COMPSCI334

Example site HTML and PHP scripts used in the lectures

Ulrich Speidel

The scripts in this document are available under:

http://www.cs.auckland.ac.nz/courses/compsci334s1t/lectures/exampleSite.zip

The following file is the index file of the bird supporters' home page, **`index.html`**:

```
<!-- This page and the associated pages are copyright to Ulrich Speidel and -->
<!-- may be used for study purposes only. Copyright violations will be      -->
<!-- prosecuted - and have been in the past!                                -->
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <title>Virtual Supporter's Club for Endangered New Zealand Birds</title>
</head>
    <body style="font-family: Arial; font-size: 10pt;">
        <h1>Virtual Supporter's Club for Endangered New Zealand Birds</h1>
        <p>Welcome to the home page of our virtual supporter's club!</p>
        <p>On this page, you can find information about our projects and about the bird species for whose protection we work:</p>
        <img src="takahe.jpg" alt="Picture of a Takahe" align="right"/>
        <table bgcolor="#ffff66">
            <tbody>
                <tr>
                    <td>
                        <h2>Endangered feathered friends:</h2>
                        <ul>
                            <li>Our "mascot": the rare
                                <a href="birds/takahe.html" style="text-decoration: none;"><b><i>takahe</i></b></a>
                            </li>
                            <li>The <a href="http://www.kiwirecovery.org.nz" style="text-decoration: none;"><b><i>kiwi</i></b></a>
                                - New Zealand's best-known bird
```

```
                    </li>
                    <li>The <a href="http://www.kakaporecovery.org.nz" style="text-decoration: none;"><b><i>kakapo</i></b></a>
                        - one of the rarest birds in the worlds
                    </li>
                </ul>
            </td>
        </tr>
        <tr>
            <td bgcolor="#ccccff">
                <div align="center">
                    <h3>Project of the Year</h3>
                    <a href="donationsForm.php">Donate towards our bird sanctuary!</a>

                    <h3>Projects of other organisations</h3>

                    <ol>
                        <li><a href="http://www.kiwirecovery.org.nz/">The Kiwi Recovery Programme</a></li>
                        <li><a href="http://http://www.kakaporecovery.org.nz/">The Kakapo Recovery Programme</a></li>
                    </ol>
                </div>
            </td>
        </tr>
    </tbody>
</table>
<br>
<table width="100%" cellpadding="2px" cellspacing="2px">
    <tbody>
        <tr>
            <td width="20%" valign="top" style="background-color: rgb(51, 51, 204); color: white;"
                onclick="top.location.replace('donationsForm.php')">
                Make a donation!
            </td>
            <td width="20%" valign="top" style="background-color: #3333CC; color: white;"
                onclick="top.location.replace('aboutUs.html')">
                About us
            </td>
            <td width="20%" valign="top" style="background-color: #33C; color: #fff;"
                onclick="top.location.replace('projects.html')">
                Other projects
            </td>
            <td width="20%" valign="top" style="background-color: rgb(51, 51, 204); color: white;"
                onclick="top.location.replace('organisations.php')">
                Other organisations
            </td>
            <td width="20%" valign="top" style="background-color: rgb(51, 51, 204); color: white;"
```

```
                    onclick="top.location.replace('elsewhere.html')">
                    Birds in other countries
                </td>
            </tr>
        </tbody>
    </table>
</body>
</html>
```

The following PHP script **`donationsForm.php`** generates the donations form:

```
<!-- This page and the associated pages are copyright to Ulrich Speidel and -->
<!-- may be used for study purposes only. Copyright violations will be      -->
<!-- prosecuted - and have been in the past!                                -->
<html>
    <head>
        <title>Donations form for endangered birds</title>
    </head>
    <body>
        <form name="donation" action="donation.php" method="post" enctype="multipart/form-data">
            <input type="hidden" name="timeOfIssue"
                value="<?php echo time();?>">
