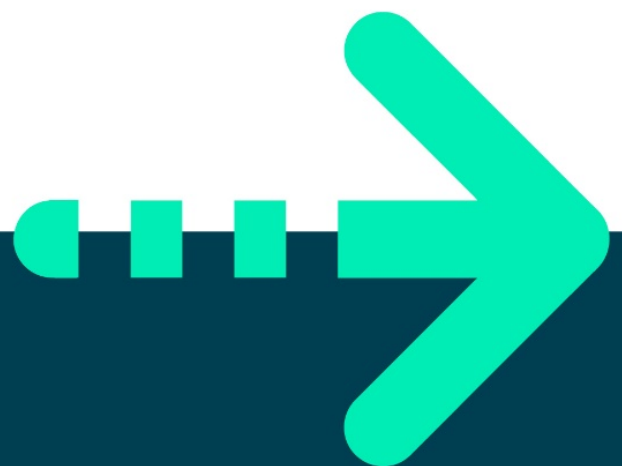




HTML AND CSS





How the Internet Works

Objectives

In this exercise you will be looking at the technologies on which the Internet is built. In particular you will be looking at:

- The HTTP protocol and the use of the form postback technique

Reference material

This exercise is based on material from the chapter “How the Internet Works”.

Overview

In this exercise, you will use a custom version of the Telnet program to make HTTP requests to a Web server; from this you will see how Web browser / Web server communication takes place. You will then look at the Internet Information Services Manager tool and machine.config / web.config files.

Estimated duration

The estimated duration for this lab is 30 minutes.

Completed solution

There is no completed solution for this lab.

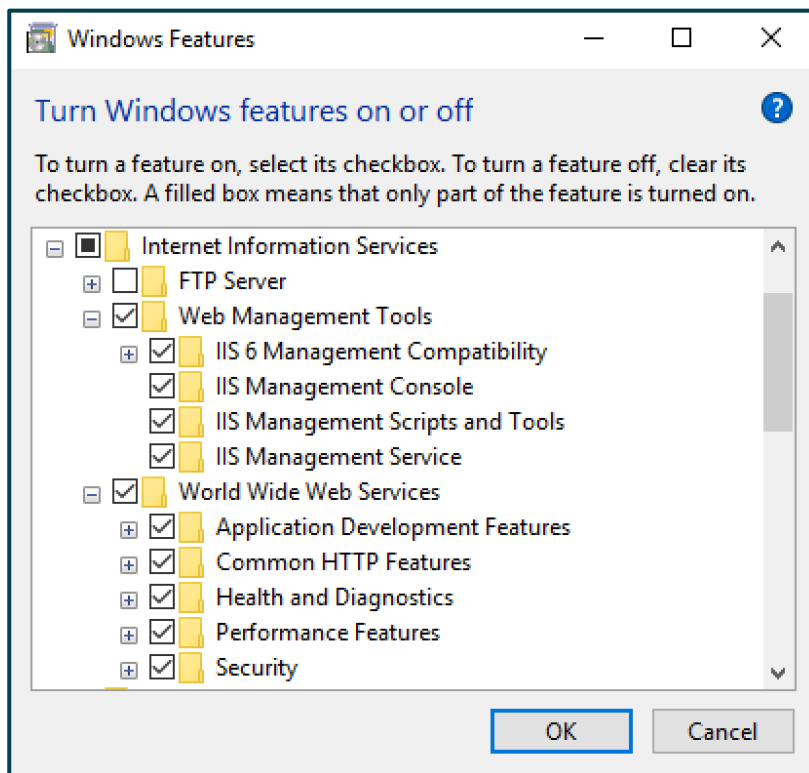


Step by step

Setting Up

For this exercise, you will need to use the IIS web server. This is installed on your computer, but is not switched on by default. Follow these instructions to start IIS:

1. On the start menu, select Control Panel, then Programs and Features
2. Select "Turn Windows Features On and Off"
3. Scroll down until you find the Internet Information Services section
4. Underneath Internet Information Services, you will find many different section and sub-sections with settings that can be turned on or off. You should leave the "FTP" section alone – but in all the other sections, go into each sub-section in turn, and switch **on** every setting:



5. If you are prompted to restart your computer then you should do so before you continue.

Understanding HTTP

Communication between the Web browser and Web server is via the HTTP protocol.

1. Copy all the files located in "1 - HTML and CSS including revision\Lab Resources\Chapter 3 - How the Internet works\Web Root" to "C:\inetpub\wwwroot". (The default website home directory is LocalDrive:\inetpub\wwwroot.).



