Laboratorio di Basi di Dati e Multimedia

Laurea in Tecnologie dell'Informazione: Multimedia

Docente: Alessandra Di Pierro
Email: dipierro@sci.univr.it

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": non distingue i caratteri minuscoli da quelli maiuscoli all'interno dei TAG
- 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
  - Se si usa `Word` si deve salvare il documento (con estensione .html) con l'opzione "*solo testo con interruzione di riga*"
- 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 TAG
- 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>
```

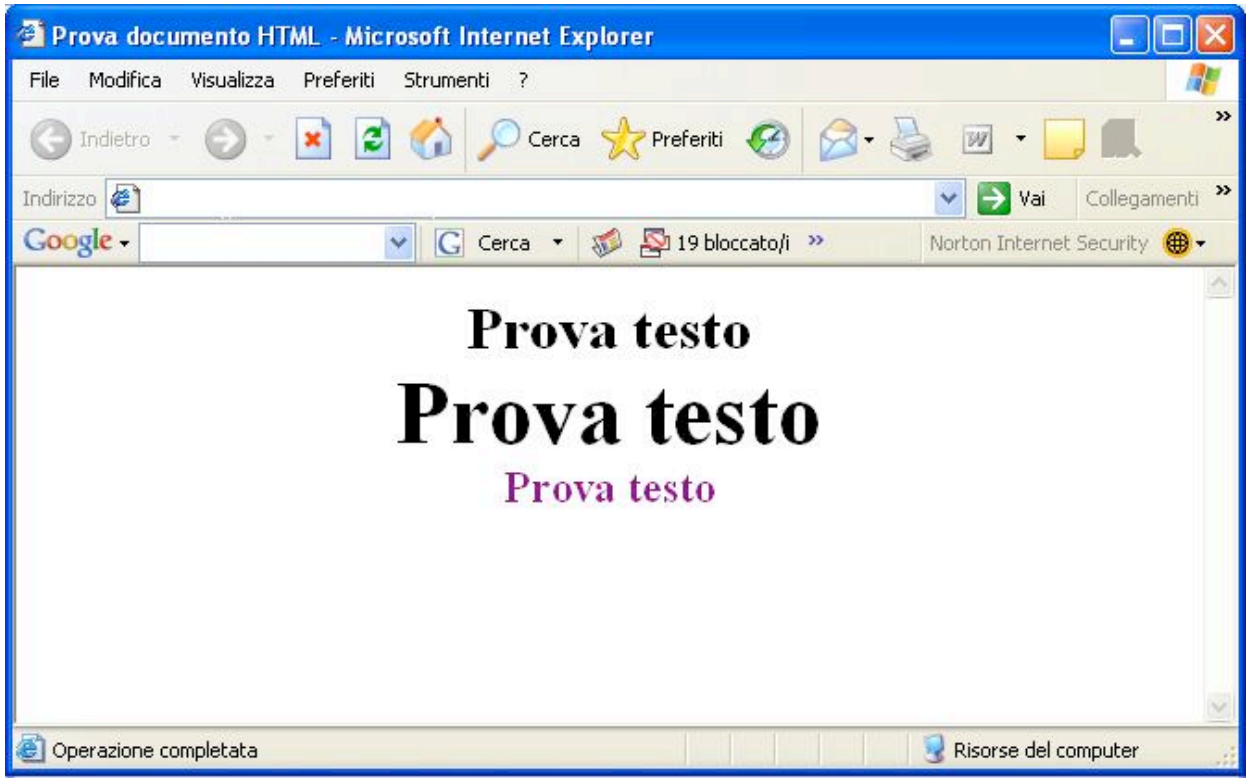
Formattazione del testo

