



# Laboratorio di Basi di Dati

**Docente:** Alberto Belussi

*Lezione 5*

# HyperText Markup Language

- ◆ Linguaggio di descrizione di testi secondo lo schema **SGML** (Standard Generalized Markup Language)
- ◆ Gli ipertesti del Web sono scritti in HTML
- ◆ HTML **non** è un linguaggio di programmazione
- ◆ HTML **non** è "case sensitive": nei nomi dei TAG non distingue i caratteri minuscoli da quelli maiuscoli.
- ◆ HTML è un linguaggio di marcatura che permette di descrivere come il contenuto di un documento verrà presentato

# File HTML

- ◆ Un documento HTML è un file in formato testo che ha estensione `.html` o `.htm`
- ◆ Il file HTML che contiene un documento è formato dal `contenuto` del documento più la `marcatura`
- ◆ La `marcatura` descrive il modo in cui il `contenuto` verrà `presentato`

**File HTML** = `contenuto` + `marcatura`

# File HTML (2)

- ◆ I documenti HTML si possono creare con degli editor di testo.
- ◆ I browser leggono i documenti HTML e li visualizzano interpretando le specifiche di formattazione (marcatura)

# HTML: concetti generali

- ◆ La marcatura prevede l'uso di etichette dette TAGS
- ◆ I TAG racchiudono il testo di cui definiscono la formattazione

`<tag> testo </tag>`

- ◆ Il significato di un tag può essere modificato tramite attributi

`<tag attributo="valore"> testo </tag>`

# Struttura del documento

- ◆ File HTML, struttura generale:  
`<html> intestazione + corpo </html>`
- ◆ Intestazione: `<head> ... </head>`  
contiene informazioni sul documento:  
titolo `<title>... </title>`
- ◆ Corpo: `<body> ... </body>`  
contiene il testo del documento e i tag per la  
presentazione

# Struttura del documento: TAG

```
<HTML>
```

```
  <HEAD>
```

```
    <TITLE>
```

```
    </TITLE>
```

```
  </HEAD>
```

```
  <BODY>
```

```
  </BODY>
```

```
</HTML>
```

# Formattazione del testo

◆ grassetto

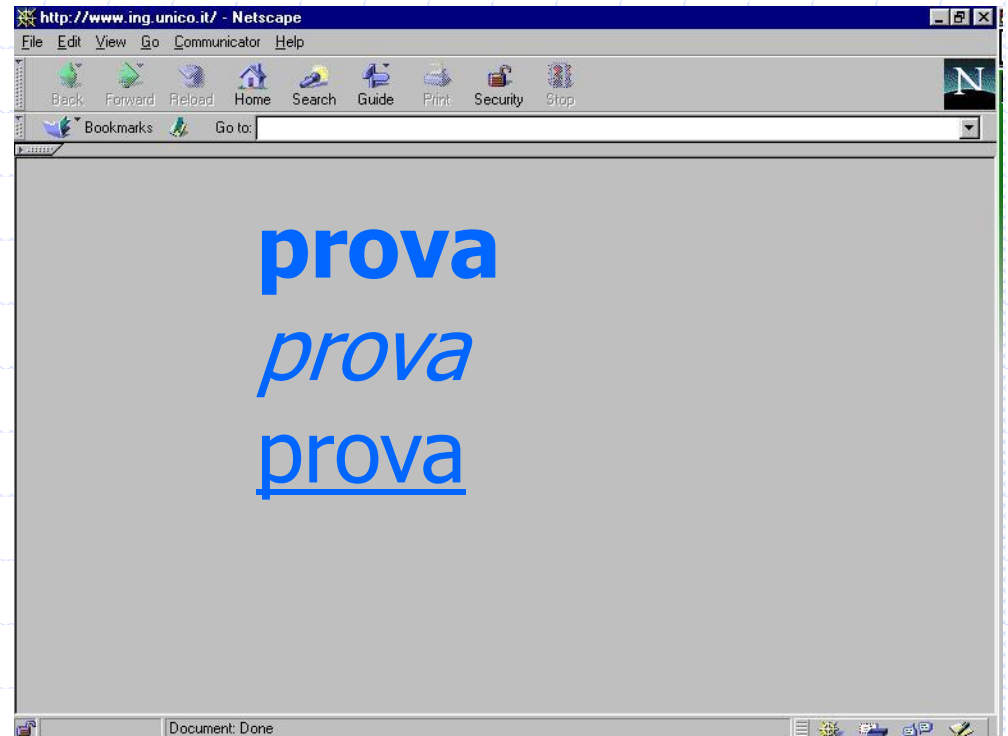
◆ corsivo

◆ sottolineato

`<b> prova </b>`

`<i> prova </i>`

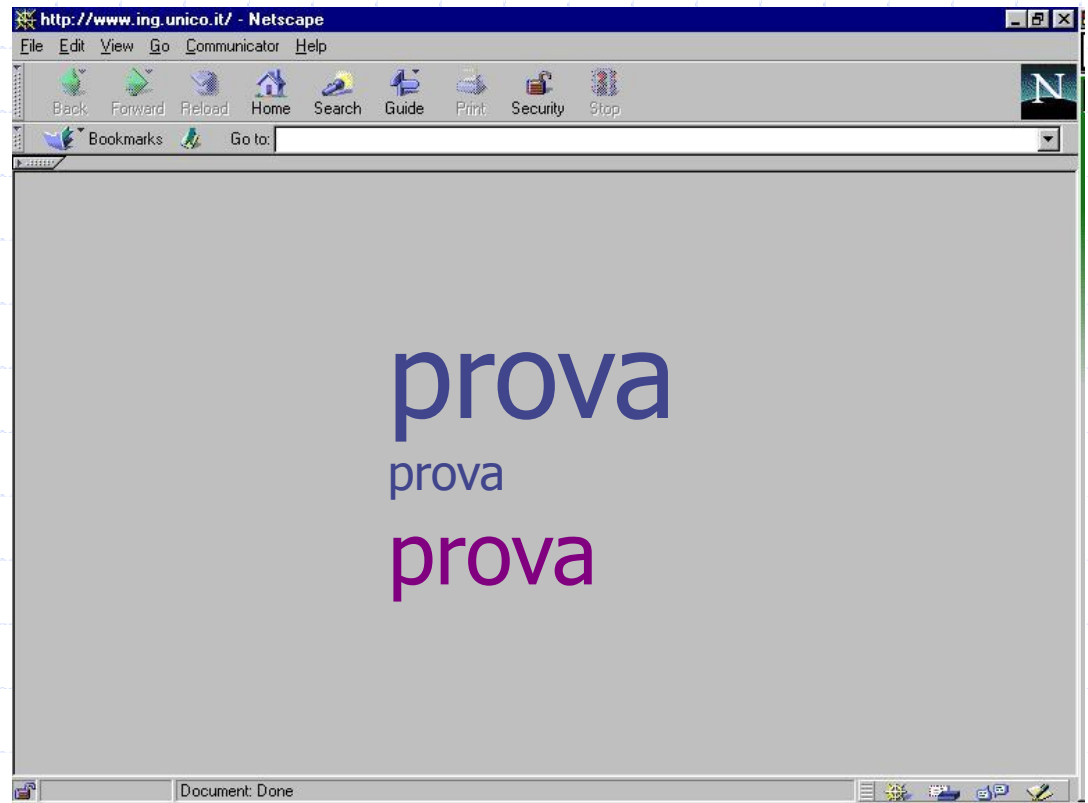
`<u> prova </u>`





# Formattazione del testo

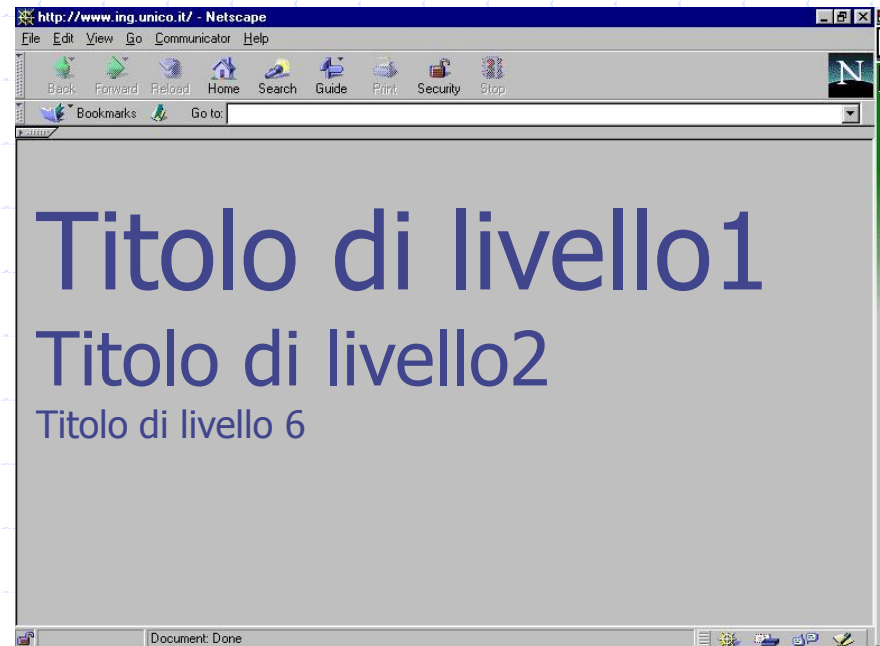
- ◆ Dimensioni: `<font size="+3"> prova</font>`  
`<font size="9"> prova</font>`
- ◆ Colore: `<font color="#800080">prova</font>`



