

# *Notes from Well House Consultants*

*These notes are written by Well House Consultants and distributed under their Open Training Notes License. If a copy of this license is not supplied at the end of these notes, please visit*

*<http://www.wellho.net/net/whcotnl.html>  
for details.*

## 1.1 Well House Consultants

Well House Consultants provides niche training, primarily but not exclusively in Open Source programming languages. We offer public courses at our training centre and private courses at your offices. We also make some of our training notes available under our "Open Training Notes" license, such as we're doing in this document here.

## 1.2 Open Training Notes License

With an "Open Training Notes License", for which we make no charge, you're allowed to print, use and distribute these notes provided that you retain the complete and unaltered license agreement with them, including our copyright statement. This means that you can learn from the notes, and have others learn from them too.

You are NOT allowed to charge (directly or indirectly) for the copying or distribution of these notes, nor are you allowed to charge for presentations making any use of them.

## 1.3 Courses presented by the author

If you would like us to attend a course (Java, Perl, Python, PHP, Tcl/Tk, MySQL or Linux) presented by the author of these notes, please see our public course schedule at

<http://www.wellho.net/course/index.html>

If you have a group of 4 or more trainees who require the same course at the same time, it will cost you less to have us run a private course for you. Please visit our onsite training page at

<http://www.wellho.net/course/otc.html>

which will give you details and costing information

## 1.4 Contact Details

Well House Consultants may be found online at

<http://www.wellho.net>

[graham@wellho.net](mailto:graham@wellho.net)

technical contact

[lisa@wellho.net](mailto:lisa@wellho.net)

administration contact

Our full postal address is

404 The Spa

Melksham

Wiltshire

UK SN12 6QL

Phone +44 (0) 1225 708225

Fax +44 (0) 1225 707126

# *HTML for Web Application Authors*

If you're providing server side applications on your web site, you'll need to have some understanding of HTML as that's how you'll be addressing your user's browser. As well as the standard HTML tags, you'll need a good understanding of forms, and perhaps you'll get involved in other extensions to the basic HTML such as frames, style sheets, javascript and embedded objects – all of which we introduce in this module.

<i>Tables</i> . . . . .	4
<i>Frames</i> . . . . .	5
<i>Forms</i> . . . . .	5
<i>Objects and images within a web page</i> . . . . .	12
<i>Style sheets</i> . . . . .	12
<i>JavaScript</i> . . . . .	13

## 2.1 Tables

Within a page of HTML, you can specify an ordered list (`<ol>`) or an unordered list (`<ul>`), ending with `</ol>` and `</ul>` respectively. Individual list items should be prefixed with `<li>`.

To give better control over formatting than a list, you may want to arrange your data into a table. Tables are enclosed in `<table>` and `</table>` tags; within tables, rows are enclosed in `<tr>` and `</tr>` tags, and within each row, data elements are enclosed in `<td>` and `</td>` tags. Many parameters can be specified to tables to control their looks. You'll find that tables are used for the layout of the majority of web sites, with one table very often nested within the cell of another. A tip if you're trying to work out why your tables don't display properly: set the attribute `border=1` on the open table tag.

Here's a table sample:

```
<html>
<head><title>Table Demo</title></head>
<body>
<h2 align=center>A Sample Table</h2>
<table border=1>
<tr><td>January</td>
<td>31</td>
<td>Winter</td>
</tr><tr><td>February</td>
<td>28 or 29</td>
<td>Winter</td>
</tr><tr><td>March</td>
<td>31</td>
<td>Winter</td>
</tr><tr><td>April</td>
<td>30</td>
<td>Spring</td>
</tr></table>
<hr>
&copy; Well House Consultants2004<br>
01225 708225
</body>
</html>
```

Month	Days	Season
January	31	Winter
February	28 or 29	Winter
March	31	Winter
April	30	Spring

