**Solutions to Selected Exercises**

## Chapter 2

## Exercise 2.1

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e2\_1.html

This is a solution to Exercise 2.1

-->

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

<head>

<title> Exercise 2.1 </title>

</head>

<body>

<h2> Ruper B. Baggins </h2>

<p>

1321 Causeway Circle <br />

Middle, Earth <br />

rbaggins@miderth.net<br />

</p>

<hr />

<h3> Bush Watcher </h3>

<p>

<em> Forest Keepers, Limited </em> <br />

14 Cranberry Way <br />

Middle, Earth <br />

<strong> (no web site yet) </strong>

</p>

</body>

</html>

## Exercise 2.3

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e2\_3.html

This is a solution to Exercise 2.3

-->

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

<head>

<title> Exercise 2.3 </title>

</head>

<body>

<h2> Ruper B. Baggins </h2>

<p>

1321 Causeway Circle <br />

Middle, Earth <br />

rbaggins@miderth.net<br /><br />

<a href = "e2\_31.html"> Mr. Baggins' Background </a>

</p>

<hr />

<h3> Bush Watcher </h3>

<p>

<em> Forest Keepers, Limited </em> <br />

14 Cranberry Way <br />

Middle, Earth <br />

<strong> (no web site yet) </strong>

</p>

</body>

</html>

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e2\_31.html

This is part of the solution to Exercise 2.3

(The second document for the background info)

-->

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

<head>

<title> Exercise 2.3 (background) </title>

</head>

<body>

<p>

Although we share the same family name, I am not in any way

related to the famous (or is it infamous) adventurer, Bilbo.

I have a lovely wife, Elvira, and two grown children, Max

and Miriam. Max has chosen to follow me in my profession,

which is described below. Miriam is a beekeeper for the town

bookkeeper, who keeps bees as a second job.

</p>

<p>

I am employed by Forest Keepers, Limited. My job, as I understand

it, is to keep an eye on the 4 acres of wild cranberries that

grow in the swamp at the edge of the village forest. I am required

to file a daily report, in triplicate, on the condition of the

cranberry bushes. To accomplish my task, I walk by and inspect

every cranberry bush in the swamp every workday. My employer provides

me with wading boots for my job. I pick up the boots at the office

every weekday morning and turn them back in, after a thorough

cleaning, after each workday.

</p>

</body>

</html>

## Exercise 2.4

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e2\_4.html

A solution to Exercise 2.4 - an unordered list

-->

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

<head>

<title> Unordered List </title>

</head>

<body>

<h3> Grocery List </h3>

<ul>

<li> milk - 2%, 2 gallons </li>

<li> bread - butter top wheat </li>

<li> cheddar cheese - sharp, 1 lb. </li>

<li> soup - vegetable beef, 3 cans </li>

<li> hamburger - 80% fat free, 2 lbs. </li>

<li> orange juice - not from concentrate, 1/2 gallon </li>

<li> eggs - large, 1 dozen </li>

</ul>

</body>

</html>

## Exercise 2.8

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e2\_8.html

A solution to Exercise 2.8 - a nested, ordered list

-->

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

<head>

<title> An Ordered List </title>

</head>

<body>

<h3> My Uncles, Aunts, and Cousins </h3>

<ol>

<li> Violet Vinelli (my mother) </li>

<li> Frederick Vinelli

<ol>

<li> Mary Vinelli </li>

<li> Betty Ann Boop </li>

<li> Bob Vinelli </li>

<li> Roger Vinelli </li>

</ol>

</li>

<li> Maxine Robinson

<ol>

<li> John Robinson </li>

<li> Patty Robinson </li>

<li> Lucille Robinson </li>

</ol>

</li>

<li> Thomas Vinelli

<ol>

<li> Albert Vinelli </li>

<li> Alison MacKinsey </li>

<li> Alton Vinelli </li>

</ol>

</li>

</ol>

<ol>

<li> Albert Alphonso (my father) </li>

<li> Herbert Alphonso

<ol>

<li> Louise Alphonso </li>

<li> Pam Alphonso </li>

<li> Fred Alphonso </li>

</ol>

</li>

<li> Ann Marie Predicate

<ol>

<li> George Predicate </li>

<li> Michael Predicate </li>

<li> Darcie Predicate </li>

</ol>

</li>

<li> Ferdinand Alphonso

<ol>

<li> Noah Alphonso </li>

<li> Leah Alphonso </li>

<li> Jo Alphonso </li>

</ol>

</li>

</ol>

</body>

</html>

## Exercise 2.9

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e2\_9.html

A solution to Exercise 2.9 - a simple table

-->

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

<head>

<title> A simple table </title>

</head>

<body>

<table border = "border">

<caption> Trees </caption>

<tr>

<th> </th>

<th> Pine </th>

<th> Maple </th>

<th> Oak </th>

<th> Fir </th>

</tr>

<tr>

<th> Average Height (feet) </th>

<td> 55 </td>

<td> 50 </td>

<td> 50 </td>

<td> 65 </td>

</tr>

<tr>

<th> Average Width (inches) </th>

<td> 18 </td>

<td> 26 </td>

<td> 24 </td>

<td> 28 </td>

</tr>

<tr>

<th> Typical Lifespan (years) </th>

<td> 150 </td>

<td> 230 </td>

<td> 310 </td>

<td> 135 </td>

</tr>

<tr>

<th> Leaf Type </th>

<td> Long needles </td>

<td> Broadleaf </td>

<td> Split leaf </td>

<td> Short needles </td>

</tr>

</table>

</body>

</html>

## Exercise 2.10

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e2\_10.html

A solution to Exercise 2.10 - a simple table

-->

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

<head>

<title> A simple table </title>

</head>

<body>

<table border = "border">

<caption> Tree Characteristics </caption>

<tr>

<td rowspan = "2" colspan = "2"> </td>

<th colspan = "4"> Tree </th>

</tr>

<tr>

<th> Pine </th>

<th> Maple </th>

<th> Oak </th>

<th> Fir </th>

</tr>

<tr>

<th rowspan = "4"> Characteristic </th>

<th> Average Height (feet) </th>

<td> 55 </td>

<td> 50 </td>

<td> 50 </td>

<td> 65 </td>

</tr>

<tr>

<th> Average Width (inches) </th>

<td> 18 </td>

<td> 26 </td>

<td> 24 </td>

<td> 28 </td>

</tr>

<tr>

<th> Typical Lifespan (years) </th>

<td> 150 </td>

<td> 230 </td>

<td> 310 </td>

<td> 135 </td>

</tr>

<tr>

<th> Leaf Type </th>

<td> Long needles </td>

<td> Broadleaf </td>

<td> Split leaf </td>

<td> Short needles </td>

</tr>

</table>

</body>

</html>

## Chapter 3

## Exercise 3.1

/\* Book Layout Style Sheet \*/

h1 {font: bold 24pt Helvetica 'Times New Roman'}

h2 {font: bold 20pt Helvetica 'Times New Roman'}

h3 {font: bold 16pt Helvetica 'Times New Roman'}

p {font: 12pt 'Times New Roman'}

## Exercise 3.2

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e3\_2.html

A solution to Exercise 3.2 - a styled table

-->

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

<head>

<title> A Styled Table </title>

<style type = "text/css">

<!--

td.win {font-size: 16pt; color: red;}

td.lose {font-size: 14pt; color: blue;}

-->

</style>

</head>

<body>

<table border = "border">

<caption style = "font-size: 18pt"> Football Scores </caption>

<tr>

<th> Team </th>

<th> Score </th>

</tr>

<tr>

<th style = "font-family: 'Century Gothic';

font-style: italic;

color: gold;"> Colorado </th>

<td class = "win"> 30 </td>

</tr>

<tr>

<th style = "font-family: 'Century Gothic';

font-style: italic;

color: red;"> Nebraska </th>

<td class = "lose"> 29 </td>

</tr>

<tr>

<th style = "font-family: 'Century Gothic';

font-style: italic;

color: grey;"> Iowa State </th>

<td class = "win"> 17 </td>

</tr>

<tr>

<th style = "font-family: 'Century Gothic';

font-style: italic;

color: blue;"> Kansas </th>

<td class = "lose"> 10 </td>

</tr>

<tr>

<th style = "font-family: 'Century Gothic';

font-style: italic;

color: purple;"> Kansas State </th>

<td class = "win"> 48 </td>

</tr>

<tr>

<th style = "font-family: 'Century Gothic';

font-style: italic;

color: green;"> Missouri </th>

<td class = "lose"> 13 </td>

</tr>

</table>

</body>

</html>

## Exercise 3.4

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e3\_4.html

A solution for Exercise 3.4 - floating a text element

-->

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

<head> <title> Floating a text element </title>

</head>

<body>

<p style = "float: left; width: 1.5in; margin-right: 10px;

margin-bottom: 10px;" >

My airplane soars like an eagle and handles like

a hummingbird.

</p>

<p>

The 210 is the flagship

single-engine Cessna aircraft. Although the 210 began as a

four-place aircraft, it soon acquired a third row of seats,

stretching it to a six-place plane. The 210 is classified

as a high-performance airplane, which means its landing

gear is retractable and its engine has more than 200

horsepower. In its first model year, which was 1960,

the 210 was powered by a 260-horsepower fuel-injected

six-cylinder engine that displaced 471 cubic inches.

The 210 is the fastest single-engine airplane ever

built by Cessna.