# Titoli

◆ I livelli di titolazione sono 6:

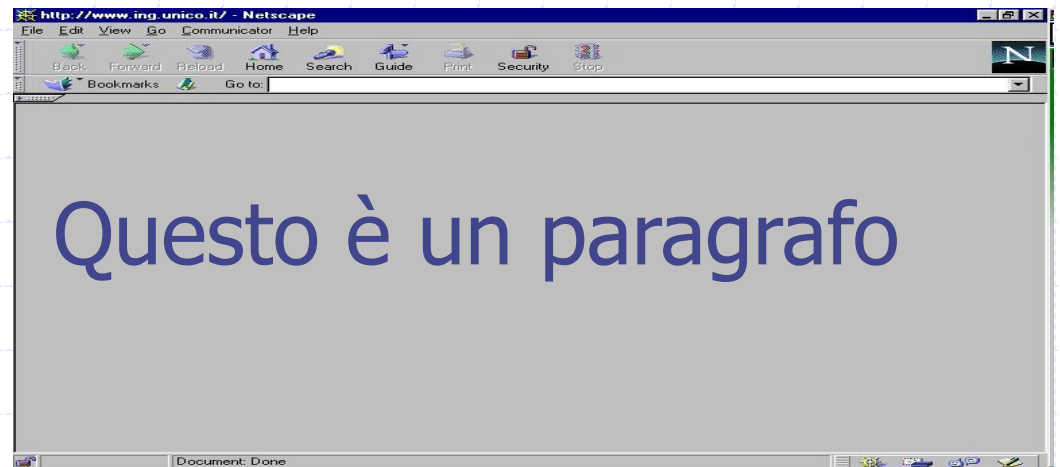
- Livello 1 (massimo) `<h1>` Titolo livello 1 `</h1>`
- Livello 2 `<h2>` Titolo livello 2 `</h2>`
- ...
- Livello 6 (minimo) `<h6>` Titolo livello 6 `</h6>`



# Paragrafi

In HTML il comando "Invio" non ha significato: il browser legge la sequenza di parole senza badare alle interruzioni di linea.

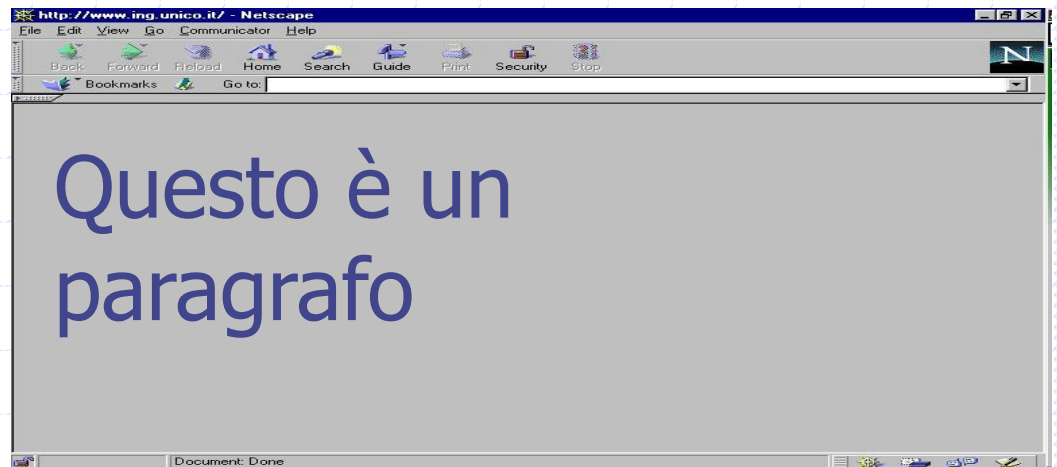
Paragrafi: `<p>`Questo è un  
paragrafo`</p>`



# Interruzione di linea

Per interrompere una linea in un punto desiderato si usa il TAG **<br>**:

**<p>**Questo è un **<br/>**paragrafo**</p>**

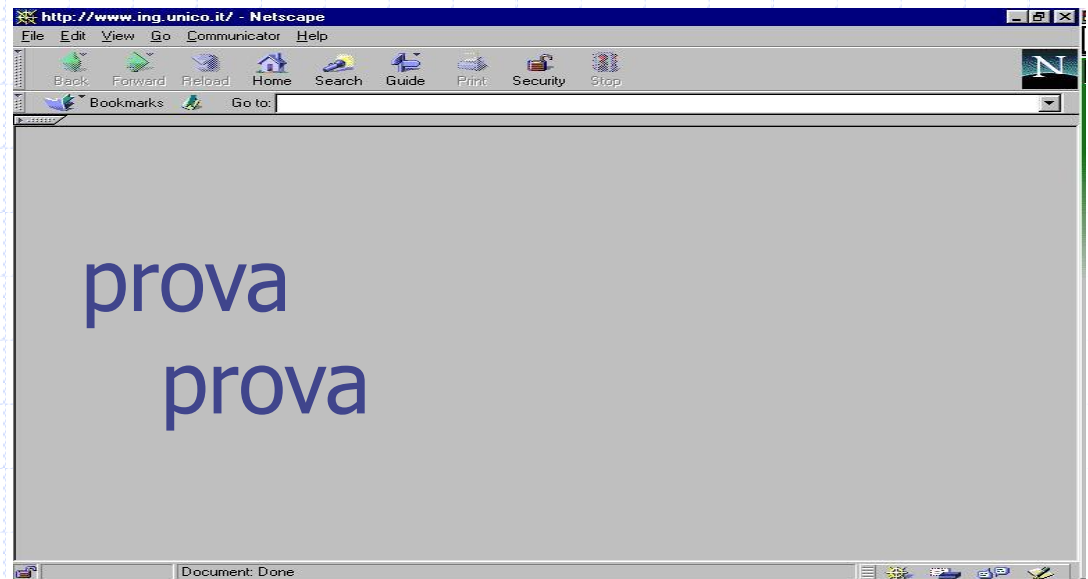


# Testo formattato

Per rendere visibili spazi aggiunti nel documento HTML ed interruzioni di linea si usa:

`<pre>prova`

`prova</pre>`



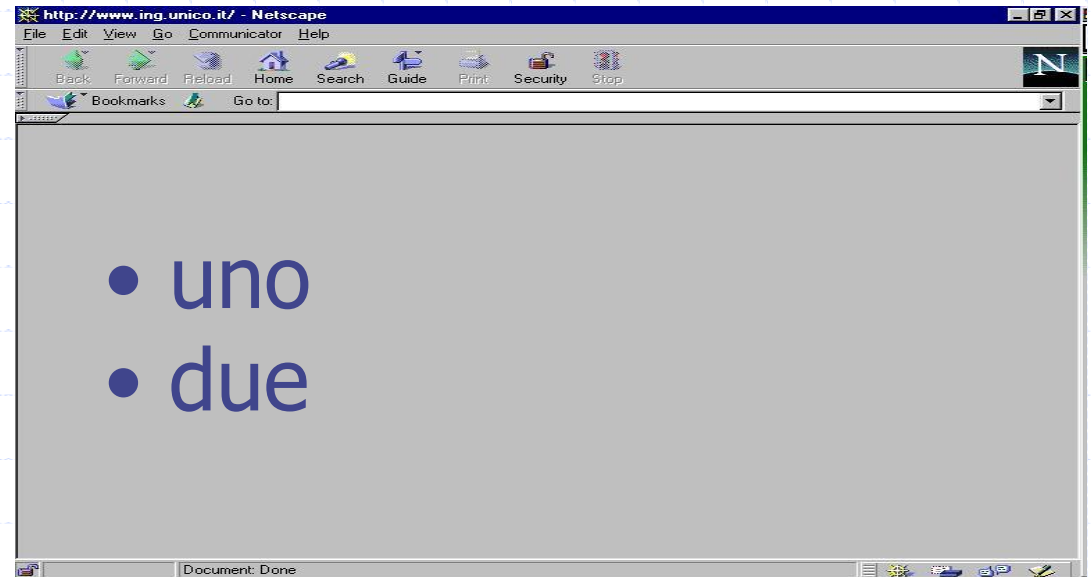
# Liste non numerate

<ul>

<li> uno </li>

<li> due </li>

</ul>



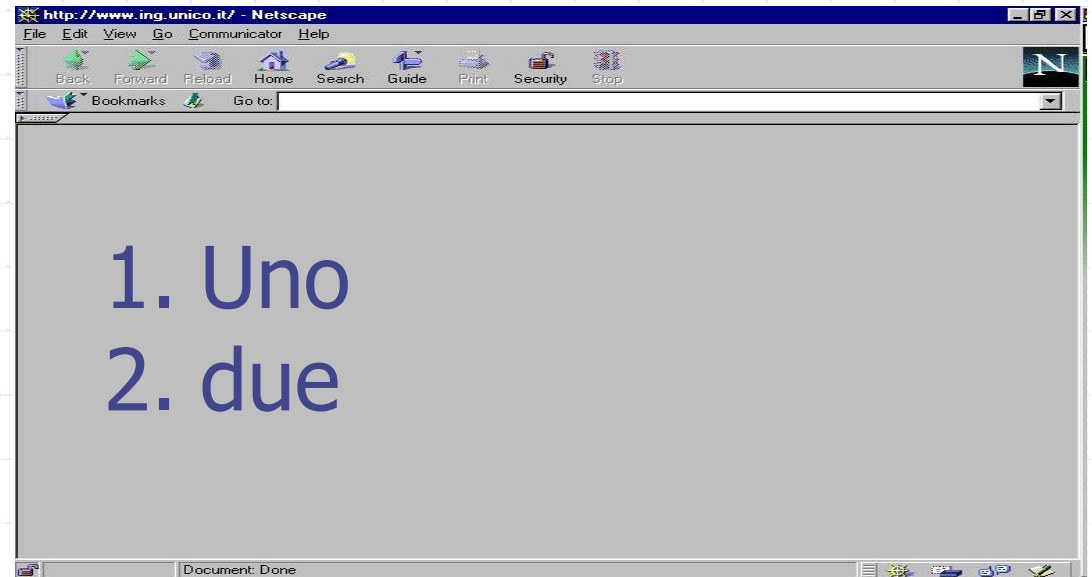
# Liste numerate

```
<ol>
```

```
<li> uno </li>
```

```
<li> due </li>
```

```
</ol>
```



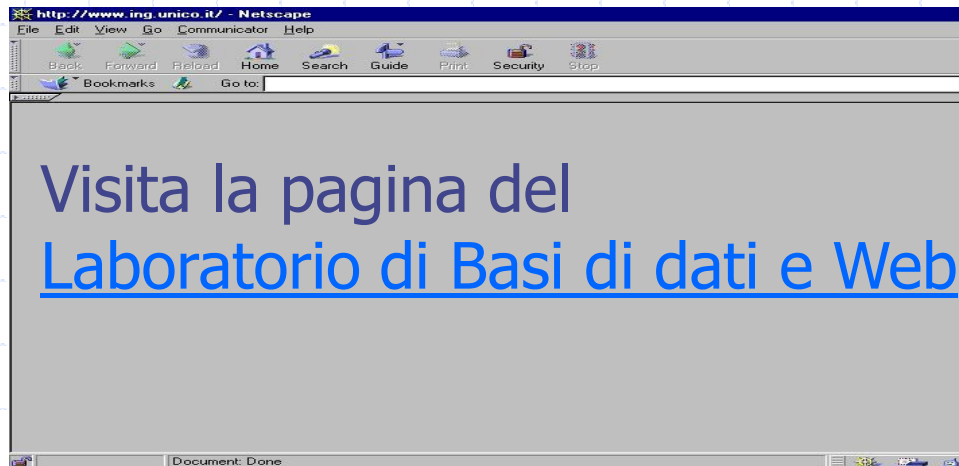
# Collegamenti ipertestuali verso altri documenti

Visita la pagina del

`<a href="http://www.scienze.univr.it/fol/main?ent=oi&cs=4&discr=&discrCd=&id=40079">`

Laboratorio di Basi di dati e Web

`</a>`



Basi di dati e web - Laboratorio (2006/2007) - Fac. Scienze matematiche fisiche e naturali - Un - Microsoft Internet Explorer

File Modifica Visualizza Preferit Strumenti ?

Indirizzo http://www.scienze.univr.it/fol/main?ent=oi&cs=4&discr=&discrCd=&id=40079

FACOLTA' DI SCIENZE MATEMATICHE, FISICHE E NATURALI  
Università degli Studi di Verona

contatti dove siamo telefono email  
cerca

Home Ateneo Home Facoltà

OFFERTA FORMATIVA

Corsi di laurea

- » Laurea in Bioinformatica
- » Laurea in Biotecnologie Agro-Industriali
- » Laurea in Informatica
- » Modaltà iscrizioni
- » Insegnamenti
  - » Calendario didattico
  - » Orario lezioni
  - » Piani didattici e Guide dello studente
  - » Calendario esami
  - » Avvisi del corso di studio e degli insegnamenti
  - » Proposte tesi e stage
  - » Organi collegiali
  - » Docenti
- » Laurea in Informatica Multimediale
- » Laurea in Matematica applicata
- » Laurea in Scienze e Tecnologie Viticole ed Enologiche
- » Corsi nella stato "a esaurimento (attivi gli anni successivi al primo)":
  - » Laurea in Tecnologie dell'Informazione: Multimedia
- » corsi di laurea specialistica

Laurea in Informatica

Basi di dati e web - Laboratorio (2006/2007)

Codice insegnamento	10099
Docente	Alberto Bellusi
crediti	2
Altri corsi di studio in cui è offerto	Laurea in Informatica Multimediale
Settore disciplinare	INF/01 - INFORMATICA
Periodo	2° Q, 3° Q

Per visualizzare la struttura dell'insegnamento a cui questo modulo appartiene, consultare » organizzazione dell'insegnamento

Orario lezioni

2° Q

Non inserito.

3° Q

Giorno	Ora	Tipo	Luogo	Note
martedì	14.30 - 17.30	esercitazione	Laboratorio didattico Delta	

Obiettivi formativi

Questo modulo dell'insegnamento di "Basi di dati e Web" si propone di introdurre lo studente ai seguenti argomenti:

- \* uso di un Database Management System (DBMS) per la creazione/gestione/interrogazione di una base di dati;
- \* realizzazione di siti web dinamici che presentano informazioni memorizzate su basi di dati.

Le tecnologie principali utilizzate nelle esercitazioni sono le servlet e le Java Server Pages (JSP), per cui si

Start Intranet locale

10.25

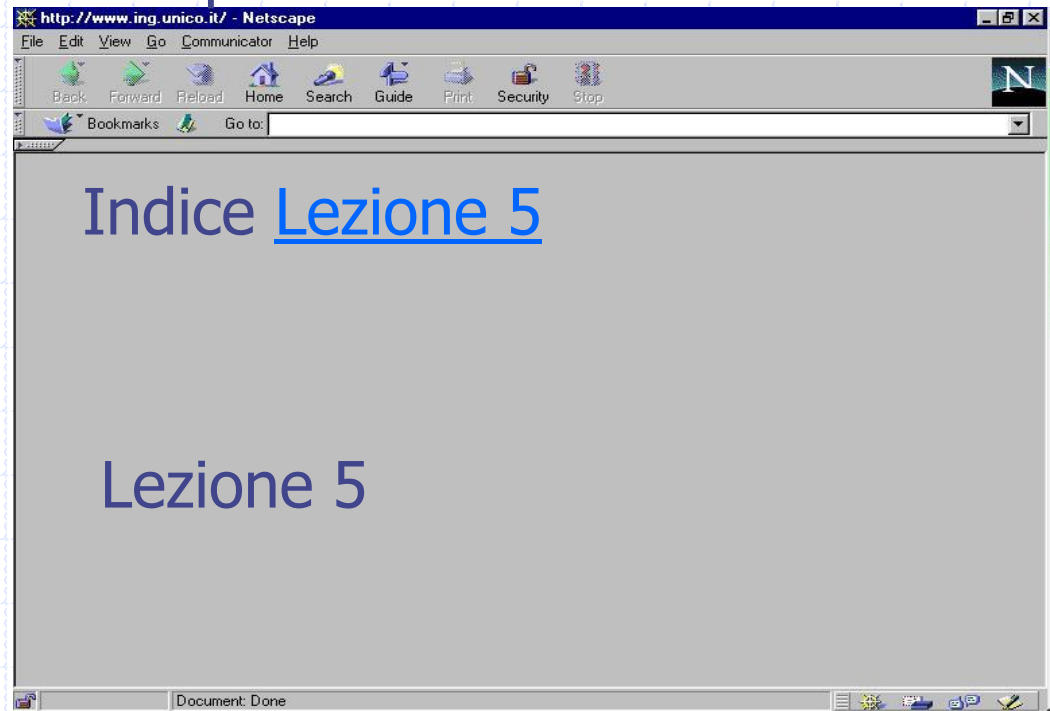


# Collegamenti ipertestuali sullo stesso documento

Indice [Lezione 5](prova.html#LEZ5)

