



Programmazione web

Lezione del 2 Marzo 2020

Prof.ssa Novella Bartolini

Ricevimento: Mercoledì ore 12:30-14:30

(si prega di prendere appuntamento per email)

Via Salaria 113, terzo piano, stanza 309

Email: bartolini@di.uniroma1.it



Informazioni sul corso

- Sito del corso
 - <https://twiki.di.uniroma1.it/twiki/view/PW/AnnoAccademico1920>
- Orario lezioni
- Orario di ricevimento
- Testi di riferimento
- Slide delle lezioni
- **Comunicazioni varie**



Orario lezioni e ricevimento

- ➔ **Lezione: Lunedì 10.15,
Giovedì 11.15**
 - Eventuali variazioni verranno segnalate sul sito

- ➔ **Ricevimento: Mercoledì dalle 12.30 alle 13.30**
 - Prenotare per email

- ➔ **Organizzazione delle lezioni (*perlopiù...*):**
 - **Lunedì: teoria**
 - **Giovedì: esercitazioni pratiche**



Testi di riferimento sul sito del corso

- ➔ [Dispensa su XHTML \(Deitel & Deitel\)](#), Prentice Hall and Deitel & Associates
- ➔ [Marty Hall, "Core Servlets and Java Server Pages"](#), Prentice Hall & Sun Microsystems
- ➔ SLIDE DEL CORSO!



Modalità di esame per tutti

- ➔ L'esame finale consiste in una prova scritta in due parti: una parte pratica di realizzazione di segmenti di un'applicazione web, e una parte di teoria.
- ➔ Il voto finale sarà la media dei voti conseguiti nella parte pratica e in quella teorica.



WIS

- ➔ Un Web-based Information System è un sistema informatico basato sul web.
- ➔ Non è un insieme di pagine web
- ➔ Ha un'elevata complessità sia in termini di dati che di applicazioni
- ➔ E' spesso integrato con sistemi diversi come DBMS, sistemi transazionali ecc.



Tipologie di servizi elettronici

➔ Informativi

- Per la fornitura su richiesta di informazioni strutturate e classificate, contenuti multimediali

➔ Di comunicazione

- Interazione bidirezionale tra individui

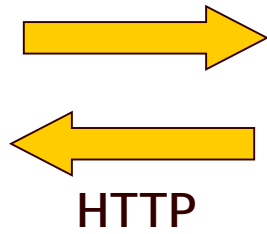
➔ Transazionali

- Per acquistare prodotti o servizi online, o per trasmettere dati

Architetture realizzative di un WIS



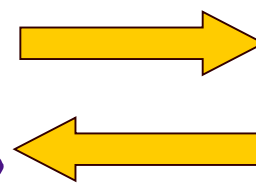
Client Browser



HTTP



Web Server



Local servers or repositories
Other remote servers



Web Browser (client)

Software per reperire pagine web, attraverso il protocollo HTTP e visualizzarle

- Invia richieste e riceve risposte secondo il protocollo HTTP
- Interpreta comandi di formattazione espressi in HTML
- Visualizza file di tipo diverso (espresso attraverso un'*estensione MIME*)



MIME

- ⇒ Multipurpose Internet Mail Extensions
- ⇒ Standard che specifica tipi di oggetto non testuali per la trasmissione in applicazioni Internet (WWW, e-mail)
- ⇒ Attraverso l'estensione MIME è possibile associare un oggetto ad un'applicazione che lo gestisca
- ⇒ MIME specifica solo il formato degli oggetti, che vengono trasmessi secondo una codifica standard (base64)
- ⇒ Standard che prevede tutte le possibili funzionalità per la trasmissione dei documenti; non è detto che un applicativo sia in grado di realizzarle tutte



Web Server

- ➔ Programma in grado di fornire pagine web (memorizzate sulla macchina su cui viene eseguito) attraverso il protocollo HTTP
- ➔ Processo demone con socket in ascolto sulla porta TCP 80
- ➔ I più diffusi:
 - Apache
 - Internet Information Server (IIS)
 - NGINX



World Wide Web Consortium W3C

W3C

Fondato nel 1994 da Tim Berners-Lee
per lo sviluppo e l'integrazione di tecnologie per il
World Wide Web.

Ente di standardizzazione

W3C Recommendations: documenti che specificano il
funzionamento delle tecnologie per il WWW (es:
Extensible HyperText Markup Language - XHTML,
Cascading Style Sheets – CSS, e Extensible Markup
Language - XML)



XHTML

- ➔ Extensible HyperText Markup Language
 - Evoluzione dell'HTML
- ➔ Documenti XHTML
 - Costituiscono la codifica sorgente delle pagine web
 - Editabili attraverso un comune editor testuale
 - Estensione del nome dei file **.html** o **.htm**
- ➔ Web server
 - Memorizza i documenti XHTML
- ➔ Web browser
 - Effettua richieste di documenti XHTML



Testo di riferimento su XHTML

➔ Presente sul sito

http://twiki.dsi.uniroma1.it/twiki/view/Lab_prog_rete/WebHome

Introduction to XHTML, Deitel & Deitel



Struttura di un documento XHTML

- ➔ Il concetto centrale è quello di tag (etichetta)
 - sintatticamente, un tag si apre con `<nometag>` e si chiude con `</nometag>`
 - tutti i tag aperti devono essere chiusi
 - tra un tag aperto e uno chiuso può trovarsi qualsiasi testo, eventualmente contenente altri tag
 - se non c'è nessun testo, allora è ammessa la sintassi `<nometag/>`
 - Un tag può contenere una lista di assegnamenti `nomeattributo="valore"` tra `<nometag e >` o `/>` (si può usare anche `'` invece di `"`)



Struttura di un documento XHTML

- ➔ La struttura del documento viene definita attraverso
 - **Start tag** (es. `<html>`)
 - Definiscono attributi attraverso espressioni *name = value*
 - **End tag** (es. `</html>`)
- ➔ Intestazione (**head**)
 - Definisce il titolo del documento
 - Include fogli di stile e di scripting
- ➔ Corpo centrale (**body**)
 - Definisce il contenuto che viene visualizzato dal browser
- ➔ Commenti XHTML
 - Iniziano con `<!--` e terminano con `-->`


```
1 <html>
2   <head>
3     <title>Internet and WWW How to Program - Welcome</title>
4   </head>
5
6   <body>
7     <p>HTML: pagina di prova </p>
8   </body>
9 </html>
```