- ✓ grassetto — Grassetto
- ✓ corsivo — Corsivo
- ✓ sottolineato — <u> Sottolineato </u>

Formattazione del testo

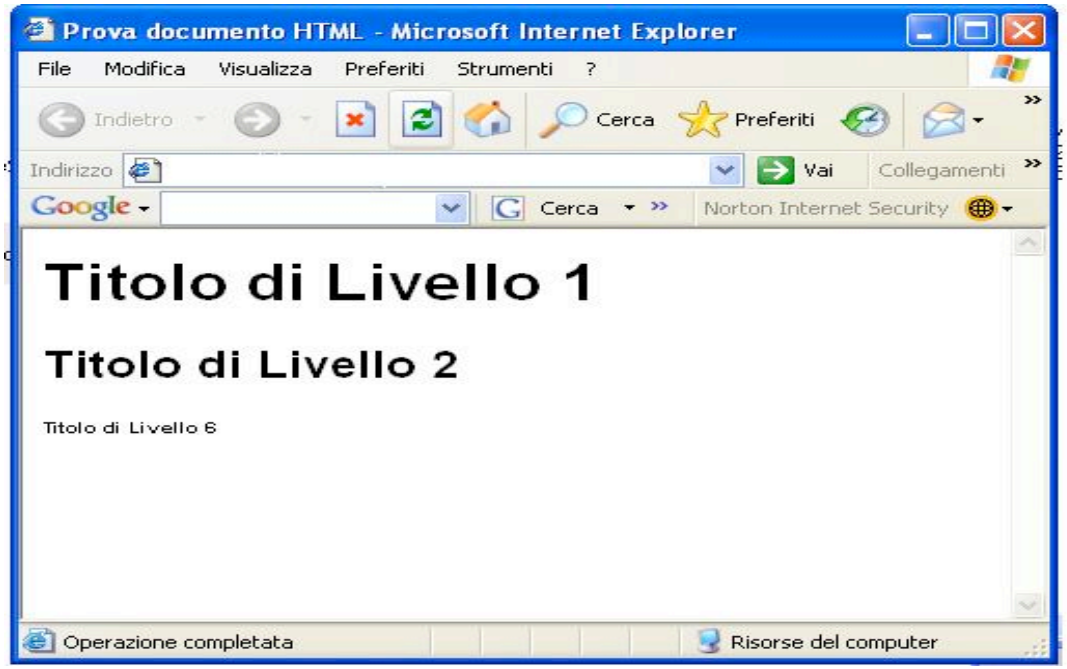
✓Dimensioni: Prova testo
 Prova testo

✓Colore: Prova testo



Titoli

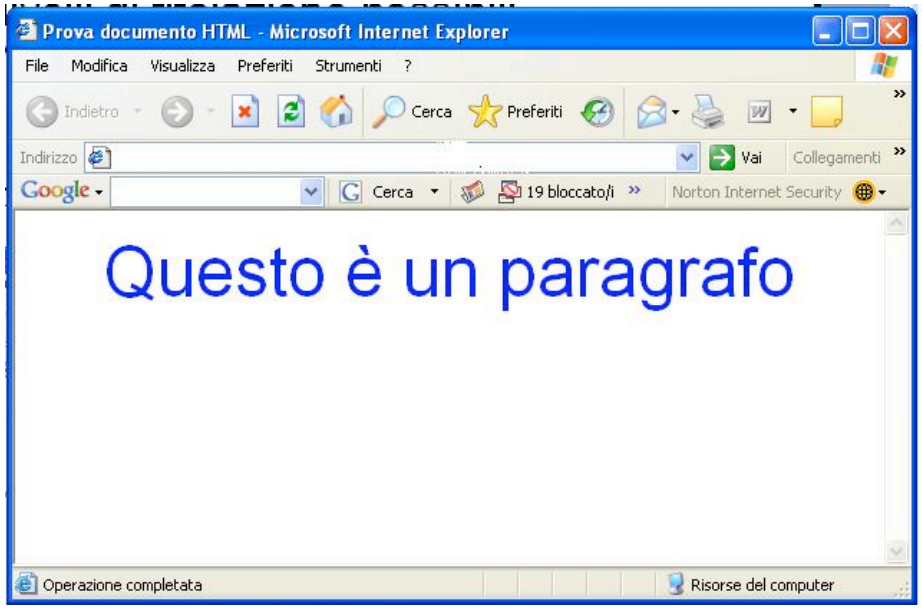
- ✓ Ci sono 6 livelli di titolazione possibili
  - ✓ Livello 1 (massimo) <h1> Titolo livello 1 </h1>
  - ✓ Livello 2 <h2> Titolo livello 2 </h2>
  - ✓ …
  - ✓ Livello 6 (minimo) <h6> Titolo livello 6 </h6>



Paragrafi

- ✓ In HTML il comando “invio” non ha significato
- ✓ Il browser legge la sequenza di parole senza considerare le interruzioni di linea
- ✓ Per definire un paragrafo:

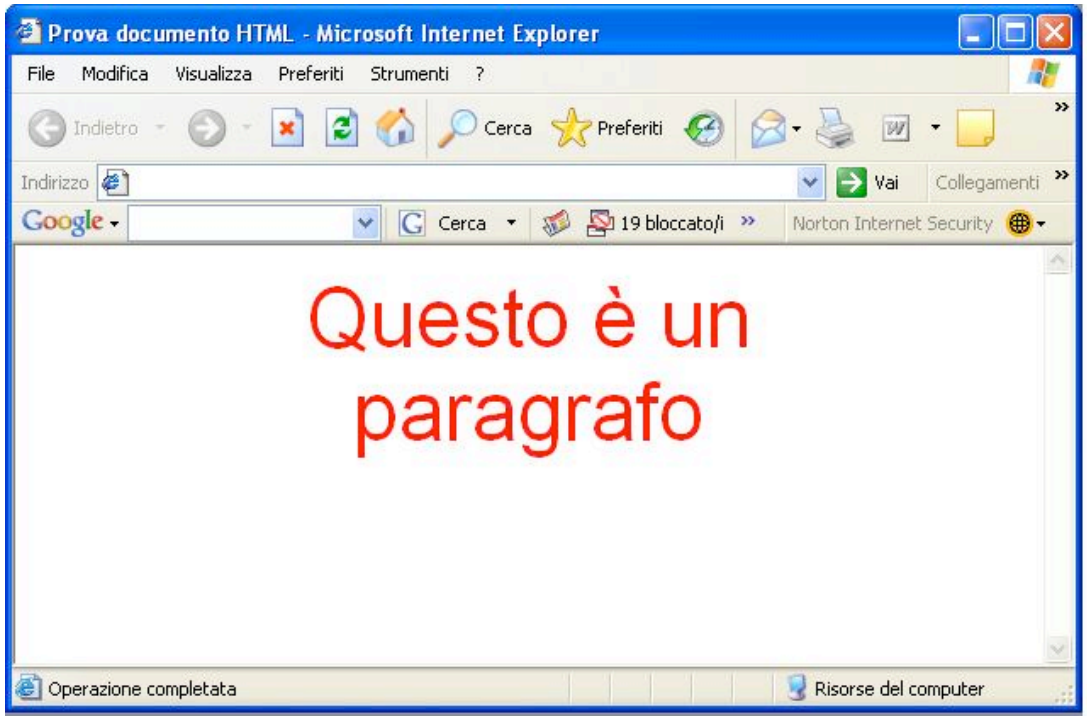
 Questo è un paragrafo



Interruzioni di linea

- ✓ Per interrompere una linea in un punto desiderato si usa il tag

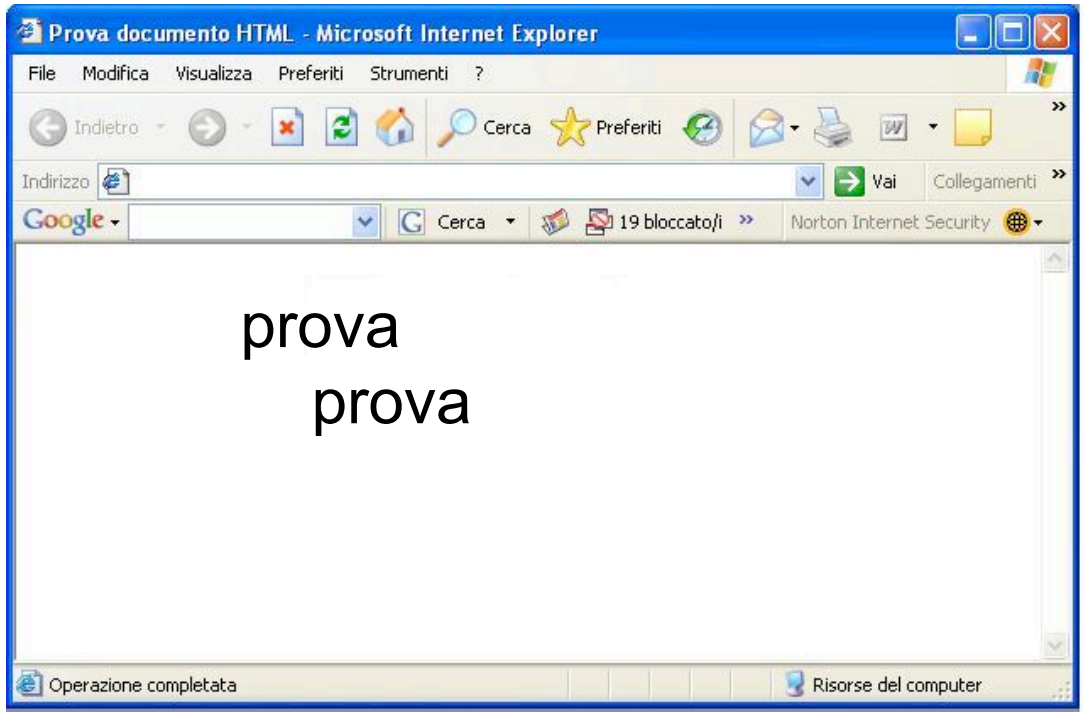
 Questo è un <br> paragrafo



Testo formattato

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

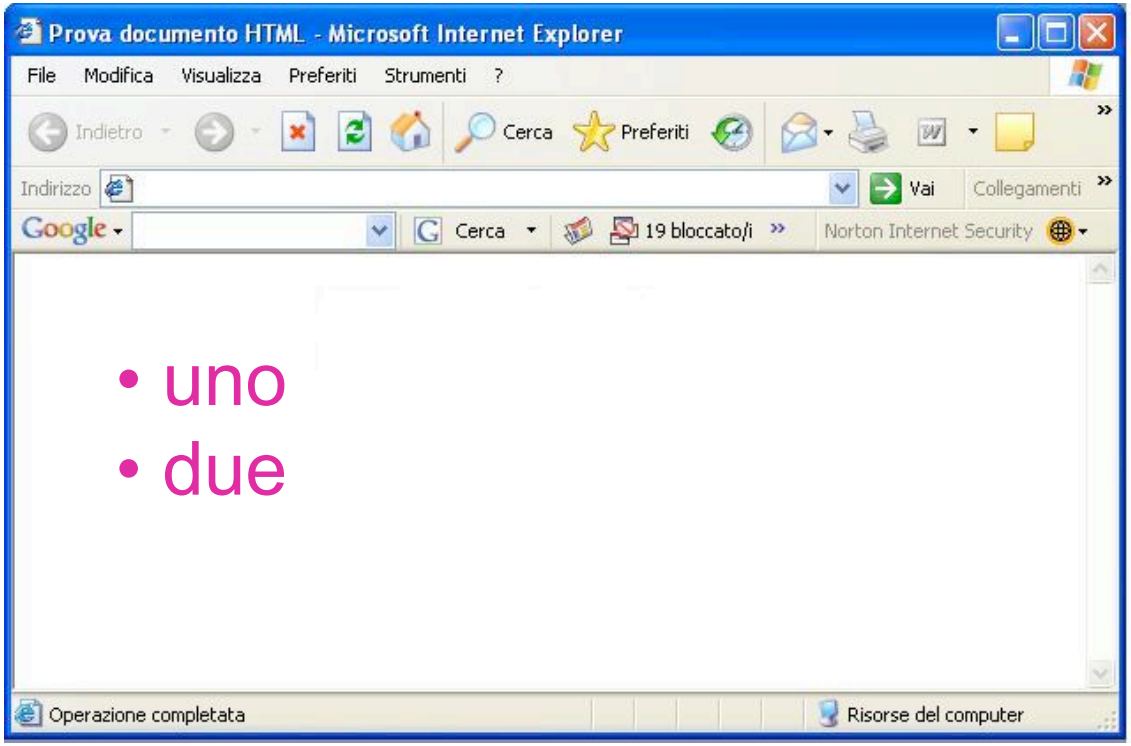
```
<pre> prova
                prova</pre>
```



Liste non numerate

✓ Sono definiti mediante i seguenti tag:

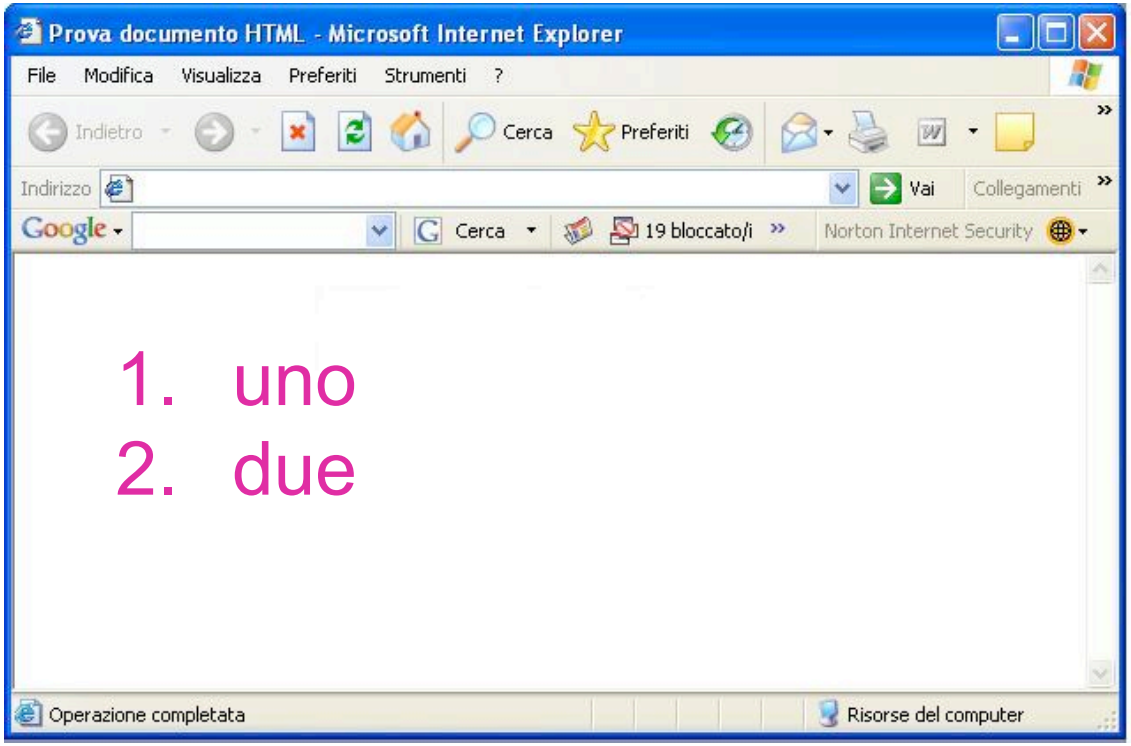
```
<ul>
    <li> uno </li>
    <li> due </li>
</ul>
```



Liste numerate

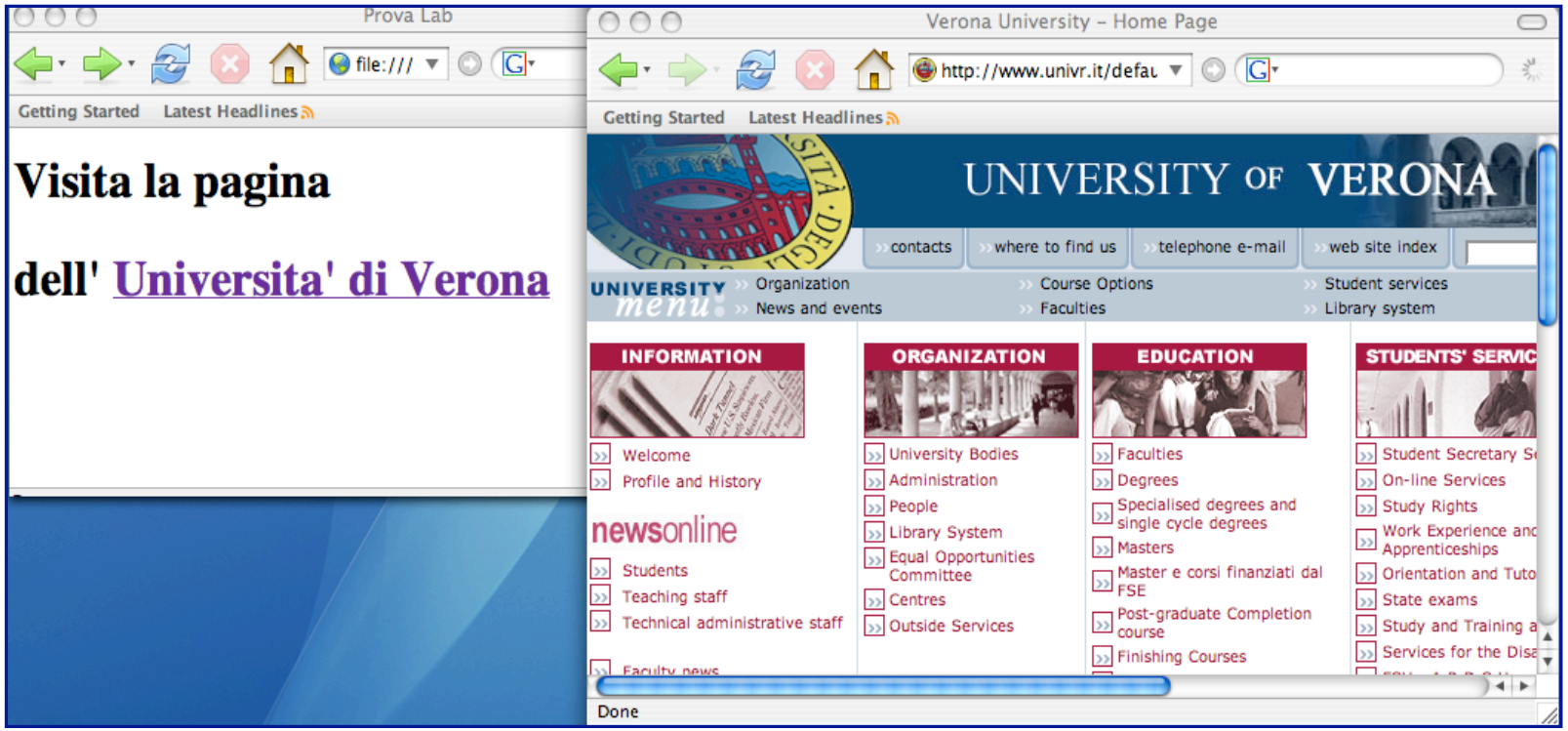
✓ Sono definite mediante i seguenti tag:

```
<ol>
    <li> uno </li>
    <li> due </li>
</ol>
```



Collegamenti Ipertestuali verso altri documenti

- Visita la pagina http://www.univr.it dell'Universita' di Verona

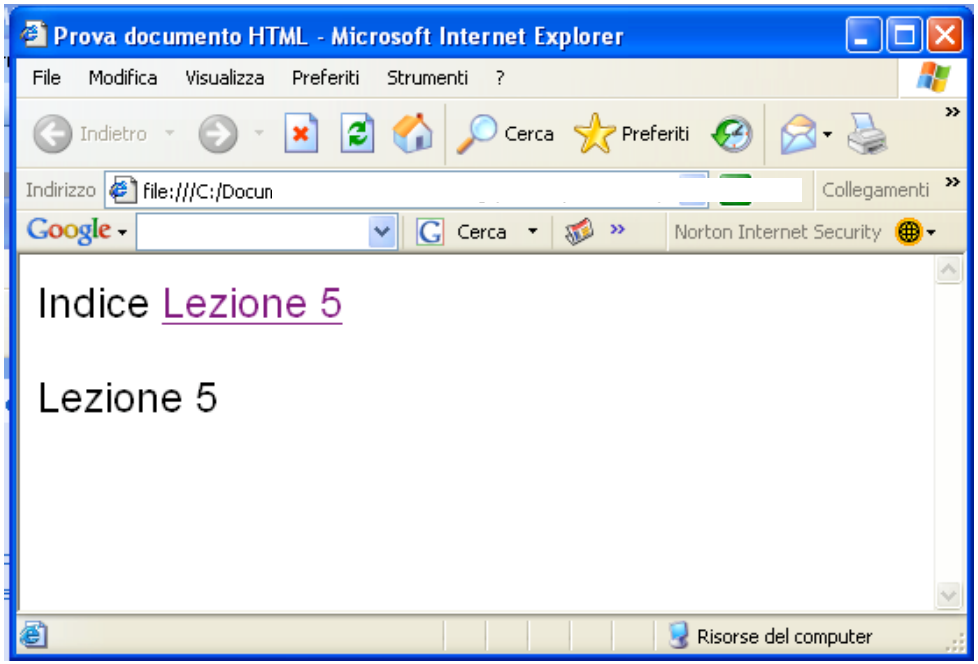


Collegamenti ipertestuali sullo stesso documento

Indice

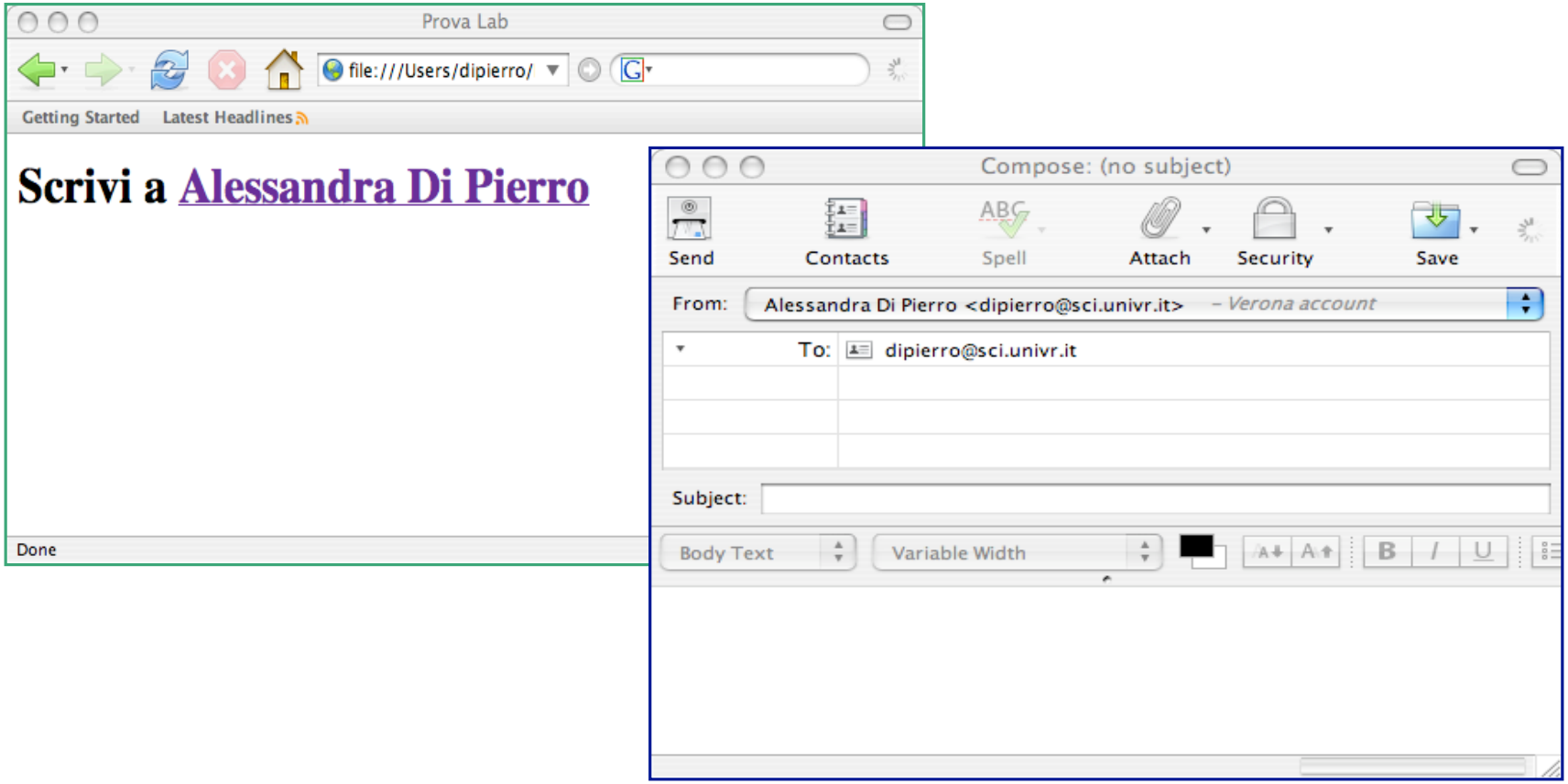
 Lezione 5

Lezione 5