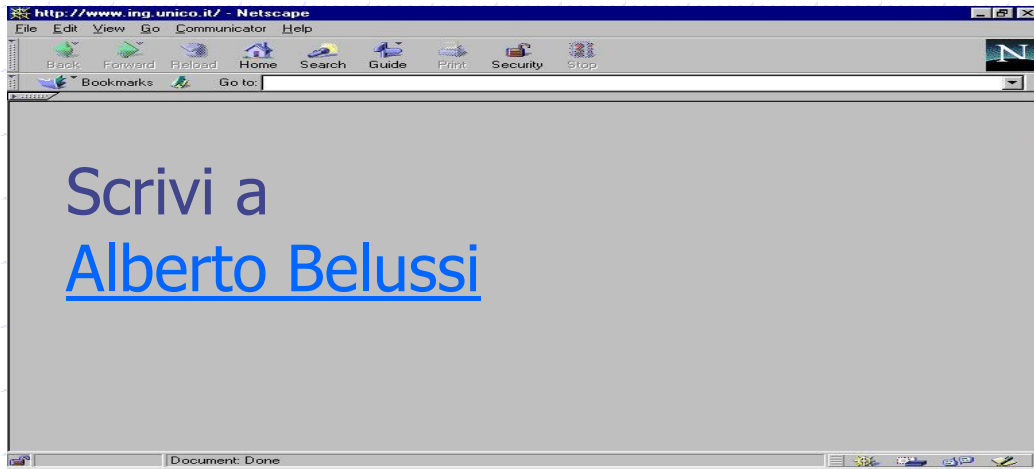
Lezione 5

File: prova.html



# Collegamenti ipertestuali

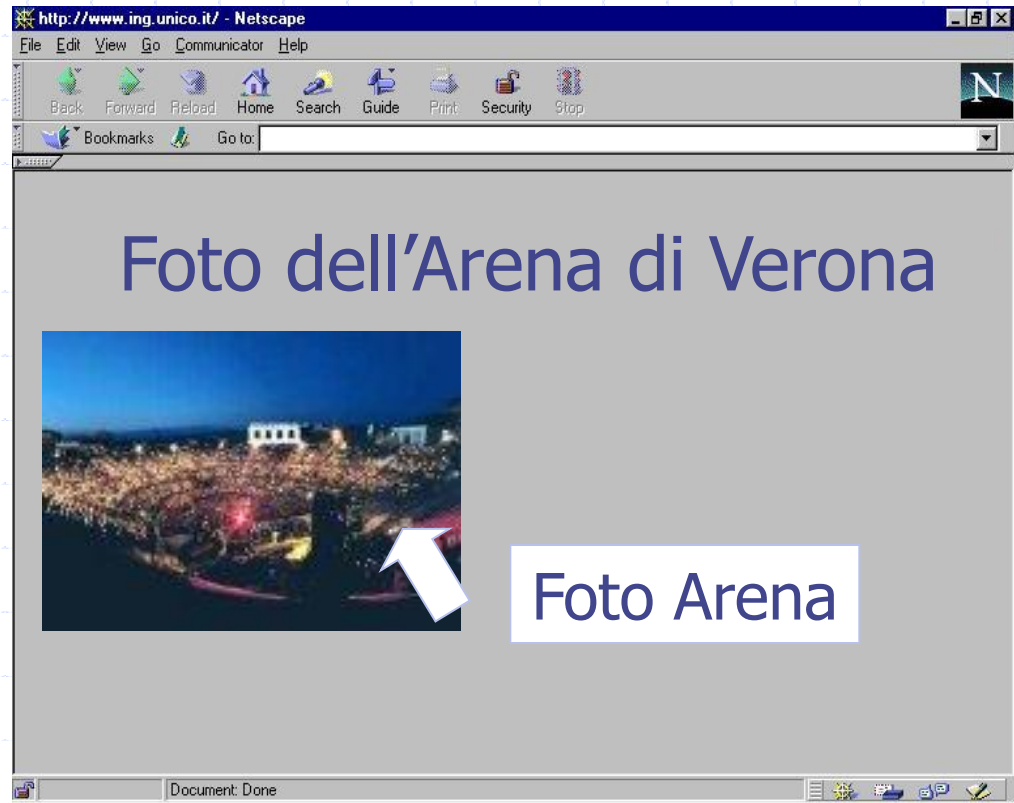
Scrivi a `<a href = "mailto:alberto.belussi@univr.it">`  
Alberto Belussi `</a>`



# Immagini

```
<p align="center" > Foto dell'Arena di Verona </p>  
<img src = "arena.jpg" width="200" height="400"  
alt="Foto Arena">
```

In numero di pixel

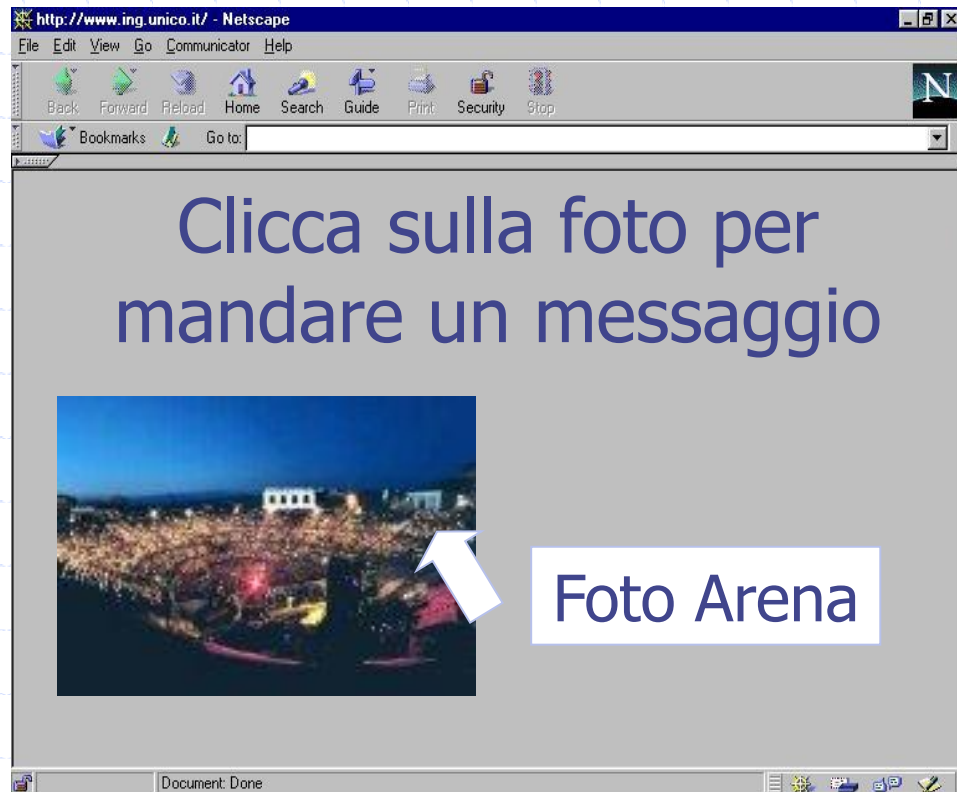


# Immagini + collegamenti

<p> Clicca sulla foto per mandare un messaggio </p>

<a href = <mailto:arena@sci.univr.it> >

<img src = "arena.gif" width="200" height="400" alt="Foto Arena">  
</a>



# Tabelle

- ◆ Per definire una tabella:

`<TABLE> ... </TABLE>`

- ◆ Per definire la didascalia della tabella (o titolo):

`<CAPTION> ... </CAPTION>`

- ◆ Per specificare una riga dentro la tabella:

`<TR> ... </TR>`

- ◆ Per definire una cella della riga di intestazione:

`<TH> ... </TH>`

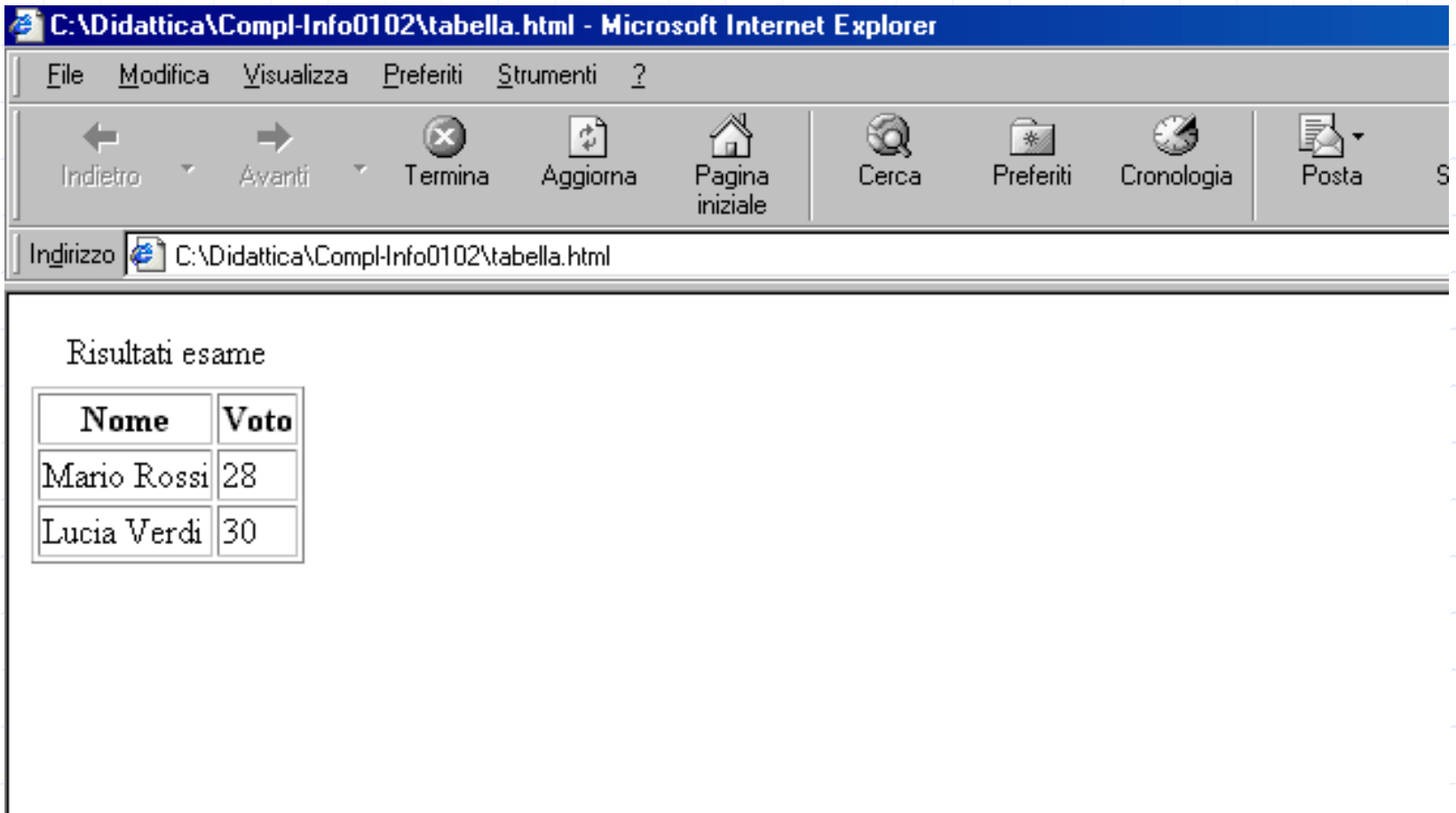
- ◆ Per definire una cella di una riga dati:

