COMP 4021
Internet Computing

HTML Forms

David Rossiter

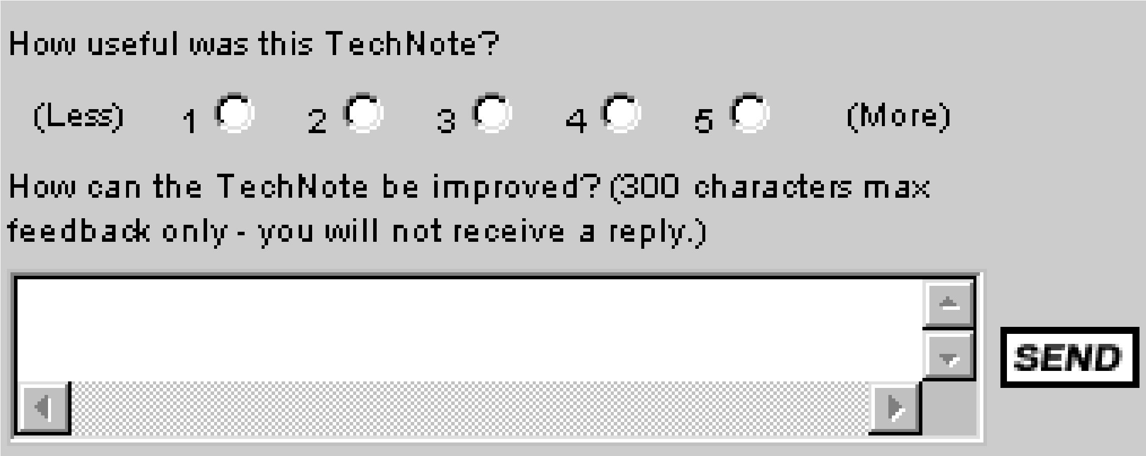
What is a Form?

- Forms are the easiest way to send data from a browser to a server program
- The user can enter various types of information, then send them to the server

- Browsers receive data from server most of the time
- Besides form, have you learnt any other situations under which data (of any kind) are sent to the server?

Common Form Usage

- Typical example:



How useful was this TechNote?

(Less) 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ (More)

How can the TechNote be improved? (300 characters max feedback only - you will not receive a reply.)

- After the *Submit* button is pressed, the data is (usually) sent to a particular program running on a particular server, somewhere on the internet

Constructing a Form

method is the method used to transfer the data to the server

- Either *get* or *post*
- Differences discussed later

action is the location of the server side script

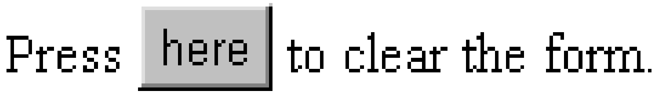
```
<form  method="get"
action="http://ihome.ust.hk/~rossiter/cgi-bin/show_environment.php"
>
.  .  .

</form>
```

HTML tags for form content go here

- As you know, you should use lower case letters for all HTML, which includes forms

Reset & Submit Buttons

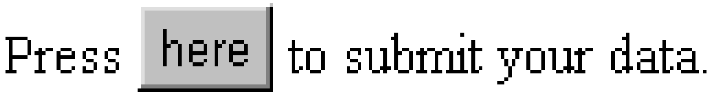
- For a reset button use *input type="reset"*
- Reset buttons are optional - not appropriate if there are only a few simple fields on the form
- Example:



Press here to clear the form.

```
Press <input type="reset"
  value="here"/> to clear the form.
```

- For a submit button use *input type=submit*
- Example:



Press here to submit your data.

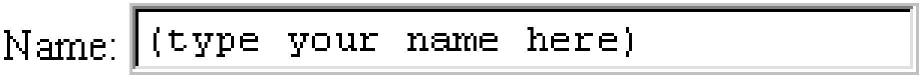
```
Press <input type="submit"
  value="here"/> to submit your data.
```

Text Input Field

- Use *input type=*"*text*" for a text field
- Various possibilities: ([ ] = optional)

```
<input [type=text] name="text-id" [size=nn]
   [maxlength=nn] [value="default text"] />
```

- Example:



Name: (type your name here)

```
Name: <input name="name" size="32"
     value="(type your name here)" maxlength="255"/>
```

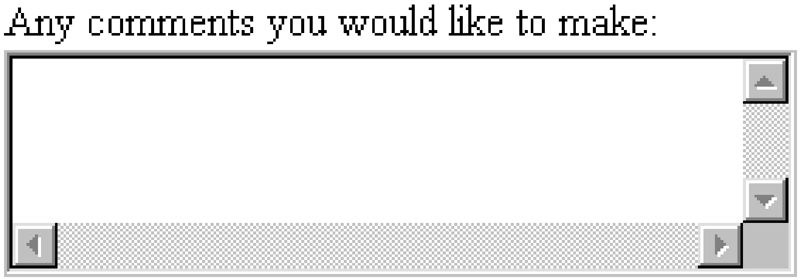
Text Area Field

- Use *textarea* for text input bigger than one line
- Various possibilities: ([ ] = optional)

```
<textarea name="text-id" [cols=mm] [rows=nn]>
   default text</textarea>
```

- Example:

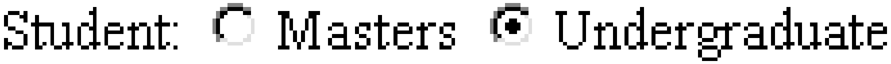
```
Any comments you would
   like to make: <br/>
<textarea name="comments"
   rows="3" cols="32"> </textarea>
```


Any comments you would like to make:

Radio Button Input

- Use *input type="radio"* for radio buttons
- Possibilities: <input type="radio" name="*radio-set-id*" value="*choice-id*"[checked] />

- Example:

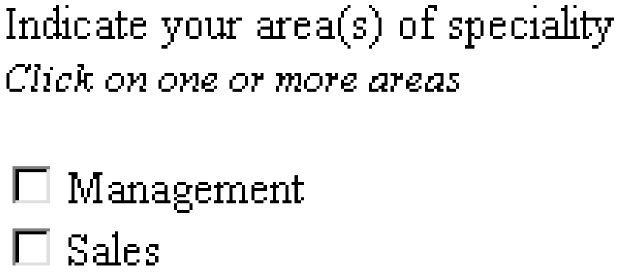

Student: ○ Masters ● Undergraduate

```
Student:
<input type="radio" name="degree" value="masters"/>Masters
<input type="radio" name="degree" value="undergraduate"
   checked/>Undergraduate
```

