Using *htmlArea* & a Database to Maintain Content on a Website

by

Peter Lavin

December 30, 2003

Overview

If you wish to develop a website that others can contribute to one option is to have text files sent as e-mail attachments, convert these to HTML and then upload the file. There is a better way. If the people making submissions to your site are capable of using a word processor then things can be done much more efficiently. This article will show you how to use a free component called *htmlArea* and a database to handle the addition and display of content on your website. Some knowledge of HTML, databases and server-side scripts is assumed.

Introduction

htmlArea is a free WYSIWYG (what you see is what you get) editor created with Javascript. It can replace “textarea” fields in a web page and using it is akin to using a word processor. Text can be manipulated by using buttons on a toolbar and options in “select” controls. Foreground and background colours may be changed, images inserted, fonts adjusted – all manner of text manipulation that you might expect from a word processor. When any changes are made to text the appropriate HTML code is inserted. Press the “enter” key and a paragraph tag will automatically be inserted, the space bar and a non-breaking space is inserted. A toggle button is provided that allows you to view the HTML source. You may also expand the control and open it in a separate window.

This item is available for free and may be downloaded from http://www.interactivetools.com/products/htmlarea/ - no strings attached and no trial period. You can even customise it if you want to. Have a look at a modified version of this control at http://www.webstationone.com/test/htmlarea/index2.html.

However, before going any further you should be aware that this component works only in Internet Explorer Version 5 or higher. This does not mean that your HTML code will not function if viewed in Netscape. It simply means that textareas will appear as textareas and not be replaced by an *htmlArea*. A beta version with cross-browser functionality is in the works but still lacks documentation.

htmlArea is a very attractive and powerful component but it’s only useful if its contents can be saved. This article will describe how to install this control and how to save and retrieve the HTML files that are created. Our solution will be implemented using PHP and MySQL though the principles will apply to any relational database and any server-side scripting language.

Download and Install

Excellent installation instructions are provided by interactivetools at their site so follow them and install *htmlArea*. However, there is one change you will probably want to make. If you follow the instructions given in the link above, you will need to place any page that uses *htmlArea* in the “htmlarea” directory. It is much more likely that you will want to access this control from a page in your server’s root directory. To do so change the line of code that reads, `_editor_url = "";` to `_editor_url = "htmlarea/";`.

You can embed an *htmlArea* control that will work from your root directory using the following code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<title>htmlArea</title>
<meta name="description" content="" />
<meta name="keywords" content="" />
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"
/>
<meta name="author" content="your name" />
<script language="Javascript1.2"><!-- // load htmlarea
_editor_url = "htmlarea/"; // URL to htmlarea files
var win_ie_ver = parseFloat(navigator.appVersion.split("MSIE")[1]);
if (navigator.userAgent.indexOf('Mac')        >= 0) { win_ie_ver = 0; }
if (navigator.userAgent.indexOf('Windows CE') >= 0) { win_ie_ver = 0; }
if (navigator.userAgent.indexOf('Opera')      >= 0) { win_ie_ver = 0; }
if (win_ie_ver >= 5.5) {
 document.write('<scr' + 'ipt src="' +_editor_url+ 'editor.js"');
 document.write(' language="Javascript1.2"></scr' + 'ipt>');
} else { document.write('<scr'+'ipt>function editor_generate() { return
false; }</scr'+'ipt>'); }
// -->
</script>
</head>
<body>
<script language="JavaScript1.2" defer>
      editor_generate('contents');
</script>
<p align="center">
<textarea name="contents" cols="40" rows="12" readonly>
</textarea>
</p>
</body>
</html>
```

Save this code, upload it to your server's root directory and try it out. If you've done everything correctly you should see an *htmlArea* in place of the textarea named "contents". Try out the various features and see what it is capable of.

Database Table

Let's define the database table that will store the contents of an *htmlArea*. We'll name the table "items" and give it the following structure:

Field Name	Data Type	Properties
id	int(11)	PRIMARY KEY

title	varchar(35)	Not Null
topic	varchar(35)	Not Null
author	varchar(15)	Not Null
contents	text	Not Null
whenadded	timestamp	
reviewed	tinyint(4)	

A few more fields than you imagined perhaps but they all do serve a purpose. The “id” field will be a unique identifier and should probably be “auto_increment” type. “title” and “author” are self explanatory and “contents” will hold what’s typed into the *htmlArea*. The ‘whenadded’ field is a ‘timestamp’ data type so it will record the date and time that a record is added. Depending upon your needs, the ‘reviewed’ field may or may not be necessary but it will allow you to vet content before it is posted to your site. For instance, a where clause in your SQL such as, “WHERE reviewed =1”, would keep items not yet reviewed from showing up on your site.

Saving Content

I don’t intend to discuss here how a form is submitted from a web page. If you’ve read this far you probably know how to do this anyway. If not there are plenty of online tutorials available on this subject. Let’s assume that our *htmlArea* control is part of a “form” and the “action” attribute has been set to a file in the current directory called “processform.php”. For the sake of clarity, the topics used in the select control below have been hardcoded. In real-life these would most likely be created dynamically (or perhaps in this case magically) from a database. The HTML code for the form in your web page might look like the following:

```
<form name="frmitem" method="post" action=" processform.php">
<table border="0" >
<td>Title: </td><td><input type="text" name="title" maxlength=35></td>
</tr>
<td>Topic:</td>
<td><select name="topic">
      <option>Ghosts</option>
      <option>Goblins</option>
      <option>Harry Potter</option>
      <option>Haunted Places</option>
      <option>Hermione</option>
      <option>Ron</option>
</select></td>
</tr>
<tr>
<td colspan=2 valign ="top">Contents: </td></tr><tr>
<td colspan=2 ><textarea name="contents"  rows="22"
cols="40"></textarea></td>
</tr>
<tr>
```

```
<td align = "center" colspan="2">
<input type="submit" value="Send" >   <input type="reset"
value="Clear"></td>
</tr>
</table>
</form>
```

The “textarea” named “contents” will become an *htmlArea* control but will not appear differently in the source code. When this form is submitted it does not need to be handled in any special way. Any formatting done by the user will be captured as HTML code in your database.

Just as your form does not require any special coding nor does your server-side script.

```
<?php
/* include files holding connection information and additional
functions */
include 'connection.php';
include 'dbfunctions.inc';
/*****************************************************************/
$user = "guest";
//retrieve information posted from form
$title=@$HTTP_POST_VARS["title"];
$description=@$HTTP_POST_VARS["description"];
$topic=@$HTTP_POST_VARS["topic"];
$contents=@$HTTP_POST_VARS["contents"];
//programmer created function from the dbfunctions.inc include file
opendatabase();
$strsql = "INSERT INTO items ".
      "VALUES(null,'$title', $topic', ".
      "'$user','$contents',null, 0)";
$connection = @ mysql_connect($hostname, $username,$password)
      or die("Cannot connect to database");
if (! mysql_selectdb($databasename, $connection))
      showerror();
if (!($result = @ mysql_query($strsql, $connection)))
      header("Location: mainpage.php?status=F");
else
      header("Location: mainpage.php?status=T");
?>
```

Again, for the sake of simplicity, we have used a literal for the value of the variable “$user” and have not verified any of the data submitted. Also, functions to open the database are not shown but are assumed to be in the “dbfunctions.inc” file.

How to Display the Saved Contents

The contents will have been saved with the appropriate HTML code and when retrieved will need to be integrated into an HTML page. Assuming a hyperlink such as the following;

```
<a href="gettext.php?itemnumber=7">About Hermione</a>,
```

the contents might be retrieved and displayed using the following “gettext.php” file :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<title>Article</title>
<meta name="description" content="" />
<meta name="keywords" content="" />
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"
/>
<meta name="author" content="" />
<link rel="stylesheet" type="text/css" href="style/template.css"
title="template" />
<!--local style elements here-->
<style type="text/css">
</style>
</head>
<body>
<!—insert the item here-->
<?php
//database information and database functions
include 'connection.php';
include 'dbfunctions.inc';
/*************************************************************/
$itemnumber=@$HTTP_GET_VARS["itemnumber"];
$strsql = "Select title, description, topic, username, contents, ".
      "Date_Format(whenadded,'%M %e,%Y') AS formatted ".
      "From item WHERE id = $itemnumber";
$connection = @ mysql_connect($hostname, $username,$password)
      or die("Cannot connect to database");
if (! mysql_selectdb($databasename, $connection))
      showerror();
if (!($result = @ mysql_query($strsql, $connection)))
      showerror();
$row=mysql_fetch_array($result);
echo "<h3>$row[title]</h3>";
echo "$row[topic]<br />";
echo "$row[contents]";
?>
</body>
</html>
```

You'll probably want to lay out your page in a more attractive fashion but the above code shows you how any formatting added by the user has been saved to the database and retrieved exactly as the user entered.

Conclusion

We have seen how the combination of *htmlArea* and a database can greatly simplify adding content to a website. This control converts a textarea into a "word processor" giving the user the freedom to format text as (s)he sees fit and it frees the webmaster from having to mark up text. You can cut out the middle man and have content added directly to your site.

About the Author

Peter Lavin runs a Web Design/Development firm in Toronto, Canada. For more information visit http://www.softcoded.com. He may be reached at peterlavin@sympatico.ca.