Collegamenti ipertestuali

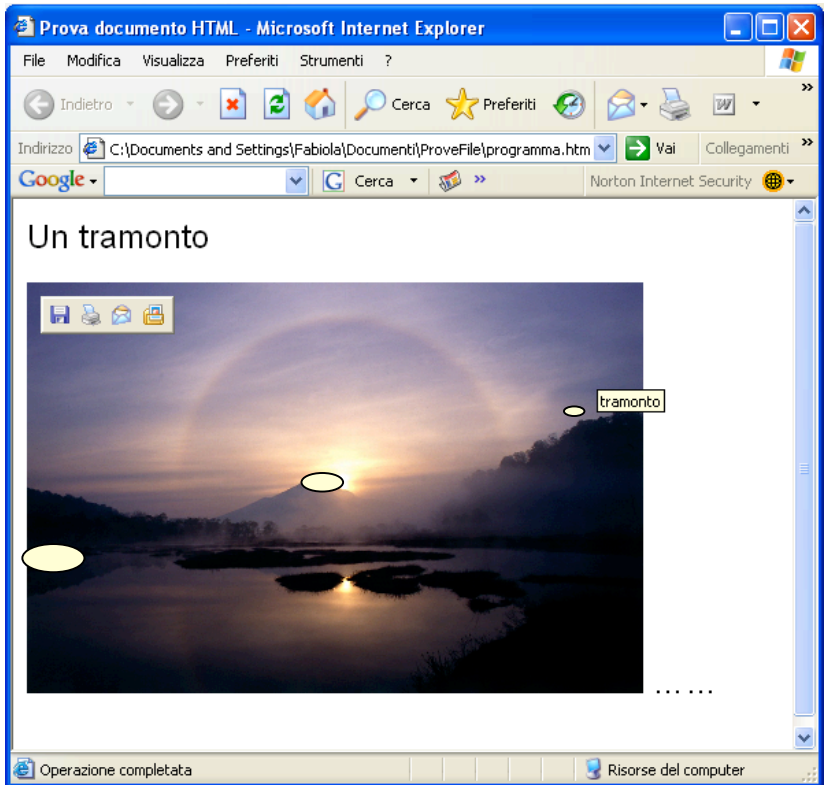
Scrivi a <a href "mailto:dipierro@sci.univr.it">Alessandra Di Pierro



Immagini

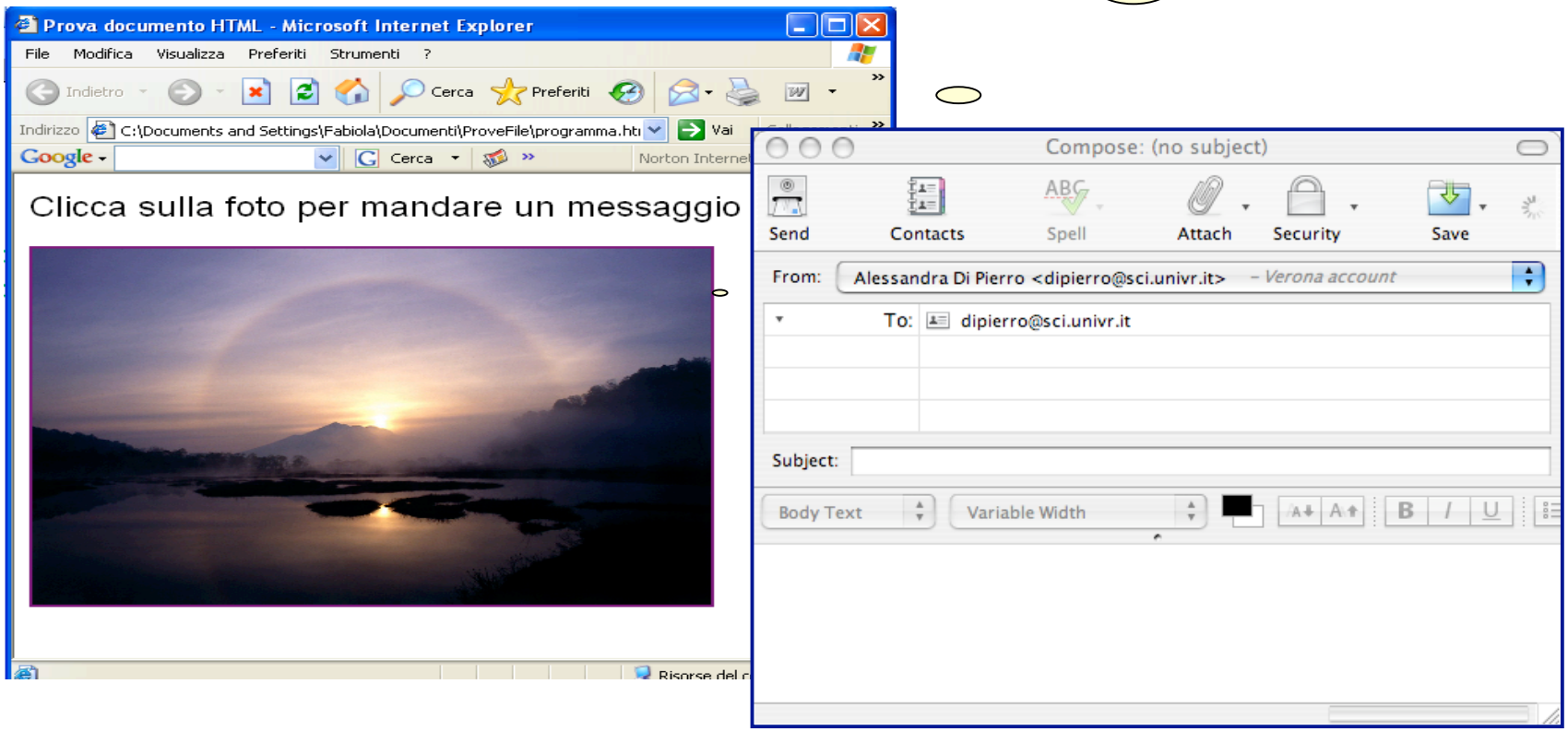
```
<p align="center">
Un tramonto
</p>
<img src =
    "tramonto.jpg"
    width="450"
    height="300"
   alt="tramonto"> ……
```

tramonto



Immagini più collegamenti