---

© Well House Consultants, 2004  
01225 708225

Figure 1 How the table might appear on a browser

Although the raw HTML source of a table is often very hard to follow, it's surprisingly easy to generate within a program using nested loops. We didn't write the above code directly, we used the following piece of PHP:

```
<html>
<head><title>Table Demo</title></head>
<body>
<h2 align=center>A Sample Table</h2>
<table border=1>
<?php
$info = array(
    array("January",31,"Winter"),
    array("February","28 or 29","Winter"),
    array("March",31,"Winter"),
    array("April",30,"Spring"));
foreach ($info as $month) {
    print "<tr>";
    foreach ($month as $cell) {
        print "<td>$cell</td>\n";
    }
    print "</tr>";
}
?>
</table>
<hr>
&copy; Well House Consultants, <?php print(date("Y")); ?><br>
01225 708225
</body>
</html>
```

## 2.2 Frames

On many sites, the browser window is divided into a series of frames. In this case, you'll have multiple HTML documents. The window as a whole will be defined in one document (a **frameset**), with each of the resulting frames being defined in its own piece of HTML. Documents called up can then replace all the frames, or one frame, or even open a new browser window.

Frames are useful tools on web sites where you wish to provide a control pane that's unchanging, and a data pane in which results are presented, but they do add to the design and development complexity and it's often better to use tables and refresh the whole page as you navigate a site.

There are also issues with the titling of framed pages which make them hard to bookmark sensibly, and with the printing of pages. It's very easy to provide a page that won't print out if you use frames, and that's why so many web sites have a "printer friendly version" option.

## 2.3 Forms

Forms are a vital part of any web site that includes executable content as they're the mechanism that allows user data entry. With a form, you define a set of user-enterable boxes between a **<form>** and **</form>** tag. Within the **<form>** tag an **action** attribute gives the URL to be called when the user indicates that the form has been completed, and a **method** attribute may be specified too.

### *Element types within a form*

There's much more you might want to do, and forms can have a range of different elements. There are basically a number of classes of elements, but within each of

them a number of attributes can be applied to tailor how they look and respond.  
 Here's a sample piece of HTML that includes most of the form elements:

```
<HTML>
<head><title>Form Element Demo</title></head>
<BODY BGCOLOR=white text=black link=blue>
<Form method=post action="/cgi-bin/envlist.pl">
<H2>Different form elements</H2>
This calls a script that lists out all fields entered, and the whole
environment<P>
<HR><!-- ##### -->
<H4>Type = Text</H4>
default text <INPUT type=text name=a_text>
<BR>
Another text <INPUT type=text maxlength=5 size=8
value="abc" name=a_t2>
<BR>
<HR><!-- ##### -->
<H4>Type = radio</H4>
Select a radio button from:<BR>
<INPUT type=radio name=b_set value=Orange>
Orange Juice<BR>
<INPUT type=radio name=b_set value=Lemon CHECKED>
Lemonade<BR>
<INPUT type=radio name=b_set value=Lime>
Rose's Lime Cordial<BR>
```

