

\exercise\RandomServlet.java
WEB-INF\web.xml

Run:

http://localhost:8080/02-servlets_api/random

Exercise 6 – Cookies Servlet

Exercise Description:

Write a Servlet that stores cookies on the user's computer. The user enters the name and value of the cookies in an HTML form and submits them to the server. The Servlet reads the parameters sent by the user and adds a new cookie to the client according to the parameters. The server prints the cookies it receives in each request.

Exercise directory:

<exercises>\cookies and sessions\

Files to use:

\exercise\CookiesServlet.java

Run:

http://localhost:8080/03-cookies_and_sessions/cookies

Exercise 7 – Logging in using a Cookie

Exercise Description:

Write a Servlet containing a Login form with two fields (Username and Password). When the user logs in, he can mark a "remember my user name" checkbox for future log-ins.
The Servlet stores the user name in a persistent Cookie (saved forever), and uses this Cookie for the next log-in to welcome the user back to the application (the user still needs to enter his password in order to login).

Exercise directory:

<exercises>\cookies and sessions\

Files to use:

\exercise\CookieFormServlet.java

Run:

http://localhost:8080/03-cookies_and_sessions/cookieForm

Exercise 8 – Shopping Cart

Exercise Description:

Write a shopping cart Servlet, based on **HttpServlet**. The Servlet will create an "Items list" (inventory) that the user can buy and stores it in the **ServletContext**. The user is allowed to add items from the inventory to his shopping cart, which is practically a list stored in the HttpSession. The Servlet shows the shopping cart's content and the total price of all the items on the cart.

Exercise directory:

<exercises>\cookies and sessions\

Files to use:

\exercise\CartServlet.java
\exercise\Item.java

Run:

http://localhost:8080/03-cookies_and_sessions/cart

Exercise 9 – Shopping Cart using a Database

Exercise Description:

Write a shopping cart Servlet, based on **HttpServlet**. The Servlet reads the available items in the shop from a database table. The user is allowed to add items to his shopping cart. The Servlet shows the shopping cart and the total price of all the items on the cart.

The design is based on the Data Access Objects (DAO) design pattern. The database is a Microsoft Access database used with a Jdbc-Odbc bridge driver.

Create an ODBC Data Source Name (DSN) named "**CartServlet**" that will point to the **.mdb** file (located in the Database directory of the exercise).

Exercise directory:

<exercises>\database\

Files to use:

\exercise\DbCartServlet.java
\exercise\Item.java
\exercise\ItemDAO.java
Access Database\Items.mdb

Run:

http://localhost:8080/04-database/dbCart

05-JSP Basics - Exercises

Exercise 10 – Hello World JSP

Exercise Description:

Write a simple “Hello World” JSP. Save and run it using the browser. Look for the generated Servlet, open it and find the "Hello World" text.

Exercise directory:

<exercises>\Jsp basic\

Files to use:

hello.jsp

Run:

http://localhost:8080/05-Jsp_basic/hello.jsp

06-JSP Syntax and Semantics - Exercises

Exercise 11– Calendar

Exercise Description:

Write a JSP that shows the current calendar parameters. The parameters to show are: day, month, year, day of week, day of year, hours, minutes, seconds, milliseconds and time zone. Use the **java.util.Calendar** class.

Exercise directory:

<exercises>\Jsp syntax\

Files to use:

calendar.jsp

Run:

http://localhost:8080/06-Jsp_syntax/calendar.jsp

Exercise 12 – User's Preferences

Exercise Description:

Write a JSP that allows the user to set his color preferences (Background and Text colors). The user uses two combo-boxes, one for background color, and one for text color. Both combo-boxes get their values from a Strings array. The preferences are kept in the session for future use.

Exercise directory:

<exercises>\Jsp syntax\

Files to use:

preferences.jsp

Run:

http://localhost:8080/06-Jsp_syntax/preferences.jsp

Exercise 13 – HTML Table Helper Bean

Exercise Description:

Write a JSP that uses a special JavaBean to draw HTML tables. The bean is called **TableHelper** and gets two parameters for each table: A two-dimensional String array (containing the table's data) and the table's background color. The html property of the bean returns the needed HTML code.

Exercise directory:

<exercises>\Jsp and Beans\

Files to use:

tables.jsp
\exercise\TableHelper.java

Run:

http://localhost:8080/07-Jsp_and_beans/tables.jsp

Exercise 14 – User Creation

Exercise Description:

Write two JSPs that handle the creation of new users. The first JSP is a simple HTML form with four fields (First name, Last name, ID and Email). The second JSP gets the request parameters and creates a User bean. It then calls the **User.validate()** method to validate the First name and Last name (Which must not be empty). The validate() method uses the JSP error page mechanism to an error, by throwing an **Exception** if a validation error occurs.

Exercise directory:

<exercises>\ Jsp and Beans\

Files to use:

newUser.jsp
showUser.jsp
error.jsp
\exercise\User.java

Run:

http://localhost:8080/07-Jsp_and_beans/newUser.jsp

Exercise 15 – Items' Views

Exercise Description:

Write a simple catalog application. The user can select how he wants to see the items in the catalog: as a list of items or as separate items. A controller Servlet receives the request and forwards it to the proper JSP.