```
<p> Clicca sulla foto per mandare un messaggio </p>
<a href "mailto:dipierro@sci.univr.it">
    <img src = "tramonto.jpg" width="450" height="300"
  alt="tramonto"> </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 di intestazione:

  <TH> … </TH>

- ✓ Per definire una cella per i 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>
```

Esempio 1: Risultato

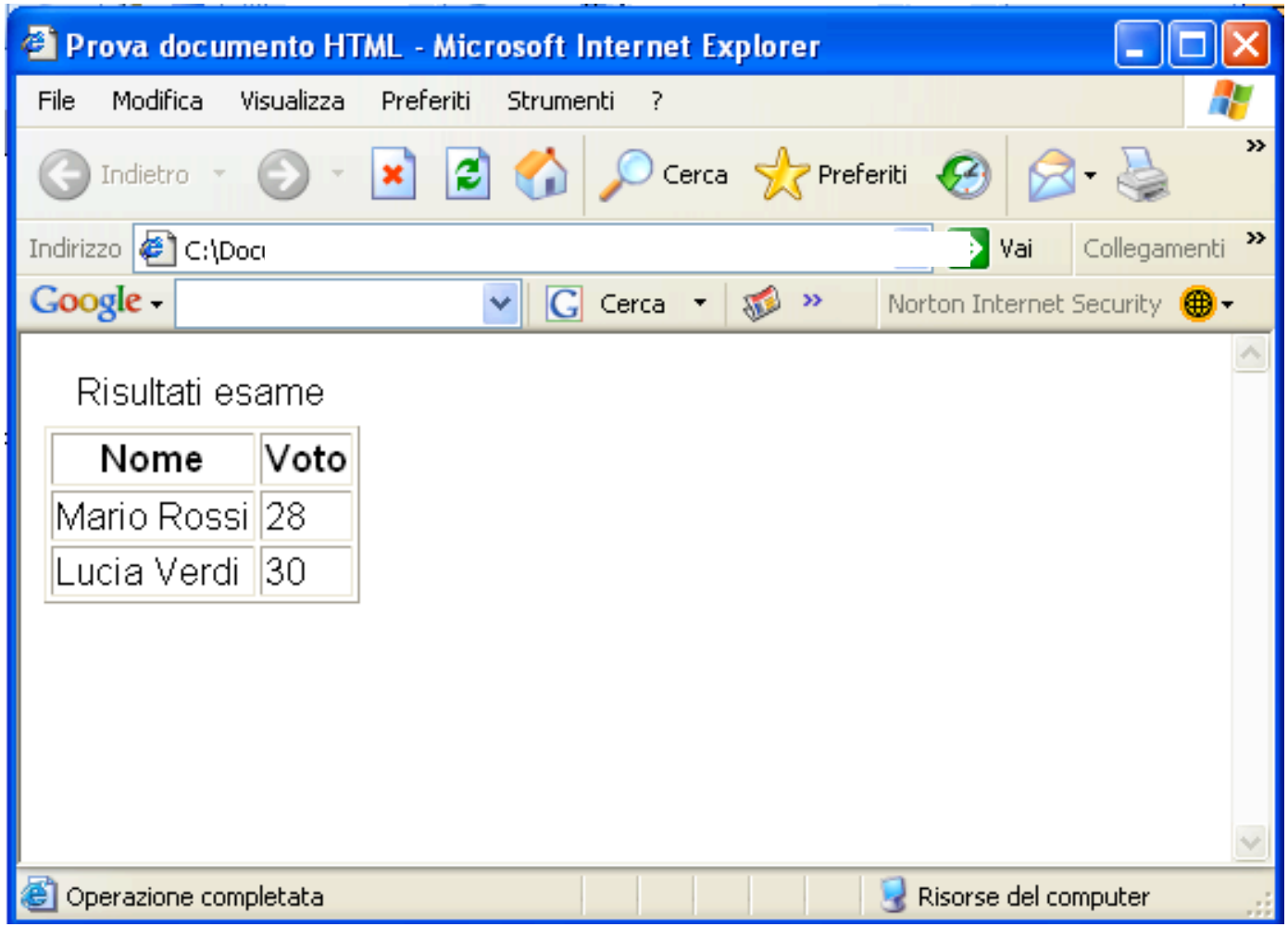
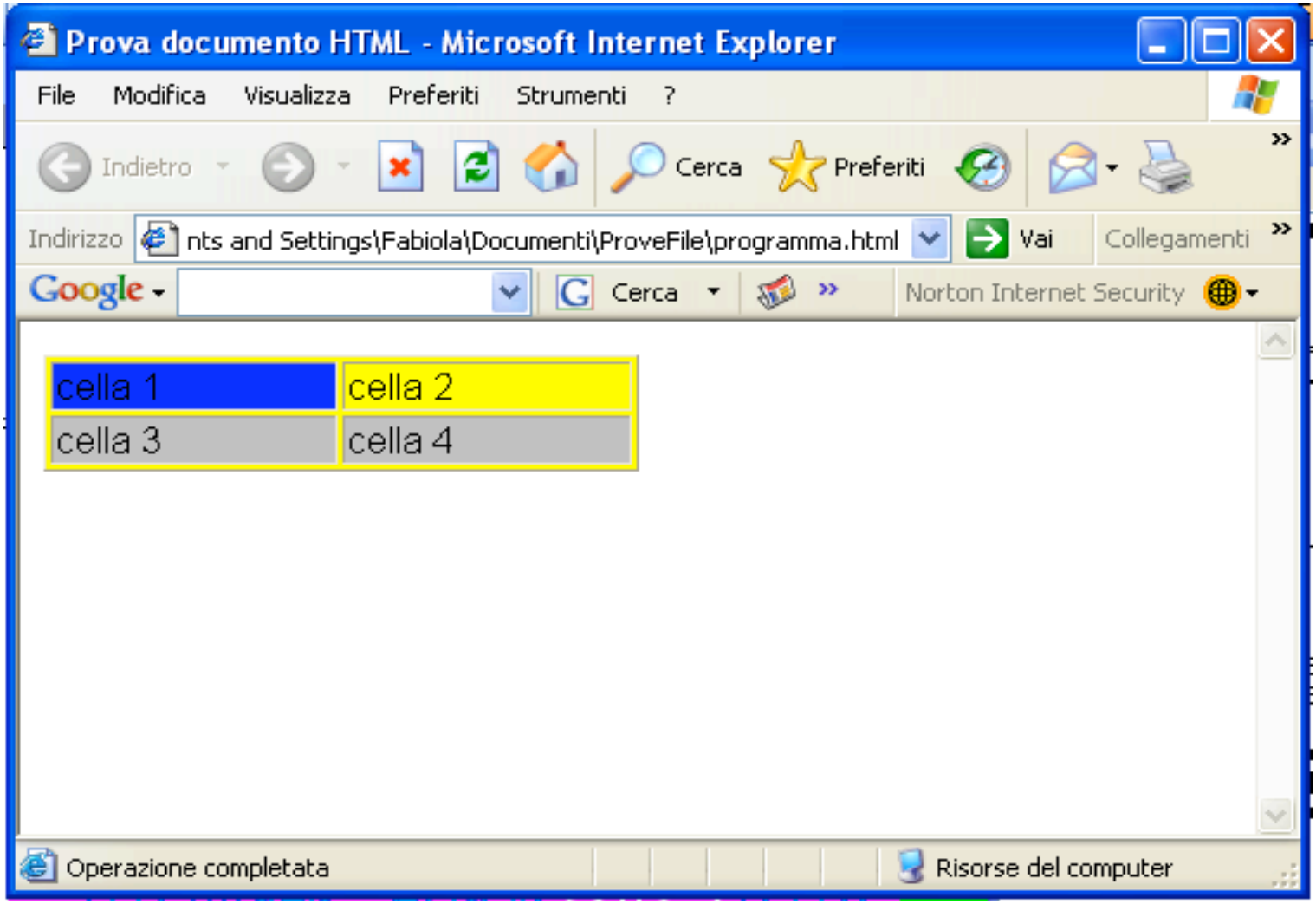


Tabelle: Esempio 2