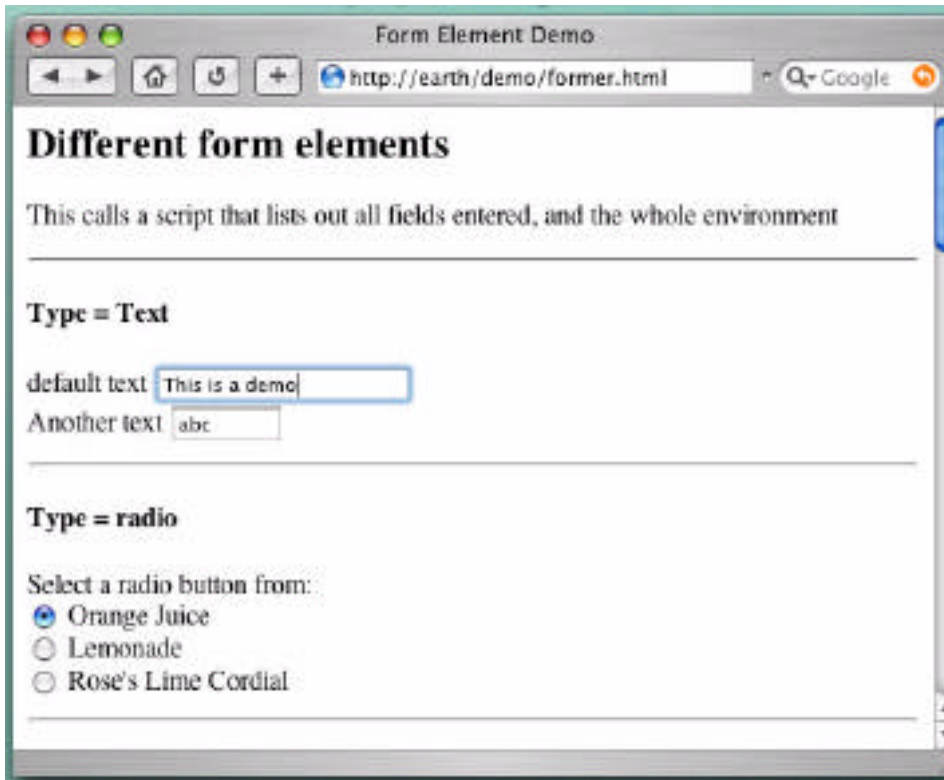


Figure 2 Display showing the form elements above

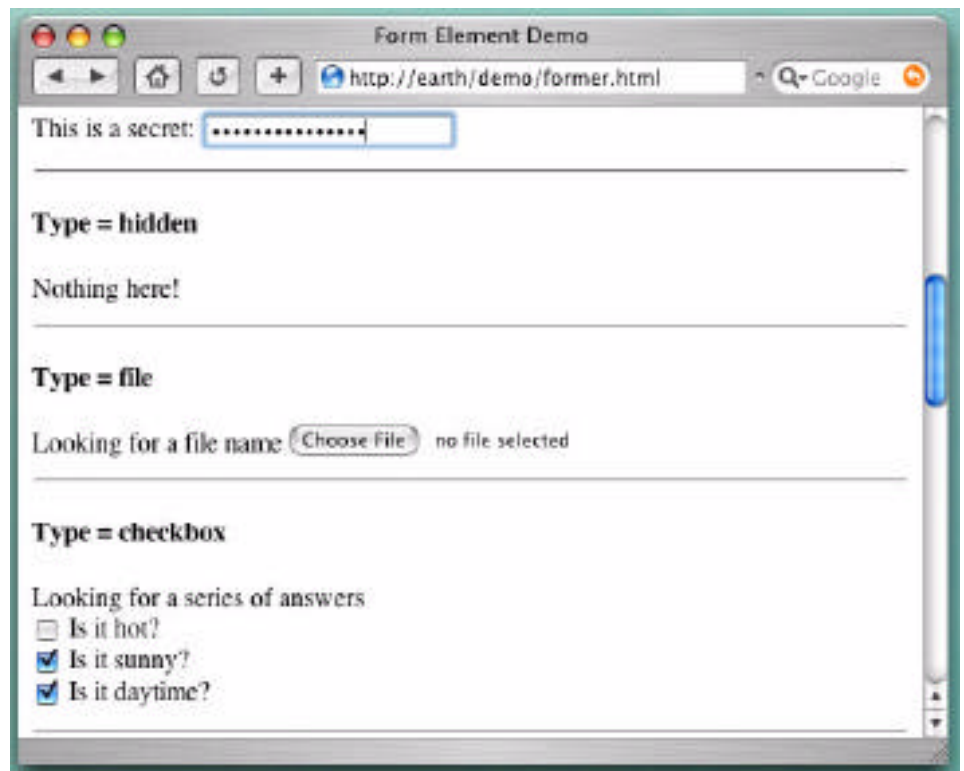
```
<HR><!-- ##### -->
<H4>Type = password</H4>
This is a secret:
<INPUT type=password name=c_pass>
<BR>
```

```

<HR>
<H4>Type = hidden</H4>
Nothing here!
<INPUT type=hidden name=ehem value="WHC_1040198606_450">
<HR>
<!-- ##### -->
<H4>Type = file</H4>
Looking for a file name
<INPUT type=file name=filer>
<HR>
<!-- ##### -->
<H4>Type = checkbox</H4>
Looking for a series of answers<BR>
<INPUT type=checkbox name=hot> Is it hot?<BR>
<INPUT type=checkbox name=sunny value="sun-on"> Is it sunny?<BR>
<INPUT type=checkbox name=daytime CHECKED>
Is it daytime?<BR>
<HR>