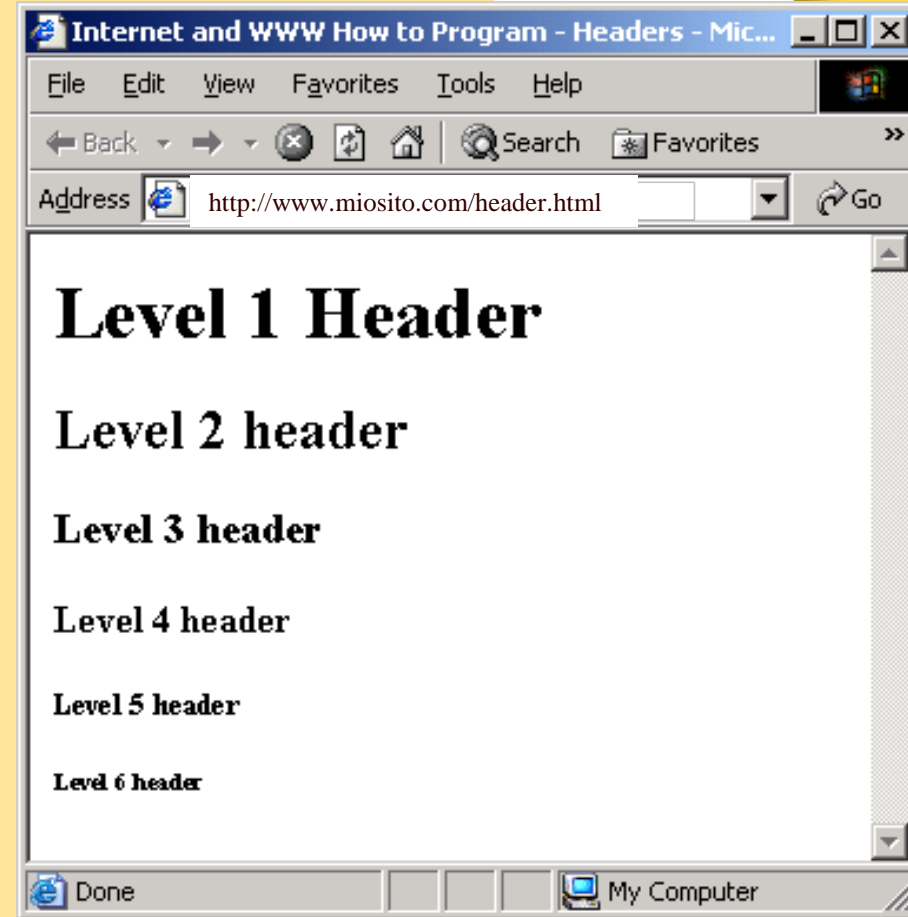




Elementi del linguaggio html

- ⇒ Sei tipi di header
 - da **h1** a **h6**

```
1 <?xml version = "1.0" ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd" >
4
5 <!-- header.html -->
6 <!-- XHTML headers-->
7
8 <html xmlns = " http://www.w3.org/1999/xhtml" >
9 <head>
10 <title>Internet and WWW How to Program - Headers</title>
11 </head>
12
13 <body>
14
15 <h1>Level 1 Header</h1>
16 <h2>Level 2 header</h2>
17 <h3>Level 3 header</h3>
18 <h4>Level 4 header</h4>
19 <h5>Level 5 header</h5>
20 <h6>Level 6 header</h6>
21
22 </body>
23 </html>
```





Hyperlinks

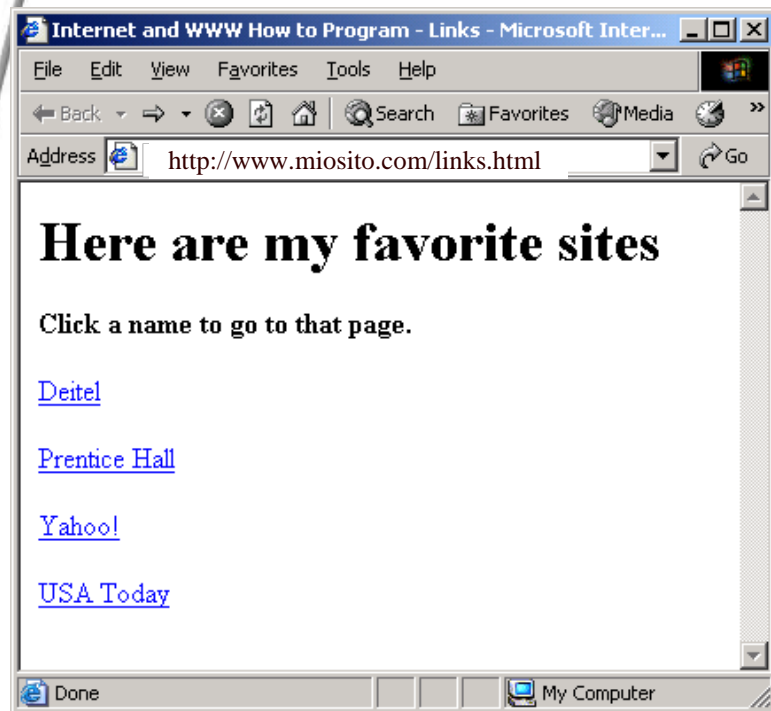
- ➔ Riferimenti ad altri file come documenti XHTML e immagini
- ➔ Sia il testo che le immagini possono costituire un hyperlink
- ➔ Vengono creati usando il tag **<a>** (anchor)
 - Attributo **href**
 - Specifica il percorso dell'oggetto del link
 - Il link può essere un indirizzo di posta elettronica, usando **mailto:**
URL