Checkbox Input

- Use *input type=*`"`*checkbox*`"` for a series of “on” or “off” switches
- Possibilities: `<input type="checkbox" name="`*`box-set-id`*`" value="`*`choice-id`*`" [checked] />`
- Example:

```
Indicate your area(s)
of specialty <br/>
```



Indicate your area(s) of speciality

Click on one or more areas

□ Management

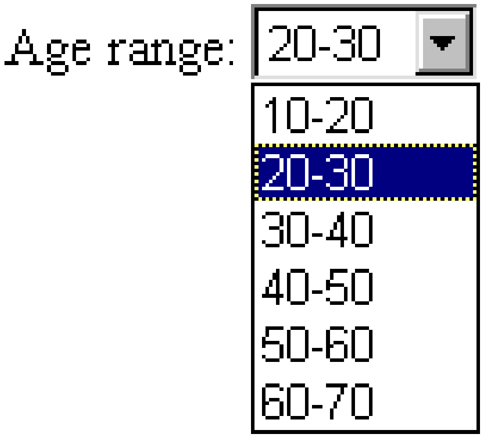
□ Sales

```
<input type="checkbox" name="speciality" value="1"/>Management<br/>
<input type="checkbox" name="speciality" value="2"/>Sales<br/>
```

Select Input

- Use *input type="select"* when one single option needs to be selected from several possibilities. Example:

```
Age range: <select name="age_range">
 <option>10-20</option>
 <option selected>20-30</option>
 <option>30-40</option>
 <option>40-50</option>
 <option>50-60</option>
 <option>60-70</option>
 </select>
```



Complete Form Code 1/5

```
<html>
<body>
<h1>HTML Form Example</h1>

<form method="get"
action="http://ihome.ust.hk/~rossiter/cgi-bin/
                                        show_environment.php">

Name:
<input name="name" size="32" value="(type your name here)"
   maxlength="255" />
```

Complete Form Code 2/5

```
<p> Student:
<input type="radio" name="degree" value="masters" />Masters
<input type="radio" name="degree" value="undergraduate"  checked />
    Undergraduate</p>
<p> Age range:

 <select name="age_range">
    <option>10-20</option>
    <option selected>20-30</option>
    <option>30-40</option>
     <option>40-50</option>
    <option>50-60</option>
     <option>60-70</option>
 </select>  </p>
```

Complete Form Code 3/5

```
<p>Indicate your area(s) of specialty<br/>
<i><small>Click on one or more areas</small></i>
<br/><br/>

<input type="checkbox" name="specialty" value="1"/>Management<br/>
<input type="checkbox" name="specialty" value="2"/>Sales<br/>
<input type="checkbox" name="specialty" value="3"/>Human Resources<br/>
<input type="checkbox" name="specialty" value="4"/>IT/ database<br/>
<input type="checkbox" name="specialty" value="5"/>IT/ networking<br/>
<input type="checkbox" name="specialty" value="6"/>IT/ internet<br/>
<input type="checkbox" name="specialty" value="7"/>IT/ general<br/>
```

Complete Form Code 4/5

```
<br/>
Any comments you would like to make:
<br/>
<textarea name="comments" rows="3" cols="32"/>
</p>
<p>
<input type="hidden" name="username" value="rossiter"/>
<input type="image" name="image" src="send_button.gif" align="centre"/>
<br/>
<br/>
</p>
```

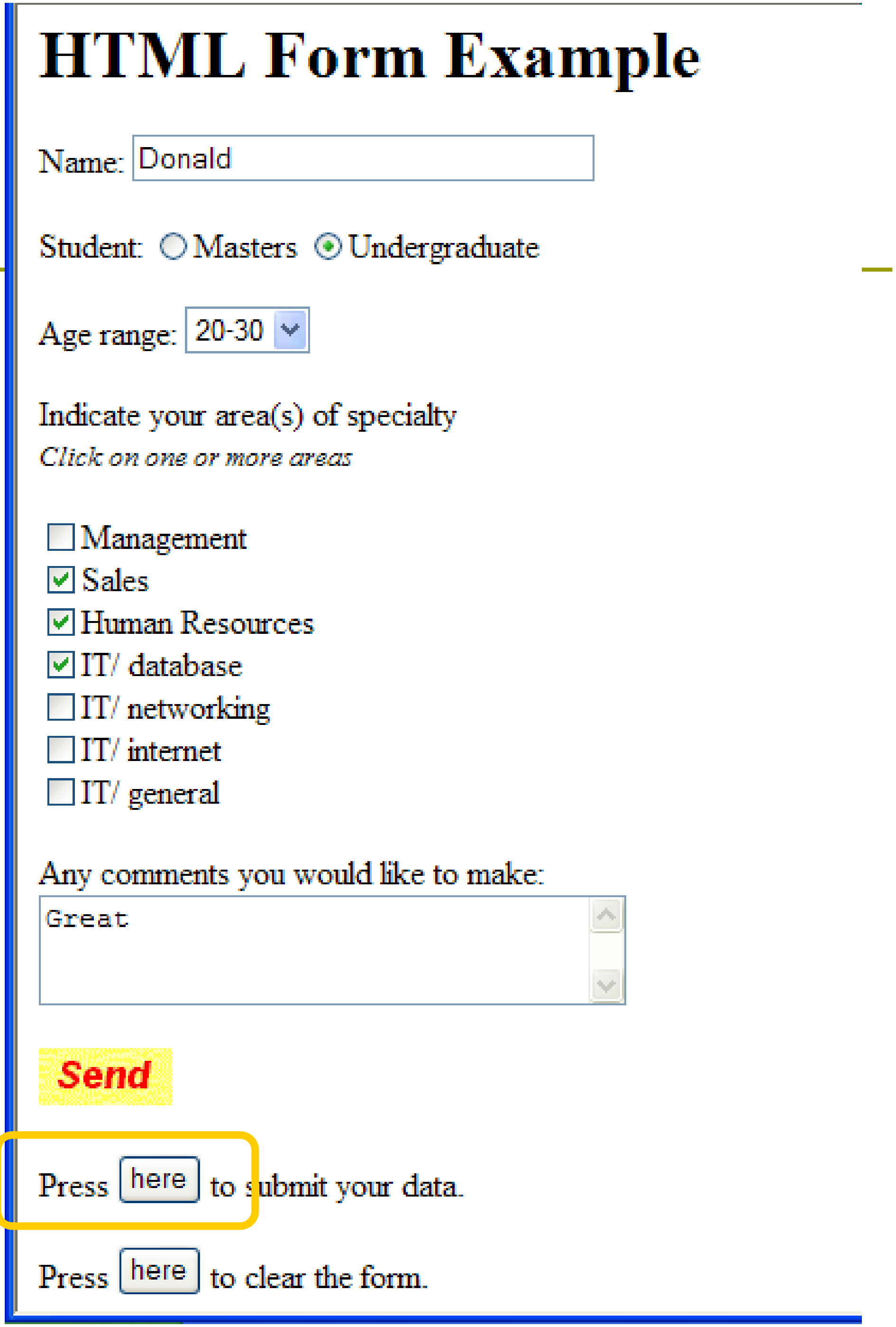
To be discussed in the next few slides

Complete Form Code 5/5

```
Press <input type="submit" value="here"/> to submit your data.
<br/>
<br/>

Press <input type="reset" value="here"/> to clear the form.
<br/>
<br/>

</form>
</body>
</html>
```