</p>

</body>

</html>

## Exercise 3.6

#### <?xml version = "1.0"?>

#### <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

#### "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

#### <!-- e36.html

#### A solution to Exercise 3.6

#### -->

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

#### <head> <title> Exercise 3.6 </title>

#### <style type = "text/css">

#### ol {list-style-type: upper-roman;}

#### ol ol {list-style-type: upper-alpha;}

#### ol ol ol {list-style-type: decimal;}

#### li.pink {color: pink}

#### li.blue {color: blue}

#### li.red {color: red}

#### </style>

#### </head>

#### <body>

#### <ol>

#### <li class = "pink"> Compact Cars

#### <ol>

#### <li> Two door

#### <ol>

#### <li> Hyundai Accent </li>

#### <li> Chevrolet Cobalt </li>

#### <li> Honda Civic </li>

#### </ol>

#### </li>

#### <li> Four door

#### <ol>

#### <li> Hyundai Accent </li>

#### <li> Chevrolet Cobalt </li>

#### <li> Honda Civic </li>

#### </ol>

#### </li>

#### </ol>

#### </li>

#### <li class = "blue"> Midsize Cars

#### <ol>

#### <li> Two door

#### <ol>

#### <li> Honda Accord </li>

#### <li> Hyundai Genesis </li>

#### <li> Nissan Altima </li>

#### </ol>

#### </li>

#### <li> Four door

#### <ol>

#### <li> Honda Accord </li>

#### <li> Dodge Avenger </li>

#### <li> Ford Fusion </li>

#### </ol>

#### </li>

#### </ol>

#### </li>

#### <li class = "red"> Sports Cars

#### <ol>

#### <li> Coupe

#### <ol>

#### <li> Jaguar XK </li>

#### <li> Ford Mustang </li>

#### <li> Nissan Z </li>

#### </ol>

#### </li>

#### <li> Convertible

#### <ol>

#### <li> Mazda Miata </li>

#### <li> Ford Mustang </li>

#### <li> Lotus Elise </li>

#### </ol>

#### </li>

#### </ol>

#### </li>

#### </ol>

#### </body>

#### </html>

## Exercise 3.12

<?xml version = "1.0" encoding = "utf-8" ?>

<!DOCTYPE html PUBLIC "-//w3c//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e312.html

A solution to Exercise 3.12

-->

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

<head> <title> Exercise 3.12 </title>

<style type = "text/css">

dt {font-family: Courier; font-size: 12pt;}

dd {font-family: 'Times New Roman'; font-size: 14pt;

font-style: italic;}

</style>

</head>

<body>

<h3> Single-Engine Cessna Airplanes </h3>

<dl>

<dt> 152 </dt>

<dd> Two-place trainer </dd>

<dt> 172 </dt>

<dd> Smaller four-place airplane </dd>

<dt> 182 </dt>

<dd> Larger four-place airplane </dd>

<dt> 210 </dt>

<dd> Six-place airplane - high performance </dd>

</dl>

</body>

</html>

#### Chapter 4

## Exercise 4.1

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e4\_1.html - A solution to Exercise 4.1

-->

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

<head>

<title> Exercise 4.1 </title>

</head>

<body>

<script type = "text/javascript">

<!--

var number, square, cube;

document.write("Number, Square, Cube <br /><br />");

for (number = 5; number < 16; number++) {

square = number \* number;

cube = number \* square;

document.write(number + ", " + square + ", " + cube + "<br />");

}

// -->

</script>

</body>

</html>

**Exercise 4.2**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e4\_2.html - A solution to Exercise 4.2

-->

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

<head>

<title> Exercise 4.2 </title>

</head>

<body>

<script type = "text/javascript">

<!--

var first = 1, second = 1, next, count;

document.write("First 20 Fibonacci Numbers <br/><br/>");

document.write("1 - 1 <br/> 2 - 1 <br/>");

for (count = 3; count <= 20; count++) {

next = first + second;

document.write(count + " - " + next + "<br/>");

first = second;

second = next;

}

// -->

</script>

</body>

</html>

**Exercise 4.4**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e4\_4.html - A solution to Exercise 4.4

-->

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

<head>

<title> Exercise 4.4 </title>

</head>

<body>

<script type = "text/javascript">

<!--

var first = 1, second = 1, next, count;

number = prompt("How many Fibonacci numbers do you want? (3-50)", "");

if (number >= 3 && number <= 50) {

document.write("First " + number + " Fibonacci Numbers <br /><br />");

document.write("1 - 1 <br/> 2 - 1 <br />");

for (count = 3; count <= number; count++) {

next = first + second;

document.write(count + " - " + next + "<br />");

first = second;

second = next;

}

}

else

document.write("Error - number not in the range 3-50");

// -->

</script>

</body>

</html>

## Exercise 4.6

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e4\_6.html - A solution to Exercise 4.6

-->

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

<head>

<title> Exercise 4.6 </title>

</head>

<body>

<script type = "text/javascript">

<!--

var first = 1, second = 1, next, count;

str = prompt("Please input your sentence", "");

var words = str.split(" ");

words = words.sort();

words\_len = words.length;

for (count = 0; count < words\_len; count++)

document.write(words[count] + "<br/>");

// -->

</script>

</body>

</html>

## Exercise 4.7

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e4\_7.html - A solution to Exercise 4.7

-->

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

<head>

<title> Exercise 4.7 </title>

<script type = "text/javascript">

<!--

// A function to compare strings for reverse alphabetic order

function dec\_order(a, b) {

if (a > b)

return -1;

else if (a < b)

return 1;

else return 0;

}

// -->

</script>

</head>

<body>

<script type = "text/javascript">

<!--

var order, str, words, word\_len, count;

// Get the input

str = prompt("Please input your sentence", "");

order = prompt("What order? (ascending or descending)", "");

// If the order is recognized, issue an error message

if (order != "descending" && order != "ascending")

document.write("Error - order is incorrectly specified <br/>");

// Otherwise, do the sort, depending on the requested order

else {

var words = str.split(" ");

if (order == "ascending")

words = words.sort();

else

words = words.sort(dec\_order);

// Write out the results

words\_len = words.length;

for (count = 0; count < words\_len; count++)

document.write(words[count] + "<br/>");

}

// -->

</script>

</body>

</html>

## Exercise 4.9

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e4\_9.html - A solution to Exercise 4.9

-->

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

<head>

<title> Exercise 4.9 </title>

<script type = "text/javascript">

<!--

// Function e\_names

// Parameter: an array of strings

// Returns: the number of given strings that end

// in either "ie" or "y"

function e\_names(names) {

var len, index, count = 0;

len = names.length;

// Loop to use pattern matching to produce the count

for (index = 0; index < len; index++) {

position1 = names[index].search(/ie$/);

position2 = names[index].search(/y$/);

if (position1 + position2 > -2)

count++;

}

return count;

}

// -->

</script>

</head>

<body>

<script type = "text/javascript">

<!--

// Function e\_names tester

var new\_names = new Array ("freddie", "bob", "mieke", "yahoo2", "georgey");

result = e\_names(new\_names);

document.write("The number of special names is: " + result + "<br/>");

// -->

</script>

</body>

</html>

## Exercise 4.14

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e4\_14.html - A solution to Exercise 4.14

-->

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

<head>

<title> Exercise 4.14 </title>

<script type = "text/javascript">

<!--

var result;

// Function reverser

// Parameter: a number

// Returns: the number with its digits in reverse order

// Note: Math.floor must be used to get the integer part

// of the division operations

function reverser(num) {

var digit, position = 0;

// If the number has just one digit, that's it

if (num < 10)

return num;

// Get the first digit

result = num % 10;

num = Math.floor(num / 10);

// Loop to produce the result for the rest

do {

digit = num % 10;

result = 10 \* result + digit;

num = Math.floor(num / 10);

} while (num >= 1);

return result;

}

// -->

</script>

</head>

<body>

<script type = "text/javascript">

<!--

// Function reverser tester

var num1 = 2468, num2 = 7;

result = reverser(num1);

document.write("The reverse of 2468 is: " + result + "<br />");

result = reverser(num2);

document.write("The reverse of 7 is: " + result + "<br />");

// -->

</script>

</body>

</html>

#### Chapter 5

**Exercise 5.1**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e51.hmtl

A solution to Exercise 5.1 - events with radio buttons

-->

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

<head>

<title> Exercise 5.1 </title>

<script type = "text/javascript" src = "e51.js" >

</script>

</head>

<body>

<h4> Choose your favorite color </h4>

<form>

<label> <input type = "radio" name = "colorButton"

value = "red"

onClick = "colorChoice('red')" />

Red </label>

<br />

<label> <input type = "radio" name = "colorButton"

value = "blue"

onClick = "colorChoice('blue')" />

Blue </label>

<br />

<label> <input type = "radio" name = "colorButton"

value = "green"

onClick = "colorChoice('green')" />

Green </label>

<br />

<label> <input type = "radio" name = "colorButton"

value = "yellow"

onClick = "colorChoice('yellow')" />

Yellow </label>

<br />

<label> <input type = "radio" name = "colorButton"

value = "orange"

onClick = "colorChoice('orange')" />

Orange </label>

</form>

</body>

</html>

// e51.js - The JavaScript solution for Exercise 5.1

//

// The event handler function to produce an alert message

// about the chosen color

function colorChoice (color) {

switch (color) {

case "red":

alert("Your favorite color is red");

break;

case "blue":

alert("Your favorite color is blue");

break;

case "green":

alert("Your favorite color is green");

break;

case "yellow":

alert("Your favorite color is yellow");

break;

case "orange":

alert("Your favorite color is orange");

break;

default:

alert("Error in JavaScript function colorChoice");

break;

}

}

**Exercise 5.2**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e5\_2.hmtl

A solution to Exercise 5.2 - events with radio buttons

-->

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

<head>

<title> Exercise 5.2 </title>

<script type = "text/javascript" src = "e521.js" >

</script>

</head>

<body>

<h4> Choose your favorite color </h4>

<form id = "colorsForm">

<p>

<label> <input type = "radio" name = "colorButton"

value = "red" /> Red

</label>

</p> <p>

<label> <input type = "radio" name = "colorButton"

value = "blue" /> Blue

</label>

</p><p>

<label> <input type = "radio" name = "colorButton"

value = "green" /> Green

</label>

</p><p>

<label> <input type = "radio" name = "colorButton"

value = "yellow" /> Yellow

</label>

</p><p>

<label> <input type = "radio" name = "colorButton"

value = "orange" /> Orange

</label>

</p>

</form>

<script type = "text/javascript" src = "e522.js" >

</script>

</body>

</html>

// e521.js - JavaScript for the solution to Exercise 5.2

//

// The event handler function to produce an alert message

// about the chosen color

function colorChoice () {

var color;

// Put the DOM address of the elements array in a local variable

var radioElement = document.getElementById("colorsForm").elements;

// Determine which button was pressed

for (var index = 0; index < radioElement.length; index++) {

if (radioElement[index].checked) {

color = radioElement[index].value;

break;

}

}

// Produce an alert message about the chosen color

switch (color) {

case "red":

alert("Your favorite color is red");

break;

case "blue":

alert("Your favorite color is blue");

break;

case "green":

alert("Your favorite color is green");

break;

case "yellow":

alert("Your favorite color is yellow");

break;

case "orange":

alert("Your favorite color is orange");

break;

default:

alert("Error in JavaScript function colorChoice");

break;

}

}

// e522.js - Handler registration for the solution to

// Exercise 5.2

var dom = document.getElementById("colorsForm");

dom.elements[0].onclick = colorChoice;

dom.elements[1].onclick = colorChoice;

dom.elements[2].onclick = colorChoice;

dom.elements[3].onclick = colorChoice;

dom.elements[4].onclick = colorChoice;