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3     "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- links.html      -->
6 <!-- Introduction to hyperlinks -->
7
8 <html >
9     <head>
10         <title>Internet and WWW How to Program - Links</title>
11     </head>
12
13     <body>
14
15         <h1>Here are my favorite sites</h1>
16
17         <p><strong>Click a name to go to that page.</strong></p>
18
19         <!-- Create four text hyperlinks -->
20         <p><a href = "http://www.deitel.com">Deitel</a></p>
21
22         <p><a href = "http://www.prenhall.com">Prentice Hall</a></p>
23
24         <p><a href = "http://www.yahoo.com">Yahoo!</a></p>
25
```

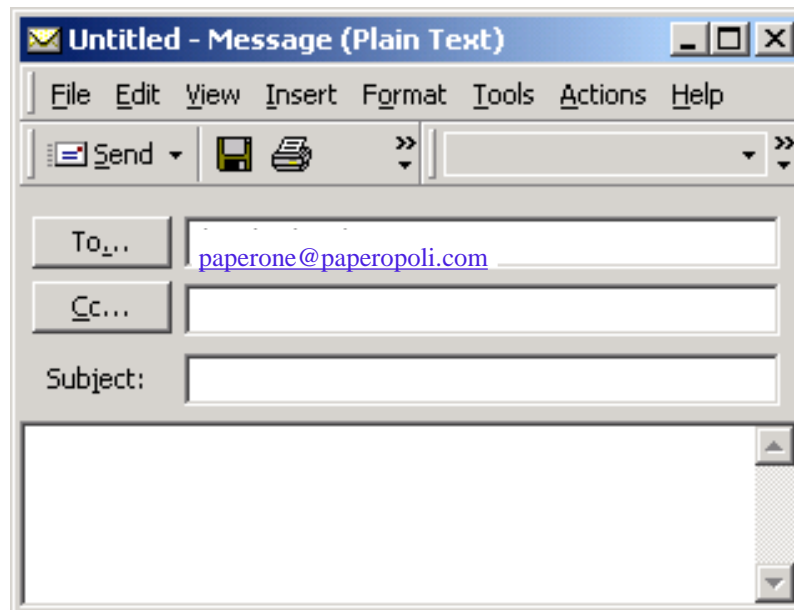
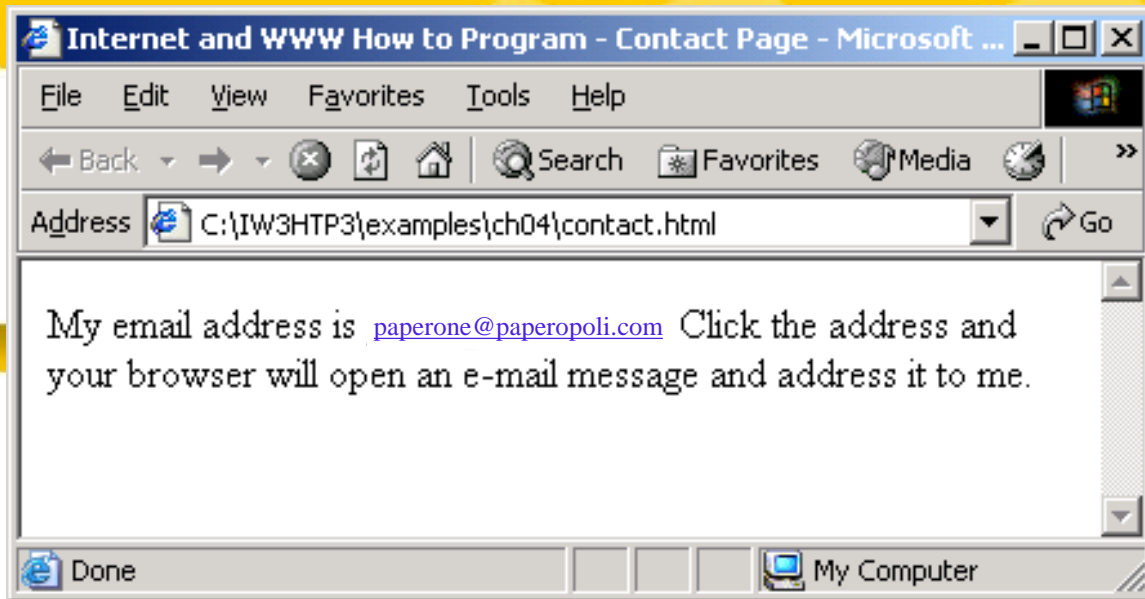
26 <p>USA Today</p>

27
28 </body>

29 </html>



```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- contact.html -->
6 <!-- Adding email hyperlinks -->
7
8 <html >
9   <head>
10     <title>Internet and WWW How to Program - Contact Page</title>
11   </head>
12
13   <body>
14
15     <p>
16       My email address is
17       <a href = "mailto:paperone@paperopoli.com">
18         paperone@paperopoli.com
19       </a>
20       Click the address and your browser will
21       open an e-mail message and address it to me.
22     </p>
23   </body>
24 </html >
```



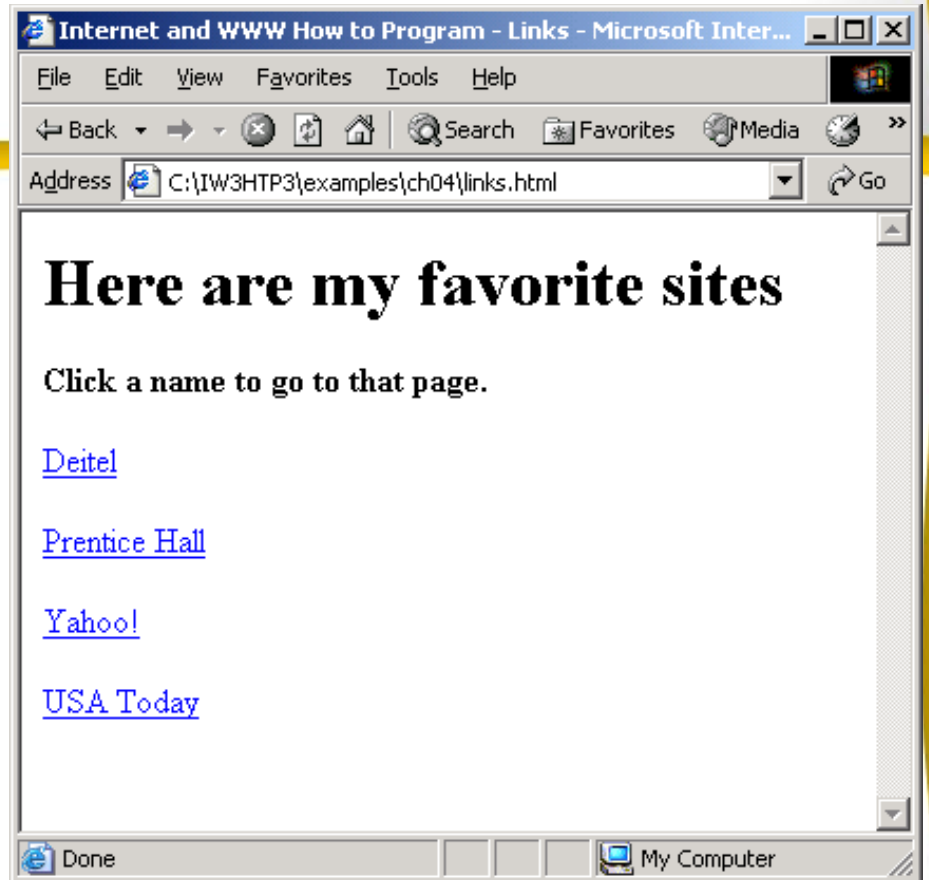


Immagini

- ⇒ Formati più comuni sul web: jpg, png, gif
- ⇒ **Si includono usando l'elemento `img`**
 - Attributo **`src`**
 - Specifica la locazione del file contenente l'immagine
 - Attributi **`width`** and **`height`**

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3     "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- nav.html          -->
6 <!-- Using images as link anchors -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10         <title>Internet and WWW How to Program - Navigation Bar
11         </title>
12     </head>
13
14     <body>
15
16         <p>
17             <a href = "links.html">
18                 <img src = "buttons/links.jpg" width = "65"
19                     height = "50" alt = "Links Page" />
20             </a><br />
21
22             <a href = "list.html">
23                 <img src = "buttons/list.jpg" width = "65"
24                     height = "50" alt = "List Example Page" />
25             </a><br />
```

```
26 <a href = "contact.html" >
27     <img src = "buttons/contact.jpg" width = "65"
28         height = "50" alt = "Contact Page" />
29 </a><br />
30
31 <a href = "header.html" >
32     <img src = "buttons/header.jpg" width = "65"
33         height = "50" alt = "Header Page" />
34 </a><br />
35
36 <a href = "table1.html" >
37     <img src = "buttons/table.jpg" width = "65"
38         height = "50" alt = "Table Page" />
39 </a><br />
40
41 <a href = "form.html" >
42     <img src = "buttons/form.jpg" width = "65"
43         height = "50" alt = "Feedback Form" />
44 </a><br />
45 </p>
46 </body>
47
48 </html>
```





