

FORM

# Strumenti a disposizione

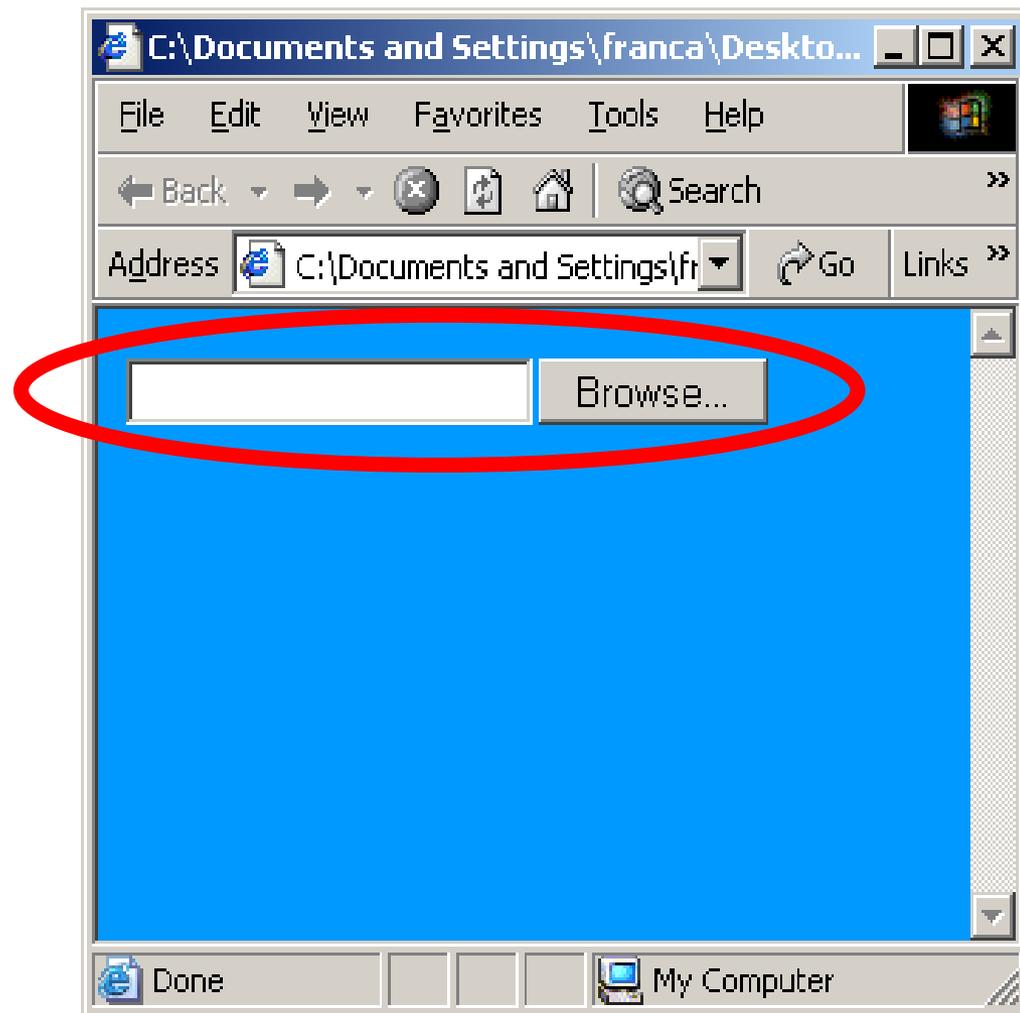
- L'HTML visto fino ad ora permette di descrivere solo documenti statici e passivi
  - Colori, immagini, testi, etc...
  - L'utente può solo interagire con i link
- Non sempre questo è sufficiente
- Un generico sistema informatico (e.g. il web) ha tre comportamenti distinti:
  1. Accetta uno o più **input**
  2. Effettua una o più operazioni (**elaborazione**)
  3. Restituisce uno o più **output**

# Esempi

- Web browser:
  1. Carica un documento HTML
  2. Ne analizza il contenuto
  3. Restituisce una rappresentazione “visiva”
- Un motore di ricerca (Google, Yahoo, ...):
  1. Riceve dall'utente un testo (stringa)
  2. Invia al server la richiesta
  3. Restituisce una pagina con i risultati
- ...

# Form

- Componente della pagina HTML destinata ad accogliere input dall'utente
- Si integra con gli altri elementi



# Tag <Form>

## Sintassi

`<form attr1="val1" attrn="valn" >`

*Testo, tag di formattazione*

`<input></input>`

`<button></button>`

`<textarea></textarea>`

`<select></select>`

...

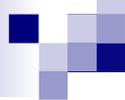
`</form>`

Nota:

- tag di chiusura
- attributi

# Attributi <form>

Attributi	
<b>action</b>	Specifica l'URI dell'agente incaricato di processare i dati della form
<b>method</b>	Specifica che metodo HTTP deve essere usato per sottomettere i dati della form (i metodi sono predefiniti: GET e POST)
<b>enctype</b>	Specifica il tipo del contenuto della form. Alcuni contenuti: "text/html", "text/plain", "image/gif", "video/mpeg", "text/css", "audio/basic" ...



<Form action="...">

L'attributo action specifica l'URI (**Uniform Resource Identifier**) dell'agente incaricato di processare i dati della form

URI puo' essere:

- 1.Indirizzo http di un programma
- 2.Path locale di un programma
- 3.mailto indirizzo

# Tag <Form>

<form attr<sub>1</sub>="val1" attr<sub>n</sub>="valn" >

Testo, tag di formattazione

**<input></input>**

<button></button>

<textarea></textarea>

<select></select>

</form>

# Tag <input>

## Sintassi

<input attr1= val1, ..., attrn=valn>

</input>

### Nota:

- attributi
- tag di chiusura NON necessario

# Attributi <input>

Attributi	
<b>type</b>	Specifica il tipo di casella che si vuole utilizzare.
<b>name</b>	Attribuisce un nome alla casella in questione. <i>Da specificare obbligatoriamente in caso si voglia usare il testo in questione.</i>
<b>maxlength</b>	<i>Dipende da type.</i>
<b>checked</b>	<i>Dipende da type.</i>
<b>src</b>	<i>Dipende da type.</i>
<b>value</b>	<i>Dipende da type.</i> <i>Da specificare obbligatoriamente in caso si voglia usare il testo in questione.</i>
<b>size</b>	<i>Dipende da type.</i>

# <input type="">

- Possibili valori di type:

**text** → per creare una casella di testo fatta da una sola riga

**password** → come text solo che al posto delle lettere vengono visualizzati dei caratteri neutri (asterischi, pallini, ...)