```

Figure 3 How the form tags above are displayed



```

<!-- ##### -->
<H4>Type = buttons</H4>
1st button <INPUT type=button name=j1 value=first><BR>
2nd button <INPUT type=button name=j2 value=second><BR>
<HR>
<!-- ##### -->
<H4>Textarea</H4>
A text area:
<TEXTAREA name=kd cols=25 rows=5>
Well House Consultants
404, The Spa
Melksham, Wiltshire
SN12 6QL
</TEXTAREA>

```

```

<HR>
<!-- ##### -->
<H4>Selects</H4>
Choose ONE of these:
<SELECT name=lssel>
<OPTION>North
<Option value=south>South
<Option SELECTED value=East>East
<Option value=W>West
</SELECT><P>
Choose several of these:
<SELECT name=lset size=3 multiple>
<OPTION selected>Swimming
<Option value=Skiing>Skiing
<Option SELECTED value="Scuba diving">Scuba
<Option value=Snorkelling>Snorkelling
<Option>Sleeping
</SELECT><P>
<HR>

```

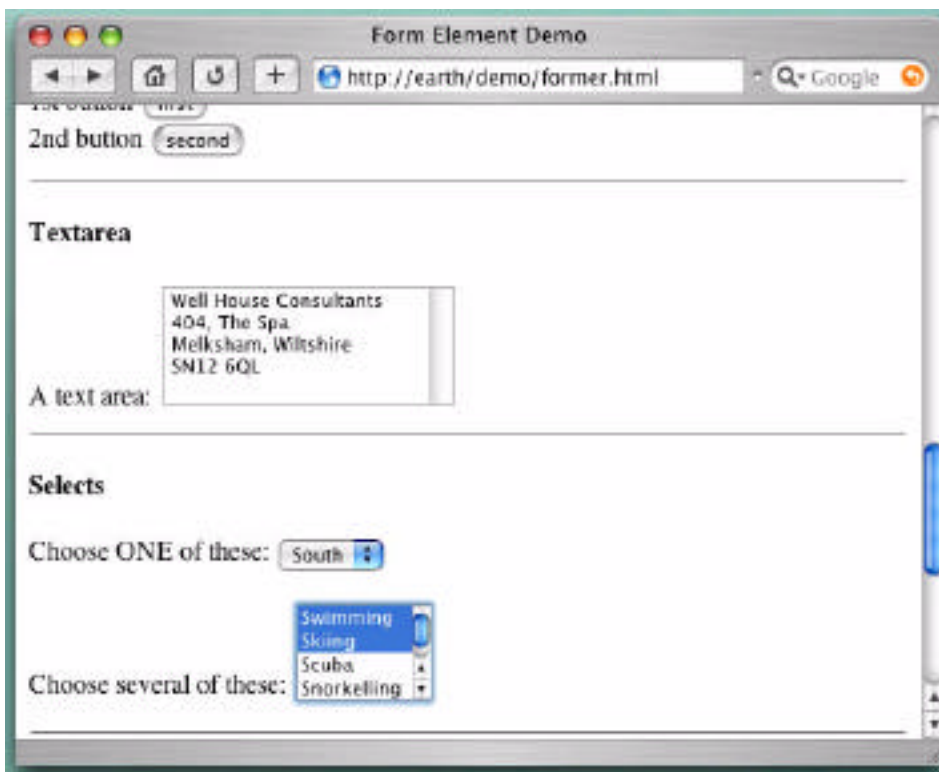


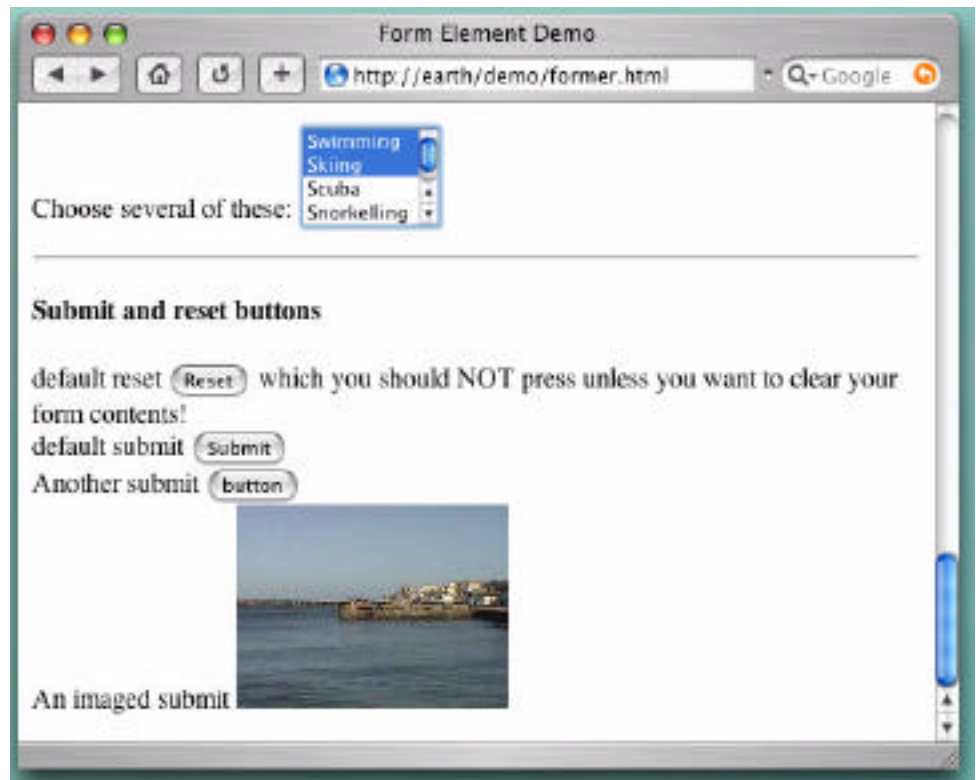
Figure 4 The textarea and select input types

```

<!-- ##### -->
<H4>Submit and reset buttons</H4>
default reset <INPUT type=reset> which you should NOT press unless you want
to clear your form contents!
<BR>
default submit <INPUT type=submit>
<BR>
Another submit <Input type=submit name=z_end value="button">
<BR>
An imaged submit <Input type=image src=swanage.jpg name=ze2>
</FORM>
</BODY>
</HTML>

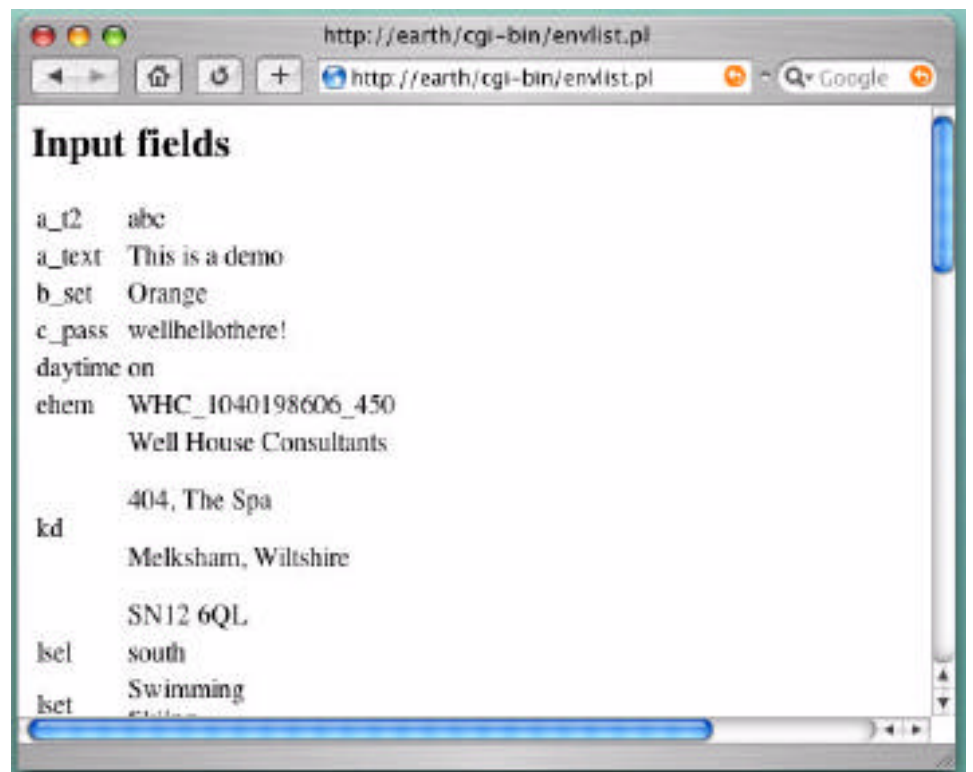
```