            <h1>Donations form</h1>
            <p>We are delighted that you are ready to donate! Please
            enter your name, your address, the amount you wish to donate,
            and your credit card data. Then click on "Send donation!"</p>
            <p><b>Name:</b> <input type="text" name="donorName" size="80"></p>
            <p><b>Address:</b><br> <textarea name="address" rows="4" cols="40" align="top"></textarea></p>
            <p><b>Amount you wish to donate:</b>
            <select name="amount">
                <?php
                    for($i = 5; $i < 101; $i = $i + 5) {
                        echo "<option value=\"".$i."\">".$i."&euro;\n";
                    }
                ?>
            </select>
             
            <b>Your photo</b> (optional): <input name="donorPhoto" type="file"></p>
            <p><b>Credit card type:</b>
                <input type="radio" name="cardType" value="Visa">Visa
                 
                <input type="radio" name="cardType" value="Mastercard">Mastercard
            </p>
            <p><b>Credit card number:</b> <input type="text" name="creditCard" size="20" maxlength="20">
             
            <b>Credit card expiry date:</b> <input type="text" name="expiry" size="4" maxlength="4"></p>
            <p>
            <b>Click here if we're permitted to publish your name:</b>
            <input type="checkbox" name="public" checked></p>
            <p><input type="submit" value="Send donation!"></p>
        </form>
    </body>
</html>
```

The following script, **`donation.php`**, takes the data from **`donationsForm.php`**, checks it and enters it into a MySQL database:

```
<!-- This page and the associated pages are copyright to Ulrich Speidel and -->
<!-- may be used for study purposes only. Copyright violations will be      -->
<!-- prosecuted - and have been in the past!                                -->
<?php

    // This script receives the data from the donations form.
    // The data is checked and shown to the user as confirmation.

    $hack = false;        // this Boolean variable indicates
                          // whether a user entered incorrect data
                          // - this could even be a hacking attempt

    // Read data from $_POST into normal variables (for convenience).
    // This happens automatically when the directive register_globals
    // in php.ini is set to on. BUT: CAUTION - this directive lets a
    // user set ANY string or number variable at the beginning of the
    // script, so you MUST initialize all variables before first use.
    // If you do not, the user can control the variable's content.

    $donorName = $_POST["donorName"];
    $address = $_POST["address"];
    $amount = $_POST["amount"];
    $cardType = $_POST["cardType"];
    $creditCard = $_POST["creditCard"];
    $expiry = $_POST["expiry"];
    $public = $_POST["public"];
    $timeOfIssue = $_POST["timeOfIssue"];

    // Data verification - ensure that the data that came from the
    // POST request is at least roughly what we expect.

    // $donorName can be an arbitrary string, but it shouldn't be empty and
    // should inculde at least one letter!
    if (!preg_match("/[A-Za-z]/",$donorName)) {
        $hack = true; $field = "Name";
    }
    // The same applies to $address
    if (!preg_match("/[A-Za-z]/",$address)) {
        $hack = true; $field = "Address";
    }
    // The amount must be an integer between 5 and 100
    if (!preg_match("/^\d*[05]$/", $amount)) { // not a multiple of 5?
```

```
    $hack = true; $field = "Amount";
}
if (($amount < 5) || ($amount > 100)) {
    $hack = true; $field = "Amount";
}
// NB: quicker way to do this is to use /^5|[1-9][05]|100$/

// The card type should be Visa or Mastercard
switch ($cardType) {
    case "Visa": break;
    case "Mastercard": break;
    default:
        $hack = true; $field = "Card type";
}
// The card number should consist of 4 groups of 4 digits each
// which are separated by nothing, a space, or a hyphen:
if (!preg_match("/^(\d{4}[\s\-]?){4}$/", $creditCard)) {
    $hack = true; $field = "Card number";
}
// See also in the book: /^(\d{4}[\s\-]?){3}\d{4}$/
// Note that if you want to accept other card types as well, the number
// of digits may differ (e.g., 12 in the case of American Express).
// A popular verification tool that deals with most real credit cards
// is CCVal. For testing with "fake" numbers, a simpler scheme like the
// one above is better.