Exercise directory:

<exercises>\ Dispatching\

Files to use:

index.jsp
item.jsp
list.jsp
\exercise\ControllerServlet.java
\exercise\Item.java

Run:

http://localhost:8080/08-Dispatching/index.jsp

11- Expression Language

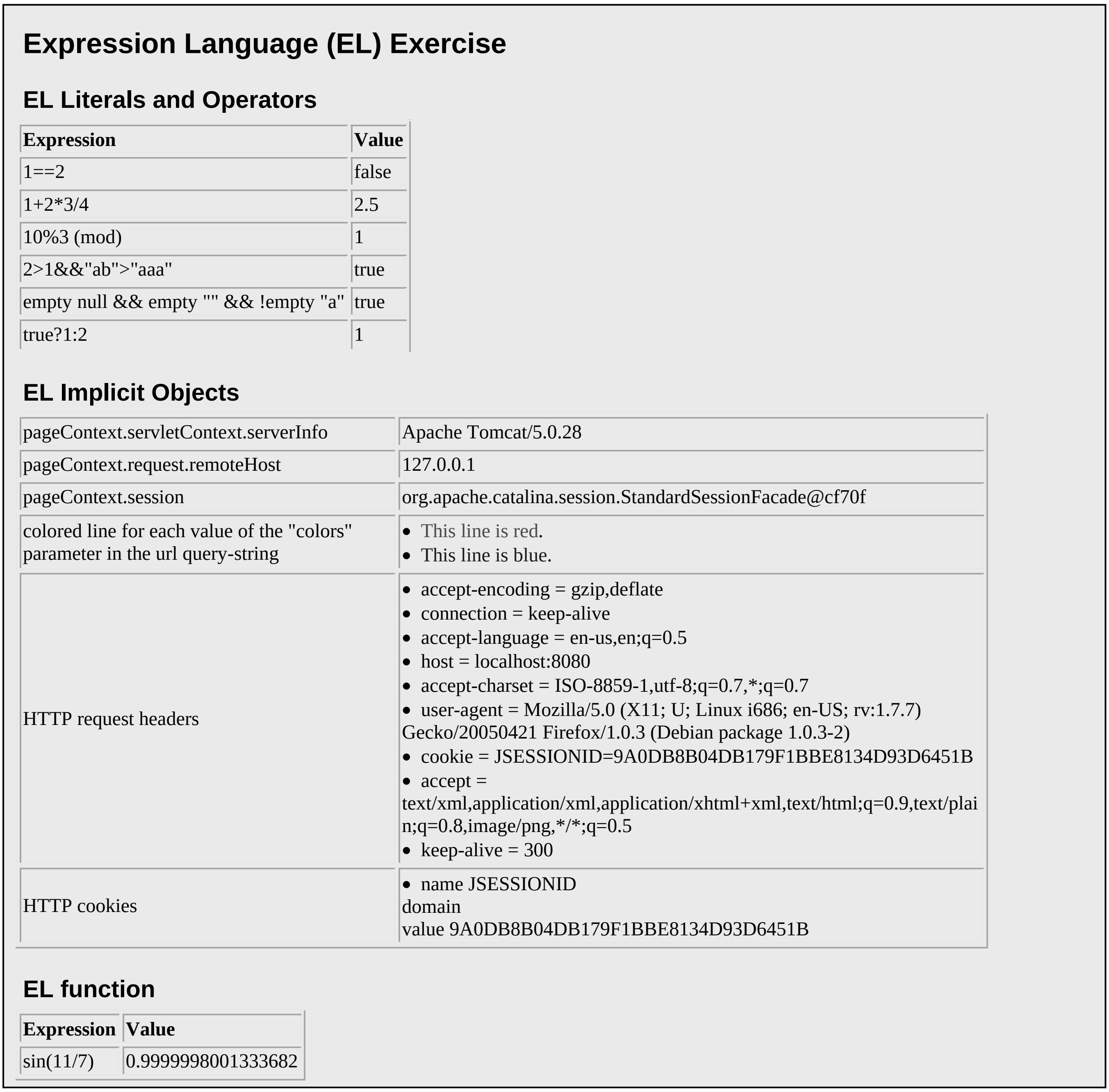
Exercise 20 – Expression Language

Exercise Description:

Fill in the missing expressions using the JSP expression language.

The first table demonstrate simple usage of literals and operands.
The second table shows you how to use pre-defined (implicit) objects - the JSP syntax required for looping in the last three rows is provided.
The last table demonstrates the usage of a user defined function, a .tld file skeleton is provided.

The final output should look similar to this:



Expression Language (EL) Exercise

EL Literals and Operators

Expression	Value
1==2	false
1+2*3/4	2.5
10%3 (mod)	1
2>1&&"ab">"aaa"	true
empty null && empty "" && !empty "a"	true
true?1:2	1

EL Implicit Objects

pageContext.servletContext.serverInfo	Apache Tomcat/5.0.28
pageContext.request.remoteHost	127.0.0.1
pageContext.session	org.apache.catalina.session.StandardSessionFacade@cf70f
colored line for each value of the "colors" parameter in the url query-string	• This line is red. • This line is blue.
HTTP request headers	• accept-encoding = gzip,deflate • connection = keep-alive • accept-language = en-us,en;q=0.5 • host = localhost:8080 • accept-charset = ISO-8859-1,utf-8;q=0.7,*;q=0.7 • user-agent = Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.7.7) Gecko/20050421 Firefox/1.0.3 (Debian package 1.0.3-2) • cookie = JSESSIONID=9A0DB8B04DB179F1BBE8134D93D6451B • accept = text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5 • keep-alive = 300
HTTP cookies	• name JSESSIONID domain value 9A0DB8B04DB179F1BBE8134D93D6451B

EL function

Expression	Value
sin(11/7)	0.9999998001333682

Exercise directory:

<exercises>\el\

Files to use:

index.jsp
WEB-INF\app.tld

Run:

http://localhost:8080/11-el/index.jsp?colors=red&colors=blue