`<TD> ... </TD>`

# Tabelle: esempio 1

```
<TABLE border="1" >
  <CAPTION> Risultati esame </CAPTION>
  <TR>
    <TH>Nome</TH>
    <TH>Voto</TH>
  </TR>
  <TR>
    <TD>Mario Rossi</TD>
    <TD>28</TD>
  </TR>
  <TR>
    <TD>Lucia Verdi</TD>
    <TD>30</TD>
  </TR>
</TABLE>
```

# Tabelle: risultato esempio 1



The screenshot shows a Microsoft Internet Explorer browser window. The title bar reads "C:\Didattica\Compl-Info0102\tabella.html - Microsoft Internet Explorer". The menu bar includes "File", "Modifica", "Visualizza", "Preferiti", "Strumenti", and "?". The toolbar contains icons for "Indietro", "Avanti", "Termina", "Aggiorna", "Pagina iniziale", "Cerca", "Preferiti", "Cronologia", and "Posta". The address bar shows "Indirizzo C:\Didattica\Compl-Info0102\tabella.html". The main content area displays the text "Risultati esame" above a table.

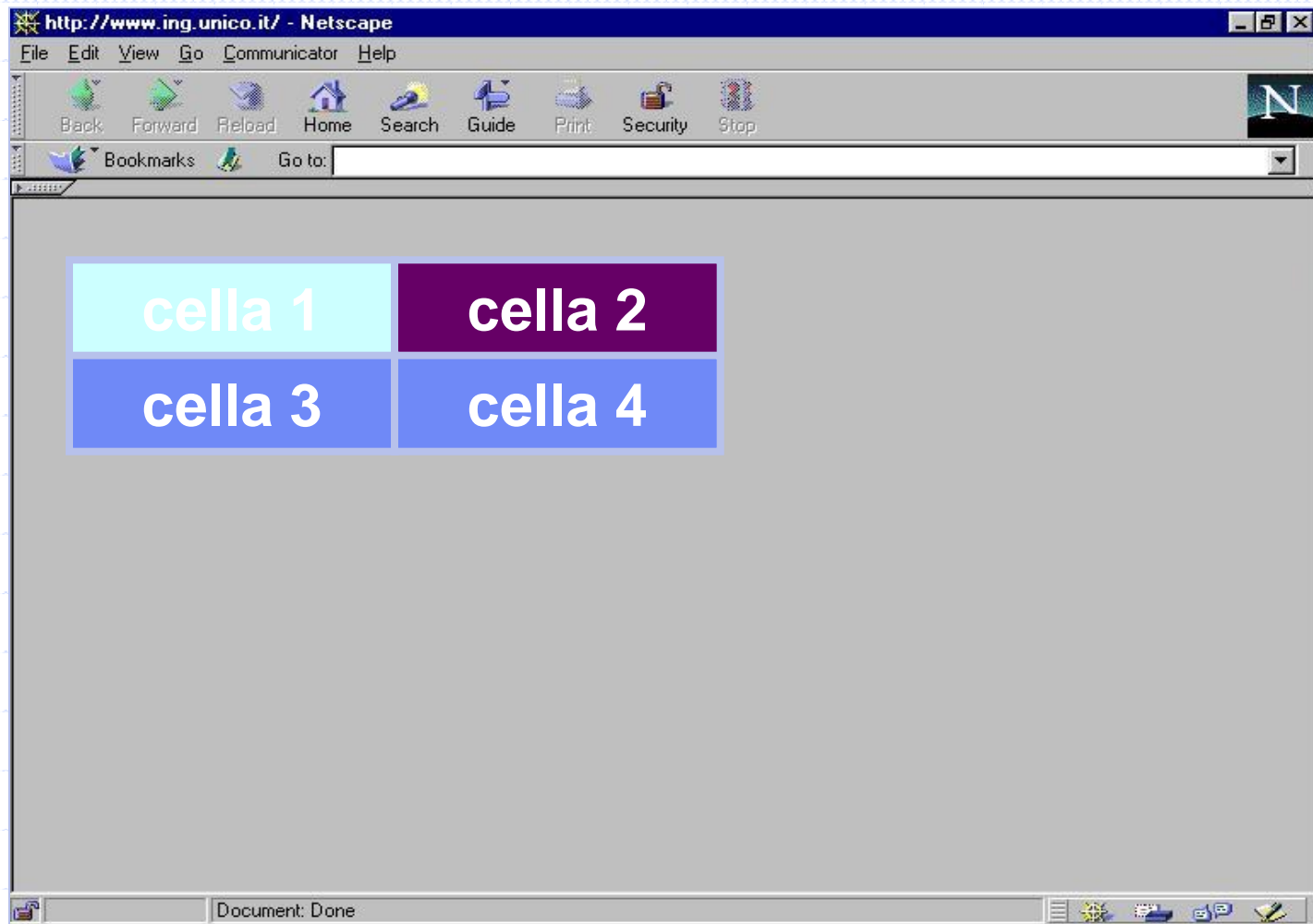
Nome	Voto
Mario Rossi	28
Lucia Verdi	30

# Tabelle: esempio 2

```
<TABLE border="1" width="50%" bgcolor="#FFFF00">
<TR>
  <TD width="50%" bgcolor="#0000FF"> cella 1</TD>
  <TD width="50%"> cella 2</TD>
</TR>
<TR bgcolor="#C0C0C0">
  <TD width="50%"> cella 3</TD>
  <TD width="50%"> cella 4</TD>
</TR>
</TABLE>
```



# Tabelle: risultato esempio 2



# Lettere accentate

- ◆ &agrave; à
- ◆ &egrave; è
- ◆ &igrave; ì
- ◆ &ograve; ò
- ◆ &ugrave; ù
- ◆ &eacute; é

- ◆ Esempio:

Il giudizio &egrave;  
pi&ugrave; che buono



Il giudizio è  
più che buono

# Formattazione via CSS

- ◆ È possibile eseguire la formattazione attraverso un foglio di stile (**C**ascading **S**tyle **S**heets - **CSS**) senza inserirla direttamente nel file HTML dove quindi rimane solo la struttura delle parti dell'ipertesto e i link che le legano tra loro.
- ◆ Le regole di formattazione espresse in un CSS possono accompagnare il file HTML oppure essere salvate in un file esterno.