**checkbox** → caselle di selezione multipla

**radio** → caselle di selezione esclusiva

**submit** → crea un pulsante per inviare informazioni ad un programma

# <input type="...">

**reset** → crea un pulsante per riportare i valori della casella ai dati iniziali

**file** → crea una finestra di controllo sui file in locale.

**hidden** → permette di passare dei parametri non visibili all'utente ai programmi della form

**image** → crea un pulsante di tipo submit con uno sfondo grafico

**button** → crea un pulsante semplice senza comportamenti predefiniti

# Esempio: *type="text"*

```
<form >
```

Inserisci il tuo nome:

```
<input type="text">
```

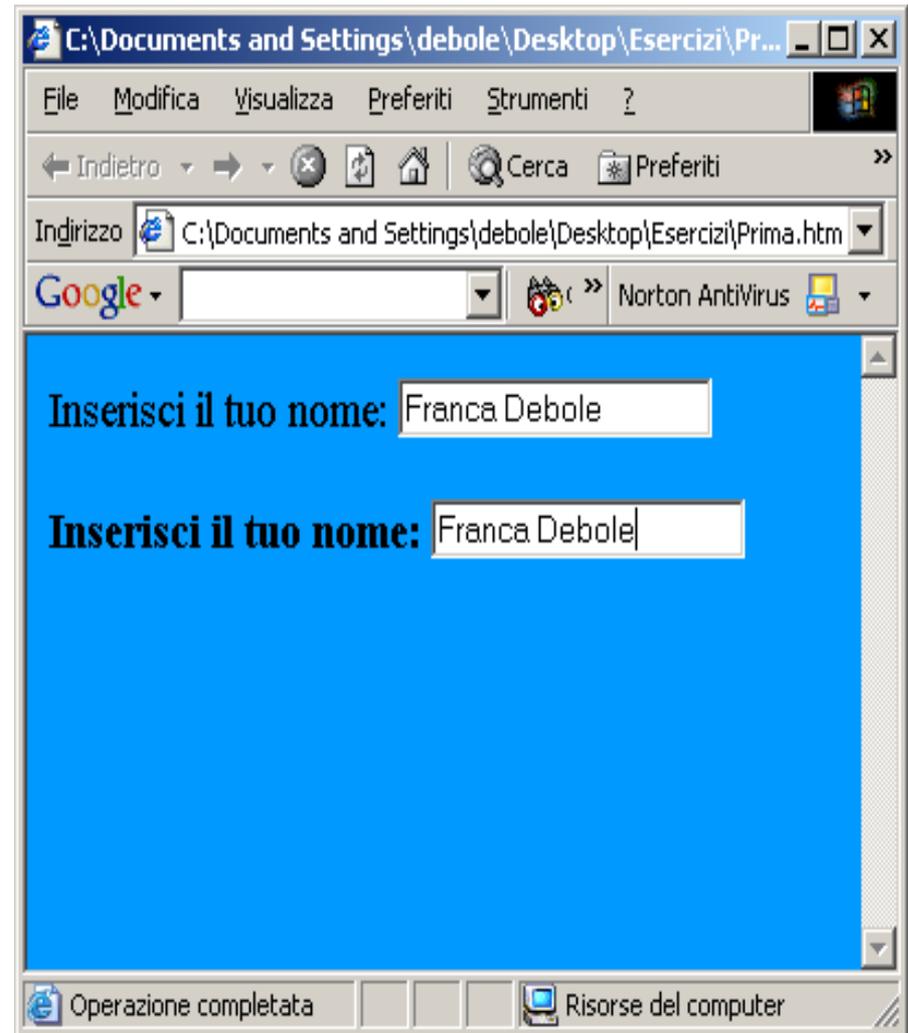
```
<br><br>
```

```
<b>Inserisci il tuo nome:
```

```
</b>
```

```
<input type="text"> <br>
```

```
</form>
```



# Esempio: *type*="password"

```
<form >
```

Inserisci il tuo nome:

```
<input type="password">
```

```
<br><br>
```

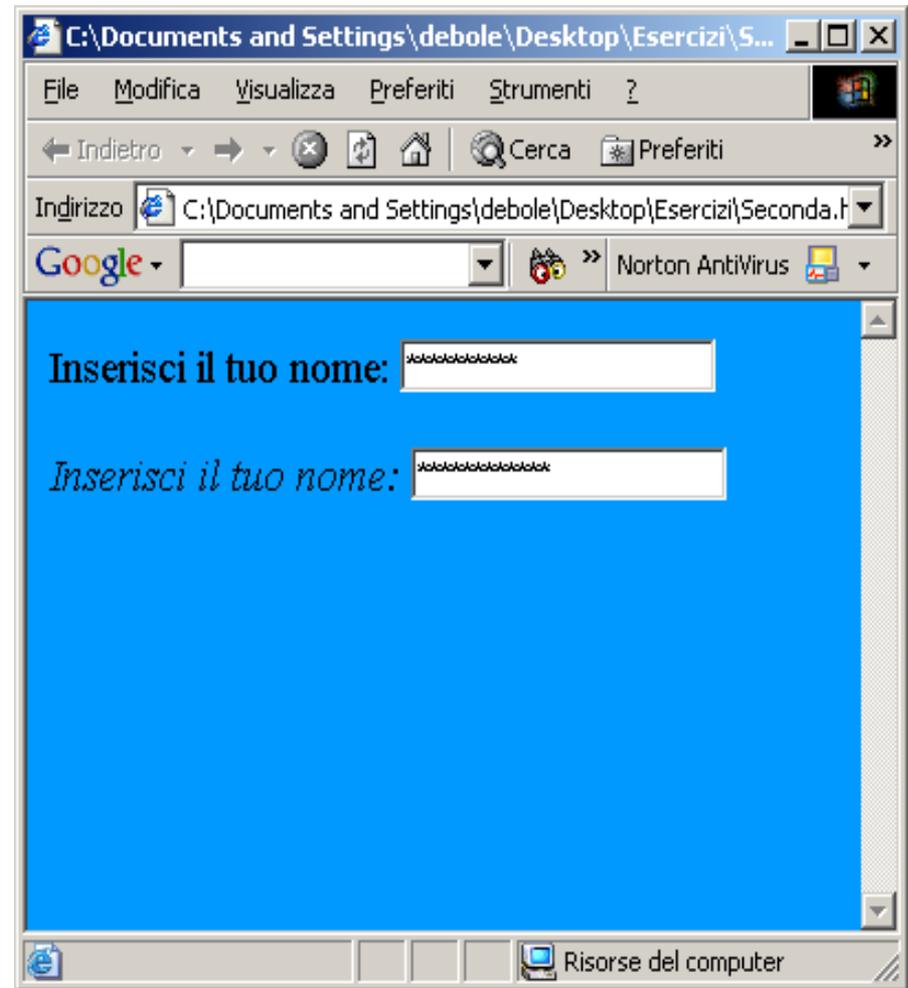
*Inserisci il tuo nome:*

```
</i>
```

```
<input type="password">
```