// The expiry date should consist of four digits, and the first two
// should be between 01 and 12, the third should (still) be a
// 0, and the fourth should now be between 5 and 9 (unless the card
// expires after 2009, which is unlikely at present).
if (!preg_match("/^(\d{2})0[3-9]$/", $expiry, $match)) {
    $hack = true; $field = "Credit card expiry date";
}
else
{
    if (($match[1] < 1) || ($match[1] > 12)) {
        $hack = true; $field = "Credit card expiry date";
    }
}
// The checkbox is either empty, "on" or someone's trying to hack us:
if (($public != "") && ($public != "on")) {
    $hack = true; $field = "Public donation";
}
// The form's time of issue must be an integer
if (!preg_match("/^\d+$/", $timeOfIssue)) {
```

```
    $hack = true; $field = "Time of issue";
}
// Has a photo been sent?
if ($_FILES["donorPhoto"]["size"] > 0) {
    // yes!
    $photo = true;
    // get extension and store it in $type
    preg_match("/(\.\w+)$/",
        $_FILES["donorPhoto"]["name"],$match);
    $type = $match[1];
    // we only permit image file extensions, so people
    // can't smuggle in PHP scripts instead
    if (in_array(
            strtolower($type),
            array(".gif",".bmp",".jpg",".png",".jpeg"))) {
        // extension is OK, but NEVER let the user determine
        // the name under which you will save the file. Create
        // a unique file name using the uniqid() function:
        $fileName = uniqid("").$type;
        // get image path relative to script location
        $imagePath =
            preg_replace("/\/[^\/]+$/","",
                        $_SERVER["SCRIPT_FILENAME"])
                        ."/images/";
        copy($_FILES["donorPhoto"]["tmp_name"],
            $imagePath.$fileName);
    }
}
else
{
    $photo = false;
}

// In the case of incorrect input, return an error message:
if ($hack) { ?>

    <html>
         <head>
              <title>Input error</title>
         </head>
         <body>
              <h1>Input error</h1>
              Your input in the field <b><?php echo $field; ?></b>
              was incorrect.
         </body>
```

```
    </html>

<?php
    exit(); // Terminate script!
}

// We are now ready to store the data in the database.
// NB: We'll skip this section until we've talked about PHP and databases
// First, we connect to the DB server
$connection = @mysql_connect("localhost","user334","myfunnypasswd")
    || die("Database connection failed!");
// The "@" in front of the mysql_connect() function suppresses any error messages
// from the function that may reveal application details to our users (such as names of
// database fields that a hacker could exploit elsewhere in an attack.
//
// If the connection is successful, $connection holds a database connection reference
// (called a "resource"). This evaluates as true and so the second part of the
// OR statement, the die() function, is never executed.

// Now we need to select the database that we wish to use. If you want to try this on the
// university server, change "donations" to whatever your UPI is.
@mysql_select_db("donations") || die("Database not available!");

// If we get to this point, we are ready to insert the data. Build the query as a
// PHP string:
$query = "insert into donations "
       . "(donationId, donorName, donorAddress, donorPhoto, donationAmount, cardType, cardNumber, cardExpiry, mayPublish, completionTime) "
       . "values ("
       . "'" . uniqid("") . "',"
       . "'$donorName',"
       . "'$address',"
       . "'$fileName',"
       . "'$amount',"
       . "'$cardType',"
       . "'$creditCard',"
       . "'$expiry',"
       . "'$public',"
       . "'" . (time() - $timeOfIssue). "')";

// Note that all of the variables above have come from escaped and verified $_POST data and
// are put inside quotes, so any SQL code that an attacker may have submitted is snugly put
// inside a string where it looks ugly but is otherwise safe!
//
// The next step is to carry out the actual insert query:
@mysql_query($query) || die("Unable to insert into database!");
```

```
// Now output the data to the user as a confirmation.
// Where applicable, use
// - stripslashes to unescape data that has been backslash-escaped by PHP
// - htmlentities() to prevent the user from injecting HTML tags into the text
//   (HTML tags such as <script> tags become invisible when loaded into a
//   browser. They can be used, e.g., my a maliciuos to introduce JavaScript
//   code to spy on another user who displays the text in his browser.