# Formattazione via CSS

Un file CSS specifica come gli elementi HTML devono essere presentati facendo riferimento direttamente ad un tipo di elemento HTML

SPECIFICA INTERNA DELLE REGOLE CSS

```
<html>
<head>
<style type="text/css">
body {background-color:#f0e4fe;}
h1 {color:blue; text-align:center;}
p {font-family:"Times New Roman";
font-size:35px;}
</style>
</head>
```

SPECIFICA ESTERNA DELLE REGOLE CSS

```
<html>
<head>
<link rel="stylesheet"
type="text/css"
href="ex1.css" />
</head>
```

FILE ex1.css:

```
body {background-color:#f0e4fe;}
h1 {color:blue; text-align:center;}
p {font-family:"Times New Roman";font-
size:35px;}
```

# Formattazione via CSS

## ◆ Regola di formattazione:

```
<selector> { <declaration>; ... <declaration>; }
```

```
<selector> ::= body|p|h1|h2| ...
```

```
<declaration> ::= <property> : <value>
```

Ad esempio

```
h1 { color:blue; text-align:center;}
```

# Background Properties

Property	Descrizione
<a href="#"><u>background-attachment</u></a>	Imposta se l'immagine di background è fissa oppure scorre con il resto della pagina ( <b>fixed</b> or <b>scroll</b> )
<a href="#"><u>background-color</u></a>	Imposta un colore di background
<a href="#"><u>background-image</u></a>	Imposta un'immagine come background ( <b>url('foto.jpg')</b> );
<a href="#"><u>background-position</u></a>	Imposta la posizione di partenza di una immagine di background ( <b>left top, left center, left bottom</b> )
<a href="#"><u>background-repeat</u></a>	Imposta come un immagine di background viene ripetuta

# Text Properties

Property	Descrizione
<a href="#">color</a>	Imposta il colore del testo
<a href="#">letter-spacing</a>	Incrementa o decrementa lo spazio tra i caratteri di un testo ( <b>3px, -2px, ...</b> )
<a href="#">line-height</a>	Imposta l'altezza della riga di testo
<a href="#">text-align</a>	Imposta l'allineamento orizzontale del testo ( <b>center, justified, ...</b> )
<a href="#">text-decoration</a>	Specifica la sottolineatura del testo ( <b>overline, underline, ...</b> )
<a href="#">text-indent</a>	Specifica l'indentazione della prima riga del testo
<a href="#">text-transform</a>	Imposta il testo tutto maiuscolo/minuscolo ( <b>uppercase, lowercase, ...</b> )
<a href="#">vertical-align</a>	Imposta l'allineamento verticale del testo ( <b>top, bottom, ...</b> )
<a href="#">word-spacing</a>	Incrementa o decrementa lo spazio tra le parole di un testo ( <b>3px, -2px, ...</b> )

# Font Properties

Property	Descrizione
<a href="#"><u>font-family</u></a>	Specifica la famiglia di font del testo
<a href="#"><u>font-size</u></a>	Specifica la dimensione del testo
<a href="#"><u>font-style</u></a>	Specifica lo stile del testo (italic, normal, ...)
<a href="#"><u>font-weight</u></a>	Specifica il peso del testo (bold, normal, ...)



# List Properties

Property	Descrizione
<a href="#"><u>list-style-image</u></a>	Specifica un'immagine come simbolo per marcare gli item della lista
<a href="#"><u>list-style-position</u></a>	Specifica se i marcatori degli item della lista dovrebbero apparire all'interno o all'esterno del flusso di contenuto
<a href="#"><u>list-style-type</u></a>	Specifica il tipo di marcatore per gli item della lista (square, circle, upper-roman, lower-alpha, ...)

# List Properties

Property	Descrizione
<a href="#"><u>list-style-image</u></a>	Specifica un'immagine come simbolo per marcare gli item della lista
<a href="#"><u>list-style-position</u></a>	Specifica se i marcatori degli item della lista dovrebbero apparire all'interno o all'esterno del flusso del contenuto
<a href="#"><u>list-style-type</u></a>	Specifica il tipo di marcatore per gli item della lista (square, circle, upper-roman, lower-alpha, ...)

Esistono molte altre proprietà che si possono indicare: si veda la documentazione sui CSS disponibile in rete, ad esempio sul sito del consorzio W3C.

# Regole di formattazione associate a classi di elementi

E' possibile definire regole di formattazione non associate ad un tipo di elemento HTML, ma con un nome (identificatore) associato, che consente di usare la regola dove si vuole richiamando il nome della regola.

In questo caso definisco una CLASSE CSS.

# Classi CSS

Nelle regole che definiscono classi il selector inizia sempre con il carattere "." seguito dal nome della classe.

Esempio:

```
.center {text-align:center;}
```

oppure

```
p.center {text-align:center;}
```

in questo secondo caso indichiamo che la classe center si può usare solo per elementi di tipo p.

# Classi CSS

Definita la classe nel file css o nella sezione opportuna del file HTML:

```
.center {text-align:center;}
```

È possibile richiamare la regola di formattazione indicando nell'elemento da formattare l'attributo class con valore il nome della classe:

```
<html><head></head>  
<body><p class="center"> Questo &egrave; un paragrafo </p>  
</body></html>
```

# FORM HTML

- ◆ Una FORM consente di specificare, all'interno di un documento HTML, una sezione in cui l'utente può inserire dei dati.
- ◆ La sintassi del tag <FORM> è la seguente:

```
<form
```

```
  action = "URL"
```

```
  method = "metodo"
```

```
  enctype = "tipo-contenuti"
```

```
  accept-charset = "set-di-caratteri"
```

```
  accept = "tipi di file in upload"
```

```
  name = "nome-modulo" >
```

} Attributi usati  
raramente

```
...
```

```
</form>
```

# Attributo ACTION

- ◆ È l'unico attributo necessario.
- ◆ Nell'attributo ACTION viene specificato l'URL della risorsa che eseguirà l'elaborazione dei dati: infatti, quando i dati sono stati inseriti e l'utente seleziona il pulsante di invio, il browser Web crea una richiesta HTTP contenente tutti i dati e la invia alla risorsa specificata nell'attributo ACTION.
- ◆ La risorsa può essere ad esempio una Servlet.

# Attributo METHOD

- ◆ Viene utilizzato per indicare il tipo di richiesta:
  - GET: i valori inseriti nella FORM vengono aggiunti all'URL della richiesta HTTP sotto forma di stringa.
    - ◆ Problemi:
      - I valori introdotti sono visibili come coppie nome/valore nella riga degli indirizzi del Browser e nei file di registrazione del server Web. Quindi GET risulta inadatto per inviare dati riservati.
      - Alcuni server e browser possono prevedere restrizioni sulla lunghezza dell'URL che può essere inviata.
  - POST: i valori inseriti nella FORM vengono forniti nello stream di input.
- ◆ L'attributo METHOD è opzionale: se non viene specificato viene usato il metodo GET.



# Elementi di INPUT

- ◆ Nel corpo di una FORM vengono descritti i vari elementi di input (o campi di input).
- ◆ Per creare elementi di input vengono utilizzati quattro tipi di tag HTML:
  - `<INPUT>`: tag generico per input semplici
  - `<SELECT>` e `<OPTION>`: per creare un menu a tendina (o casella a discesa)
  - `<TEXTAREA>`: per le caselle di testo multiriga
  - `<BUTTON>`: per creare pulsanti

# <INPUT>

<INPUT

name = "*nome del campo*"

type = "[**text** | **password** | **checkbox** |  
**radio** | **hidden** | **submit** | **reset**]"

value = "*valore iniziale*"

size = "*dimensione*"

maxlength = "*numero massimo di caratteri*" />

# Attributi di <INPUT>

- ◆ name: utilizzato per assegnare un identificatore al campo
- ◆ type: indica il tipo (o controllo) dell'elemento. Se non viene specificato si presume sia TEXT
- ◆ value: può essere utilizzato per assegnare un valore iniziale all'elemento
- ◆ size: indica le dimensioni dell'elemento in pixel o in caratteri (per i campi di testo)
- ◆ maxlength: indica il numero massimo di caratteri che possono essere digitati nell'elemento

# Il controllo TEXT

- ◆ Usato per introdurre un'unica riga di input.

- ◆ Sintassi:

<INPUT

type = **"text"** name = "*nome*"

value = "*valore-iniziale*"

size = "*dimensione*"

maxlength = "*numero massimo di caratteri*" />

- ◆ Esempio.

Testo HTML:

```
<input name="testo" type="text" value="testo iniziale" size="15" />
```

Resa del browser:

# Il controllo PASSWORD

- ◆ E' una variante del controllo TEXT.
- ◆ I caratteri introdotti non vengono visualizzati, ma vengono mascherati da \*

- ◆ Sintassi:

```
<INPUT  
  type = "password" name = "nome"  
  value = "valore-iniziale"  
  size = "dimensione"  
  maxlength = "numero massimo di caratteri" />
```

- ◆ Esempio.

Testo HTML:

```
<input name="passwd" type="password" value="1g%34D9$" size="15"  
maxlength="10" />
```

Resa del browser:

# Il controllo CHECKBOX

◆ Casella di controllo che consente di presentare un'opzione che può essere vera o falsa.

◆ Sintassi:

```
<INPUT  
  type = "checkbox" name = "nome"  
  value = "valore-associato" checked />
```

◆ Esempio

Testo HTML:

```
<input type="checkbox" name="vehicle1" value="Bike" /> Ho la bicicletta<br />
```

```
<input type="checkbox" name="vehicle2" value="Car" checked /> Ho l'auto<br/>
```

Resa del browser:

<input type="checkbox"/> Ho la bicicletta
<input checked="" type="checkbox"/> Ho l'auto

# Il controllo RADIO

◆ Casella di controllo che consente di presentare un'opzione che può essere vera o falsa. Rispetto a checkbox richiede che sia specificata dall'utente una sola scelta: il funzionamento è mutuamente esclusivo. N.B.: la mutua esclusione si ottiene tra elementi con lo stesso nome.