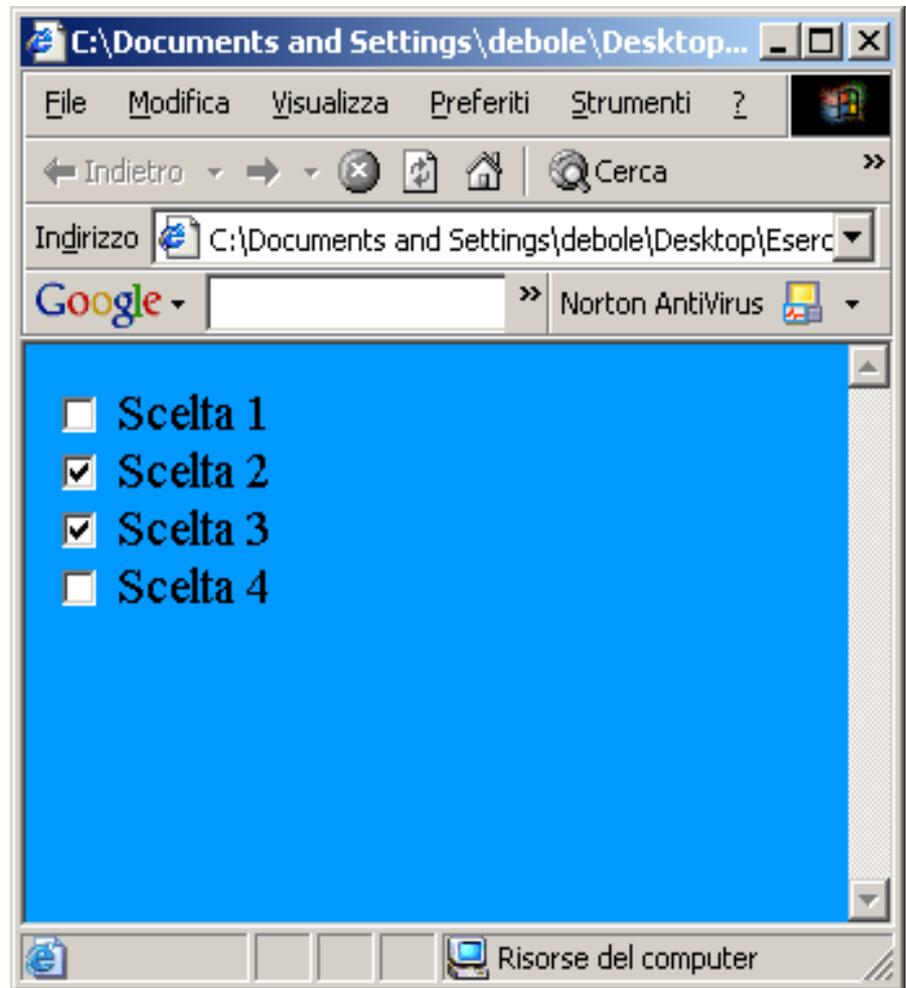
```
<br>
```

```
</form>
```



# Esempio: *type*="checkbox"

```
<form>  
<input type="checkbox">  
  Scelta 1<br>  
<input type="checkbox">  
  Scelta 2<br>  
<input type="checkbox">  
  Scelta 3<br>  
<input type="checkbox">  
  Scelta 4<br>  
</form>
```



# Esempio: *type="radio"*

```
<form>
```

```
<input type="radio">
```

```
  Scelta 1<br>
```

```
<input type="radio">
```

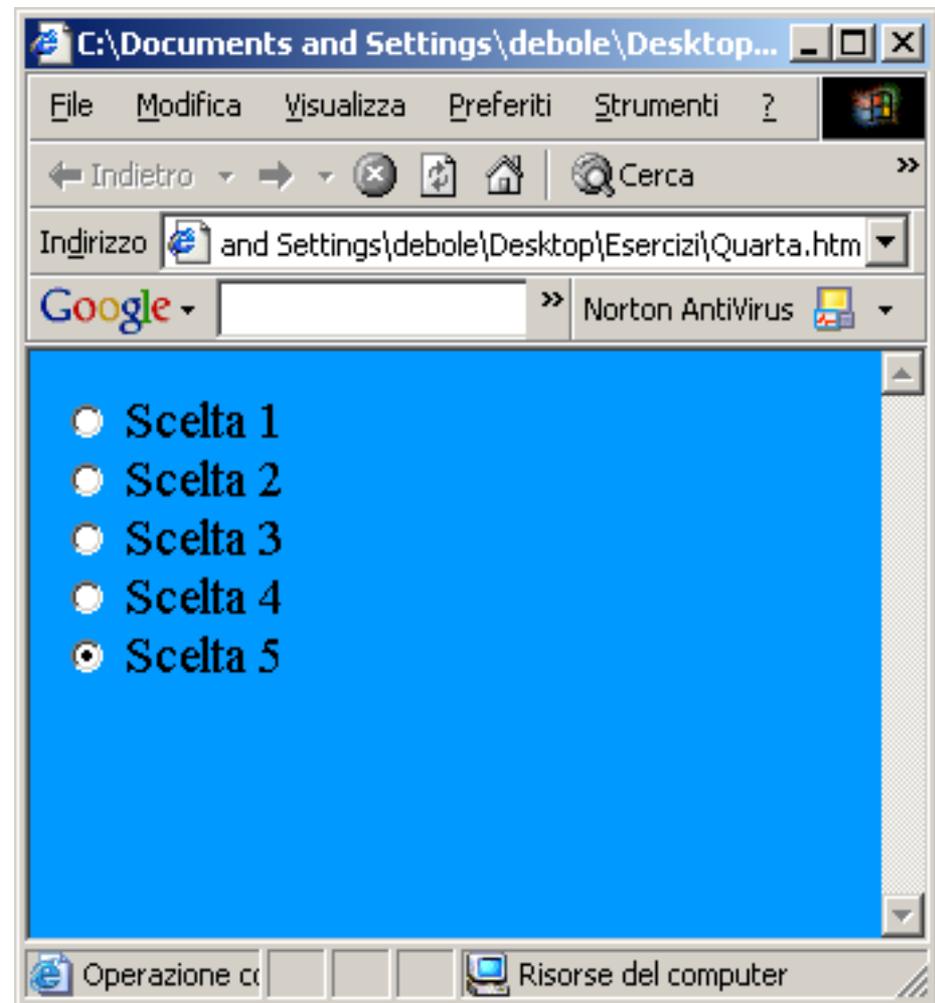
```
  Scelta 2<br>
```

```
....
```

```
<input type="radio">
```

```
  Scelta 5<br>
```

```
</form>
```