```
<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>
```

Esempio 2: Risultato



Lettere accentate

- ✓ à à
- ✓ è è
- ✓ ì ì
- ✓ ò ò
- ✓ ù ù
- ✓ é è

- Esempio:
  Il giudizio &egrave pi&ugrave che buono → Il giudizio è più che buono

FORM HTML

- Una FORM consente di specificare, all'interno di un documento HTML, una sezione in cui l'utente può inserire dei dati

- Ha 3 scopi principali:

  - Raggruppare sintatticamente gli elementi di input
  - Identificare il programma del server che si occuperà della gestione dei dati inviati
  - Specificare i valori inviati e la modalità di invio

FORM HTML: Sintassi

- La sintassi del tag <FORM> è la seguente:

```
<form
        action = "URI"
        method = "metodo"
        enctype = "tipo-contenuti"
        accept-charset = "set-di-caratteri"
        accept = "tipi-di contenuti"
        name = "nome-modulo" >
    ...
</form>
```

Attributo ACTION

- **Unico attributo necessario!**

- Nell'attributo action viene specificato l'URI (*Uniform Resource Identifier*) del programma di elaborazione
  - il browser Web crea una richiesta HTTP contenente tutti i dati che compaiono nella form e la invia al programma di elaborazione

- Il programma di elaborazione può essere una Servlet o una JSP

Attributo METHOD

- Il protocollo HTTP fornisce vari tipi di richieste per il trasferimento, prelevamento e cancellazione di file e per operazioni diagnostiche

- I tipi di richieste utilizzabili per moduli HTTP sono GET e POST. Il tipo della richiesta viene specificato mediante l'attributo method

Attributo METHOD (2)

- Una richiesta GET o POST viene interpretata dal server Web come la richiesta di prelevare il documento specificato nell'identificatore URI
- Se il server Web è configurato per gestire Servlet, interpreta una richiesta di queste risorse come la richiesta di eseguire il relativo programma
- L'output prodotto da questo programma viene poi rinviato al richiedente, esattamente come fosse stato richiesto un documento statico

Attributo METHOD (3)

- La differenza fra i metodi GET e POST è il modo in cui viene fornito l'input al processo server
  - GET: i valori inseriti dall'utente vengono aggiunti all'identificatore URI sotto forma di stringa
  - POST: i valori vengono forniti come parte della richiesta HTTP, mentre l'URI non viene modificato

Attributo METHOD (4)

- Problemi del metodo GET
  - Vengono aggiunti valori all'identificatore URI della richiesta che saranno visibili come coppie nome/valore nella riga degli indirizzi del browser e nei file di registrazione del server Web

    Inadatto per dati riservati
  - Alcuni server e browser possono prevedere restrizioni sulla lunghezza dell'indirizzo che può essere inviato
- L'attributo **method** **è opzionale**: se non viene specificato viene impiegato il metodo GET

Elementi di INPUT

- Nel corpo di una FORM vengono descritti i vari campi di input

- Per creare elementi di input vengono utilizzati quattro tipi di tag HTML:

  - <INPUT>: tag generico usato per vari tipi di elementi
  - <SELECT> e <OPTION>: per creare un menu o una casella a discesa
  - <TEXTAREA>: per le caselle di testo multiriga
  - <BUTTON>: per creare pulsanti

Tag <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 del campo. Se non viene specificato si presume sia TEXT
- value: può essere utilizzato per assegnare un valore iniziale al campo
- size: indica le dimensioni del campo in pixel o in caratteri (per i campi di testo)
- maxlength: indica il numero massimo di caratteri che possono essere digitati nel campo

Il controllo TEXT

- Forma più semplice e comune del tag <INPUT> usata per introdurre un'unica riga di testo

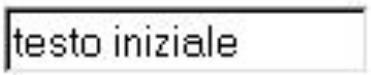
```
<INPUT
    type = "text"
    value = "valore-iniziale"
    size = "dimensione"
    maxlength = "numero massimo di caratteri" >
```

Esempio:

Testo HTML:

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

Resa del browser:



Il controllo PASSWORD

- Variante del controllo TEXT; i caratteri introdotti non vengono visualizzati, ma vengono mascherati da *

```
<INPUT
   type = "password"
   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

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

Esempio

Testo HTML:

```
A<input name="scelta1" type="checkbox" value="A">
B<input name="scelta2" type="checkbox" value="B" checked>
```

Resa del browser:

Il controllo RADIO

- Pulsante di selezione che consente di presentare un'opzione che può essere vera o falsa
- Il funzionamento del pulsante è <u>mutuamente esclusivo</u>

```
<INPUT
    type = "radio"
    name = "nome"
    value = "valore-iniziale"
    checked >
```

Esempio

Testo HTML:

```
A<input name="opzione_esclusiva" type="radio" value="A"
checked>
B<input name="opzione_esclusiva" type="radio" value="B">
```

Resa del browser:

A ◉ B ○

Il controllo HIDDEN

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

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

Esempio

Testo HTML:

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

Resa del browser:

Il controllo SUBMIT

- Per inviare i dati inseriti al server

```
<INPUT
    type = "submit"
    value = "valore-iniziale" >
```

Esempio

Testo HTML:

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

Resa del browser:

Il controllo RESET

- Per riportare tutti i controlli al valore iniziale.

```
<INPUT
        type = "reset"
        value = "valore-iniziale" >
```

Esempio

Testo HTML:

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

Resa del browser:

Attributo TYPE

Attributo Type	Resa del Browser	Testo HTML
text	testo iniziale	<input name="testo" type="text" value="testo iniziale" size="15">
password	**********	<input name="passwd" type="password" value="1g%34D9$" size="15" maxlength="10">
checkbox	A B	A<input name="scelta1" type="checkbox" value="A"> B<input name="scelta2" type="checkbox" value="B" checked>
radio	A B	A<input name="opzione_esclusiva" type="radio" value="A" checked> B<input name="opzione_esclusiva" type="radio“ value="B">
hidden		<input name="variabile_nascosta" type="hidden" value="1234">
submit	Invia query	<input type="submit“ value=“Invia query”>
reset	Annulla	<input type="reset" value="Annulla">

<SELECT> e <OPTION>

- I tag *select* e *option* consentono di creare un elenco di elementi selezionabili in un menù

```
<SELECT
   name = "nome"
   size = "numero di elementi visibili" [multiple]>
   <OPTION
         value = "valore"
         [selected]>
   </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="rosso" selected>rosso</option>
    <option value="verde">verde</option>
    <option value="giallo">giallo</option>
    <option value="bianco">bianco</option>
    <option value="nero">nero</option>
</select>
```

Resa del browser:

<SELECT>: Esempio 2

Codice HTML:

```
<select name="lista" multiple>
    <option value="blue">blue</option>
    <option value="rosso" selected>rosso</option>
    <option value="verde">verde</option>
    <option value="giallo" selected>giallo</option>
    <option value="bianco">bianco</option>
    <option value="nero">nero</option>
</select>
```

Resa del browser:

<TEXTAREA>

- Utile per inserire commenti o altro testo libero per il quale non basta una sola riga

```
<TEXTAREA
   name = "nome"
   rows = "numero di righe"
   cols = "numero di colonne">
   testo
</TEXTAREA>
```