**Exercise 5.4**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e54.html

A solution to Exercise 5.4

-->

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

<head>

<title> Exercise 5.4 </title>

<script type = "text/javascript" src = "e54.js" >

</script>

</head>

<body>

<h3> Order Form </h3>

<form name = "orderForm" onSubmit = "finish()">

<p>

<label> <input type = "text" name = "apples"

size = "3"

onChange = "appleHandler()" />

Apples

</label>

</p><p>

<label> <input type = "text" name = "oranges"

size = "3"

onChange = "orangeHandler()" />

Oranges

</label>

</p><p>

<label> <input type = "text" name = "bananas"

size = "3"

onChange = "bananaHandler()" />

Bananas

</label>

</p><p>

<input type = "reset" name = "reset" />

<input type = "submit" name = "submit" />

</p>

</form>

</body>

</html>

// e54.js - The JavaScript file for the solution

// to Exercise 5.4

var total = 0;

// The event handler functions for the text boxes

function appleHandler() {

var number = document.orderForm.apples.value;

total = total + number \* 0.59;

}

function orangeHandler() {

var number = document.orderForm.oranges.value;

total = total + number \* 0.49;

}

function bananaHandler() {

var number = document.orderForm.bananas.value;

total = total + number \* 0.39;

}

// The event handler function to produce the total cost message

// when "submit" is clicked

function finish() {

total = total \* 1.05;

alert("Thank you for your order \n" +

"Your total cost is: $" + total + "\n");

}

**Exercise 5.5**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e55.html

The XHTML part of a solution to Exercise 5.5

-->

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

<head>

<title> Exercise 5.5 </title>

<script type = "text/javascript" src = "e551.js">

</script>

</head>

<body>

<h3> Order Form </h3>

<form name = "" onSubmit = "finish()">

<p>

<label> <input type = "text" id = "apples" size = "3" />

Apples

</label>

</p><p>

<label> <input type = "text" id = "oranges" size = "3" />

Oranges

</label>

</p><p>

<label> <input type = "text" id = "bananas" size = "3" />

Bananas

</label>

</p><p>

<input type = "reset" name = "reset" />

<input type = "submit" name = "submit" />

</p>

</form>

<script type = "text/javascript" src = "e552.js">

</script>

</body>

</html>

// e551.js -- The JavaScript part of the solution

// to Exercise 5.5

var total = 0;

// The event handler functions for the text boxes

function appleHandler() {

var myApple = document.getElementById("apples");

var number = myApple.value;

if (number < 0 || number > 99) {

alert("Error - the quantity you entered in not valid" +

"\n [It is not in the range of 0 - 99] \n" +

"Please enter a valid quantity");

myApple.focus();

myApple.select();

return false;

}

else {

total = total + number \* 0.59;

return true;

}

}

function orangeHandler() {

var myOrange = document.getElementById("oranges");

var number = myOrange.value;

if (number < 0 || number > 99) {

alert("Error - the quantity you entered in not valid" +

"\n [It is not in the range of 0 - 99] \n" +

"Please enter a valid quantity");

myOrange.focus();

myOrange.select();

return false;

}

else {

total = total + number \* 0.39;

return true;

}

}

function bananaHandler() {

var myBanana = document.getElementById("bananas");

var number = myBanana.value;

if (number < 0 || number > 99) {

alert("Error - the quantity you entered in not valid" +

"\n [It is not in the range of 0 - 99] \n" +

"Please enter a valid quantity");

myBanana.focus();

myBanana.select();

return false;

}

else {

total = total + number \* 0.49;

return true;

}

}

// The event handler function to produce the total cost message

// when "submit" is clicked

function finish() {

total = total \* 1.05;

alert("Thank you for your order \n" +

"Your total cost is: $" + total + "\n");

}

// e552.js - The JavaScript code to register the

// handlers

document.getElementById("apples").onchange = appleHandler;

document.getElementById("oranges").onchange = orangeHandler;

document.getElementById("bananas").onchange = bananaHandler;

**Exercise 5.6**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e56.html

A solution to Exercise 5.6

-->

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

<head>

<title> Exercise 5.6 </title>

<script type = "text/javascript" src = "e561.js" >

</script>

</head>

<body>

<h3> Order Form </h3>

<form name = "orderForm" onSubmit = "finish()">

<p>

<label> <input type = "text" id = "apples"

size = "3" /> Apples

</label>

</p><p>

<label> <input type = "text" id = "oranges"

size = "3" /> Oranges

</label>

</p><p>

<label> <input type = "text" id = "bananas"

size = "3" /> Bananas

</label>

</p><p>

<input type = "reset" name = "reset" />

<input type = "submit" name = "submit" />

</p>

</form>

<script type = "text/javascript" src = "e562.js" >

</script>

</body>

</html>

// e561.js - Event handlers for the solution to

// Exercise 5.6

var total = 0;

// The event handler functions for the text boxes

function appleHandler() {

var dom = document.getElementById("apples");

var number = dom.value;

if (number < dom.min || number > dom.max) {

alert("Error - the quantity you entered in not valid" +

"\n [It is not in the range of " + dom.min +

" to " + dom.max + "] \n" +

"Please enter a valid quantity");

dom.focus();

dom.select();

return false;

}

else {

total = total + number \* 0.59;

return true;

}

}

function orangeHandler() {

var dom = document.getElementById("oranges");

var number = dom.value;

if (number < dom.min || number > dom.max) {

alert("Error - the quantity you entered in not valid" +

"\n [It is not in the range of " + dom.min +

" to " + dom.max + "] \n" +

"Please enter a valid quantity");

dom.focus();

dom.select();

return false;

}

else {

total = total + number \* 0.39;

return true;

}

}

function bananaHandler() {

var dom = document.getElementById("bananas");

var number = dom.value;

if (number < dom.min || number > dom.max) {

alert("Error - the quantity you entered in not valid" +

"\n [It is not in the range of " + dom.min +

" to " + dom.max + "] \n" +

"Please enter a valid quantity");

dom.focus();

dom.select();

return false;

}

else {

total = total + number \* 0.49;

return true;

}

}

// The event handler function to produce the total cost message

// when "submit" is clicked

function finish() {

total = total \* 1.05;

alert("Thank you for your order \n" +

"Your total cost is: $" + total + "\n");

}

// e562.js - The body part of the JavaScript for the

// solution to Exercise 5.6

// Get DOM addresses of the text boxes

var appleDom = document.getElementById("apples");

var orangeDom = document.getElementById("oranges");

var bananaDom = document.getElementById("bananas");

// Set the onchange properties for the event handlers

appleDom.onchange = appleHandler;

orangeDom.onchange = orangeHandler;

bananaDom.onchange = bananaHandler;

// Add properties for minimum and maximum values for the text boxes

appleDom.max = 99;

appleDom.min = 0;

orangeDom.max = 99;

orangeDom.min = 0;

bananaDom.max = 99;

bananaDom.min = 0;

#### Chapter 6

##### Exercise 6.1

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e6\_1.html

A solution to Exercise 6.1

-->

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

<head>

<title> Exercise 6.1 </title>

<style type = "text/css">

/\* A style for the paragraph of text \*/

.regtext {position: absolute; top: 100px; left: 100px;

font-family: Times; font-size: 14pt; width: 330px}

/\* A style for the image \*/

.img {background-image: url(c172.gif); position: absolute;

left: 190px; top: 180px; width: 100px}

</style>

</head>

<body>

<p class = "img">

<br/><br/><br/><br/><br/>

</p>

<p class = "regtext">

I was born on July 4th,

1976, in Huckabee, Alaska.

I have three brothers and

a sister, all older than I.

My sister, Mary, is 26 years old.

She lives in Kalkan, Montana.

My oldest brother, Ron, is 32

years old. He lives in Huckabee.

My youngest brother, Max, is

28 years old. He lives in Pinkee,

Wyoming. My middle brother, Fred,

is 30 years old. He lives in

Kinkyhollow, Nebraska.

My parents, who are both still

living, still live in Huckabee.

</p>

</body>

</html>

## Exercise 6.2

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e62.html

The XHTML part of a solution to Exercise 6.2

-->

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

<head>

<title> Exercise 6.2 </title>

<style type = "text/css">

/\* A style for the paragraph of text \*/

.regtext {position: absolute; top: 150px; left: 100px;

font-family: Times; font-size: 14pt; width: 330px}

/\* A style for the image \*/

.img {background-image: url(c172.gif); position: absolute;

left: 100px; top: 150px; width: 100px}

</style>

<script type = "text/javascript" src = "e62.js" >

</script>

</head>

<body>

<h2> Background Image Position Control Buttons </h2>

<p>

<form name = "moveControl">

<label> <input type = "radio" name = "choser"

checked = "checked"

onclick = "moveIt('picture', 150, 100)" />

Northwest

</label>

<p/><p>

<label> <input type = "radio" name = "choser"

onclick = "moveIt('picture', 150, 300)" />

Northeast

</label>

</p><p>

<label> <input type = "radio" name = "choser"

onclick = "moveIt('picture', 300, 300)" />

Southeast

</label>

<p/><p>

<label> <input type = "radio" name = "choser"

onclick = "moveIt('picture', 300, 100)" />

Southwest

</label>

</p>

</form>

<p class = "img" id = "picture">

<br/><br/><br/><br/><br/>

</p>

<p class = "regtext">

I was born on July 4th,

1976, in Huckabee, Alaska.

I have three brothers and

a sister, all older than I.

My sister, Mary, is 26 years old.

She lives in Kalkan, Montana.

My oldest brother, Ron, is 32

years old. He lives in Huckabee.

My youngest brother, Max, is

28 years old. He lives in Pinkee,

Wyoming. My middle brother, Fred,

is 30 years old. He lives in

Kinkyhollow, Nebraska.

My parents, who are both still

living, still live in Huckabee.