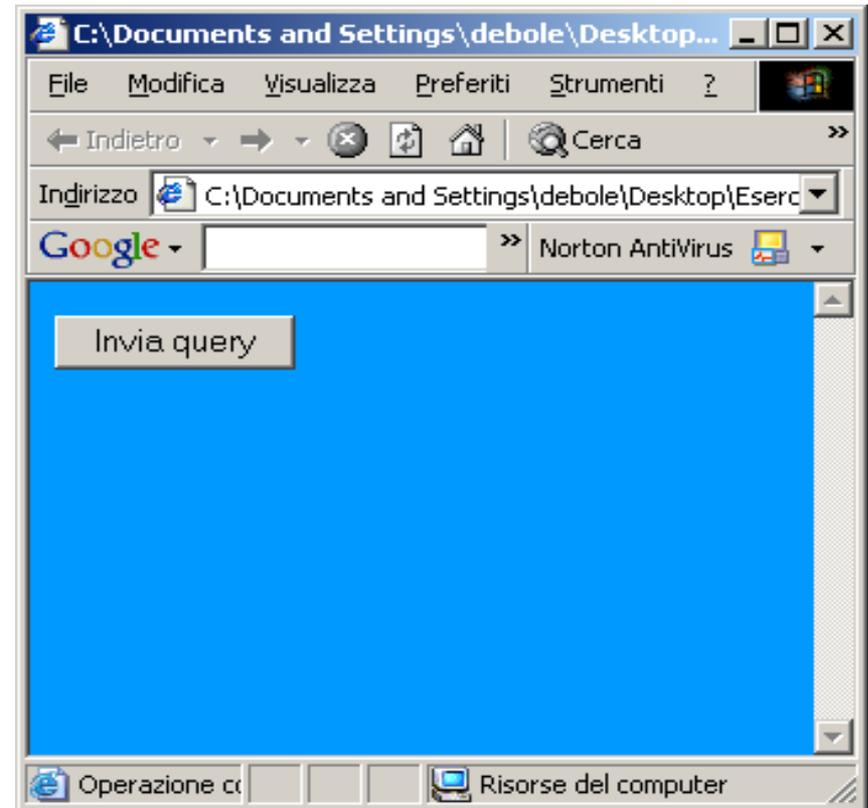
# Esempio: *type*="submit"

```
<form>
```

```
<input type="submit">
```

```
<br>
```

```
</form>
```



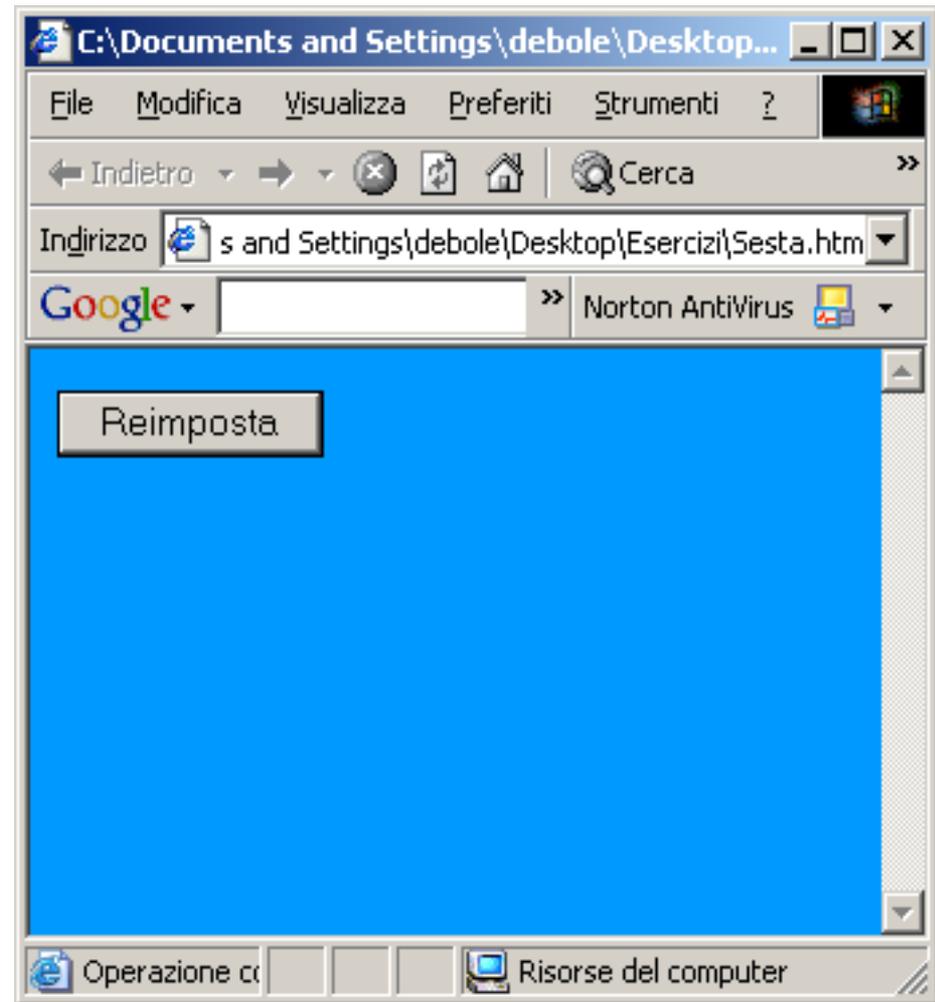
# Esempio: *type="reset"*

```
<form>
```

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

```
<br>
```

```
</form>
```