Figure 5 Various submit buttons, including an image



Because it can be used to display the contents of any form, we've pointed this set of form examples at the environment listing script from the previous section. Here's an example of a response from a completed form:

Figure 6 Some of the form fields



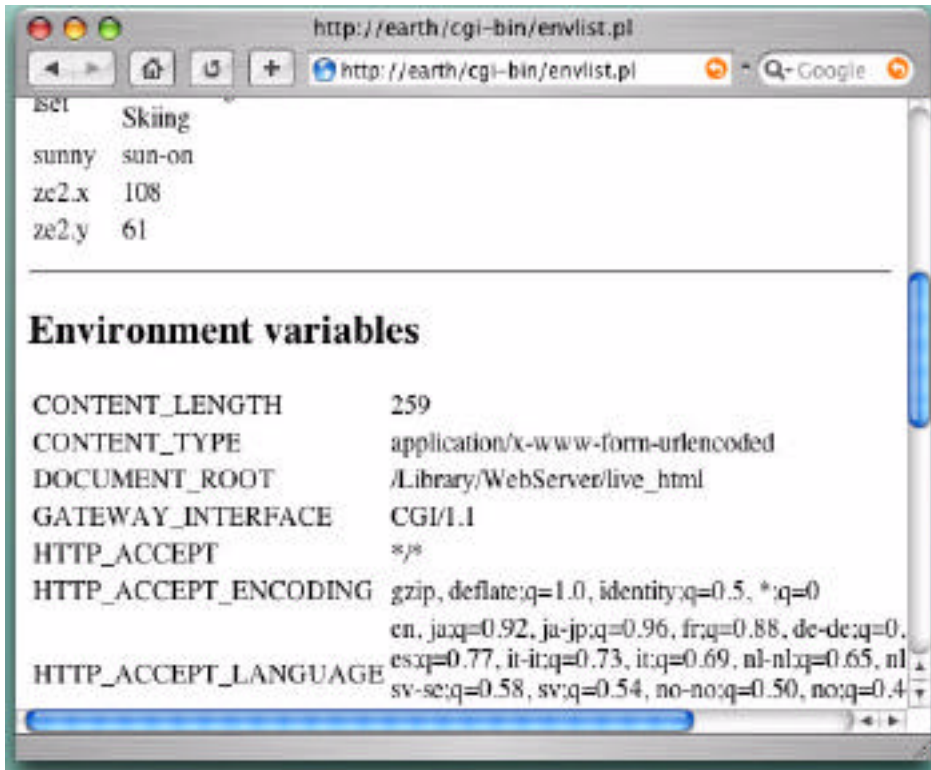


Figure 7 Further form fields, and some other information

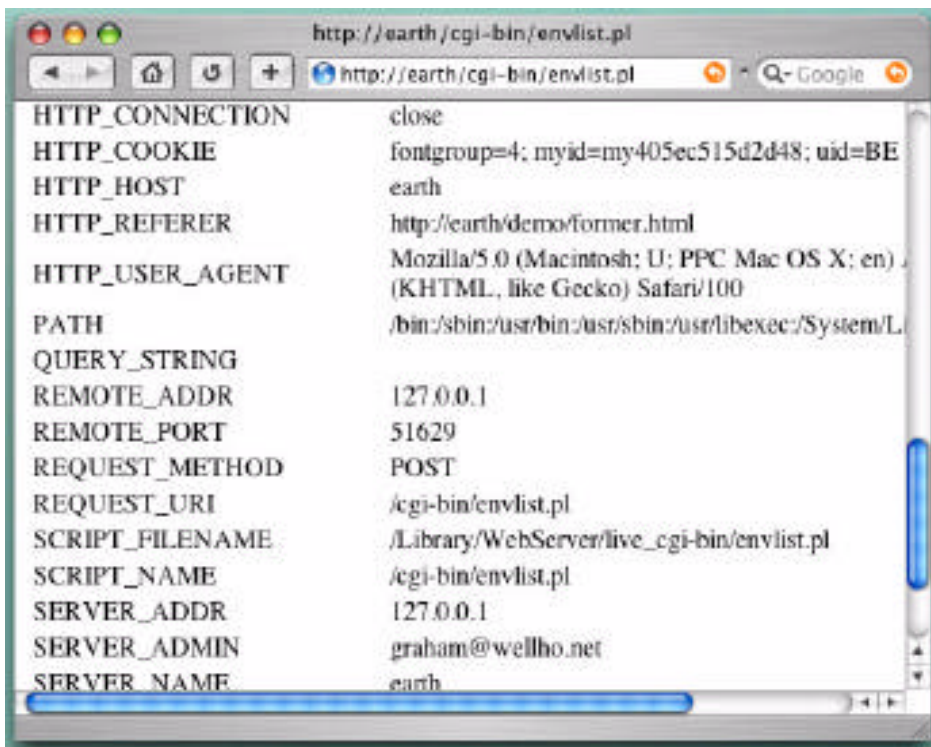


Figure 8 When you submit a form, your IP address, browser type, cookies and other information are passed back too!

*The script used*

So that you have a complete, workable example, here's the program that our form pointed to; it happens to be written in Perl.

```
#!/usr/bin/perl

# ANY form - list variables and environment

if ($ENV{"REQUEST_METHOD"} eq "POST") {
    read (STDIN,$buffer,$ENV{'CONTENT_LENGTH'});
} else {
    $buffer = $ENV{"QUERY_STRING"};
}

@userdata = split (/&/,$buffer);
foreach (@userdata) {
    ($field,$value) = /(.*)=(.*)/;
    $value=~ tr/+// ;
    $value=~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C",hex($1))/eg;
    $input{$field} .= "\n" if ($input{$field});
    $input{$field} .= $value;
}

# input fields

print "Content-type: text/html\n\n";
print "<BODY BGCOLOR=white TEXT=BLACK><H2>Input fields</H2><TABLE>";

foreach (sort keys %input) {
    print ("<TR><TD>",web($_), "</TD>");
    print ("<TD>",web($input{$_}), "</TD></TR>\n");