Tag di formattazione del testo

- ➔ **del**
 - Testo barrato
- ➔ **sup**
 - Testo formattato come apice
- ➔ **sub**
 - Testo formattato come pedice
- ➔ **br**
 - Interruzione di linea
- ➔ **hr**
 - Linea orizzontale



Tabelle XHTML

- ➔ Si usano per organizzare dati in righe e colonne
- ➔ Tag **table** per definire una tabella
 - Attributo **border**
 - » Specifica la larghezza del bordo della tabella in pixel
 - Elemento **caption**
 - » Introduce una didascalia
- L'elemento **tr** definisce una riga della tabella
- Le celle di dati sono definite con il tag **td**



Tablelle XHTML

- Intestazione della tabella definita attraverso il tag **thead**, sezione conclusiva definita con l'elemento **tfoot**
 - L'elemento **th** (definisce le colonne della sezione di intestazione e della sezione conclusiva)
- Il corpo della tabella è definito con l'elemento **tbody**


```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- table1.html -->
6 <!-- Creating a basic table -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>A simple XHTML table</title>
11   </head>
12
13   <body>
14
15     <!-- the <table> tag opens a table -->
16     <table border = "1" width = "40%">
17
18
19
20     <!-- the <caption> tag summarizes the table's -->
21     <!-- contents (this helps the visually impaired) -->
22     <caption><strong>Price of Fruit</strong></caption>
23
```



```
24 <! -- the <thead> is the first section of a table -- >
25 <! -- it formats the table header area -- >
26 <thead>
27     <tr>                <!-- <tr> inserts a table row -- >
28         <th>Fruit </th> <!-- insert a heading cell -- >
29         <th>Price </th>
30     </tr>
31 </thead>
32
33 <! -- the <tfoot> is the last section of a table -- >
34 <! -- it formats the table footer -- >
35 <tfoot>
36     <tr>
37         <th>Total </th>
38         <th>$3.75</th>
39     </tr>
40 </tfoot>
41
42 <! -- all table content is enclosed -- >
43 <! -- within the <tbody> -- >
44 <tbody>
45     <tr>
46         <td>Apple </td> <!-- insert a data cell -- >
47         <td>$0.25</td>
48     </tr>
```



```
49     <tr>
50
51         <td> Orange</td>
52         <td>$0.50</td>
53     </tr>
54
55     <tr>
56         <td>Banana</td>
57         <td>$1.00</td>
58     </tr>
59
60     <tr>
61         <td>Pineapple </td>
62         <td>$2.00</td>
63     </tr>
64     </tbody>
65
66 </table>
67
68 </body>
69 </html>
```



A screenshot of a Microsoft Internet Explorer browser window displaying a simple XHTML table. The window title is "A simple XHTML table - Microsoft Internet Explorer". The address bar shows the local file path: "C:\IW3HTP3\examples\ch05\table1.html". The table content is as follows:

Fruit	Price
Apple	\$0.25
Orange	\$0.50
Banana	\$1.00
Pineapple	\$2.00
Total	\$3.75

The browser interface includes a menu bar (File, Edit, View, Favorites, Tools, Help), a toolbar with navigation buttons (Back, Forward, Stop, Refresh, Home, Search, Favorites, Media), and a status bar at the bottom showing "Done" and "My Computer".



Form XHTML

➔ Elemento **form**

– Attributo **method**

- Specifica come inviare i dati del form al Web Server
- **method** = “**post**”
 - I dati del form vengono inclusi nel messaggio di richiesta
- **method** = “**get**”
 - I dati del form vengono appesi alla fine dell’URL

– Attributo **action**

- Specifica l’indirizzo di destinazione della richiesta, ovvero uno script sul Web server