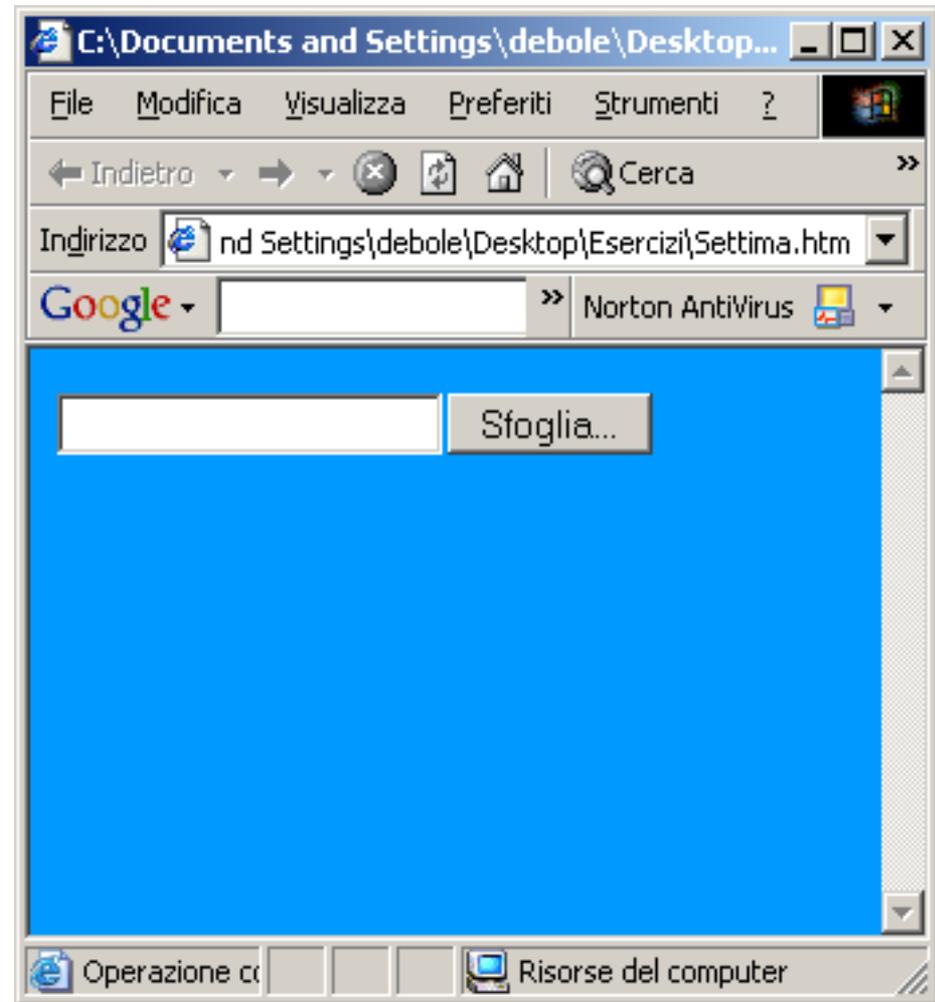
# Esempio: *type*="file"

```
<form>
```

```
<input type="file">
```

```
<br>
```

```
</form>
```



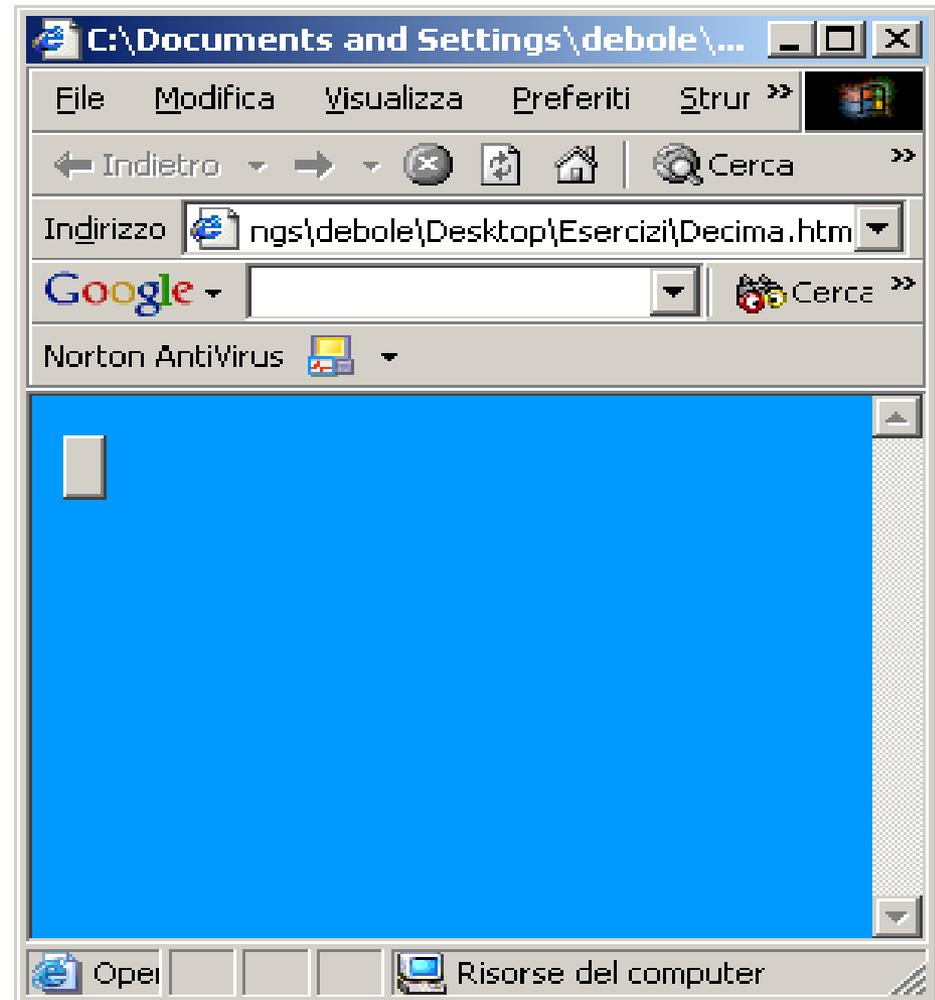
# Esempio: *type="button"*

```
<form>
```

```
<input type="button">
```

```
<br>
```

```
</form>
```



# Attributi al variare di type

## maxlength

Quando `type` ha valore “text” or “password” questo attributo specifica quanti caratteri l’utente può inserire.

## checked

Quando `type` ha valore “radio” or “checkbox” questo attributo specifica quale voce è selezionata.

## src

Quando `type` ha valore “image” specifica il percorso dell’immagine.