Complete Form Display

- After making the selections shown on the right the form is then submitted by clicking on the submit button
- The response is shown in the following two slides



HTML Form Example

Name: Donald

Student: ○ Masters ● Undergraduate

Age range: 20-30

Indicate your area(s) of specialty

Click on one or more areas

□ Management
☑ Sales
☑ Human Resources
☑ IT/ database
□ IT/ networking
□ IT/ internet
□ IT/ general

Any comments you would like to make:

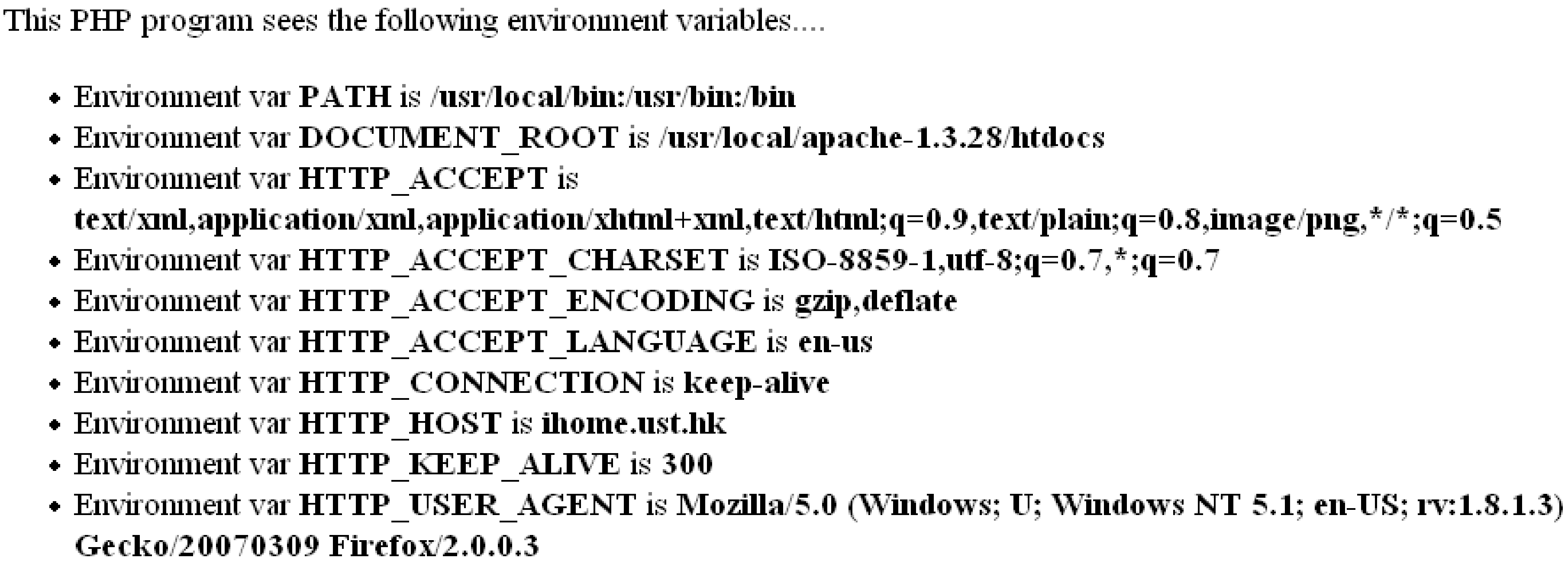
Great

Send

Press here to submit your data.

Press here to clear the form.