2. Open a browser window (e.g. IE or Firefox), navigate to the page <http://localhost/Home.html> and wait for the page to be displayed. From the browser, view the source of the page (**IE: View > Source; FF: View > Page Source**).
3. Next you are going to request the same page manually using a Telnet emulation program, *CoursewareFolder\02 Web Application Architectures\Resources\QATelnet.exe*. Launch this now and type in the following command in the top half of the window:

```
GET /Home.html HTTP/1.0
```

You'll need to press enter after the blank line at the bottom; this blank line is important, because it indicates the end of the HTTP Request header. Click the **Send!** menu; you should see the response appear in the bottom half of the window.

4. If this Telnet program was going to display the page, rather than show you the HTML source, would it have all the necessary data to do so? Think back to how the page looked in the browser back in step 1.
5. The page itself contains an image; you can see the `` tag in the HTML returned. You have requested the page, but not the image (logo.gif), and hence you could not display the full page as yet. To request the image you would need to request it separately:

```
GET /logo.gif HTTP/1.0
```

Delete the current request and try this new request now, and you should see the content of the GIF image appear in the bottom half of the Telnet program. Note therefore, that some requests will request the page, whereas others will request supporting information for that page. Other common examples include stylesheets and JavaScript files.

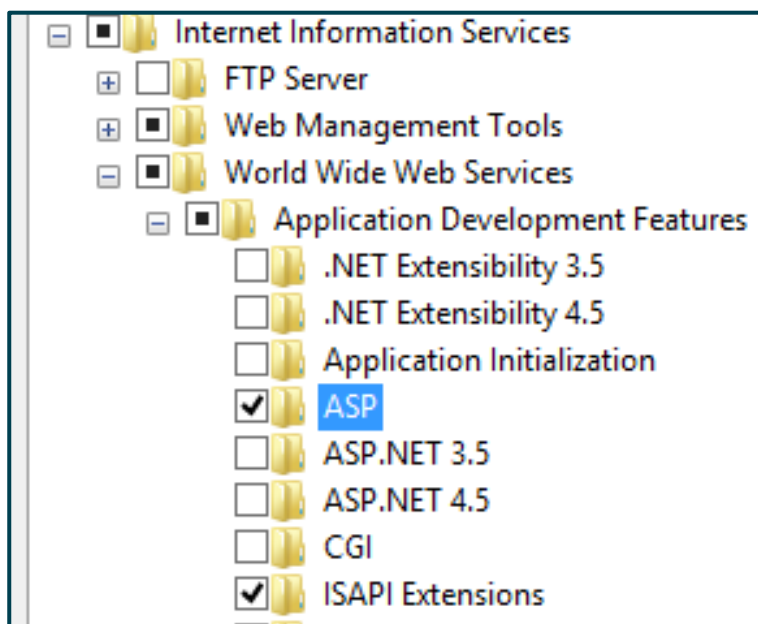
Postbacks

When a Web page presents a form to the user and the user presses a Submit button, the data is sent back to the Web server. A common technique is to submit the data back to the same page¹.

6. Open a browser window (e.g. IE or Firefox), navigate to the page <http://localhost/NameAndAddress.asp> and wait for the page to be displayed. Type in your details and press submit.

If you get the message: *"The page you are requesting cannot be served because of the extension configuration"* then you'll need to configure IIS for classic asp pages. Simply do the following steps:

- a) open the Control Panel and then select Programs and Features->Turn Windows features on or off->.ASP



7. Next we are going to request the same page manually. Run the Telnet emulation program, **CoursewareFolder\02 Web Application Architectures\Resources\QATelnet.exe**, and type the following command in the top half of the window:

```
GET /NameAndAddress.asp HTTP/1.0
```

Note that the blank line on the second line is important. When you typed in the GET command, click the **Send!** menu; you should see the response appear in the bottom half of the window.

8. The Web page that is returned contains a form with the Method set to POST and the Action set to the same URL as the page. This page will post back to itself. In the top window of the Telnet program, delete the GET command and type in the following command:

¹ Strictly this is back to the same URL as it's a new instance of the page.



```
POST /NameAndAddress.asp HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 26

Name=Noddy&Address=Toytown
```

The Content-Length value must equal the number of characters in the form data.

Housekeeping

In the previous part, the NameAndAddress page knew your details because you submitted them along with the request. The NameAndAddress page could then write out your details in the corresponding response. Let's look at another variation of this:

9. Open a browser window, navigate to <http://localhost/State.asp> and wait for the page to be displayed. Type in some data into the textbox and press submit. This page is similar to the NameAndAddress page except that it displays the textbox for both the GET and the POST. What do you notice about the content of the textbox on postback?

10. We have a version written in classic ASP that will maintain the state of the textbox, this file is called state2.asp; you can try this page in a browser if you like. Keep C:\InetPub\wwwroot\state.aspx open in Notepad, and also open the file C:\InetPub\wwwroot\state2.asp open in Notepad. Compare the code that determines whether the page was requested via a postback in the two files.

11.