</p>

</body>

</html>

// e62.js - The JavaScript file for the solution to

// Exercise 6.2

/\* A function to move an element \*/

function moveIt(movee, newTop, newLeft) {

dom = document.getElementById(movee).style;

/\* Change the top and left properties to perform the move \*/

dom.top = newTop + "px";

dom.left = newLeft + "px";

}

## Exercise 6.3

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e63.html

The XHTML part of a solution to Exercise 6.3

-->

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

<head>

<title> Exercise 6.3 </title>

<style type = "text/css">

/\* A style for the paragraph of text \*/

.regtext {position: absolute; top: 200px; left: 100px;

font-family: Times; font-size: 14pt; width: 330px}

</style>

<script type = "text/javascript" src = "e63.js" >

</script>

</head>

<body>

<h2> Background Image Visibility Control Buttons </h2>

<form name = "visibilityControl">

<p>

<label> <input type = "checkbox" name = "choser"

onclick = "flipImage('northwest')" />

Northwest

</label>

<p/><p>

<label> <input type = "checkbox" name = "choser"

onclick = "flipImage('northeast')" />

Northeast

</label>

</p><p>

<label> <input type = "checkbox" name = "choser"

onclick = "flipImage('southeast')" />

Southeast

</label>

</p><p>

<label> <input type = "checkbox" name = "choser"

onclick = "flipImage('southwest')" />

Southwest

</label>

</p>

</form>

<p id = "northwest" style = "background-image: url(c172.gif);

visibility: hidden; position: absolute;

left: 100px; top: 200px; width: 100px">

<br/><br/><br/><br/><br/>

</p>

<p id = "northeast" style = "background-image: url(c172.gif);

visibility: hidden; position: absolute;

left: 300px; top: 200px; width: 100px">

<br/><br/><br/><br/><br/>

</p>

<p id = "southeast" style = "background-image: url(c172.gif);

visibility: hidden; position: absolute;

left: 300px; top: 350px; width: 100px">

<br/><br/><br/><br/><br/>

</p>

<p id = "southwest" style = "background-image: url(c172.gif);

visibility: hidden; position: absolute;

left: 100px; top: 350px; width: 100px">

<br/><br/><br/><br/><br/>

</p>

<p class = "regtext">

I was born on July 4th,

1976, in Huckabee, Alaska.

I have three brothers and

a sister, all older than I.

My sister, Mary, is 26 years old.

She lives in Kalkan, Montana.

My oldest brother, Ron, is 32

years old. He lives in Huckabee.

My youngest brother, Max, is

28 years old. He lives in Pinkee,

Wyoming. My middle brother, Fred,

is 30 years old. He lives in

Kinkyhollow, Nebraska.

My parents, who are both still

living, still live in Huckabee.

</p>

</body>

</html>

// e63.js - The JavaScript part of a solution to

// Exercise 6.3

/\* A function to change the visibility of an element \*/

function flipImage(img) {

dom = document.getElementById(img).style;

/\* Change the visibility property \*/

if (dom.visibility == "visible" || dom.visibility == "show")

dom.visibility = "hidden";

else

dom.visibility = "visible";

}

## Exercise 6.5

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e65.html

A solution to Exercise 6.5

-->

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

<head>

<title> Exercise 6.5 </title>

<style type = "text/css" >

.planes {position: absolute; top: 100; left: 0; z-index: 0;" />

</style>

<script type = "text/javascript" src = "e65.js" >

</script>

</head>

<body>

<p>

<a href = "JAVASCRIPT:toTop('C172')">

1

</a>

</p><p>

<a href = "JAVASCRIPT:toTop('cix')">

2

</a>

</p><p>

<a href = "JAVASCRIPT:toTop('C182')">

3

</a>

</p><p>

<img class = "planes" id = "C172" src = "c172.gif"

alt = "(Picture of a C172)" />

<img class = "planes" id = "cix" src = "cix.gif"

alt = "(Picture of a Citation airplane)" />

<img class = "planes" id = "C182" src = "c182.gif"

alt = "(Picture of a C182)" />

</p><p></p>

</body>

</html>

// e65.js - The JavaScript part of a solution to

// Exercise 6.5

var top = "C172";

function toTop(newTop) {

// Get DOM addresses for the new top and the old top elements

domTop = document.getElementById(top).style;

domNew = document.getElementById(newTop).style;

// Set the zIndex properties of the two elements

domTop.zIndex = "0";

domNew.zIndex = "10";

top = newTop;

}

**Chapter 7**

## Exercise 7.1

<?xml version = "1.0" encoding = "utf-8"?>

<!-- cars.dtd - a document type definition for

the cars.xml document

A solution to Exercise 7.1

-->

<!ELEMENT car\_catalog (car+)>

<!ELEMENT car (make, model, year, color, engine,

number\_of\_doors, transmission\_type, accessories)>

<!ELEMENT make (#PCDATA)>

<!ELEMENT model (#PCDATA)>

<!ELEMENT year (#PCDATA)>

<!ELEMENT color (#PCDATA)>

<!ELEMENT engine (number\_of\_cylinders, fuel\_system)>

<!ELEMENT number\_of\_doors (#PCDATA)>

<!ELEMENT transmission\_type (#PCDATA)>

<!ELEMENT accessories (#PCDATA)>

<!ATTLIST accessories radio CDATA #REQUIRED>

<!ATTLIST accessories air\_conditioning CDATA #REQUIRED>

<!ATTLIST accessories power\_windows CDATA #REQUIRED>

<!ATTLIST accessories power\_steering CDATA #REQUIRED>

<!ATTLIST accessories power\_brakes CDATA #REQUIRED>

<!ENTITY c "Chevrolet">

<!ENTITY f "Ford">

<!ENTITY d "Dodge">

<!ENTITY h "Honda">

<!ENTITY n "Nisson">

<!ENTITY t "Toyota">

**Exercise 7.2**

<?xml version = "1.0" encoding = "utf-8"?>

<!-- cars.xml - A solution to Exercise 7.2

-->

<!DOCTYPE car\_catalog SYSTEM "cars.dtd">

<?xml-stylesheet type = "text/css" href = "cars.css"?>

<car\_catalog>

<car>

<year> 1997 </year>

<make> &c; </make>

<model> Impala </model>

<color> Light blue </color>

<engine>

<number\_of\_cylinders> 8 cylinder

</number\_of\_cylinders>

<fuel\_system> multi-port fuel injected </fuel\_system>

</engine>

<number\_of\_doors> 4 door </number\_of\_doors>

<transmission\_type> 4 speed automatic

</transmission\_type>

<accessories radio = "yes" air\_conditioning = "yes"

power\_windows = "yes"

power\_steering = "yes"

power\_brakes = "yes" />

</car>

<car>

<year> 1965 </year>

<make> &f; </make>

<model> Mustang </model>

<color> White </color>

<engine>

<number\_of\_cylinders> 8 cylinder

</number\_of\_cylinders>

<fuel\_system> 4BBL carburetor </fuel\_system>

</engine>

<number\_of\_doors> 2 door </number\_of\_doors>

<transmission\_type> 3 speed manual </transmission\_type>

<accessories radio = "yes" air\_conditioning = "no"

power\_windows = "no" power\_steering = "yes"

power\_brakes = "yes" />

</car>

<car>

<year> 1985 </year>

<make> &t; </make>

<model> Camry </model>

<color> Blue </color>

<engine>

<number\_of\_cylinders> 4 cylinder

</number\_of\_cylinders>

<fuel\_system> fuel injected </fuel\_system>

</engine>

<number\_of\_doors> 4 door </number\_of\_doors>

<transmission\_type> 4 speed manual </transmission\_type>

<accessories radio = "yes" air\_conditioning = "yes"

power\_windows = "no" power\_steering = "yes"

power\_brakes = "yes" />

</car>

</car\_catalog>

**Exercise 7.4**

<!-- cars.css - a style sheet for the cars.xml document

A solution to Exercise 8.4

-->

car {display: block; margin-top: 15px; color: blue;}

year, make, model {color: red; font-size: 16pt;}

color {display: block; margin-left: 20px; font-size: 12pt;}

engine {display: block; margin-left: 20px;}

number\_of\_cylinders {font-size: 12pt;}

fuel\_system {font-size: 12pt;}

number\_of\_doors {display: block; margin-left: 20px; font-size: 12pt;}

transmission\_type {display: block; margin-left: 20px; font-size: 12pt;}

**Exercise 7.5**

<?xml version = "1.0" encoding = "utf-8"?>

<!-- xslcar.xsl

A solution to Exercise 7.5

-->

<xsl:stylesheet xmlns:xsl = "http://www.w3.org/TR/WD-xsl"

xmlns = "http://www.w3.org/TR/REC-html40">

<xsl:template match = "/">

<h2> Car Description </h2>

<span style = "font-style: italic"> Year: </span>

<xsl:value-of select = "car\_catalog/car/year" /> <br />

<span style = "font-style: italic"> Make: </span>

<xsl:value-of select = "car\_catalog/car/make" /> <br />

<span style = "font-style: italic"> Model: </span>

<xsl:value-of select = "car\_catalog/car/model" /> <br />

<span style = "font-style: italic"> Color: </span>

<xsl:value-of select = "car\_catalog/car/color" /> <br />

<span style = "font-style: italic"> Cylinders: </span>

<xsl:value-of select =

"car\_catalog/car/engine/number\_of\_cylinders" /> <br />

<span style = "font-style: italic"> Fuel system: </span>

<xsl:value-of select = "car\_catalog/car/engine/fuel\_system" />

<br />

<span style = "font-style: italic"> Doors: </span>

<xsl:value-of select = "car\_catalog/car/number\_of\_doors" />

<br />

</xsl:template>

</xsl:stylesheet>

## Exercise 7.6

<?xml version = "1.0" encoding = "utf-8"?>

<!-- xslcars.xsl

A solution to Exercise 7.6

-->

<xsl:stylesheet xmlns:xsl = "http://www.w3.org/TR/WD-xsl"

xmlns = "http://www.w3.org/TR/REC-html40">

<xsl:template match = "/">

<h2> Car Description </h2>

<xsl:for-each select = "car\_catalog/car">

<span style = "font-style: italic"> Year: </span>

<xsl:value-of select = "year" /> <br />

<span style = "font-style: italic"> Make: </span>

<xsl:value-of select = "make" /> <br />

<span style = "font-style: italic"> Model: </span>

<xsl:value-of select = "model" /> <br />

<span style = "font-style: italic"> Color: </span>

<xsl:value-of select = "color" /> <br />

<span style = "font-style: italic"> Cylinders: </span>

<xsl:value-of select = "engine/number\_of\_cylinders" />

<br />

<span style = "font-style: italic"> Fuel system: </span>

<xsl:value-of select = "engine/fuel\_system" /> <br />

<span style = "font-style: italic"> Doors: </span>

<xsl:value-of select = "number\_of\_doors" />

<br /><br />

</xsl:for-each>

</xsl:template>

</xsl:stylesheet>

#### Chapter 9

## Exercise 9.1

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e9\_1.php

A solution to Exercise 9.1

-->

<html>

<head> <title> e9\_1.php </title>

<?php

function unique($strings) {

$uniqueStrings = array();

foreach ($strings as $string) {

foreach ($uniqueStrings as $uString) {

if ($string == $uString) break;

}

if ($string != $uString)

$uniqueStrings[] = $string;

}

return $uniqueStrings;

}

?>

</head>

<body>

<?php

$str = array(42, 24, 2, 4, 42, 24, 2, 4, 24, 42, 42, 24);

$uStr = unique($str);

foreach ($uStr as $st)

print ("$st <br />");

?>

</body>

</html>

**Exercise 9.3**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e9\_3.php

Uses a function to determine the three most frequently

occurring strings in a given array of strings and returns

them in an array.