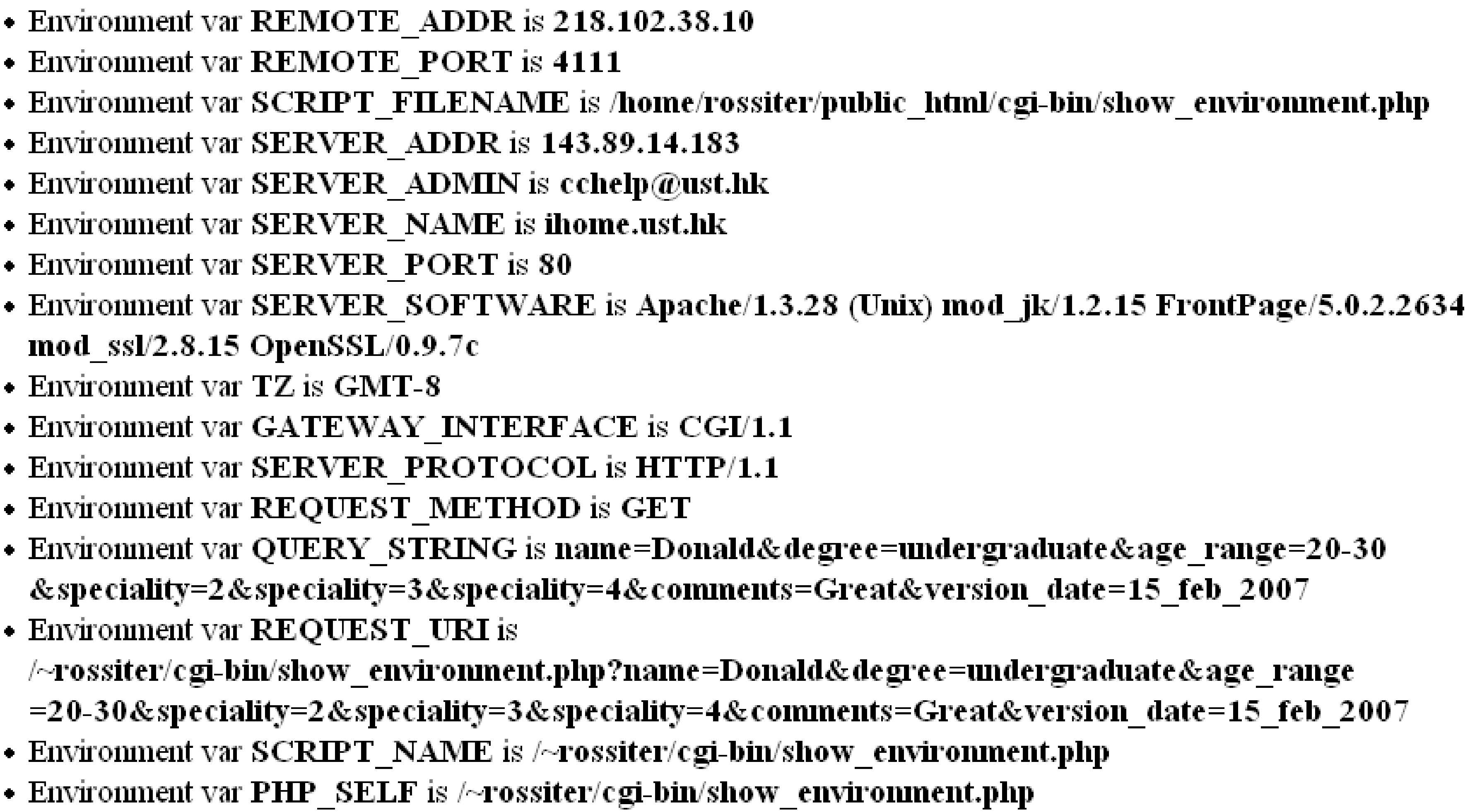
Data Seen By Server Side Program 1/2



This PHP program sees the following environment variables....

- Environment var **PATH** is **/usr/local/bin:/usr/bin:/bin**
- Environment var **DOCUMENT_ROOT** is **/usr/local/apache-1.3.28/htdocs**
- Environment var **HTTP_ACCEPT** is **text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5**
- Environment var **HTTP_ACCEPT_CHARSET** is **ISO-8859-1,utf-8;q=0.7,*;q=0.7**
- Environment var **HTTP_ACCEPT_ENCODING** is **gzip,deflate**
- Environment var **HTTP_ACCEPT_LANGUAGE** is **en-us**
- Environment var **HTTP_CONNECTION** is **keep-alive**
- Environment var **HTTP_HOST** is **ihome.ust.hk**
- Environment var **HTTP_KEEP_ALIVE** is **300**
- Environment var **HTTP_USER_AGENT** is **Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.3) Gecko/20070309 Firefox/2.0.0.3**

Data Seen By Server Side Program 2/2



- Environment var **REMOTE_ADDR** is **218.102.38.10**
- Environment var **REMOTE_PORT** is **4111**
- Environment var **SCRIPT_FILENAME** is **/home/rossiter/public_html/cgi-bin/show_environment.php**
- Environment var **SERVER_ADDR** is **143.89.14.183**
- Environment var **SERVER_ADMIN** is **cchelp@ust.hk**
- Environment var **SERVER_NAME** is **ihome.ust.hk**
- Environment var **SERVER_PORT** is **80**
- Environment var **SERVER_SOFTWARE** is **Apache/1.3.28 (Unix) mod_jk/1.2.15 FrontPage/5.0.2.2634 mod_ssl/2.8.15 OpenSSL/0.9.7c**
- Environment var **TZ** is **GMT-8**
- Environment var **GATEWAY_INTERFACE** is **CGI/1.1**
- Environment var **SERVER_PROTOCOL** is **HTTP/1.1**
- Environment var **REQUEST_METHOD** is **GET**
- Environment var **QUERY_STRING** is **name=Donald&degree=undergraduate&age_range=20-30&speciality=2&speciality=3&speciality=4&comments=Great&version_date=15_feb_2007**
- Environment var **REQUEST_URI** is **/~rossiter/cgi-bin/show_environment.php?name=Donald&degree=undergraduate&age_range=20-30&speciality=2&speciality=3&speciality=4&comments=Great&version_date=15_feb_2007**
- Environment var **SCRIPT_NAME** is **/~rossiter/cgi-bin/show_environment.php**
- Environment var **PHP_SELF** is **/~rossiter/cgi-bin/show_environment.php**

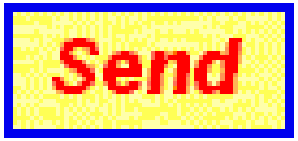
The Fields Which Contain Form Data

- In this example, the GET method is being used to send data to the server
- You can see from the last two slides that the QUERY_STRING field contains the form data
- Because the GET method is being used, the REQUEST_URI field also has the form data
- The server side program can extract the data from either of those strings

Images in Forms

- If you just want to put an image in a form for visual effect and nothing more, just use the usual `<img src="image.gif" />` instruction
- If you want to use an image to act as a submit button, use *input type="image"*
- For example:

```
<input type="image" name="image"
      src="send_button.gif"/>
```



Images in Forms

- Possible syntax:

```
   <input type="image"
name="image-id" src="image-url" [align=alignment] />
```

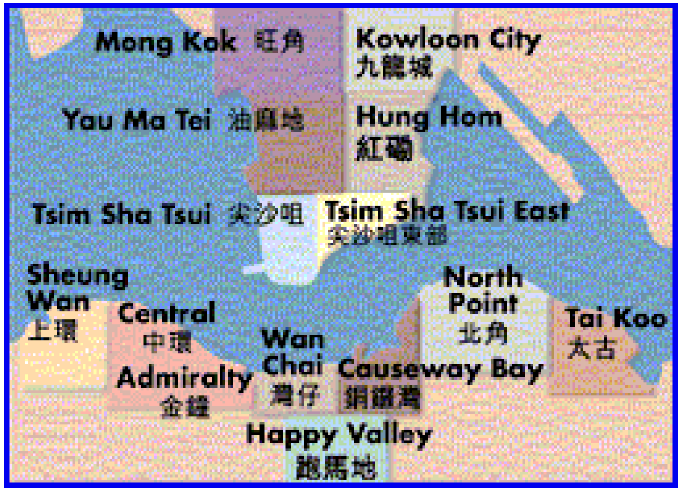
- There is another advantage to using images
- The exact (x, y) position that the user clicks on is automatically sent to the server side program along with all the other form data
- The server side program may choose to use it or ignore it

Images in Forms

- Example clickable map:

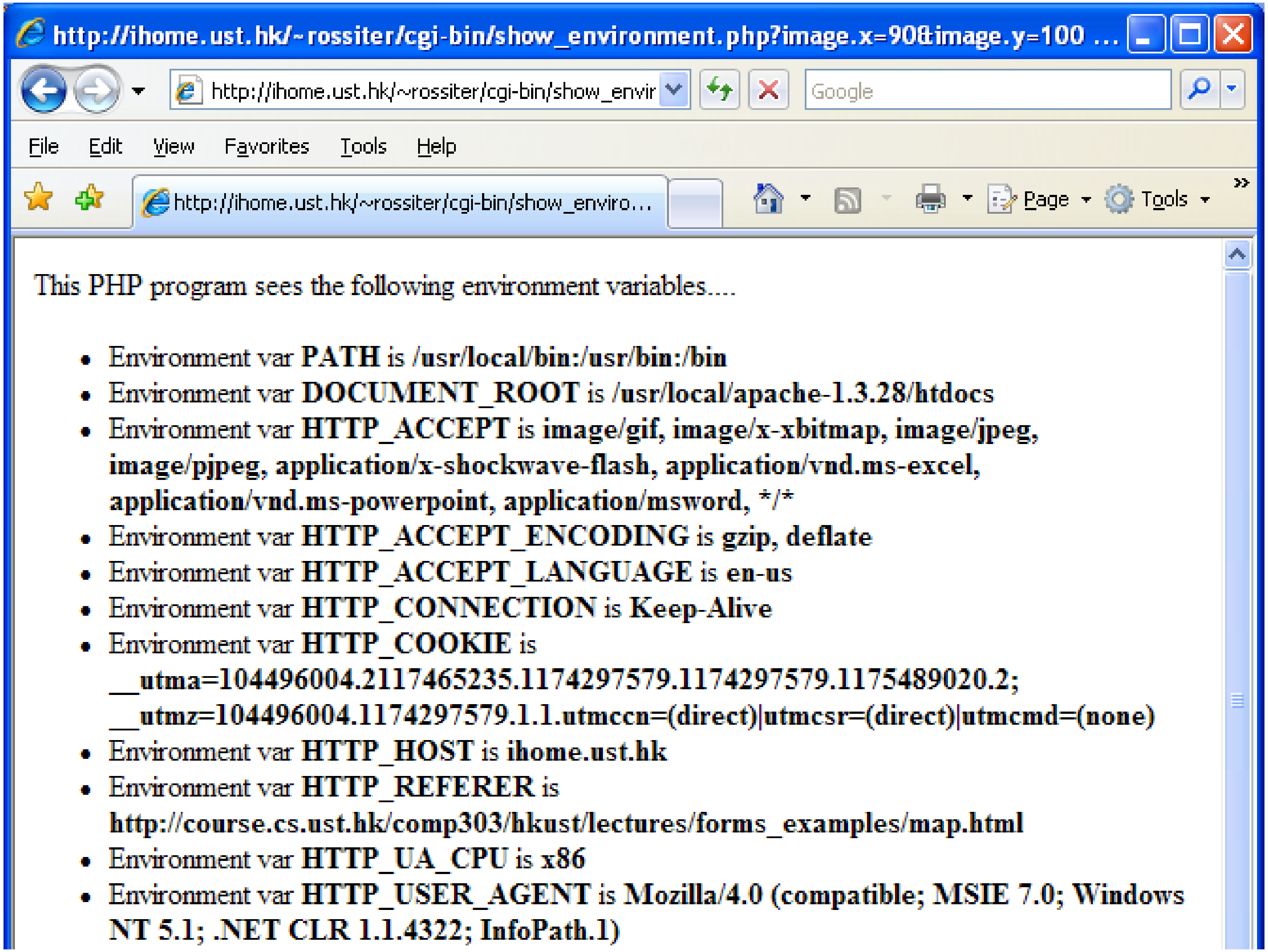
```
What is your favourite
Hong Kong shopping area?
<br/>

<form method="get"
action="http://ihome.ust.hk/~rossiter/cgi-bin/
                                  show_environment.php”>
<input type="image" name="image" src="harbour.gif"/>
</form>
```

What is your favourite Hong Kong shopping area?



- After clicking on the image data is sent to the server program
- This particular server side program simply lists all the data it knows about – see next two slides