➔ Elemento **input**

- Specifica i dati da fornire allo script che elabora il form

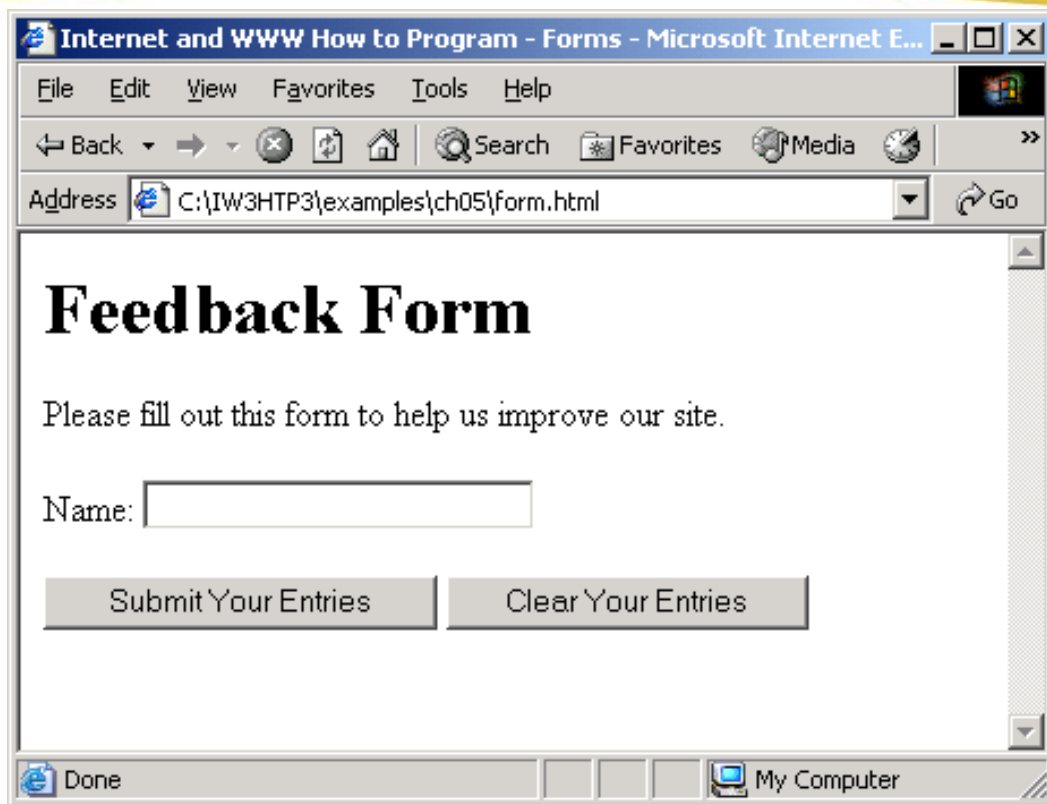
```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3     "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5
6 <!-- Form Design Example 1 -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10         <title>Internet and WWW How to Program - Forms</title>
11     </head>
12
13     <body>
14
15         <h1>Feedback Form</h1>
16
17         <p>Please fill out this form to help
18             us improve our site.</p>
19
20         <!-- this tag starts the form, gives the -->
21         <!-- method of sending information and the -->
22         <!-- location of form scripts -->
23         <form method = "post" action = "/cgi-bin/formmail">
24
```

```
25 <p>
26 <! -- hidden inputs contain non - visual -- >
27 <! -- information -- >
28 <input type = "hidden" name = "recipient"
29 value = "deitel@deitel.com" />
30 <input type = "hidden" name = "subject"
31 value = "Feedback Form" />
32 <input type = "hidden" name = "redirect"
33 value = "main.html" />
34 </p>
35
36 <!-- <input type = "text"> inserts a text box -- >
37 <p><label> Name:
38 <input name = "name" type = "text" size = "25"
39 maxlength = "30" />
40 </label></p>
41
42 <p>
43 <!-- input types "submit" and "reset" insert -- >
44 <!-- buttons for submitting and clearing the -- >
45 <!-- form's contents -- >
46 <input type = "submit" value =
47 "Submit Your Entries" />
48 <input type = "reset" value =
49 "Clear Your Entries" />
50 </ p>
```

52 </form>

54 </body>

55 </html>





XHTML Form (continua)

- Elemento **textarea**
 - Inserisce un'area di testo
 - Attributi **rows** e **cols**
- Elemento **input** di tipo **password**
 - Inserisce un'area di testo che non viene visualizzata dal browser
- Elemento **input** di tipo **checkbox** (quadrato)
 - Abilita la selezione di un elenco di opzioni (nessuna, una o più di una)
- Elemento **select**
 - Fornisce una lista “drop-down” di elementi
 - Elemento **option**
 - Definisce gli elementi ad una lista drop-down
 - Attributo **selected** specifica quale item mostrare come selezionato
- Elemento **input** di tipo **radio** (cerchietto)
 - Permette di selezionare o deselezionare un'opzione (nessuna o una)


```
1 <?xml version = "1.0" ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd" >
4
5 <!-- Fig. 5.4: form2.html -->
6 <!-- Form Design Example 2 -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title> Internet and WWW How to Program - Forms </title>
11 </head>
12
13 <body>
14
15 <h1>Feedback Form </h1>
16
17 <p>Please fill out this form to help
18 us improve our site. </p>
19
20 <form method = "post" action = "/cgi-bin/formmail" >
21
```

```
22 <p>
23     <input type =   "hidden" name = "recipient"
24         value =   "deitel@deitel.com"   />
25     <input type =   "hidden" name = "subject"
26         value =   "Feedback Form"   />
27     <input type =   "hidden" name = "redirect"
28         value =   "main.html"   />
29 </p>
30
31 <p><label> Name:
32     <input name =   "name" type = "text" size = "25" />
33 </label></p>
34
35 <!-- <textarea> creates a multiline textbox      -- >
36 <p><label> Comments:<br />
37     <textarea name =   "comments" rows = "4" cols = "36">
38 Enter your comments here.
39     </textarea>
40 </label></p>
41
```

```
42 <!-- <input type = "password"> inserts a -->
43 <!-- textbox whose display is masked with -->
44 <!-- asterisk characters -->
45 <p><label>E-mail Address:
46     <input name = "email" type = "password"
47         size = "25" />
48 </label></p>
49
50 <p>
51     <strong>Things you liked:</strong><br />
52
53     <label>Site design
54     <input name = "thingsliked" type = "checkbox"
55         value = "Design" /></label>
56
57     <label>Links
58     <input name = "thingsliked" type = "checkbox"
59         value = "Links" /></label>
60
61     <label>Ease of use
62     <input name = "thingsliked" type = "checkbox"
63         value = "Ease" /></label>
64
65     <label>Images
66     <input name = "thingsliked" type = "checkbox"
67         value = "Images" /></label>
```

Notare che (con l'input type checkbox) possono essere inviati più valori in corrispondenza dello stesso nome di parametro

```
68     <label> Source code
69     <input name = "thingsliked" type = "checkbox"
70         value = "Code" /></label>
71
72 </p>
73
74 <p>
75     <input type = "submit" value =
76     "Submit Your Entries" />
77     <input type = "reset" value =
78     "Clear Your Entries" />
79 </p>
80
81 </form>
82
83 </body>
84 </html>
```

Internet and WWW How to Program - Forms - Microsoft Internet Expl...

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media

Address C:\IW3HTP3\examples\ch05\form2.html Go

Feedback Form

Please fill out this form to help us improve our site.

Name:

Comments:

E-mail Address:

Things you liked:
Site design Links Ease of use Images Source code

Done My Computer

Internet and WWW How to Program - Forms - Microsoft Internet Expl...

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media

Address C:\IW3HTP3\examples\ch05\form2.html Go

Feedback Form

Please fill out this form to help us improve our site.