-->

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

<head> <title> e9\_3.php </title>

</head>

<body>

<?php

// Function the\_three

// Parameter: an array of string

// Returns: an array of the three strings that occur most often

// in the given array

function the\_three($in\_array) {

// Create the empty word frequency array

$freq = array();

// Loop to count the words (either increment or initialize to 1)

foreach ($in\_array as $word) {

$keys = array\_keys($freq);

if(in\_array($word, $keys))

$freq[$word]++;

else

$freq[$word] = 1;

}

arsort($freq);

$new\_keys = array\_keys($freq);

return array($new\_keys[0], $new\_keys[1], $new\_keys[2]);

} #\*\* End of the\_three

// Main test driver

$test\_array = array("apples", "are", "good", "for", "you", "or",

"don't", "you", "like", "apples", "or", "maybe", "you", "like",

"oranges", "better", "than", "apples");

// Call the function

$tbl = the\_three($test\_array);

// Display the words and their frequencies

print "<br /> The Three Most Frequently Occurring Words<br /><br />";

$sorted\_keys = array\_keys($tbl);

sort($sorted\_keys);

foreach ($sorted\_keys as $word)

print "$tbl[$word] <br />";

?>

</body>

</html>

**Exercise 9.5**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e9\_5.php

Use a function to find the first four-digit number

in a given string.

-->

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

<head> <title> e9\_5.php </title>

</head>

<body>

<?php

// Function first\_four

// Parameter: a string containing numbers separated by spaces

// Returns: the first four-digit number found in the string;

// returns zero if the string has no four-digit

// number

function first\_four($str) {

// Split the given string into its numbers

$numbers = preg\_split("/ +/", $str);

foreach ($numbers as $numb)

if (strlen($numb) == 4)

return $numb;

// If there was no four-digit number, return false

return false;

} #\*\* End of first\_four

// Main test driver

$test\_str = "22 1 444 66 333 4444 55555 66666 2 1 9 555";

// Call the function

$result = first\_four($test\_str);

// If the returned value has four digits, display it

if ($result != 0)

print("The first four-digit number is: $result <br />");

else

print("There was no four-digit number in the string <br />");

?>

</body>

</html>

**Exercise 9.7**

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e9\_7.php

Uses a function to determine the three most frequently

occurring words in the given string, where the words

are delimited on the left by spaces and on the right

by commas, periods, or question marks

-->

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

<head> <title> e9\_7.php </title>

</head>

<body>

<?php

// Function the\_three

// Parameter: a string containing words that are delimited

// on the left by spaces and on the right by

// commas, periods, or question marks

// Returns: an array of the three words that occur most often

// in the given array

function the\_three($in\_str) {

// Split the string into its words

preg\_match\_all("/ +[A-Za-z]+[,.?]/u", $in\_str, $word\_list);

foreach ($word\_list[0] as $word)

print("The words are: $word <br />");

// Create the empty word frequency array

$freq = array();

// Loop to count the words (either increment or initialize to 1)

foreach ($word\_list[0] as $word) {

// First, get rid of the delimiters

preg\_match("/[A-Za-z]+/", $word, $match);

$word = $match[0];

// Now, get an array of the keys

$keys = array\_keys($freq);

// Set the frequency for the work

if(in\_array($word, $keys))

$freq[$word]++;

else

$freq[$word] = 1;

}

// Sort the frequency array in reverse order of values

arsort($freq);

// Get the keys and return the first three

$new\_keys = array\_keys($freq);

return array($new\_keys[0], $new\_keys[1], $new\_keys[2]);

} #\*\* End of the\_three

// Main test driver

$test\_str = " apples. are, good? for, you, or,

don't? you, like? apples, or. maybe. you, like,

oranges, better. than. apples?";

// Call the function

$tbl = the\_three($test\_str);

// Display the words and their frequencies

print "<br /> The Three Most Frequently Occurring Words<br /><br />";

$sorted\_keys = array\_keys($tbl);

sort($sorted\_keys);

foreach ($sorted\_keys as $word)

print "$tbl[$word] <br />";

?>

</body>

</html>

**Exercise 9.9**

<?xml version = "1.0" encoding = "utf-8" ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e99.html

A solution to Exercise 9.9

-->

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

<head> <title> Exercise 9.9 </title>

</head>

<body>

<a href = "e99.php"> Please give me a greeting </a>

</body>

</html>

<?xml version = "1.0" encoding = "utf-8" ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e99.php

The php part of a solution to Exercise 9.9

-->

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

<head> <title> Exercise 9.9 </title>

</head>

<body>

<?php

$greetings = array("Hello!", "Hej!", "Hallo!", "Hola!",

"Bonjour!");

$sub = rand(0, 4);

print "$greetings[$sub]";

?>

</body>

</html>

**Chapter 10**

**Exercise 10.3**

<?xml version = "1.0" encoding = "utf-8" ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- e103.html

This is the XHTML part of a solution to

Exercise 10.3. Uses Ajax and the name to fill

in the customer's address, given his or her name

-->

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

<head> <title> Popcorn Sales Form (Ajax) </title>

<style type = "text/css">

img {position: absolute; left: 400px; top: 50px;}

</style>

<script type = "text/JavaScript" src = "e103.js">

</script>

</head>

<body>

<h2> Welcome to Millenium Gynmastics Booster Club Popcorn

Sales

</h2>

<form action = "">

<!-- A borderless table of text widgets for name and address -->

<table>

<tr>

<td> Buyer's Name: </td>

<td> <input type = "text" name = "name"

onblur = "getAddress(this.value)"

size = "30" />

</td>

</tr>

<tr>

<td> Street Address: </td>

<td> <input type = "text" name = "address"

id = "address"

size = "30" />

</td>

</tr>

<tr>

<td> Zip code: </td>

<td> <input type = "text" name = "zip"

size = "10" />

</td>

</tr>

<tr>

<td> City </td>

<td> <input type = "text" name = "city"

size = "30" />

</td>

</tr>

<tr>

<td> State </td>

<td> <input type = "text" name = "state"

size = "30" />

</td>

</tr>

</table>

<img src = "popcorn.jpg" alt = "picture of popcorn" />

<p />

<!-- The submit and reset buttons -->

<p>

<input type = "submit" value = "Submit Order" />

<input type = "reset" value = "Clear Order Form" />

</p>

</form>

</body>

</html>

// e103.js

// Ajax JavaScript code for the e103.html document

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// function getAddress

// parameter: name

// action: create the XMLHttpRequest object, register the

// handler for onreadystatechange, prepare to send

// the request (with open), and send the request,

// along with the name, to the server

// includes: the anonymous handler for onreadystatechange,

// which is the receiver function, which gets the

// response text, and puts it in the document

function getAddress(name) {

alert ("The value of name in getAddress is:" + name + "\n");

var xhr = new XMLHttpRequest();

// Register the embedded receiver function as the handler

xhr.onreadystatechange = function () {

if (xhr.readyState == 4) {

var result = xhr.responseText;

alert ("responseText is: " + result + "\n");

document.getElementById("address").value = result;

}

}

xhr.open("GET", "getStreet.php?name=" + name, true);

xhr.send(null);

}

<?php

// getStreet.php

// Gets the form value from the "name" widget, looks up the

// address for that name, and prints it for the

// form

$addresses = array("John Beagle" => "1301 East 12th Street",

"Fred Magillicoti" => "12361 Bellview Road",

"John Smith" => "127 North Main Street",

"Mike Mutt" => "422 Dogtail Road",

"Pussy Cat" => "8913 Alans Alley"

);

header("Content-Type: text/plain");

$name = $\_GET["name"];

if (array\_key\_exists($name, $addresses))

print $addresses[$name];

else

print " ";

?>

**Exercise 10.5**

<!-- e105.html

Solution to Exercise 10.5

-->

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

<head> <title> Popcorn Sales Form (Ajax-Dojo) </title>

<style type = "text/css">

img {position: absolute; left: 400px; top: 50px;}

</style>

<script type = "text/javascript"

src = "dojo/dojo.js">

</script>

<script type = "text/javascript" src = "e105.js">

</script>

</head>

<body>

<h2> Welcome to Millenium Gynmastics Booster Club Popcorn

Sales

</h2>

<form action = "">

<!-- A borderless table of text widgets for name and address -->

<table>

<tr>

<td> Buyer's Name: </td>

<td> <input type = "text" name = "name"

size = "30" />

</td>

</tr>

<tr>

<td> Street Address: </td>

<td> <input type = "text" name = "street"

size = "30" />

</td>

</tr>

<tr>

<td> Zip code: </td>

<td> <input type = "text" name = "zip"

size = "10"

onblur = "getPlace(this.value)" />

</td>

</tr>

<tr>

<td> City </td>

<td> <input type = "text" name = "city"

id = "city" size = "30" />

</td>

</tr>

<tr>

<td> State </td>

<td> <input type = "text" name = "state"

id = "state" size = "30" />

</td>

</tr>

</table>

<img src = "popcorn.jpg" alt = "picture of popcorn" />

<p />

<!-- The submit and reset buttons -->

<p>

<input type = "submit" value = "Submit Order" />

<input type = "reset" value = "Clear Order Form" />

</p>

</form>

</body>

</html>

// e105.js

// Ajax JavaScript code for the e105.html document

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// function getPlace

// parameter: zip code

// action: call dojo.io.bind to do the Ajax request for

// the city and state

function getPlace(zip) {

dojo.io.bind({

url: "getCityState.php?zip=" + zip,

load: function (type, data, evt) {

var place = data.split(', ');

if (dojo.byId("city").value == "")

dojo.byId("city").value = place[0];

if (dojo.byId("state").value == "")

dojo.byId("state").value = place[1];

},

error: function (type, data, evt) {

alert("Error in request, returned data: " + data);

},

method: "GET",

mimetype: "text/plain"

} );

}

**Chapter 11**

**Exercise 11.2**

<%--

Document : index (of the greet5 (Exercise 11.2) project)

Created on : Mar 14, 2009, 8:23:49 PM

Author : bob

Purpose : Display one of five different

greetings on each call (Exercise 11.2)