}

print "</TABLE>";

# environment variables

print "<HR><H2>Environment variables</H2><TABLE>";

foreach (sort keys %ENV) {
    print ("<TR><TD>",web($_), "</TD>");
    print ("<TD>",web($ENV{$_}), "</TD></TR>\n");
}

print "</TABLE>";

#####

sub web {
    $_ = $_[0];
    s/&/&amp;/g;
    s/</&lt;/g;
    s/>/&gt;/g;
    s/\n\r/\n/g; # dos to standard
    s/\r/\n/g; # mac to standard
    s/\n{2,}/<P>/g;
    s/\n/<BR>/g;
    $_;
}

```

## 2.4 Objects and images within a web page

Executable content and graphic images (in Java applets, COM objects and .gif and .jpg images) can also be included on your web page. They use the `<applet>` and `</applet>`, `<object>` and `</object>` or the `<img>` and `</img>` tag respectively. In each case, the tag requires you to give a parameter specifying which applet is to be run or which image is to be displayed, and if that applet or image isn't presently loaded by the browser, it will request it of the host.

If you're learning about Java Server programming, Servlets, JSP, Struts or similar technology, note that you will probably not get involved with `<applet>` (now deprecated) or `<object>` tags. With these technologies, your Java code will be run on your server computer and the output sent to the browser will be pure HTML. Good news, since it allows Java applications to be run from browsers that have Java disabled or do not have a Java plugin.