◆ Sintassi:

```
<INPUT
```

```
  type = "radio" name = "nome"
```

```
  value = "valore-iniziale" checked />
```

◆ Esempio.

Testo HTML:

```
<input type="radio" name="sex" value="male" /> Maschio
```

```
<input type="radio" name="sex" value="female" /> Femmina
```

Resa del browser:

Maschio

Femmina

# Il controllo HIDDEN

- ◆ Campo non visualizzato.
- ◆ Utilizzato per creare un parametro di valore costante (ad esempio un codice che una Servlet potrà utilizzare come chiave per accedere ad una tabella)

- ◆ Sintassi:

```
<INPUT  
  type = "hidden" name = "nome"  
  value = "valore-iniziale" />
```

- ◆ Esempio.

Testo HTML:

```
<input name="variabile_nascosta" type="hidden" value="1234" />
```



# Il controllo SUBMIT

- ◆ Per inviare i dati inseriti al server.

- ◆ Sintassi:

```
<INPUT  
  type = "submit"  
  value = "nome-pulsante" />
```

- ◆ Esempio.

Testo HTML:

```
<input type="submit" value="Invia query" />
```

Resa del browser:



# Il controllo RESET

- ◆ Per riportare tutti i controlli al valore iniziale.

- ◆ Sintassi:

```
<INPUT  
  type = "reset"  
  value = "nome-pulsante" />
```

- ◆ Esempio.

Testo HTML:

```
<input type="reset" value="Annulla" />
```

Resa del browser:



# Attributo TYPE

Attributo Type	Resata del Browser	Testo HTML
text	<input type="text" value="testo iniziale"/>	<code>&lt;input name="testo" type="text" value="testo iniziale" size="15"/&gt;</code>
password	<input type="password" value="XXXXXXXXXX"/>	<code>&lt;input name="passwd" type="password" value="1g%34D9\$" size="15" maxlength="10"/&gt;</code>
checkbox	A <input type="checkbox"/> B <input checked="" type="checkbox"/>	<code>A&lt;input name="scelta1" type="checkbox" value="A"/&gt; B&lt;input name="scelta2" type="checkbox" value="B" checked/&gt;</code>
radio	A <input checked="" type="radio"/> B <input type="radio"/>	<code>A&lt;input name="opzione_esclusiva" type="radio" value="A" checked/&gt; B&lt;input name="opzione_esclusiva" type="radio" value="B"/&gt;</code>
hidden		<code>&lt;input name="variabile_nascosta" type="hidden" value="1234"/&gt;</code>
submit	<input type="submit" value="Invia query"/>	<code>&lt;input type="submit" value="Invia query" /&gt;</code>
reset	<input type="reset" value="Annulla"/>	<code>&lt;input type="reset" value="Annulla"/&gt;</code>

# <SELECT> e <OPTION>

```
<SELECT  
  name = "nome"  
  size = "numero di elementi visibili" [multiple]  
  <OPTION  
    value = "valore"  
    [selected] > descrizione valore  
  </OPTION>  
  ...  
</SELECT>
```

# Attributi di <SELECT> e <OPTION>

## ◆ SELECT:

- name: utilizzato per assegnare un nome al controllo
- size: indica il numero di elementi visibili contemporaneamente, ovvero l'altezza del menu.
- multiple: consente all'utente di selezionare più elementi

## ◆ OPTION:

- value: specifica il valore restituito quando viene selezionato un certo elemento.
- selected: se presente, preseleziona l'elemento.

# <SELECT>: esempio 1

Codice HTML:

```
<select name="lista">  
  <option value="blue">blue</option>  
  <option value="red" selected>rosso</option>  
  <option value="green">verde</option>  
  <option value="yellow">giallo</option>  
  <option value="white">bianco</option>  
  <option value="black">nero</option>  
</select>
```

Resa del browser:



# <SELECT>: esempio 2

Codice HTML:

```
<select name="lista" multiple>  
  <option value="blue">blue</option>  
  <option value="red" selected>rosso</option>  
  <option value="green">verde</option>  
  <option value="yellow" selected>giallo</option>  
  <option value="white">bianco</option>  
  <option value="black">nero</option>  
</select>
```

Resa del browser:



# <TEXTAREA>

<TEXTAREA

name = "*nome*"

rows = "*numero di righe*"

cols = "*numero di colonne*">

*testo*

</TEXTAREA>



# Attributi di <TEXTAREA>

- ◆ name: assegna il nome al campo
- ◆ rows: specifica il numero di righe visualizzate nella finestra del browser per l'area di testo
- ◆ cols: specifica la larghezza in caratteri dell'area di testo da visualizzare.

# <TEXTAREA>: esempio 1

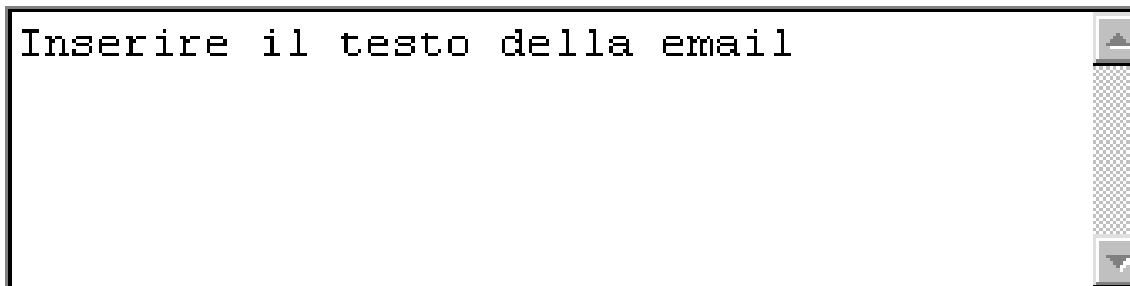
Codice HTML:

```
<textarea name="email" rows="5" cols="40">
```

Inserire il testo della email

```
</textarea>
```

Resa del browser:



Inserire il testo della email

A screenshot of a web browser window showing a text area. The text area is a rectangular box with a thin border and a vertical scrollbar on the right side. Inside the box, the text "Inserire il testo della email" is displayed in a monospaced font. The scrollbar has a small upward-pointing arrow at the top and a downward-pointing arrow at the bottom.

# Ulteriori controlli sulla form

I dati inseriti in una form dall'utente che la compila possono essere controllati direttamente dal browser (lato client) attraverso codice Javascript preparato sul lato server da chi ha implementato la form.

# Javascript

JavaScript è il linguaggio di scripting per il Web più popolare.

È un linguaggio interpretato e funziona con i principali browser presenti sul mercato: Internet Explorer, Firefox, Chrome, Opera e Safari.

# Javascript

## Cosa può fare Javascript?

- ◆ **JavaScript fornisce ai progettisti HTML uno strumento di programmazione** – I progettisti HTML non sono di solito programmatori, tuttavia Javascript è un linguaggio di scripting con una sintassi molto semplice. È quindi utilizzabile da tutti con poco sforzo.
- ◆ **JavaScript può inserire testo dinamico in una pagina HTML** – Un comando JavaScript come il seguente: `document.write("<h1>" + name + "</h1>")` può inserire il valore di una variabile di tipo testo in una pagina HTML.
- ◆ **JavaScript può reagire agli eventi** – Uno script JavaScript può essere impostato per andare in esecuzione quando accade qualcosa; tale evento scatenante può essere: la fine del caricamento della pagina nel browser, o il fatto che l'utente abbia cliccato su un elemento HTML.
- ◆ **JavaScript può leggere e scrivere elementi HTML** – Uno script JavaScript può leggere e cambiare il contenuto di un elemento HTML.
- ◆ **JavaScript può essere usato per validare i dati** – Uno script JavaScript può essere usato per validare i dati inseriti dall'utente in una form prima di inviarli al server. Questo risparmia lavoro sul server.
- ◆ **JavaScript può essere usato per identificare il tipo di browser che sta caricando la pagina HTML**
- ◆ **JavaScript può essere usato per creare e gestire cookies sul client**

# Esempi di codice Javascript

Utilizzare l'editor per testare gli script:

[http://www.w3schools.com/js/tryit.asp?filename=tryjs\\_events](http://www.w3schools.com/js/tryit.asp?filename=tryjs_events)

```
<html>
  <body>

  <h1>My First Web Page</h1>

  <script type="text/javascript">
    document.write("<p>" + Date() + "</p>");
  </script>

  </body>
</html>
```

# Esempi di codice Javascript

```
<html>
  <body>

  <h1>My First Web Page</h1>

  <p id="demo">Prova</p>

  <script type="text/javascript">
    document.getElementById("demo").innerHTML=Date();
  </script>

  </body>
</html>
```

# Javascript

JavaScript è un linguaggio ad oggetti.

Esso rappresenta tutto il documento HTML come un oggetto contenitore di altri oggetti uno per ogni elemento HTML presente nel documento.

Questa strutturazione ad oggetti del documento HTML prende il nome di:

**HTML DOM OBJECTS**



# HTML DOM Document object

L'oggetto di riferimento è l'oggetto *document* che consente di accedere e manipolare il contenuto dell'intero documento HTML.