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type"

content="text/html; charset=UTF-8">

<title>Exercise 11.2</title>

</head>

<body>

<form action = "Greet5" method = "get">

<p>

Press the button to get a greeting

<input type = "submit" value = "Get greeting" />

</p>

</form>

</body>

</html>

/\* Greet5 servlet (Exercise 11.2) \*/

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class Greet5 extends HttpServlet {

protected void processRequest(HttpServletRequest request,

HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

String [] messages = {"Howdy!", "Hej!", "Hallo!", "Hola!",

"Bonjour!"};

int sub;

try {

/\* TODO output your page here \*/

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet Greet5</title>");

out.println("</head>");

out.println("<body>");

sub = (int) (Math.random() \* 5);

out.println("<strong>" + messages[sub] + "</strong>");

out.println("</body>");

out.println("</html>");

} finally {

out.close();

}

}

protected void doGet(HttpServletRequest request,

HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

}

**Exercise 11.4**

<%--

Document : index - for the solution to Exercise 11.4

Created on : May 24, 2008, 8:37:33 PM

Author : bob

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type"

content="text/html; charset=UTF-8">

<title>Exercise 11.4</title>

</head>

<body>

<form method="POST" action="Survey">

<h2>

Welcome to the Consumer Electronics Purchasing Survey

</h2>

<p />

<h4> Your Gender: </h4>

<p>

<label>

<input type="radio" name="gender" value="female"

checked="checked" />

Female <br />

</label>

<label>

<input type="radio" name="gender" value="male" />

Male <br /> <br /> <br />

</label>

</p>

<p>

<label>

<input type="radio" name="vote" value="0" />

TV <br />

</label>

<label>

<input type="radio" name="vote" value="1" />

Digital Camera <br />

</label>

<label>

<input type="radio" name="vote" value="2" />

MP3 player <br />

</label>

<label>

<input type="radio" name="vote" value="3" />

DVD player/recorder <br />

</label>

<label>

<input type="radio" name="vote" value="4" />

Camcorder <br />

</label>

<label>

<input type="radio" name="vote" value="5" />

PDA <br />

</label>

<label>

<input type="radio" name="vote" value="6"

checked="checked" />

Other <br /> <br />

</label>

<input type = "submit" value = "Submit Vote" />

<input type = "reset" value = "Clear Vote Form" />

</p>

</form>

</body>

</html>

/\* e114 - Servlet for Exercise 11.4 \*/

import java.io.\*;

import java.net.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class Survey extends HttpServlet {

protected void processRequest(HttpServletRequest request,

HttpServletResponse response)

throws ServletException, IOException {

int votes[] = null;

int index;

int sumFemales, sumMales;

int vote;

File survdat = new File("survdat.ser");

String gender;

String products[] = {" TV", "Digital Camera", "MP3 player",

"DVD player/recorder", "Camcorder", "PDA", Other"};

// Set the content type for the response output and get a

// writer

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

// Create the initial part of the response document

out.println("<html>");

out.println("<head>");

out.println("<title>Return message</title>");

out.println("</head>");

out.println("<body>");

// Synchronize a block for the votes file access

synchronized (this) {

// If the file already exists, read in its data

try {

if (survdat.exists()) {

ObjectInputStream indat = new

ObjectInputStream(

new FileInputStream(survdat));

votes = (int[]) indat.readObject();

indat.close();

}

// If the file does not exist (this is the first

// vote), create the votes array

else {

votes = new int[15];

votes[14] = 0;

}

} catch (Exception e) {

e.printStackTrace();

}

// Get the gender of the survey respondee

gender = request.getParameter("gender");

// Increment the visits counter

votes[14]++;

// Add the consumer electronics vote of the response to

// the votes array

vote = Integer.parseInt(request.getParameter("vote"));

if (gender.equals("male")) {

vote += votes.length / 2;

}

votes[vote]++;

//Write updated votes array to disk

ObjectOutputStream outdat = new ObjectOutputStream(

new FileOutputStream(survdat));

outdat.writeObject(votes);

outdat.flush();

outdat.close();

} //\*\* end of the synchronized block

// Create the initial response information

out.println(

"<h3> Thank you for participating in the");

out.println(" Consumer Electronics Survey </h3>");

out.println("You are visitor number " + votes[14] +

"<br />");

out.println("<h4> Current Survey Results: </h4>");

// Create the total votes return information for female

// respondents

out.println("<h5> For Female Respondents </h5>");

// First, sum the votes for females and males

sumFemales = 0;

sumMales = 0;

for (index = 0; index < votes.length / 2; index++) {

sumFemales += votes[index];

sumMales += votes[index + votes.length / 2];

}

for (index = 0; index < votes.length / 2; index++) {

out.print(products[index]);

out.print(": ");

if (sumFemales > 0) {

out.print((100 \* votes[index]) / sumFemales);

out.println("%");

}

else

out.println(0);

out.println("<br />");

}

// Create the total votes return information for male

// respondents

out.println("<h5> For Male Respondents </h5>");

for (index = votes.length / 2; index < votes.length; index++) {

out.print(products[index - votes.length / 2]);

out.print(": ");

if (sumMales > 0) {

out.print((100 \* votes[index]) / sumMales);

out.println("%");

}

else

out.println(0);

out.println("<br />");

}

out.close();

}

protected void doPost(HttpServletRequest request,

HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

}

**Exercise 11.7**

<%--

Document : index for Exercise 11.7

Created on : May 25, 2008, 9:17:57 PM

Author : bob

Presents a ballot to the user and calls

the VoteCounter2 servlet for form handling

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type"

content="text/html; charset=UTF-8">

<title> Ballot </title>

</head>

<body>

<form action="VoteCounter2" method="POST">

<h3> Please choose one candidate for dogcatcher </h3>

<p>

<input type="radio" name="vote1" value="Dogman" />

Daren Dogman <br />

<input type="radio" name="vote1" value="Taildragger" />

Timmy Taildragger <br />

<input type="radio" name="vote1" value="Dogpile" />

Don Dogpile <br />

</p>

<h3> Please choose one candidate for catcatcher </h3>

<p>

<input type="radio" name="vote2" value="Catland" />

Kitty Catland <br />

<input type="radio" name="vote2" value="El Gato" />

Al El Gato <br />

<input type="radio" name="vote2" value="Katnip" />

Kitten Katnip <br />

<input type="radio" name="vote2" value="Cat" />

Tommie Cat <br />

<input type="radio" name="vote2" value="Feline" />

Fred Feline <br />

</p><p>

<input type = "submit" value = "Submit ballot" />

</p>

</form>

</body>

</html>

// VoteCounter2.java - Exercise 11.7

// This servlet processes the ballot form, returning a

// document asking for a new vote if no vote was made on the

// ballot. For legitimate ballots, the vote is added to

// the current totals, and those totals are presented to

// the user in a return document.

// A cookie is returned to the voter, recording the fact

// that a vote was received. The servlet examines all votes

// for cookies to ensure that there is no multiple voting.

// The voting data file, votesdat2.dat, is stored on the Web

// server.

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class VoteCounter2 extends HttpServlet {

Cookie cookies[] = null;

int index;

PrintWriter out;

protected void processRequest(HttpServletRequest request,

HttpServletResponse response)

throws ServletException, IOException {

Cookie newCookie;

int votes[] = null;

int sumCatVotes;

int sumDogVotes;

String vote1;

String vote2;

File votesdat2 = new File("votesdat2.ser");

String dogCandidates[] = {"Daren Dogman", "Timmy Taildragger",

"Don Dogpile"};

String catCandidates [] = {"Kitty Catland", "Al El Gato",

"Kitten Katnip", "Tommie Cat", "Fred Feline"};

// Get cookies from the request

cookies = request.getCookies();

// Check to see if there was a vote on the form

vote1 = request.getParameter("vote1");

vote2 = request.getParameter("vote2");

if (vote1 == null || vote2 == null) { //\*\* No legal vote

// Create the return document

makeHeader(response);

out.println(

"You did not vote for both <br />");

out.println(

"Please mark the ballot completely and resubmit");

} //\*\* end of if (vote == null) ...

else { //\*\* There was a vote

// Check to see if this client voted before

if (!votedBefore()) {

// No previous vote, so get the contents of the file

// (if the file already exists)

// Synchronize block for file input/output

synchronized (this) {

if (votesdat2.exists()) {

ObjectInputStream indat =

new ObjectInputStream(

new FileInputStream(votesdat2));

// We need the try/catch here because

// readObject can throw ClassNotFoundException

try {

votes = (int[]) indat.readObject();

} catch (ClassNotFoundException problem) {

problem.printStackTrace();

}

} //\*\* end of if(votesdat.exists() ...

// If the file does not exist (this is the first

// vote), create the votes array

else {

votes = new int[8];

}

// Add the new vote to the votes array

if (vote1.equals("Dogman"))

votes[0]++;

else if (vote1.equals("Taildragger"))

votes[1]++;

else votes[2]++;

if (vote2.equals("Catland"))

votes[3]++;

else if (vote2.equals("El Gato"))

votes[4]++;

else if (vote2.equals("Katnip"))

votes[5]++;

else if (vote2.equals("Cat"))

votes[6]++;

else votes[7]++;

// Write updated votes array to disk

ObjectOutputStream outdat = new ObjectOutputStream(

new FileOutputStream(votesdat2));

outdat.writeObject(votes);

outdat.flush();

outdat.close();

} //\*\* end of synchronized block

// Attach a cookie to the response

newCookie = new Cookie("iVoted", "true");

newCookie.setMaxAge(5); //\*\* Set to 5 for testing

response.addCookie(newCookie);

// Write a response message

makeHeader(response);

out.println("Your vote has been received");

out.println(

"<br /><br /> Current Voting Percentages: <br />");

// Compute the total dog and cat votes

sumCatVotes = 0;

sumDogVotes = 0;

for (index = 0; index < 3; index++)

sumDogVotes += votes[index];

for (index = 3; index < 8; index++)

sumCatVotes += votes[index];

// Create the dog votes percentages

out.println("<h3> For Dogcatcher </h3>");

for (index = 0; index < 3; index++) {

out.print(dogCandidates[index]);

out.print(": ");

if (sumDogVotes > 0)

out.println(

((100 \* votes[index]) / sumDogVotes) +

"% <br />");

else

out.println(0 + "% <br />");

}

// Create the cat votes percentages

out.println("<br /> <br /> <h3> For Catcatcher </h3>");

for (index = 3; index < 8; index++) {

out.print(catCandidates[index - 3]);

out.print(": ");

if (sumCatVotes > 0)

out.println(((100 \* votes[index]) /

sumCatVotes) + "% <br />");

else

out.println(0 + "% <br />");

}

} //\*\* end of if (!votedBefore() ...

else { // The client voted before

// Write a response message

makeHeader(response);

out.println(

"Your vote is illegal - you have already voted!");

} // end of else clause - client voted before

} // end of else (there was a vote)

// Finish response document and close the stream

out.println("</body> </html>");

out.close();

} //\*\* end of ProcessRequest

//-----------------------------------------------------------------

// Method votedBefore - return true if the client voted before;

// false otherwise

boolean votedBefore() {

if (cookies == null || cookies.length == 0) {

return false;

} else {

// Check the cookies to see if this user voted before

for (index = 0; index < cookies.length; index++) {

if (cookies[index].getName().equals("iVoted")) {

return true;

}

} // end of for (index = 0; ...

return false;

} //\*\* end of if (cookies == null ...