//   htmlentities() converts "<" and ">" into &lt; and &gt;, which are
//   harmless
// - nl2br() to insert <br/> tags before newlines for correct HTML display
//
// Note that we do not need to use any of these functions on the card type,
// card number, or expiry date. We have already verfied above that they do not
// contain backslashes or HTML tags.
//
?>
<html>
    <body>
        <h1>Dear <?php echo htmlentities(stripslashes($donorName)); ?></h1>
        <p>Thank you very much for your donation of
        <?php echo $amount; ?>&euro;.
        A receipt will be sent to your address</p>
        <p><b><?php
            echo nl2br(htmlentities(stripslashes($address)));
        ?></b></p>
        <p>The donation will be debited to your
        <b><?php echo $cardType; ?></b> credit card:</p>
        <p><b><?php echo $creditCard; ?></b>
        expiring on <b><?php echo $expiry; ?></b>.</p>
        <?php
        if ($photo) { ?>
            <p>Your photo is shown below:<br>
            <img src="<?php echo "images/".$fileName; ?>"
                 alt="Image of <?php echo htmlentities(stripslashes($donorName)); ?>"/></p>
        <?php
        }
        ?>
        <p>We note that we are
        <?php
            if ($public == "") {
                echo "not";
            }
        ?>
        permitted to publish your name.</p>
        <p>You took <?php echo (time() - $timeOfIssue); ?>
```

```
        seconds to complete our form</p>.
    </body>
</html>
```

The following script **`donationsList.php`** is an example of doing a select query in MySQL from PHP:

```
<?php
    // Output HTTP headers to prevent caching - this
    // ensures (hopefully) that we will alwys get the
    // latest version
    header("Expires: Sun, 11 Aug 2002 05:00:00 GMT");
    header("Last-Modified: " . gmdate("D, d M Y H:i:s") . " GMT");
    header("Cache-Control: no-store, no-cache, must-revalidate");
    header("Cache-Control: post-check=0, pre-check=0", false);
    header("Pragma: no-cache");
?>
<html>
    <body style="font-family: Arial, sans-serif; font-size: 10pt;">
<?php
    // This script must only be accessible to the treasurer,
    // so let's ask for a login and password

    if (count($_POST) == 0) {
        // output login form
?>
        <h3>Hello treasurer! Please log in:</h3>
        <form action="donationsList.php" method="post">
            Login: <input type="text" name="login"><br/>
            Password: <input type="password" name="pass"><br/>
            <input type="submit" value="Log me in!">
        </form>
<?php
    }
    else
    {
        // check login and password
        if (($_POST["login"] != "joe") ||
            ($_POST["pass"] != "moneybag")) {
            // output error message and finish HTML
?>
        <h3>Login unknown or password incorrect!</h3>
    </body>
</html>
<?php
            exit(); // exit script here
        }
        // Login OK, so let's output the list
        $sumOfDonations = 0; // initialize a sum counter
```

```
        // Read the data from the database.
        // First, we connect to the DB server
        $connection = @mysql_connect("localhost","user334","myfunnypasswd")
            || die("Database connection failed!");
        // The "@" in front of the mysql_connect() function suppresses any error messages
        // from the function that may reveal application details to our users (such as names of
        // database fields that a hacker could exploit elsewhere in an attack.
        //
        // If the connection is successful, $connection holds a database connection reference
        // (called a "resource"). This evaluates as true and so the second part of the
        // OR statement, the die() function, is never executed.

        // Now we need to select the database that we wish to use. If you want to try this on the
        // university server, change "donations" to whatever your UPI is.
        @mysql_select_db("donations") || die("Database not available!");

        // If we get to this point, we are ready to read the data.
        // Build the query as a PHP string:
        $query = "select * from donations";
        // execute the query
        $result = mysql_query($query);

        // Output list header in HTML
        // and arrange list in a table
?>
        <h1>Our generous donors</h1>
        <table border="1" cellpadding="5"
            style="font-family: Arial, sans-serif; font-size: 10pt;">
            <tr>
                <th>
                    Amount
                </th>
                <th>
                    Details
                </th>
                <th>
                    Donor photo
                </th>
            </tr>
<?php
        // Read each record from the query result using
        // mysql_fetch_assoc(), which returns false if
        // there are no records left
        while ($donationRecord = mysql_fetch_assoc($result)) {
```

```
            // While there are records left, mysql_fetch_assoc()
            // returns one record from the query result as an
            // associative array each time the function is called.