## 2.5 Style sheets

Although you can write specific font and colour directives within your HTML, you'll be encouraged in a new application to use style (also known as style sheets, cascaded style sheets or css). By using style sheets, you can define, in a single file, the look and feel for your whole web site, which you then apply to every page with a single directive.

The Well House Consultants web site uses style sheets, and here are some sample code lines from the current version:

- Within the head of the each page we specify:  
`<link href="/WHC.css" rel="stylesheet" type="text/css">`
- Text throughout the site is decorated with span tags such as:  
`<span class="body">`
- And the classes are defined in the .css file, for example:  

```
.body {  
    font-family: Optima, Verdana, Arial, Helvetica;  
    font-size: 10pt;  
    font-style: normal;  
    line-height: 14px;  
    font-weight: normal;  
    font-variant: normal;  
    color: #000000;  
}
```

Our web site actually goes somewhat further. By changing the style sheet file for individual users, we can let each visitor to our site choose their own look and feel from a set of four alternatives we have provided. Not only does this make it easier for people to adjust our site to suit their needs, but it also helps fulfil our obligations under the UK's disability discrimination act.

## 2.6 JavaScript

You can also embed code on JavaScript in a web page, and this gives an element of local interaction to the page if you wish. For example, you could program a select box to automatically submit if it's changed.

The code to define the selection element of a form (an example from our web site):

```
<select name=jump onChange="leap(this.form)">
```

The JavaScript is defined within the page head, carefully written in an HTML comment tag so that users of browsers who don't understand JavaScript don't see the sources!

```
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_jumpMenu(targ,selObj,restore){ //v3.0

eval(targ+".location='"+selObj.options[selObj.selectedIndex].value+"'");
  if (restore) selObj.selectedIndex=0;
}
function leap(where) {where.submit()}
//-->
</script>
```

 **Exercise**

# *License*

*These notes are distributed under the **Well House Consultants Open Training Notes License**. Basically, if you distribute it and use it for free, we'll let you have it for free. If you charge for its distribution of use, we'll charge.*

### 3.1 Open Training Notes License

Training notes distributed under the **Well House Consultants Open Training Notes License** (WHCOTNL) may be reproduced for any purpose PROVIDE THAT:

- This License statement is retained, unaltered (save for additions to the change log) and complete.
- No charge is made for the distribution, nor for the use or application thereof. This means that you can use them to run training sessions or as support material for those sessions, but you cannot then make a charge for those training sessions.
- Alterations to the content of the document are clearly marked as being such, and a log of amendments is added below this notice.
- These notes are provided "as is" with no warranty of fitness for purpose. Whilst every attempt has been made to ensure their accuracy, no liability can be accepted for any errors of the consequences thereof.

Copyright is retained by Well House Consultants Ltd, of 404, The Spa, Melksham, Wiltshire, UK, SN12 6QL - phone number +44 (1) 1225 708225. Email contact - Graham Ellis (graham@wellho.net).

Please send any amendments and corrections to these notes to the Copyright holder - under the spirit of the Open Distribution license, we will incorporate suitable changes into future releases for the use of the community.

If you are charged for this material, or for presentation of a course (Other than by Well House Consultants) using this material, please let us know. It is a violation of the license under which this notes are distributed for such a charge to be made, except by the Copyright Holder.

If you would like Well House Consultants to use this material to present a training course for your organisation, or if you wish to attend a public course is one is available, please contact us or see our web site - <http://www.wellho.net> - for further details.

Change log  
Original Version, Well House Consultants, 2004

Updated by: \_\_\_\_\_ on \_\_\_\_\_  
Updated by: \_\_\_\_\_ on \_\_\_\_\_  
Updated by: \_\_\_\_\_ on \_\_\_\_\_  
Updated by: \_\_\_\_\_ on \_\_\_\_\_  
Updated by: \_\_\_\_\_ on \_\_\_\_\_  
Updated by: \_\_\_\_\_ on \_\_\_\_\_

*License Ends.*