} //\*\* end of votedBefore

//-----------------------------------------------------------------

// Method makeHeader - get the writer and produce the

// response header

void makeHeader(HttpServletResponse response)

throws IOException {

// Set content type for response and get a writer

response.setContentType("text/html");

out = response.getWriter();

// Write the response document head and the message

out.println("<html><head>");

out.println(

"<title> Return message </title></head><body>");

} //\*\* end of makeHeader

// Method doPost - just calls processRequest

protected void doPost(HttpServletRequest request,

HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

} //\*\* end of VoteCounter2

**Chapter 12**

**Exercise 12.1**

<!-- e121.aspx

A solution to Exercise 12.1

-->

<%@ Page language="c#" %>

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

<head> <title> Exercise 12.1 </title>

</head>

<body>

<h3> Days, hours, and minutes in my life </h3>

<%

string msg, days, hours, minutes;

DateTime rightnow, birthdate;

TimeSpan timeSpan;

// Set date/time of right now and new years day

rightnow = DateTime.Now;

birthdate = new DateTime(1989, 2, 19);

// Compute the difference in time/dates

timeSpan = rightnow.Subtract(birthdate);

// Compute and display the differences in days, hours, and minutes

days = timeSpan.Days.ToString();

msg = string.Format("Days: {0}, ", days);

Response.Write(msg);

hours = timeSpan.Hours.ToString();

msg = string.Format("Hours: {0}, ", hours);

Response.Write(msg);

minutes = timeSpan.Minutes.ToString();

msg = string.Format("Minutes: {0} <br />", minutes);

Response.Write(msg);

%>

</body>

</html>

**Exercise 12.3**

<!-- e123.aspx

A solution to Exercise 12.3.

A modification of controls.aspx.

Adds text box for the user's address and a drop-down list

for the user's favorite category of music, which is populated

in the code-behind file.

-->

<%@ Page Language="C#" AutoEventWireup="true"

CodeFile="e123.aspx.cs" Inherits="\_Default" %>

<!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 id="Head1" runat="server">

<title>E123</title>

</head>

<body>

<form id="form1" runat="server">

<div>

Name: <asp:TextBox ID="name" runat="server"></asp:TextBox>

<br /><br />

Address: <asp:TextBox ID="address" runat="server"></asp:TextBox>

Favorite color:

<asp:DropDownList ID="color" runat="server"

AutoPostBack="true"

OnSelectedIndexChanged="colorItemSelected">

</asp:DropDownList>

<br /><br />

Favorite category of music:

<asp:DropDownList ID="music" runat="server"

AutoPostBack="true"

OnSelectedIndexChanged="musicItemSelected">

</asp:DropDownList>

<br /><br />

<asp:Label ID="colorMessage" runat="server" Text=""></asp:Label>

<br /><br />

<asp:Label ID="musicMessage" runat="server" Text=""></asp:Label>

</div>

</form>

</body>

</html>

// e123.aspx.cs

// The code-behind file for e123.aspx

using System;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

color.Items.Add(new ListItem("blue"));

color.Items.Add(new ListItem("red"));

color.Items.Add(new ListItem("green"));

color.Items.Add(new ListItem("yellow"));

music.Items.Add(new ListItem("rock"));

music.Items.Add(new ListItem("rap"));

music.Items.Add(new ListItem("country"));

music.Items.Add(new ListItem("classical"));

music.Items.Add(new ListItem("jazz"));

}

}

protected void colorItemSelected(object sender, EventArgs e)

{

string newMsg = string.Format(

"Hi {0}; your favorite color is {1}",

name.Text, color.SelectedItem);

colorMessage.Text = newMsg;

}

protected void musicItemSelected(object sender, EventArgs e)

{

string newMsg = string.Format(

"Hello {0}; your favorite category of music is {1}",

name.Text, music.SelectedItem);

musicMessage.Text = newMsg;

}

}

**Exercise 12.4**

<!-- e124.aspx

A solution to Exercise 12.4

-->

<%@ Page language="c#" %>

<html>

<head> <title> Exercise 12.4 </title>

</head>

<body>

<form runat = "server">

<p>

Your name:

<asp:TextBox id = "name" runat = "server" />

<asp:RequiredFieldValidator

ControlToValidate = "name"

Display = "Static"

runat = "server"

ErrorMessage = "Please enter your name">

</asp:RequiredFieldValidator>

<br />

Your address:

<asp:TextBox id = "address" runat = "server" />

<asp:RegularExpressionValidator

ControlToValidate = "address"

Display = "Static"

runat = "server"

ErrorMessage = "Address must begin with a number"

ValidationExpression = "\d+ [A-Z][a-z]\*">

</asp:RegularExpressionValidator>

<br />

Your phone number:

<asp:TextBox id = "phone" runat = "server" />

<asp:RegularExpressionValidator

ControlToValidate = "phone"

Display = "Static"

runat = "server"

ErrorMessage = "Phone number form must be ddd-ddd-dddd"

ValidationExpression = "\d{3}-\d{3}-\d{4}">

</asp:RegularExpressionValidator>

<br />

Your age:

<asp:TextBox id = "age" runat = "server" />

<asp:RangeValidator

ControlToValidate = "age"

Display = "Static"

runat = "server"

MaximumValue = "110"

MinimumValue = "10"

Type = "Integer"

ErrorMessage = "Age must be in the range of 10 to 110">

</asp:RangeValidator>

<br />

Your Social Security number:

<asp:TextBox id = "ssn" runat = "server" />

<asp:RegularExpressionValidator

ControlToValidate = "ssn"

Display = "Static"

runat = "server"

ErrorMessage =

"Social security number must have the form: ddd-dd-dddd"

ValidationExpression = "\d{3}-\d{2}-\d{4}">

</asp:RegularExpressionValidator>

<br />

<input type = "submit" value = "Submit" />

</p>

</form>

</body>

</html>

**Chapter 13**

**Exercise 13.2**

No changes to access\_cars.php are required to handle UPDATE and INSERT SQL commands. The mysql\_query function takes SQL commands as its parameter, whether they are queries, updates, or inserts. For example, we could have:

$query =

"INSERT INTO Corvettes VALUES (38, ′convertible′, 17.1, 1993, 15)";

$result = mysql\_query($query);

To determine the number of rows that were affected by the INSERT command, the mysql\_affected\_rows function is used (with the result as its parameter).

**Exercise 13.3**

// e13\_3.java

// This servlet receives an SQL command from its HTML document,

// connects to the cars database, and performs the command on

// the database. If the command was a query, it returns an HTML

// table of the results of the query

// This is a solution to Exercise 13.3

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.\*;

import java.util.\*;

import java.sql.\*;

public class e13\_3 extends HttpServlet {

private Connection myCon;

private Statement myStmt;

// The init method - instantiate the db driver, connect to the

// db, and create a statement for an SQL command

public void init() {

// Instantiate the driver for MySQL

try {

Class.forName("org.gjt.mm.mysql.Driver").newInstance();

}

catch (Exception e) {

e.printStackTrace();

}

// Create the connection to the cars db

try {

myCon = DriverManager.getConnection (

"jdbc:mysql://localhost/cars?user=root");

}

catch (SQLException e) {

e.printStackTrace();

}

// Create the statement for SQL queries

try {

myStmt = myCon.createStatement();

}

catch (Exception e) {

e.printStackTrace();

}

} //\*\* end of the init method

// The doPost method - get the query, perform it, and produce

// an HTML table of the results

public void doPost(HttpServletRequest request,

HttpServletResponse response)

throws ServletException, IOException {

ResultSet result;

String command, colName, dat;

int numCols, index;

ResultSetMetaData resultMd;

// Get the SQL command

command = request.getParameter("Command");

// Set the MIME type and get a writer

response.setContentType("text/html");

PrintWriter out = response.getWriter();

// Create the initial html and display the request

out.println("<html>");

out.println("<head><title>JDBCServlet</title></head>");

out.println("<body>");

out.print("<p><b>The command is: </b>" + command + "</p>");

// If the command is a query, use executeQuery on it

if (command.substring(0, 5).equalsIgnoreCase("SELECT")) {

try {

result = myStmt.executeQuery(command);

// Get the result's metadata and the number of result rows

resultMd = result.getMetaData();

numCols = resultMd.getColumnCount();

// Produce the table header and caption

out.println("<table border>");

out.println("<caption> <b> Query Results </b> </caption>");

out.println("<tr>");

// Loop to produce the column headings

for (index = 1; index <= numCols; index++) {

colName = resultMd.getColumnLabel(index);

out.print("<th>" + colName + "</th>");

}

out.println("</tr>");

// Loop to produce the rows of the result

while (result.next()) {

out.println("<tr>");

// Loop to produce the data of a row of the result

for (index = 1; index <= numCols; index++) {

dat = result.getString(index);

out.println("<td>" + dat + "</td>");

} //\*\* end of for (index = 0; ...

out.println("</tr>");

} //\*\* end of while (result.next()) ...

out.println("</table>");

} //\*\* end of try

catch (Exception e) {

e.printStackTrace();

} //\*\* end of catch

} //\*\* end of the then clause of if (command.substring...

else { //\*\* It is a non-query command

try {

myStmt.executeUpdate(command);

}

catch (Exception e) {

e.printStackTrace();

}

} //\*\* end of else clause of if (command.substring...

out.println("</body></html>");

} //\*\* end of doPost method

} //\*\* end of class e13\_3

**Chapter 14**

**Exercise 14.1**

# e141.rb - A solution to Exercise 14.1

# Get input

puts "Please type the three input numbers"

a = gets.to\_i

b = gets.to\_i

c = gets.to\_i

# Compute result

result = 10 \* a \* b - ((c - 1) / 17.44)

# Display the result

puts "The result is: #{result}"

**Exercise 14.3**

# e143.rb - A solution to Exercise 14.3

# Get input

puts "Please type the three names, on separate lines"

name1 = gets

name2 = gets

name3 = gets

# Pass one - compare/interchange both pairs

if name2 < name1

temp = name1

name1 = name2

name2 = temp

end

if name3 < name2

temp = name2

name2 = name3

name3 = temp

end

# Pass two - compare/interchange the first two

if name2 < name1

temp = name1

name1 = name2

name2 = temp

end

# Display results

puts "The names in order are: \n #{name1} #{name2} #{name3}"