            // Read out data into convenience variables, using
            // the name of the respective database field as a key
            // into the associative array. This is done with extract():
            extract($donationRecord);
            // NB: You should never do this with data from $_POST, $_GET
            // or $_COOKIE, because it lets your users overwrite any
            // existing variables.

            // Add this donation to the total
            $sumOfDonations += $donationAmount;
?>
            <tr valign="top">
                <td>
<?php
            // output amount of donation
            echo $donationAmount."&euro ";
            if (!$mayPublish) {
                echo "anonymous";
            }
?>
                </td>
                <td>
<?php
            // output donor details
            echo "<p><b>".htmlentities($donorName)."</b></p>";
            echo "<p>".nl2br(htmlentities($donorAddress))."</p>";
?>
                    <table bgcolor="yellow"
                        style="font-family: Arial, sans-serif; font-size: 10pt;">
                        <tr>
                            <td>
<?php
            // output credit card details
            echo $cardType;
?>
                            </td>
                            <td>
<?php
            echo $cardNumber;
?>
                            </td>
                        </tr>
```

```
                        <tr>
                            <td>
<?php
            echo $cardExpiry;
?>
                            </td>
                            <td>
                        </tr>
                    </table>
                </td>
                <td>
<?php
            // output donor file
            if ($donorPhoto) {
                echo "<img width=\"100\"
                    src=\"images/".$donorPhoto."\"
                    alt=\"Image of ".htmlentities(stripslashes($donorName))."\">";
            }
            else
            {
                echo "no photo";
            }
?>
                </td>
            </tr>
<?php
        } // end of while loop
?>
            <tr>
                <td>
                    <b>Total of all donations until today:</b>
                </td>
                <th>
                    <b><?php echo $sumOfDonations; ?> &euro;</b>
                </th>
            </tr>
        </table>
<?php
    }
?>
    </body>
</html>
```

The following script **`thermometer.php`** is an example of graphics generation with PHP:

```
<?php

    // thermometer.php
    // (c) 5-5-2006 Ulrich Speidel
    // This graphics script outputs a PNG image with a thermometer
    // that reflects the sum of all donations to date
    // NOTE: This script requires the GD extension of PHP to be
    // enabled!

    $total = 1200; // dummy donations level for simulation purposes
    /* Uncomment the following block if the data is coming from the database
    // Read the data from the database.
    // First, we connect to the DB server
    $connection = @mysql_connect("localhost","user334","myfunnypasswd")
        || die("Database connection failed!");
    // The "@" in front of the mysql_connect() function suppresses any error messages
    // from the function that may reveal application details to our users (such as names of
    // database fields that a hacker could exploit elsewhere in an attack.
    //
    // If the connection is successful, $connection holds a database connection reference
    // (called a "resource"). This evaluates as true and so the second part of the
    // OR statement, the die() function, is never executed.

    // Now we need to select the database that we wish to use. If you want to try this on the
    // university server, change "donations" to whatever your UPI is.