Alcuni oggetti contenuti (collezioni di):

*document.anchors[]*: è la collezione di tutti i link del documento

Alcuni metodi:

*document.getElementById("id")*: restituisce l'elemento con l'id passato come parametro.

Per ulteriori dettagli consultare:

[http://www.w3schools.com/jsref/dom\\_obj\\_document.asp](http://www.w3schools.com/jsref/dom_obj_document.asp)

# Javascript per validare

Esempio di come strutturare il codice Javascript nel contesto di una form:

File *greeting.html*

```
<html>
  <head>
    <script type="text/javascript">
      function greeting()
      {
        alert("Welcome " + document.forms["frm1"]["fname"].value + "!")
      }
    </script>
  </head>
  <body>
    What is your name?<br />
    <form name="frm1" action="greeting.html" onsubmit="greeting()">
      <input type="text" name="fname" />
      <input type="submit" value="Submit" />
    </form>
  </body>
</html>
```

# Javascript per validare

Esempio di controllo di compilazione di un elemento:

File *NotEmpty.html*

```
<html>
  <head>
    <script type="text/javascript">
      function validateForm() {
        var x=document.forms["myForm"]["fname"].value
        if (x==null || x=="") {
          alert("First name must be filled out");
          return false;
        }
      }
    </script>
  </head>
  <body>
    <form name="myForm" action="NotEmpty.html" onsubmit="return validateForm()"
      method="post">
      First name: <input type="text" name="fname">
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

Recupero il dato

Eseguo il controllo

In caso di violazione restituisco  
false e allerto l'utente



# Fine HTML

# Il Protocollo HTTP

- ◆ HTTP (Hypertext Transfer Protocol) è il “linguaggio” utilizzato per controllare l’invio di documenti HTML via Internet.
- ◆ Il protocollo HTTP prescrive le regole mediante le quali i browser effettuano le richieste e i server forniscono le relative risposte.
- ◆ Documentazione: RFC 2616 (<http://www.w3.org/Protocols/rfc2616/rfc2616.html>) versione aggiornata delle specifiche del protocollo HTTP versione 1.1.

# La richiesta HTTP

- ◆ HTTP è un protocollo *senza stati* a richieste e risposte.
- ◆ Senza stati significa che il server Web non ricorda nulla delle richieste pervenute in precedenza dallo stesso client: il protocollo considera semplicemente la richiesta attuale di un documento e la risposta costituita dal documento stesso.

# La richiesta HTTP (2)

## ◆ Operazioni di base:

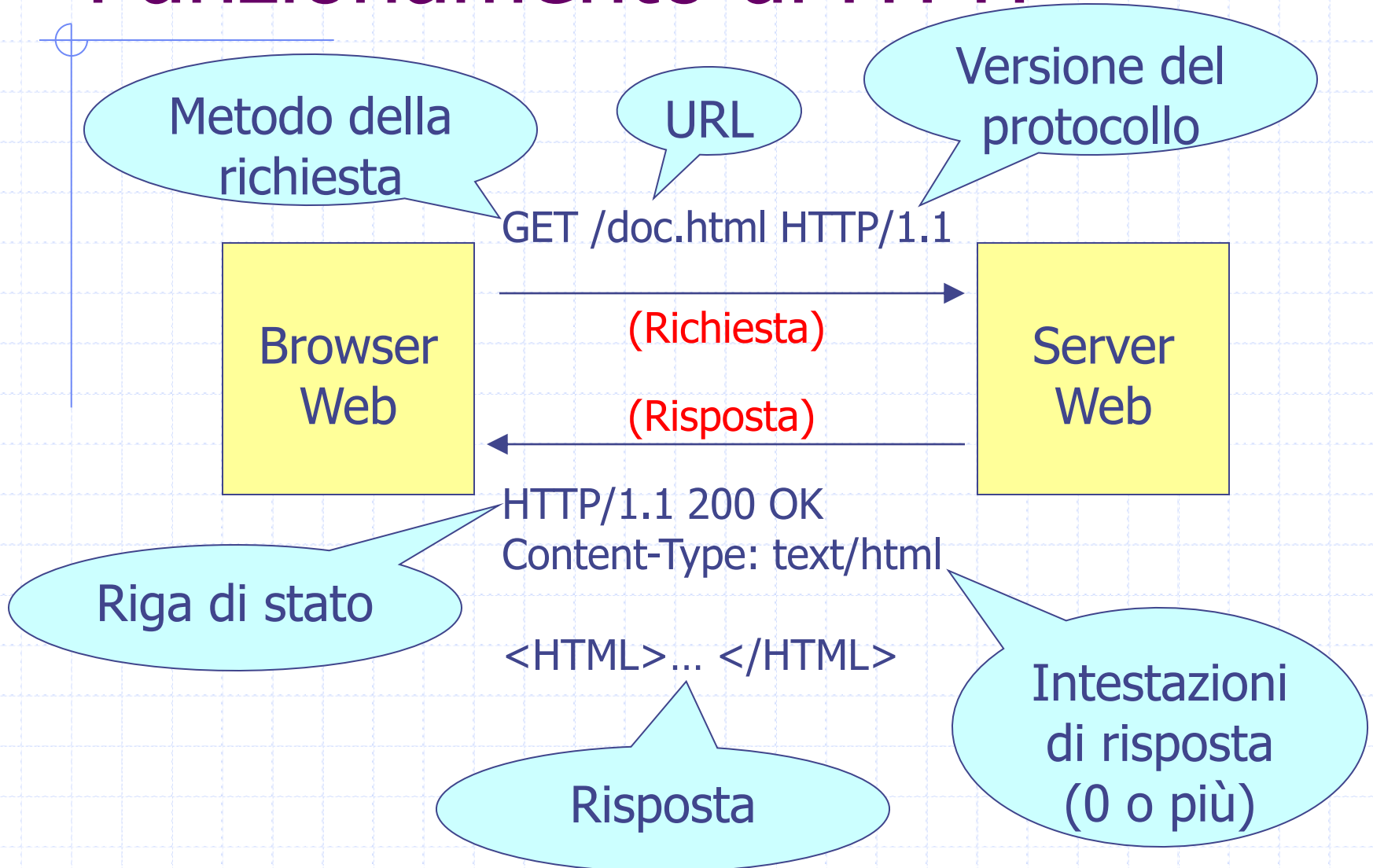
1. Un'applicazione client (browser Web) apre una connessione verso la porta HTTP del server Web (normalmente la porta 80).
2. Il client invia una richiesta attraverso la connessione aperta.
3. Il server Web analizza la richiesta ed individua la risorsa specificata.
4. Il server invia una copia della risorsa.
5. Il server chiude la connessione.

# Connessione al Server Web

- ◆ Normalmente un server Web riceve le richieste sulla porta 80, in questo caso l'indirizzo <http://www.scienze.univr.it/fol/main?ent=oi&cs=4&discr=&discrCd=&id=28310> fa riferimento al documento `main?ent=oi&cs=4&discr=&discrCd=&id=28310` sul server Web in esecuzione sull'host [www.scienze.univr.it](http://www.scienze.univr.it) e operante sulla porta standard 80.
- ◆ Se invece il server Web utilizzasse la porta 8080, l'indirizzo dovrebbe essere: <http://www.scienze.univr.it:8080/fol/main?ent=oi&cs=4&discr=&discrCd=&id=28310>



# Funzionamento di HTTP



# Esempio

- ◆ Sulla riga di indirizzo del browser viene digitato `http://www.scienze.univr.it/fol/main?ent=oi&cs=4&discr=&discrCd=&id=40079`
- ◆ Il browser web apre una connessione sulla porta 80 del server web [www.scienze.univr.it](http://www.scienze.univr.it)
- ◆ Il browser web invia la riga `GET fol/main? ent=oi&cs=4&discr=&discrCd=&id=40079 HTTP/1.0` seguita da una riga vuota

# Esempio (2)

◆ Il server web restituisce la risposta:

HTTP/1.1 200 ok

Date: Mon, 31 Mar 2003 14:27:43 GMT

...

Content-Length: 1619

Content-Type: text/html

<HTML>

<HEAD>

<TITLE> </TITLE>

</HEAD>

<BODY >

...

</BODY>

</HTML>

# Esempio (3)

- ◆ Il browser analizza la riga di stato e trova il codice di stato **200 ok** che indica che la richiesta ha avuto successo.
- ◆ Il browser analizza le intestazioni di risposta che indicano che verranno inviati 1619 byte di codice HTML.
- ◆ Il browser legge il codice HTML e visualizza il risultato.
- ◆ Se il codice HTML contiene riferimenti ad altre risorse che devono essere caricate con il documento, allora il browser invia una richiesta per ogni risorsa necessaria.

# Consegna Esercitazione 5

Inviare via email al docente una cartella zippata contenenti i file HTML e i fogli di stile CSS richiesti nell'esercitazione.

**ES5-<Matricola>.zip**

Il messaggio dovrà soddisfare il seguente formato:

- Oggetto: <Matricola> - Esercitazione 5
- Contenuto: <Matricola> - <Cognome> - <Nome>
- Allegato: file di nome ES5-<Matricola>.zip

Il messaggio email va spedito entro le 23.59 del giorno 22 aprile 2013.