**Exercise 14.5**

# e145.rb - A solution to Exercise 14.5

# Create the two arrays

pos\_list = Array.new

neg\_list = Array.new

# Loop to read input and build the two new arrays

while next\_one = gets

next\_one = next\_one.to\_i

# If the number is positive, put it in the positive array

if next\_one > 0

pos\_list.push(next\_one)

elsif next\_one < 0

neg\_list.push(next\_one)

end

end

# Display resulting arrays

puts "The positive array is:"

for value in pos\_list

puts value

end

puts "The negative array is:"

for value in neg\_list

puts value

end

**Chapter 15**

**Exercise 15.5**

class AdderController < ApplicationController

def the\_form

end

# result method - fetch data and compute the sum

def result

# Fetch the form values

@first = params[:first].to\_i

@second = params[:second].to\_i

# Compute the sum

@sum = @first + @second

end

end

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- the\_form.html.erb - the initial template for e155

-->

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

<head>

<title> The initial template for Exercise 15.5 </title>

</head>

<body>

<h2> Please input two integers </h2>

<form action = result method = "post">

<label> <input type = "text" name = "first"

size = "5" />

First number

</label>

<br />

<label> <input type = "text" name = "second"

size = "5" />

Second number

</label>

<br /> <br />

<input type = "submit" value = "Submit data" />

<br />

<input type = "reset" value = "Clear form" />

</form>

</body>

</html>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- result.html.erb - the template for the result of e155

-->

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

<head>

<title> The result template for Exercise 15.5 </title>

</head>

<body>

<p> The sum of your input is: <%= @sum %> </p>

</body>

</html>

**Exercise 15.6**

# main\_controller.rb - for the fbplayers application

# Exercise 15.6

class MainController < ApplicationController

# welcome method - the initial view

def welcome

end

# result method - fetches values for the result view

def result

@age1 = params[:age1]

@age2 = params[:age2]

@team = params[:team]

@selected\_players = Player.find(:all, :conditions =>

["age >= ? and age <= ? and team = ?",

@age1, @age2, @team])

end

end

<!-- welcome.html.erb - initial view for the fbplayers application

(Exercise 15.6)

-->

<!-- The form to collect input from the user about their interests -->

<form action = "result" method = "post" >

<span class = "labels"> From age: </span>

<input type = "text" size = "4" name = "age1" />

<span class = "labels"> To age: </span>

<input type = "text" size = "4" name = "age2" />

<span class = "labels"> Team: </span>

<input type = "text" size = "12" name = "team" />

<br />

<input type = "submit" value = "Submit request" /> <br />

<input type = "reset" value = "Reset form" /> <br />

</form>

<!-- result.html.erb - the result of the user request for

information about players (Exercise 15.6)

-->

<p>

<!-- Display what the user asked for -->

Players aged from <%= @age1 %> to <%= @age2 %>

with the <%= @team %> team

</p>

<!-- Display the results of the request in a table -->

<table border = "border">

<tr>

<th> Player </th>

<th> Age </th>

</tr>

<!-- Put the cars in @selected\_players in the table -->

<% @selected\_players.each do |player| %>

<tr>

<td> <%= player.name %> </td>

<td> <%= player.age %> </td>

</tr>

<% end %> <!-- end of do loop -->

</table>

#### Appendix C

## Exercise C.1

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- ec\_1.html

To test the applet, ec\_1.java

Solution to Exercise 7.1

Legal styles are BOLD, PLAIN, and ITALIC

-->

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

<head>

<title> Test ec\_1.java </title>

</head>

<body>

<p>

<object codetype = "application/java"

code = "ec\_1.class"

width = "600"

height = "100">

<param name = "size"

value = "35" />

<param name = "font"

value = "Courier" />

<param name = "style"

value = "ITALIC" />

</object>

</p>

</body>

</html>

/\* ec\_1.java

An applet to illustrate parameters

\*/

import java.applet.\*;

import javax.swing.\*;

import java.awt.\*;

// The panel class on which the message will be painted

class MessagePanel2 extends JPanel {

Font myFont = new Font(ec\_1.myFont, ec\_1.myStyle,

ec\_1.mySize);

public void paintComponent(Graphics grafObj) {

super.paintComponent(grafObj);

grafObj.setFont(myFont);

grafObj.drawString("Welcome to my home page!", 50, 50);

}

}

// The ec\_1 applet

public class ec\_1 extends JApplet {

static int mySize, myStyle;

static String myFont;

public void init() {

Container messageArea = getContentPane();

String pString;

// Get the fontsize parameter

pString = getParameter("size");

// If it's null, set the size to 30, otherwise

// use the parameter value

if (pString == null)

mySize = 30;

else mySize = Integer.parseInt(pString);

// Get the font parameter

pString = getParameter("font");

// If it's null, set the font to 'Times Roman',

// otherwise, use the parameter value

if (pString == null)

else myFont = pString;

// Get the font style parameter

pString = getParameter("style");

// If it's null, set the font to PLAIN, otherwise

// use the parameter value

if (pString == null)

myStyle = Font.PLAIN;

else if (pString.equals("BOLD"))

myStyle = Font.BOLD;

else if (pString.equals("ITALIC"))

myStyle = Font.ITALIC;

// Instantiate the panel with the message and add it to

// the content pane

MessagePanel2 myMessagePanel = new MessagePanel2();

messageArea.add(myMessagePanel);

}

}

## Exercise c.3

<?xml version = "1.0" encoding = "utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- ec\_3.html

To test the applet, Olympic

A solution to Exercise c.3

-->

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

<head>

<title> Test Olympic </title>

</head>

<body>

<p>

<object codetype = "application/java"

code = "Olympic.class"

width = "460"

height = "180">

</object>

</p>

</body>

</html>

/\* Olympic.java

A solution to Exercise c.3 - the applet

\*/

import java.applet.\*;

import java.awt.\*;

import javax.swing.\*;

// The panel class for drawing

class MyPanel extends JPanel {

public void paint(Graphics grafObj) {

Font myFont = new Font("TimesRoman", Font.ITALIC, 24);

super.paintComponent(grafObj);

grafObj.setColor(Color.green);

grafObj.drawRect(10, 10, 440, 120);

grafObj.setColor(Color.blue);

grafObj.drawOval(20, 20, 100, 100);

grafObj.drawOval(100, 20, 100, 100);

grafObj.drawOval(180, 20, 100, 100);

grafObj.drawOval(260, 20, 100, 100);

grafObj.drawOval(340, 20, 100, 100);

grafObj.setColor(Color.red);

grafObj.setFont(myFont);

grafObj.drawString("United States Olympic Committee", 60, 160);

}

}

// The Olympic applet

public class Olympic extends JApplet {

Container olympicArea = getContentPane();

MyPanel newPanel = new MyPanel();

// The init method for the applet - adds the panel

// to the content area of the applet

public void init() {

olympicArea.add(newPanel);

}

}

## Exercise c.4

/\* Mexfood.java

A solution to Exercise c.4

\*/

import java.awt.\*;

import java.applet.\*;

import javax.swing.\*;

public class Mexfood extends JApplet {

// Create a content pane

Container contentPane = getContentPane();

// The init method, which does everything in this applet

public void init() {

// Create a panel object and set its layout manager to put

// the components in a column

JPanel myPanel = new JPanel();

myPanel.setLayout(new GridLayout(4, 3, 10, 10));

myPanel.setBackground(Color.cyan);

// Create checkboxes and textboxes for the different food items

// and add them to the panel

TextField t1 = new TextField(2);

TextField t2 = new TextField(2);

TextField t3 = new TextField(2);

TextField t4 = new TextField(2);

JLabel l1 = new JLabel("Tacos Quantity:");

JLabel l2 = new JLabel("Chalupas Quantity:");

JLabel l3 = new JLabel("Burritos Quantity:");

JLabel l4 = new JLabel("Nachos Quantity:");

myPanel.add(l1);

myPanel.add(t1);

myPanel.add(l2);

myPanel.add(t2);

myPanel.add(l3);

myPanel.add(t3);

myPanel.add(l4);

myPanel.add(t4);

// Now add the panel

contentPane.add(myPanel);

} // End of init()

} // End of Mexfood applet

**Exercise c.5**

/\* FoodCost.java

A solution to Exercise c.5

\*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

import javax.swing.\*;

// The applet

public class FoodCost extends JApplet implements ActionListener {

private JPanel myPanel = new JPanel();

private JTextField t1, t2, t3, t4, t5;

private JLabel l1, l2, l3, l4, l5;

private double totalCost = 0.0;

Container contentPane = getContentPane();

public void init() {

// Create a panel object and set its layout manager to put

// the components in a column

myPanel.setLayout(new GridLayout(5, 2, 10, 10));

myPanel.setBackground(Color.cyan);

// Create checkboxes and textboxes for the different food items

// and add them to the panel

l1 = new JLabel("Tacos Quantity:");

l2 = new JLabel("Chalupas Quantity:");

l3 = new JLabel("Burritos Quantity:");

l4 = new JLabel("Nachos Quantity:");

l5 = new JLabel("Total Cost:");

t1 = new JTextField(2);

t2 = new JTextField(2);

t3 = new JTextField(2);

t4 = new JTextField(2);

t5 = new JTextField(6);

myPanel.add(l1);

myPanel.add(t1);

myPanel.add(l2);

myPanel.add(t2);

myPanel.add(l3);

myPanel.add(t3);

myPanel.add(l4);

myPanel.add(t4);

myPanel.add(l5);

myPanel.add(t5);

// Register the event handler

t1.addActionListener(this);

t2.addActionListener(this);

t3.addActionListener(this);

t4.addActionListener(this);

// Now add the panel

contentPane.add(myPanel);

} // End of init()

// The event handler

public void actionPerformed(ActionEvent e) {

// Determine which textbox has been changed and add the cost

int quantity;

Object source = e.getSource();

if (source == t1) {

quantity = Integer.parseInt(t1.getText());

totalCost = totalCost + quantity \* 0.79;

} else if (source == t2) {

quantity = Integer.parseInt(t2.getText());

totalCost = totalCost + quantity \* 1.19;

} else if (source == t3) {

quantity = Integer.parseInt(t3.getText());

totalCost = totalCost + quantity \* 1.39;

} else if (source == t4) {

quantity = Integer.parseInt(t4.getText());

totalCost = totalCost + quantity \* 1.29;

}

String cost = Double.toString(totalCost);

t5.setText("$" + cost);

repaint();

} //\* end of method ActionEvent

} // End of Mexfood applet