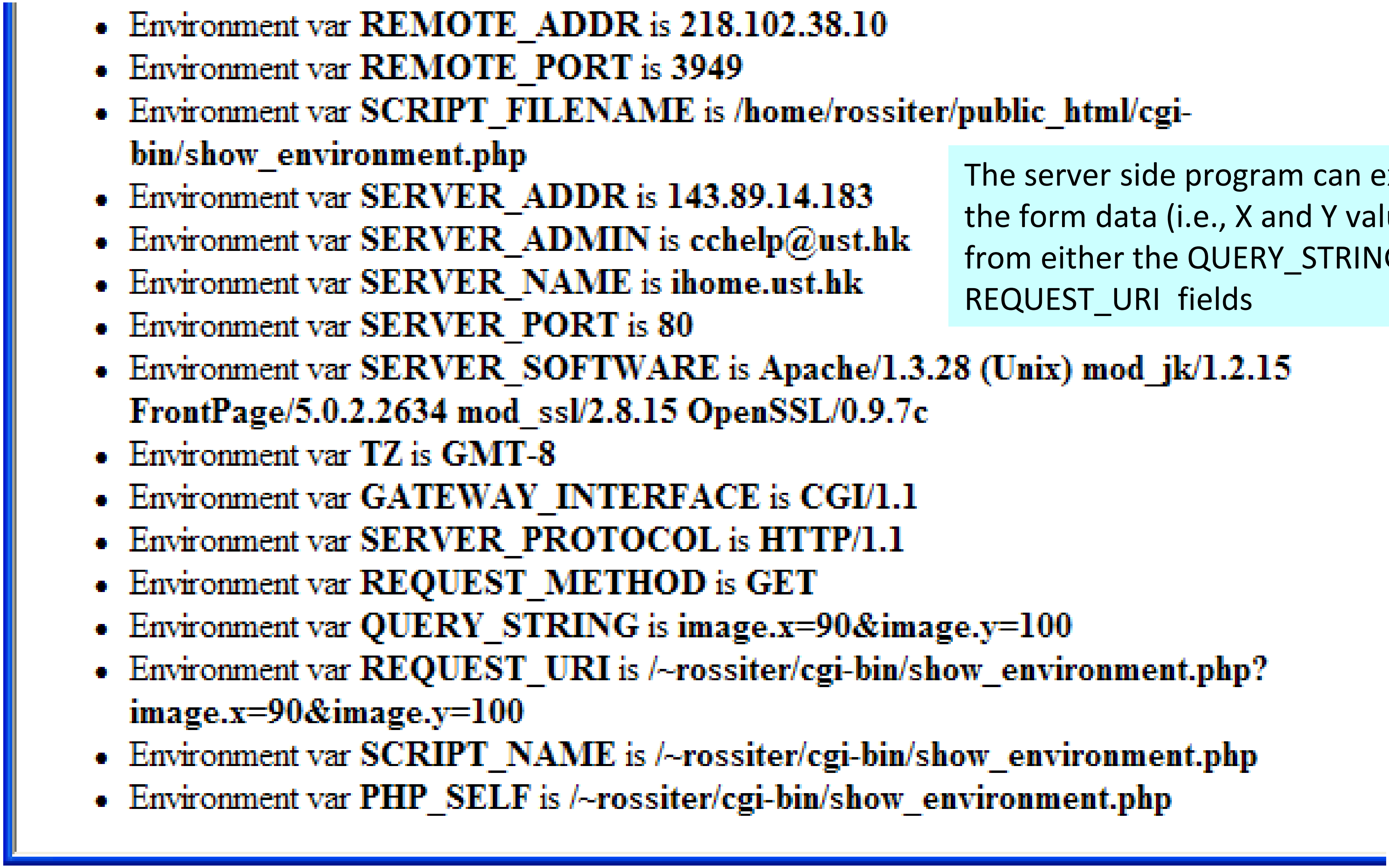
Data Seen By Server Side Program 1/2


http://ihome.ust.hk/~rossiter/cgi-bin/show_environment.php?image.x=90&image.y=100 ...
http://ihome.ust.hk/~rossiter/cgi-bin/show_envir
Google
File
Edit
View
Favorites
Tools
Help
http://ihome.ust.hk/~rossiter/cgi-bin/show_enviro...
Page
Tools

This PHP program sees the following environment variables....

- Environment var **PATH** is **/usr/local/bin:/usr/bin:/bin**
- Environment var **DOCUMENT_ROOT** is **/usr/local/apache-1.3.28/htdocs**
- Environment var **HTTP_ACCEPT** is **image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */***
- Environment var **HTTP_ACCEPT_ENCODING** is **gzip, deflate**
- Environment var **HTTP_ACCEPT_LANGUAGE** is **en-us**
- Environment var **HTTP_CONNECTION** is **Keep-Alive**
- Environment var **HTTP_COOKIE** is **__utma=104496004.2117465235.1174297579.1174297579.1175489020.2; __utmz=104496004.1174297579.1.1.utmccn=(direct)|utmcsr=(direct)|utmcmd=(none)**
- Environment var **HTTP_HOST** is **ihome.ust.hk**
- Environment var **HTTP_REFERER** is **http://course.cs.ust.hk/comp303/hkust/lectures/forms_examples/map.html**
- Environment var **HTTP_UA_CPU** is **x86**
- Environment var **HTTP_USER_AGENT** is **Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 1.1.4322; InfoPath.1)**

Data Seen By Server Side Program 2/2



```
• Environment var REMOTE_ADDR is 218.102.38.10
• Environment var REMOTE_PORT is 3949
• Environment var SCRIPT_FILENAME is /home/rossiter/public_html/cgi-
  bin/show_environment.php
• Environment var SERVER_ADDR is 143.89.14.183
• Environment var SERVER_ADMIN is cchelp@ust.hk
• Environment var SERVER_NAME is ihome.ust.hk
• Environment var SERVER_PORT is 80
• Environment var SERVER_SOFTWARE is Apache/1.3.28 (Unix) mod_jk/1.2.15
  FrontPage/5.0.2.2634 mod_ssl/2.8.15 OpenSSL/0.9.7c
• Environment var TZ is GMT-8
• Environment var GATEWAY_INTERFACE is CGI/1.1
• Environment var SERVER_PROTOCOL is HTTP/1.1
• Environment var REQUEST_METHOD is GET
• Environment var QUERY_STRING is image.x=90&image.y=100
• Environment var REQUEST_URI is /~rossiter/cgi-bin/show_environment.php?
  image.x=90&image.y=100
• Environment var SCRIPT_NAME is /~rossiter/cgi-bin/show_environment.php
• Environment var PHP_SELF is /~rossiter/cgi-bin/show_environment.php
```

The server side program can extract the form data (i.e., X and Y values) from either the QUERY_STRING or REQUEST_URI fields