Name:

Comments:

E-mail Address:

Things you liked:
Site design Links Ease of use Images Source code

Done My Computer

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3     "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5
6
7
8 <html xml ns = "http://www.w3.org/1999/xhtml">
9     <head>
10         <title>Internet and WWW How to Program - Forms</title>
11     </head>
12
13     <body>
14
15         <h1>Feedback Form</h1>
16
17         <p>Please fill out this form to help
18             us improve our site.</p>
19
20         <form method = "post" action = "/Folder/servlet_iniziale">
21
22             <p>
23                 <input type = "hidden" name = "recipient"
24                     value = "deitel@deitel.com" />
25                 <input type = "hidden" name = "subject"
```

```
26     value = "Feedback Form" />
27     <input type = "hidden" name = "redirect"
28         value = "main.html" />
29 </p>
30
31 <p><label> Name:
32     <input name = "name" type = "text" size = "25" />
33 </label></p>
34
35 <p><label> Comments:<br />
36     <textarea name = "comments" rows = "4"
37         cols = "36"></textarea>
38 </label></p>
39
40 <p><label> E-mail Address:
41     <input name = "email" type = "password"
42         size = "25" /></label></p>
43
44 <p>
45     <strong> Things you liked: </strong><br />
46
47     <label> Site design
48         <input name = "thingsliked" type = "checkbox"
49             value = "Design" /></label>
50
```

51 <label> Links

52 <input name = "thingsliked" type = "checkbox"
53 value = "Links" /></label>

54
55 <label> Ease of use

56 <input name = "thingsliked" type = "checkbox"
57 value = "Ease" /></label>

58
59 <label> Images

60 <input name = "thingsliked" type = "checkbox"
61 value = "Images" /></label>

62
63 <label> Source code

64 <input name = "thingsliked" type = "checkbox"
65 value = "Code" /></label>

66 </p>

67
68 <!-- <input type = "radio" /> creates a radio -- >

69 <!-- button. The difference between radio buttons -- >

70 <!-- and checkboxes is that only one radio button -- >

71 <!-- in a group can be selected. -- >

72 <p>

73 How did you get to our site?:

74

75 <label> Search engine

76 <input name = "howtosite" type = "radio"
77 value = "search engine" checked = "checked" />

78 </label>

79
80 <label> Links from another site

81 <input name = "howtosite" type = "radio"
82 value = "link" /></label>

83
84 <label> Deitel.com Web site

85 <input name = "howtosite" type = "radio"
86 value = "deitel.com" /></label>

87
88 <label> Reference in a book

89 <input name = "howtosite" type = "radio"
90 value = "book" /></label>

91
92 <label> Other

93 <input name = "howtosite" type = "radio"
94 value = "other" /></label>

95
96 </p>
97

```
98 <p>
99   <label>Rate our site:
100
101   <!-- the <select> tag presents a drop-down -->
102   <!-- list with choices indicated by the -->
103   <!-- <option> tags -->
104   <select name = "rating">
105     <option selected = "selected">Amazing</option>
106     <option>10</option>
107     <option>9</option>
108     <option>8</option>
109     <option>7</option>
110     <option>6</option>
111     <option>5</option>
112     <option>4</option>
113     <option>3</option>
114     <option>2</option>
115     <option>1</option>
116     <option>Awful</option>
117   </select>
118
119   </label>
120 </p>
121
```



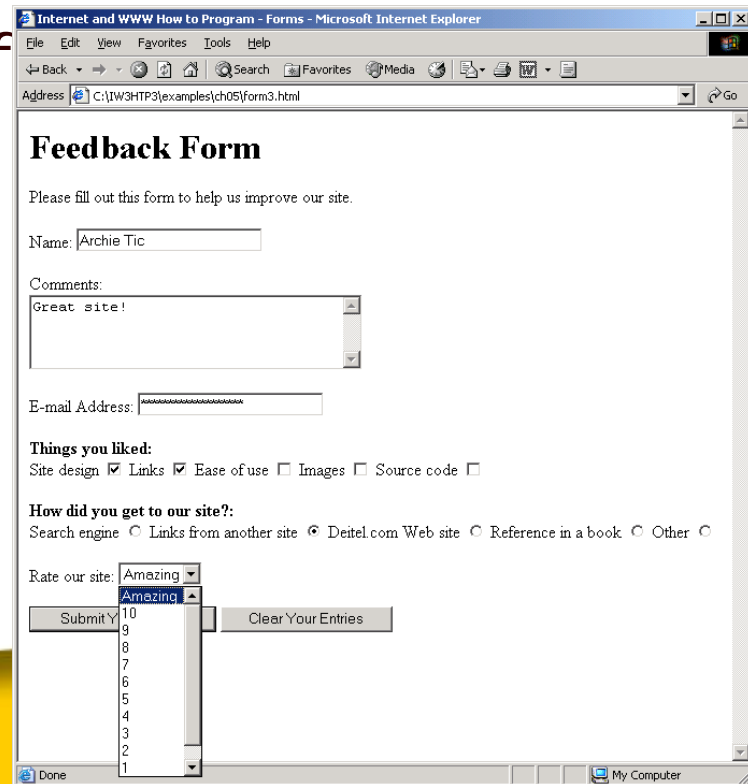
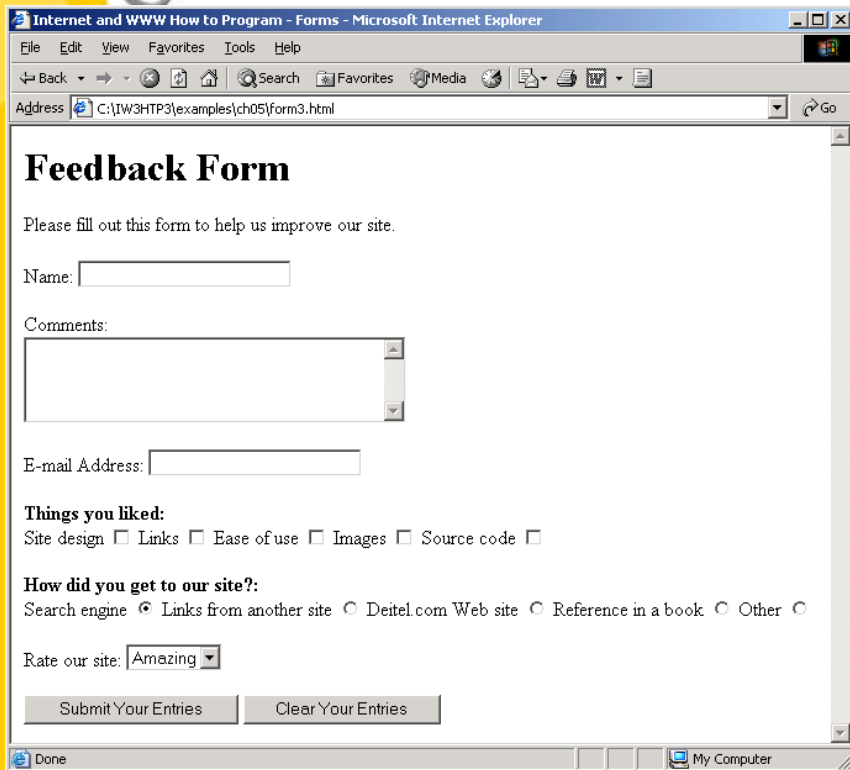
```