Attributi di <TEXTAREA>

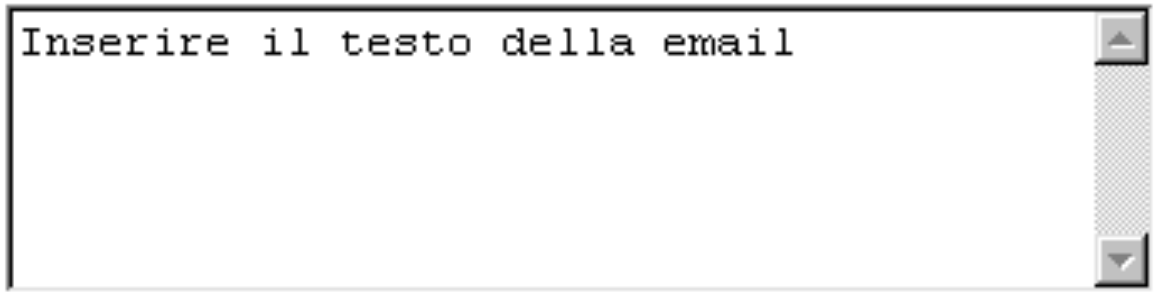
- name: assegna il nome al campo
- rows: specifica il numero di righe visualizzate nella finestra del browser per l'area di testo
- cols: specifica la larghezza in caratteri dell'area di testo da visualizzare

<TEXTAREA>: Esempio 1

Codice HTML:

```
<textarea name="email" rows="5" cols="40">
    Inserire il testo della email
</textarea>
```

Resa del browser:



Panoramica sul protocollo HTTP

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. Queste regole includono:

  – Richiedere un documento per nome
  – Accordarsi sul formato dei dati
  – Determinare l’identita’ dell’utente
  – Indicare i risultati di una richiesta

- Documentazione: RFC 2616 (http://www.freesoft.org/CIE/RFC/index.htm) 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)

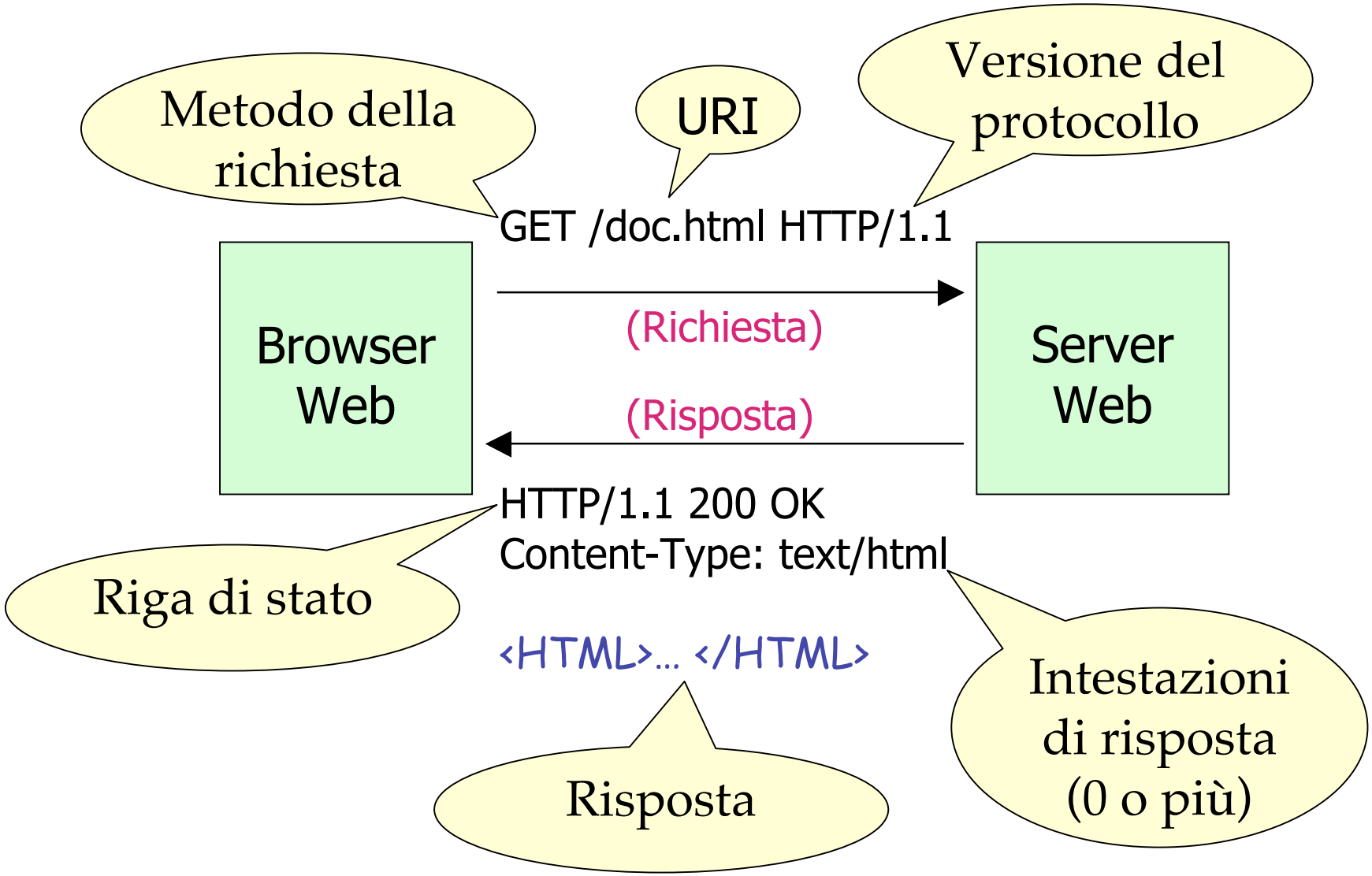
- Operazioni di base:
  - Un’applicazione client (browser Web) apre una connessione verso la porta HTTP del server Web (normalmente la porta 80)
  - Il client invia una richiesta attraverso la connessione aperta
  - Il server Web analizza la richiesta ed individua la risorsa specificata
  - Il server invia una copia della risorsa
  - 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://profs.sci.univr.it/~dipierro/index.html fa riferimento al documento ~dipierro/index.html sul server Web in esecuzione sull'host profs.sci.univr.it e operante sulla porta standard 80

- Se invece il server Web utilizzasse la porta 8080, l'indirizzo dovrebbe essere: http://profs.sci.univr.it:8080/~dipierro/index.html

Funzionamento di HTTP



Esempio

- Sulla riga di indirizzo del browser viene digitato http://profs.sci.univr.it/~dipierro/index.html

- Il browser web apre una connessione sulla porta 80 del server web profs.sci.univr.it

- Il browser web scrive la riga

  GET ~dipierro/index.html HTTP/1.0

  seguita da una riga vuota

Esempio (2)

- Il server web restituisce la risposta:

```
HTTP/1.1 200 ok
Date: Mon, 30 Apr 2006 14:27:43 GMT
…
Content-Length: 1619
Content-Type: text/html

<HTML>
<HEAD>
<TITLE> Pagina personale </TITLE>
</HEAD>
<BODY text="#FFFFFF" bgcolor="#000000" link="#FFFF99"
   vlink="#FFCCCC" alink="#FFCC33">

      …
</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

Riepilogo

- Il protocollo HTTP non ha stati, ovvero non passa alcuna informazione da una richiesta alla successiva

- L’ambiente JSP fornisce metodi per risolvere questa situazione