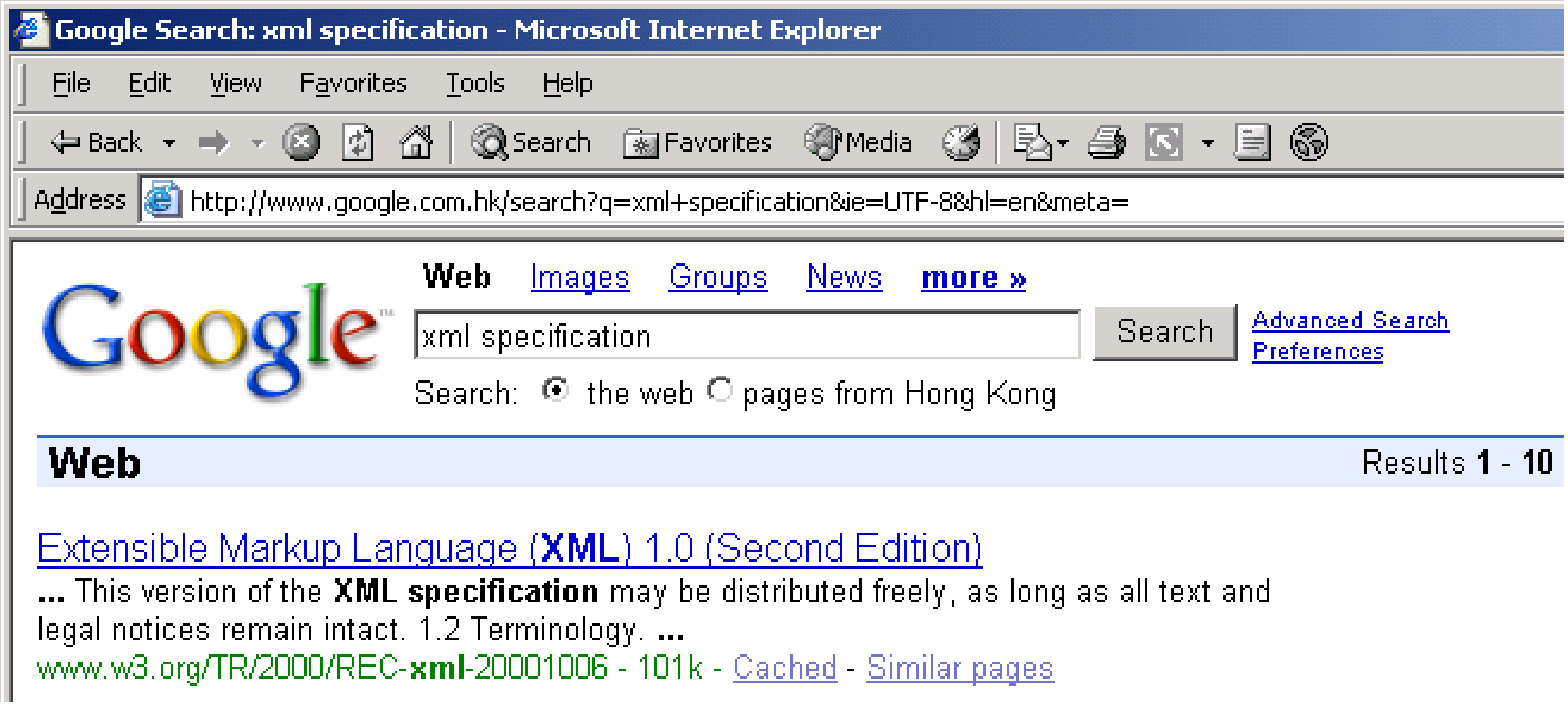
Get vs. Post

- A reminder - there’s two ways to send information from client to server; ‘get’ or ‘post’

```
<form  method="get"
action="http://ihome.ust.hk/~rossiter/cgi-bin/show_environment.php">

. . .

</form>
```

Example Get

- When the form data is sent, all the parameter names and values are shown on the browser location line



Google Search: xml specification - Microsoft Internet Explorer

Address: http://www.google.com.hk/search?q=xml+specification&ie=UTF-8&hl=en&meta=

xml specification

Search: ● the web ○ pages from Hong Kong

Web Results **1** - **10**

Extensible Markup Language (**XML**) 1.0 (Second Edition)

... This version of the **XML specification** may be distributed freely, as long as all text and legal notices remain intact. 1.2 Terminology. ...

www.w3.org/TR/2000/REC-**xml**-20001006 - 101k - Cached - Similar pages

The Get Method

- Good points
  - All the parameter names and values are shown - helpful for debugging a system you are developing
  - Easy for a program to build up the single text sequence and send it
- Bad points
  - Maximum length of a URL is about 2000 characters, sometimes is not enough
  - Users can see all the parameter names and values on the browser location line, even the ‘hidden’ ones
  - Users can easily change the values and send them to your system, could cause trouble for you

The Post Method

- When the form data is sent using the post method, all the parameter names and values, etc, are added after the HTTP header
- Good points
  - User can’t see any parameter names/ values
  - There is no limit on how many bytes you can send from client to server i.e. can send many large MP3 files, etc, using the Post method
- Bad points
  - A little bit harder to program for some languages (on the server side)

Form + JavaScript

JavaScript Processing

- Commonly, forms are ‘dead’
  - User just types/ selects, then submits
- Can do client-side processing of form data
  - Could use JavaScript to do some kind of action, such as assess checking data when an event occurs
  - Don’t actually *have* to send data to server, could use a form just for text input to JavaScript, etc

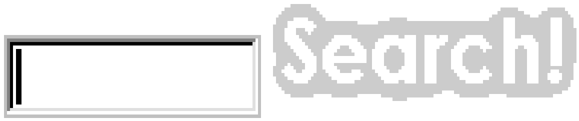
Useful JavaScript Events

- You already know lots of events for handling the mouse/keyboard, etc - here are some events which can be used for handling forms:
- onsubmit - fires when the user submits the form data (fires just before the data is actually sent)
- onreset - fires when the user resets all the fields in a form (fires just before the data is actually reset)
- onchange - fires just after the contents of something is changed
- onselect – fires when user selects text in a text field

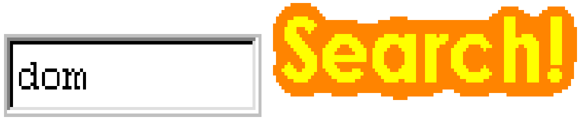
Example JavaScript Processing

- Example JavaScript checking when user triggers *onsubmit*:
  - Checking student ID field has correct number of characters
  - Checking telephone number field only has digits

- Example JavaScript change of image when user triggers *onchange*:
  - Nothing yet entered, dull image:



  - As soon as text entered, bright image:


dom

Hidden Fields

Hidden Fields

- Hidden fields are useful for implementing ‘system’ variables shared between the server and the client
- A hidden field is also useful for the client to send information to the server (without using the standard input fields such as text fields and selection boxes)

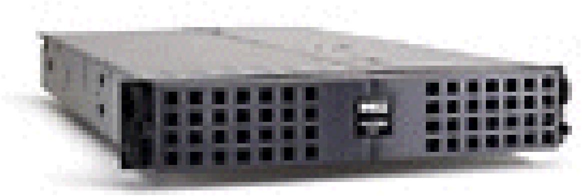
Hidden Fields

- Use *input type="hidden"* for hidden fields; e.g.
  - <input type="hidden" name="creation_time“ value="15_2_2010_14:32"/>
- The field does not appear on screen; hence, users would not see it or know about it unless using “view source”
- The user can’t change the value of the field
- This method can be used for storing ‘state’ information (i.e., which both browser and server code can read and change)
  - This can be very useful, particularly if the user/ browser does not permit cookies on their machine
  - However, cookies and sessions are more commonly used, to be discussed later

Example Hidden Field Usage

1) HTML page is generated by server

```
<input type="hidden" name="username"
     value="rossiter" /> . . .
```



2) Hidden field is sent to browser across internet (inside the HTML)

3) User enters form data as usual and presses *Submit*

4) Data is sent to server across internet, including the hidden field

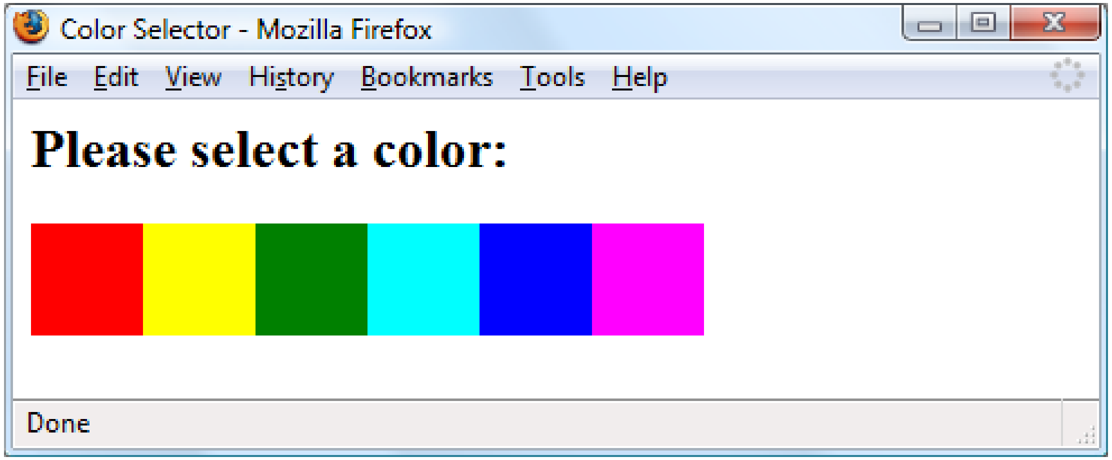
5) Server program looks at hidden field, responds appropriately:

. . . Thanks rossiter, here’s some other web information for you

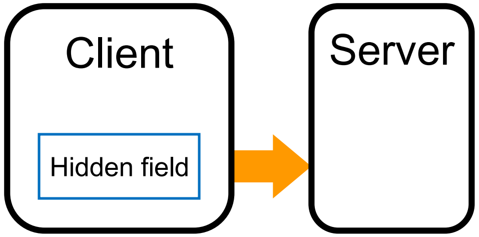
. . .