122 <p>
123     <input type =      "submit"  value =
124     "Submit Your Entries"      />
125     <input type =      "reset"  value = "Clear Your Entries"  />
126 </p>
127
128 </form>
129
130 </bo dy>
131 </html>

```

FORMS.HTML

of





Meta-elementi

- ➔ Specificano informazioni su un documento attraverso l'uso del tag **<meta>**
- ➔ Attributo **name**
 - Identifica il tipo di metalemento
 - **“keywords”** (**name = “keywords”**)
 - Fornisce ai motori di ricerca un elenco di parole con cui indicizzare la pagina
 - **“description”** (**name = “description”**)
 - Fornisce la descrizione del sito
- ➔ Attributo **content**
 - Definisce il contenuto del meta-tag, (es. la lista delle keywords o la descrizione)

```
1 <?xml version = "1.0" ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd" >
4
5 <!-- Fig. 5.8: main.html -- >
6 <!-- <meta> tag -- >
7
8 <html xmlns = "http://www.w3.org/1999/xhtml" >
9 <head>
10 <title> Internet and WWW How to Program - Welcome</title>
11
12 <!-- <meta> tags provide search engines with -- >
13 <!-- information used to catalog a site -- >
14 <meta name = "keywords" content = "Web page, design,
15 XHTML, tutorial, personal, help, index, form,
16 contact, feedback, list, links, frame, deitel" />
17
18 <meta name = "description" content = "This Web site will
19 help you learn the basics of XHTML and Web page design
20 through the use of interactive examples and
21 instruction." />
22
23 </head>
24
```

25 <body>

26

27 <h1>Welcome to Our Web Site!</h1>

28

29 <p>We have designed this site to teach about the wonders
30 of XHTML. XHTML is
31 better equipped than HTML to represent complex
32 data on the Internet. XHTML takes advantage of
33 XML's strict syntax to ensure well-formedness. Soon you
34 will know about many of the great new features of
35 XHTML.</p>

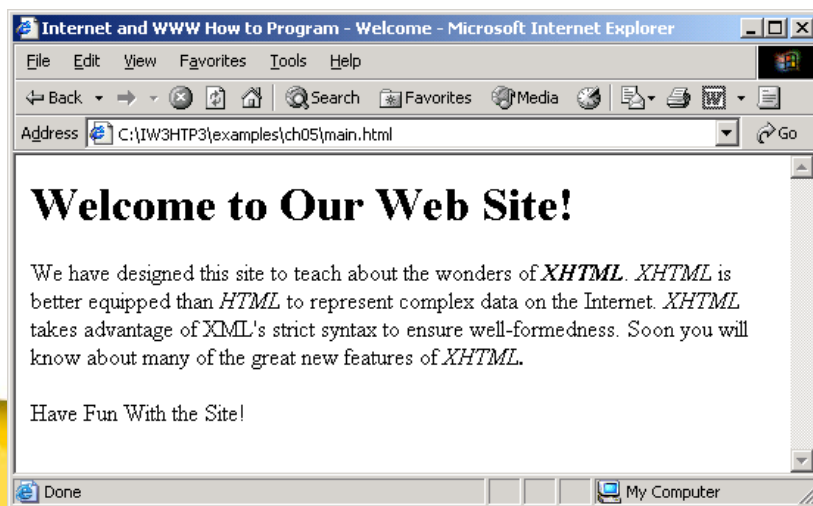
36

37 <p>Have Fun With the Site!</p>

38

39 </body>

40 </html>





Interazione Client-Server WEB

- ➔ Sempre basata sul protocollo HTTP indipendentemente dalla soluzione adottata dal lato del server
- ➔ L'URL oggetto della richiesta è usato per selezionare la risorsa lato server che si vuole usare



Protocollo HTTP

⇒ Protocollo stateless

- Ad ogni richiesta del client segue una risposta del server.
Nessuna correlazione tra richieste successive.

⇒ Quando un client invia una richiesta al server, specifica un comando

⇒ La prima linea della richiesta contiene il nome del comando, un URL e la versione del protocollo in uso:

```
GET /main.html HTTP/1.1
```




Protocollo HTTP (segue)

- ➔ Alla richiesta vengono appese informazioni opzionali
 - versione del browser,
 - i tipi di file che possono essere elaborati...

User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:12.0) Gecko/20100101 Firefox/12.0

Accept: image/gif, image/jpeg, text/*, */*



Protocollo HTTP (segue)

⇒ Il server elabora la richiesta e spedisce una risposta, specificando la versione del protocollo e uno status code (header della risposta):

HTTP/1.1 200 OK

- Altre informazioni inviate nell'header della risposta
 - Server software
 - Content-type della risposta



Protocollo HTTP: GET & POST

- ➔ GET e POST sono i comandi HTTP utilizzati più frequentemente
- ➔ Progettati inizialmente per scopi diversi
 - GETting information
 - POSTing information
- ➔ GET: informazioni utili per formulare la richiesta appese all'URL
- ➔ POST: informazioni incluse nel corpo della richiesta



Protocollo HTTP: GET

- ➔ I parametri della richiesta sono visibili
- ➔ La lunghezza della query string è limitata dal browser
 - Molti browser non consentono l'uso di query string di più di 240 caratteri
- ➔ La richiesta può essere inserita nei bookmark e ripetuta quante volte si vuole



Protocollo HTTP: **POST**

- ➔ I parametri della richiesta non sono visibili all'utente
- ➔ La quantità di dati che si possono inviare è illimitata (anche megabytes)
- ➔ Le richieste di POST non possono essere inserite nei bookmark, né spedite via email o ricaricate.



Protocollo HTTP: GET e POST

- ➔ La distinzione funzionale con cui i metodi erano stati progettati si è persa **MA** ricorda
- ➔ GET
 - parametri visibili
 - lista breve
 - inseribile nei bookmark e ripetibile
- ➔ POST:
 - parametri nascosti
 - lunghezza illimitata
 - non inseribile nei bookmark e non ripetibile
- ➔ Non usare richieste di GET per gestire ordini o aggiornamenti di un database



Caratteristiche del protocollo HTTP

E' il protocollo per il trasferimento di iper-testi (HyperText Transfer Protocol)

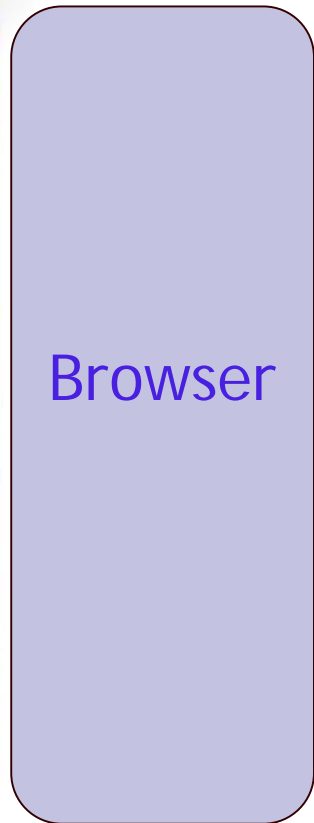
- ➔ Corrisponde al livello applicativo (stack iso/osi)
 - Presuppone un protocollo (TCP) orientato alla connessione
- ➔ Meccanismo di Richiesta/Risposta (Client/Server)
- ➔ E' possibile un trasferimento bi-direzionale di informazioni
 - il client può inviare informazioni a un server tramite un form, il server invia (di solito) pagine web al client
- ➔ Basato sul meccanismo di naming degli URI
- ➔ **Senza stato**
 - **Ogni richiesta HTTP è autonoma, il server non tiene una cronologia delle richieste**



Gestione della sessione tramite parametri hidden



Gestione della sessione tramite parametri hidden



GET /miaservlet?Nome=Giovanni