    @mysql_select_db("donations") || die("Database not available!");
    // Query sum of all donations
    $query = "select sum(donationAmount) from donations";
    $queryResult = mysql_query($query);
    $record = mysql_fetch_assoc($queryResult);
    $total = $record["sum(donationAmount)"];
    //*/

    // Headers to set content type and prevent caching
    header("Content-type: image/png");
    header("Expires: Wednesday, 18-April-02 00:00:00 GMT");
    header("Pragma: no-cache");

    // image parameters

    $target = 2000; // target in Euro
    $step = 200; // scale step in Euro
```

```
$image_width = 100;
$image_height = 400;

$height = 340; // height of thermometer in pixels
$width = 30; // width of thermometer in pixels
$x = 60; // horizontal position of thermometer
         // (centre of "tank")
$y = 370; // vertical position of thermometer
          // (centre of "tank")
$mark = 20; // mark length in pixels

// scale length in pixels (scale end is 1.5
// thermometer widths below the upper
// thermometer end)
$scale = $height - 1.5 * $width;
// vertical position of the mark that
// indicates the target (scale end)
$max = $y - $scale;
// pixels per Euro:
$ppe = $scale / $target;
// pixels per step:
$pps = $step * $ppe;
// vertical position at which the straight part of
// the pipe turns into an arc
$pipeend = $y - $height + 0.5 * $width;

// generate image
$image = @ImageCreate($image_width, $image_height)
    or die("Cannot initialize GD Image Stream");
$background_colour = ImageColorAllocate ($image, 255, 255, 255);
$text_colour = ImageColorAllocate ($image, 0, 0, 102);
$image_colour = ImageColorAllocate ($image, 0, 0, 204);
$fill_colour = ImageColorAllocate ($image, 204, 0, 51);

// Draw thermometer "tank":
ImageArc($image, $x, $y,
         1.5 * $width + 1,
         1.5 * $width + 1,
         0, 360,
         $image_colour);
// fill tank
ImageFill($image, $x, $y, $fill_colour);
// ... and open it to top
ImageArc($image, $x, $y,
```

```
         1.5 * $width + 1,
         1.5 * $width + 1,
         230, 310,
         $fill_colour);

// draw pipe
ImageLine($image,
          $x - $width/2, $y - $width/2,
          $x - $width/2, $pipeend,
          $image_colour);
ImageLine($image,
          $x + $width/2, $y - $width/2,
          $x + $width/2, $pipeend,
          $image_colour);
ImageArc($image, $x, $pipeend,
         $width, $width,
         180, 0, $image_colour);

// draw scale

$sy = $y;
$euro = 0;

while ($sy > $max) {
    $sy = $sy - $pps;
           $euro = $euro + $step;
    ImageLine($image,
          $x - $width/2, $sy,
          $x - $width/2 - $mark, $sy,
          $image_colour);
    ImageString($image,
                3, $x - $width - $mark, $sy,
                $euro, $text_colour);

}

// calculate thermometer level
$level = $y - $ppe * $total;

// draw thermometer level
// (fill colour)
ImageLine($image,
      $x - $width/2, $level,
      $x + $width/2, $level,
      $fill_colour);
```

```
    // fill pipe to level
    ImageFill($image, $x, $level + 1, $fill_colour);

    // output image
    ImagePng($image);

?>
```

The following SQL script is used to set up the database:

```
# COMPSCI334 example database setup script

# To use this on your MySQL database at uni, rename the database from "donations" into your UPI
# in the following line, save the file and run the command
#
# >mysql -u <your_upi> -p <your_upi> < donationsdb.dql
#
# from the Unix prompt on the DB server machine.

use donations;

# Example for a personalized database:
#
#    use ugue001
#

#
# Table definitions - this database uses only a single table: donations, which stores details of donations
#

#
# First, drop any existing table of that name (do NOT change the name of the table to your UPI!)
#

DROP TABLE IF EXISTS donations;

#
# Then, create a new table (do NOT change the name of the table to your UPI!)
#

CREATE TABLE donations (
  donationId varchar(23) NOT NULL,
  donorName varchar(255),
  donorAddress longtext,
  donorPhoto varchar(27),
  donationAmount int(3),
  cardType varchar(30),
  cardNumber varchar(20),
  cardExpiry varchar(30),
  mayPublish varchar(3),
  completionTime varchar(20),
  PRIMARY KEY (donationId),
```

```
  KEY (donorName),
  KEY (cardNumber),
  KEY (mayPublish)
);

#
# Now put some dummy data into the database
#

INSERT INTO donations (donationId, donorName, donorAddress, donorPhoto,
  donationAmount, cardType, cardNumber, cardExpiry, mayPublish, completionTime)
  values ("1","Fred Fish","Big Ponds","a46b748f11a53.jpg","55","Visa","1111222233334444","34125094","yes",30);

#
# Done!
#
```