# Attributi al variare di type

## value

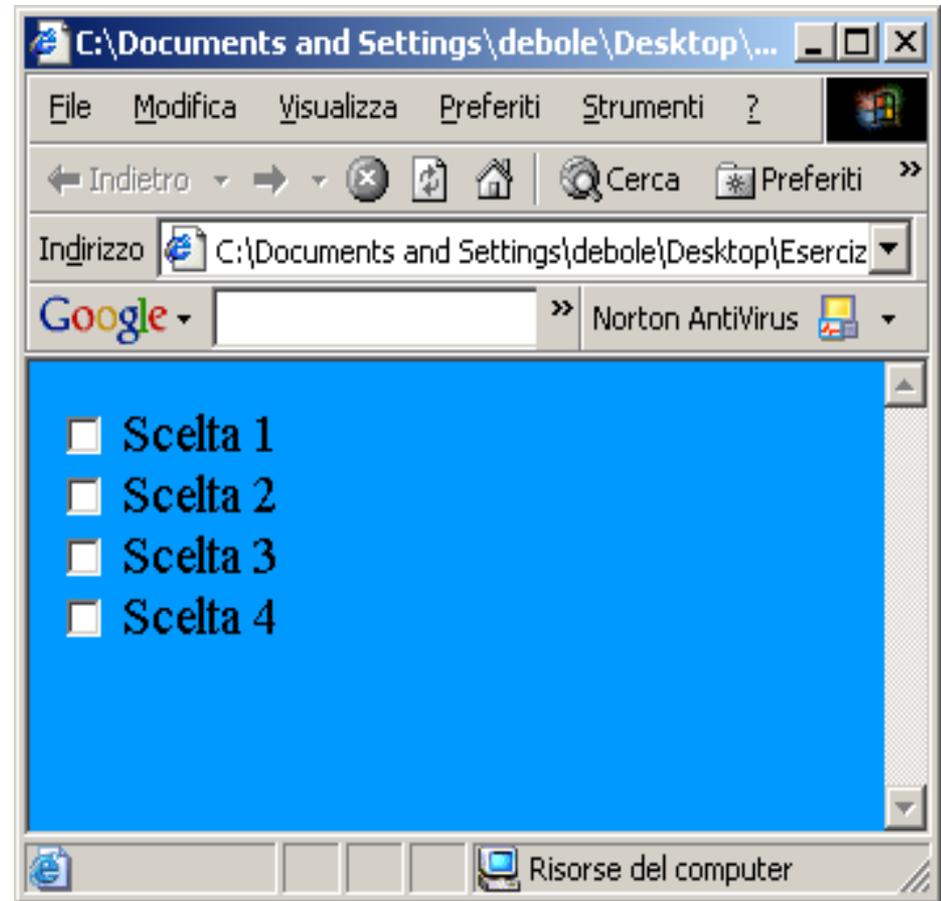
Specifica il valore iniziale del tipo di casella usata. Generalmente opzionale per `type` con valore “radio” o “checkbox” è obbligatorio.

## size

Permette di specificare la larghezza della casella. Come unità di misura vengono utilizzati i pixel tranne nel caso in cui `type` ha valore “text” o “password”, allora in questi casi si usa un numero intero che definisce il numero di caratteri.

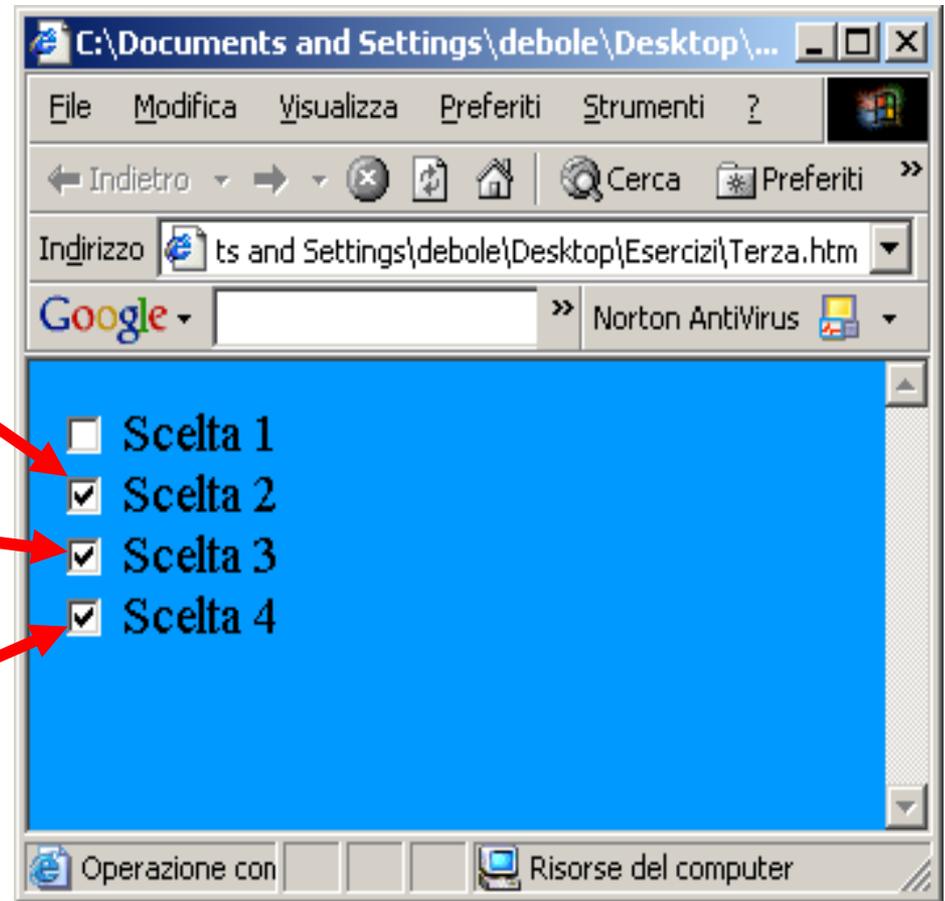
# Attributo: *checked*

```
<form>  
<input type="checkbox">  
  Scelta 1<br>  
<input type="checkbox">  
  Scelta 2<br>  
<input type="checkbox">  
  Scelta 3<br>  
<input type="checkbox">  
  Scelta 4<br>  
</form>
```



# Attributo: *checked*

```
<form>  
<input type="checkbox">  
  Scelta 1<br>  
<input type="checkbox"  
  checked> Scelta 2<br>  
<input type="checkbox"  
  checked> Scelta 3<br>  
<input type="checkbox"  
  checked> Scelta 4<br>  
</form>
```