```
<html>...<form ... method="GET">
<input type="hidden" value="Giovanni" name="Nome">
<input type="text" name="cognome">
<input type="submit" value="Submit" />
</form>...</html> (*)
```

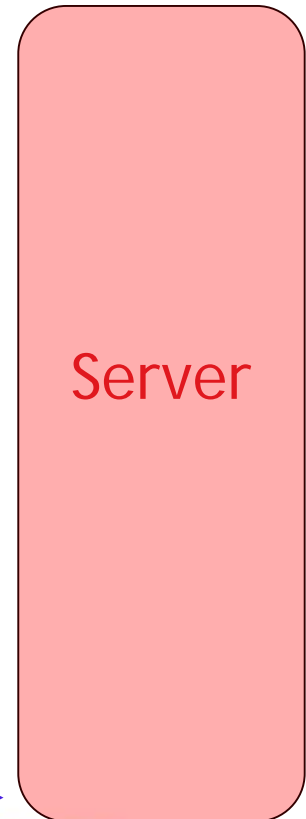
GET /miaservlet?Nome=Giovanni&Cognome=Rossi

n.b.: l'utente ha scritto solo il cognome

```
<html>...<form ... method="GET">
<input type="hidden" value="Giovanni" name="Nome">
<input type="hidden" value="Rossi" name="Cognome">
<input type="text" name="indirizzo">
<input type="submit" value="Submit" />
</form>...</html> (*)
```

GET /miaservlet?Nome=Giovanni&Cognome=Rossi
&indirizzo="Via Roma 12"

n.b.: l'utente ha scritto solo l'indirizzo



(*) Il testo della pagina html viene generato dinamicamente dalla servlet in funzione dei parametri ricevuti nella richiesta, sia che fossero hidden sia che fossero visibili nel form



Sviluppo di applicazioni di rete

- ⇒ Descrizione del paradigma client-server
- ⇒ Implicazioni del paradigma client-server nel funzionamento dei sistemi web
 - Funzionamento di un web server
 - Generazione di pagine dinamiche con tecnologie lato client e lato server



Tecnologie lato client o lato server?

lato client all'interno di Applet, o attraverso metodi di scripting dal lato client (JavaScript), o in applicazioni stand-alone.

Richiedono il supporto del client

lato server associato a Servlet, agli Enterprise JavaBean, agli Agenti, alle API ed ai servizi di Transaction Management, agli Application Server ed ai protocolli di programmazione distribuita (es. CORBA)

Non richiedono alcun supporto da parte del client



Perché soluzioni lato server

⇒ Soluzioni lato client

- problemi di prestazioni e di portabilità

⇒ Soluzioni lato server

- accesso a informazioni che sono disponibili esclusivamente dal lato del server (es. database etc.)
- minimi requisiti in termini di capacità di calcolo e storage dal lato del client (è il server a fare il grosso del lavoro)
- il client non deve avere altro che il browser per interpretare le pagine html



Altre soluzioni lato server

➔ ASP

- Consente l’inserimento di codice all’interno delle pagine HTML (HTML-embedded), che viene eseguito dal server
- E’ ottimizzato per la generazione di piccole porzioni di contenuto dinamico

➔ PHP

- Consente l’uso di codice “HTML-embedded”
- Uso di un linguaggio interamente nuovo e poca disponibilità di API rispetto a JSP e Servlet



Servlet

- ➔ Programma applicativo che viene eseguito da un server
 - Accoglie ed elabora richieste provenienti dal client (attraverso comandi http: POST o GET, ma anche attraverso altri protocolli)
 - Produce una risposta HTTP contenente codice HTML generato dinamicamente
- ➔ Molto diffuse per applicazioni che fanno uso di database
 - Thin clients (client molto semplici che richiedono supporto minimo)
 - Meccanismo request/response



Servlet: meccanismi implementati

- ➔ Identificazione della sessione, ovvero di una serie di richieste provenienti da un'unica coppia utente/browser
- ➔ Accesso alle funzioni native di autenticazione fornite dal sistema operativo
 - e a seconda dell'engine altri come...
- ➔ Clustering delle sessioni (per load balancing)
- ➔ Supporto profilazione persistente di utenti tramite JDBC
- ➔ Connettività a database



Servlet e Servlet Container

- ➔ Una servlet è una classe Java (che implementa l'interfaccia **Servlet**).
- ➔ Un servlet container può ospitare più servlet (con relativi alias).
- ➔ Quando una servlet viene invocata per la prima volta, il servlet engine genera un **thread** Java che inizializza l'oggetto Servlet.
 - Questo *persiste* per tutta la durata del processo relativo al servlet container (salvo esplicita de-allocazione).
- ➔ Ogni servlet è un thread all'interno del Servlet Container (vs CGI dove viene eseguito un processo esterno)



Possibili usi di una Servlet

- ➔ Supportare richieste multiple in modo concorrente, come per esempio conferenze on-line, servizi di comunicazione
- ➔ Aggiornare, eliminare o consultare dati contenuti in un DB remoto tramite protocollo TCP/IP
- ➔ Applicazioni basate sull'autenticazione degli utenti, gestione di sessioni di navigazione con creazione di oggetti persistenti
- ➔ Ottimizzazione delle prestazioni tramite redirectione di richieste ad altre Servlet in altri server, allo scopo di bilanciare il carico di lavoro