Example – Color Selector

- An HTML form does not provide a simple input field which handles color selection
- We can build a color selector using several DIVs and then send the color using a hidden field


Color Selector - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Done

Please select a color:



Example 1

```
<html>
 <head>
  <title>Color Selector</title>
  <style>
   div {
    position: absolute;
    width: 50px;
    height: 50px;   }
  </style>
  <script>
  function select(color) {
   document.getElementById("color").value=color;
   document.getElementById("color").form.submit();  }
  </script>
 </head>
```

Set hidden field value and submit the form using JavaScript

Continued in next slide

Example 1: Form + JavaScript

```
<body> <h2>Please select a color:</h2>
  <div style="position:relative">
    <div style="background-color:red;left:0px"
       onclick="select('red')"></div>
    <div style="background-color:yellow;left:50px"
       onclick="select('yellow')"></div>
    <div style="background-color:green;left:100px"
       onclick="select('green')"></div>
    <div style="background-color:cyan;left:150px"
       onclick="select('cyan')"></div>
    <div style="background-color:blue;left:200px"
       onclick="select('blue')"></div>
    <div style="background-color:magenta;left:250px"
       onclick="select('magenta')"></div>  </div>
  <form action="..." method="GET">
    <input type="hidden" name="color" id="color" value="" />
  </form> </body>  </html>
```

The DIVs for a ⇐
simple color
selector

⇓

onclick passes the
selected color to
the “select”
function

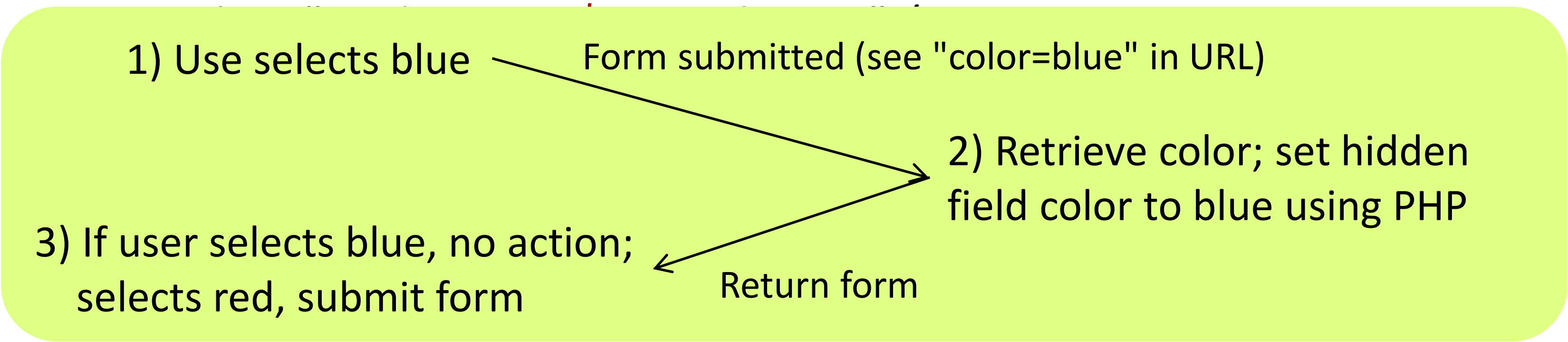
⇓

"select" function
sets the "value"
attribute to the
selected color ⇒

Setting Hidden Field in the Server

- In Example 1, the selected color is always sent to the server even when the same color is clicked several times
- It is better to send the selected color only when it is different from the color previously selected color
- In Example 2, the server sends back the user's last selected color to the browser by setting a hidden field's value with PHP:

```
<input type="hidden" name="color" id="color"
[illegible]
```



1) Use selects blue

Form submitted (see "color=blue" in URL)

2) Retrieve color; set hidden field color to blue using PHP

Return form

3) If user selects blue, no action; selects red, submit form

Example 2: Adding PHP code in Server Program

```
<?php
   $my_color = $_GET['color'];  // get "color" submitted from browser
?>
<html>  <head>
      . . .
      <script>
      function select(color) {
         var fld = document.getElementById("color");
         if (color != fld.value) {
            if (!confirm("Do you want to change " +
                                 fld.value + " to " + color + "?"))
               return;
            fld.value=color;
            fld.form.submit();
         }  }
      </script>   </head>
```

New color

Retrieve hidden form field

Previous color (set by php via hidden field)

Form is submitted only if the value is changed to a new value

Continued in next slide

Example 2 Cont.

```
<body>   <h2>Please select a color:</h2>
   <div style="position:relative">
     <div style="background-color:red;left:0px"
        onclick="select('red')"></div>
     <div style="background-color:yellow;left:50px"
        onclick="select('yellow')"></div>
     <div style="background-color:green;left:100px"
        onclick="select('green')"></div>
     <div style="background-color:cyan;left:150px"
        onclick="select('cyan')"></div>
     <div style="background-color:blue;left:200px"
        onclick="select('blue')"></div>
     <div style="background-color:magenta;left:250px"
        onclick="select('magenta')"></div>   </div>
  <form action="..." method="GET">
   <input type="hidden" name="color" id="color"
       value="<?php print $my_color; ?>" /> </form></body></html>
```

The color hidden field with a value set by the server

$my_color is set in PHP code (top of previous slide) to return user's previously selected color

Take Home Message

- Form is a very natural and popular way to present information to users and solicit input from them
- Hidden fields can be used to embed data in the form without users seeing the data (e.g., encode user name in a hidden field)
- Hidden fields in forms are very useful in passing states of the clients to the server
  - Those states can be passed to the server when the form is submitted
  - No additional action is required other than submitting the form (which the user will do anyway); user indeed does not know about this