# Attributo: *size*

```
<form>
```

```
Inserisci il tuo nome:<br>
```

```
<input type="text"><br>
```

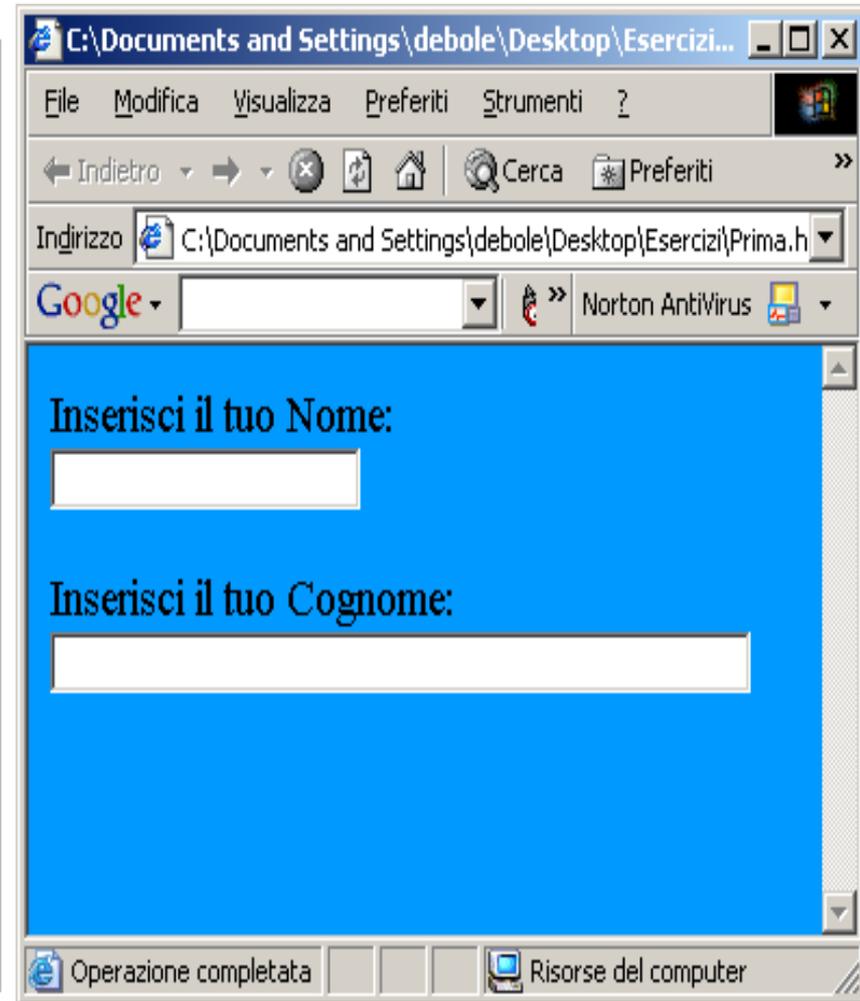
```
Inserisci il tuo Cognome:
```

```
<br>
```

```
<input type="text" size="50">
```

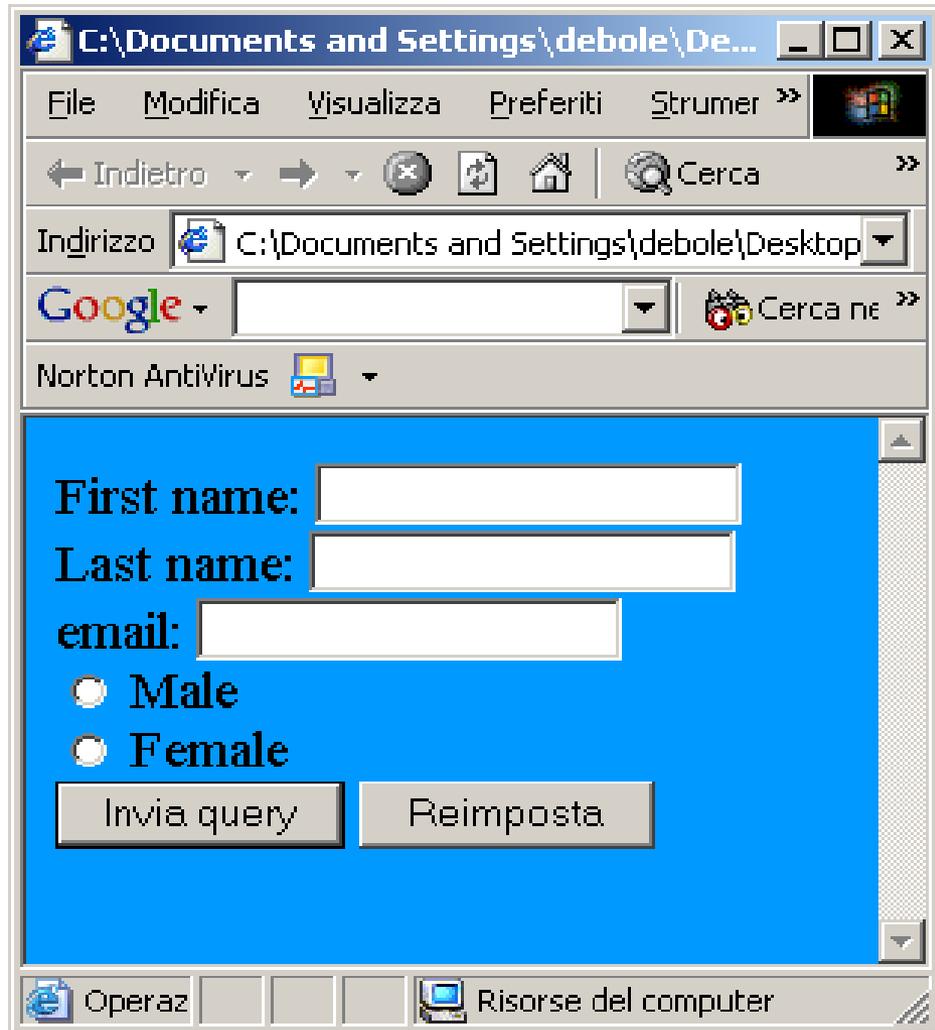
```
<br><br>
```

```
</form>
```



# Esempio: <FORM>

```
<FORM>
<P>
  First name: <INPUT type="text">
<BR>
  Last name: <INPUT type="text">
<BR>
  email: <INPUT type="text">
<BR>
  <INPUT type="radio"> Male
<BR>
  <INPUT type="radio"> Female
<BR>
  <INPUT type="submit">
<INPUT type="reset">
</P>
</FORM>
```



The screenshot shows a web browser window with a blue background. The form contains the following elements:

- First name:
- Last name:
- email:
- Male
- Female
- Invia query
- Reimposta

# Tag <Form>

<form attr<sub>1</sub>="val1" attr<sub>n</sub>="valn" >

Testo, tag di formattazione,

<input></input>

**<button></button>**

<textarea></textarea>

<select></select>

</form>

# Tag <Button>

```
<form>
```

```
<button type="button">
```

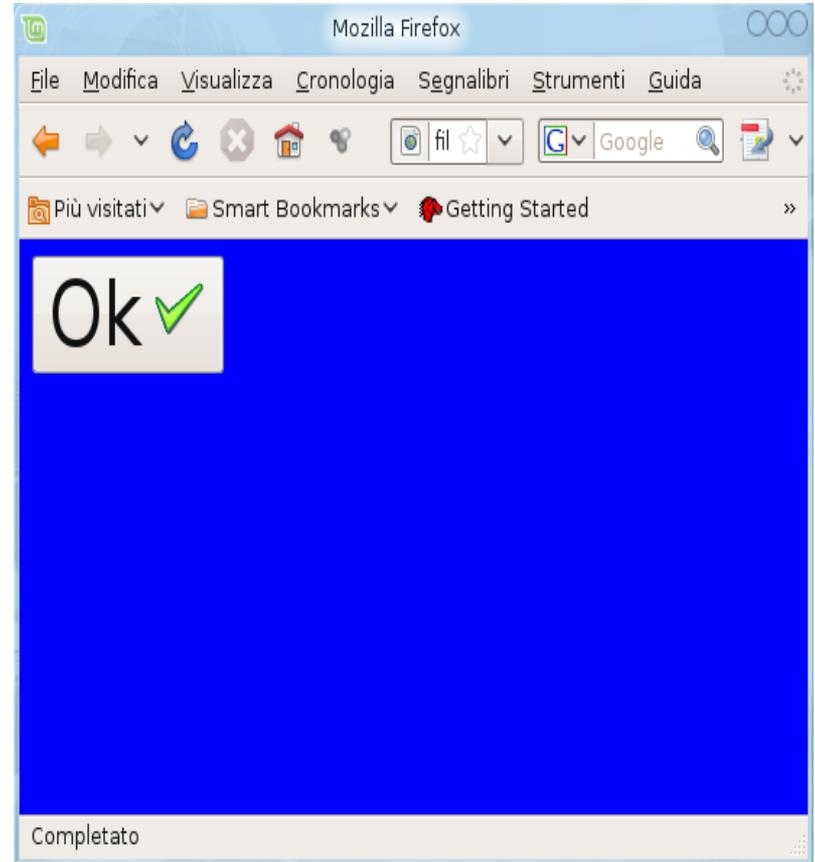
```
<font size="20">Ok</font>
```

```

```

```
</button>
```

```
</form>
```



# Tag <Form>

<form attr<sub>1</sub>="val1" attr<sub>n</sub>="valn" >

Testo, tag di formattazione,

<input></input>

<button></button>

**<textarea></textarea>**

<select></select>

</form>

# Tag <Textarea>

## Sintassi

<textarea attr1= val1, ..., attrn=valn>

*Testo non interpretato*

</textarea>

## Semantica

Textarea permette di avere una casella di testo composta da più righe

Nota:

- Tag di chiusura
- attributi

# Attributi <textarea>

Attributi	
<b>name</b>	Attribuisce un nome alla casella in questione.
<b>rows</b>	Specifica il numero di linee da rendere visibili. Se l'utente inserisce del testo che supera il numero di linee visibili, il browser automaticamente userà una barra di scorrimento per rendere visibile tutto il testo.
<b>cols</b>	Specifica il numero di colonne da rendere visibili. Se l'utente inserisce del testo che supera il numero di colonne visibili, il browser automaticamente userà una barra di scorrimento per rendere visibile tutto il testo.

# Attributi: rows, cols

```
<form>
```

```
<p>
```

```
<textarea rows="10" cols="40">
```

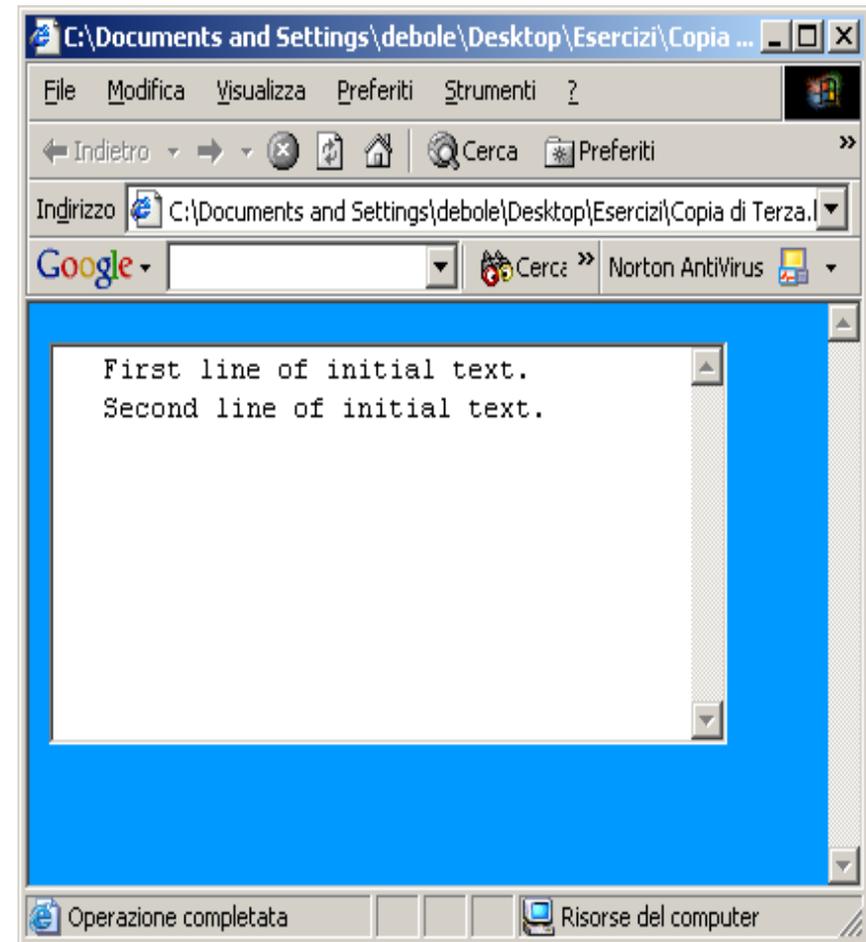
```
  First line of initial text.
```

```
  Second line of initial text.
```

```
</textarea>
```

```
</p>
```

```
</form>
```



# Attributi: rows, cols

```
<form>
```

```
<p>
```

```
<textarea rows="5"cols="20">
```

First line of initial text.

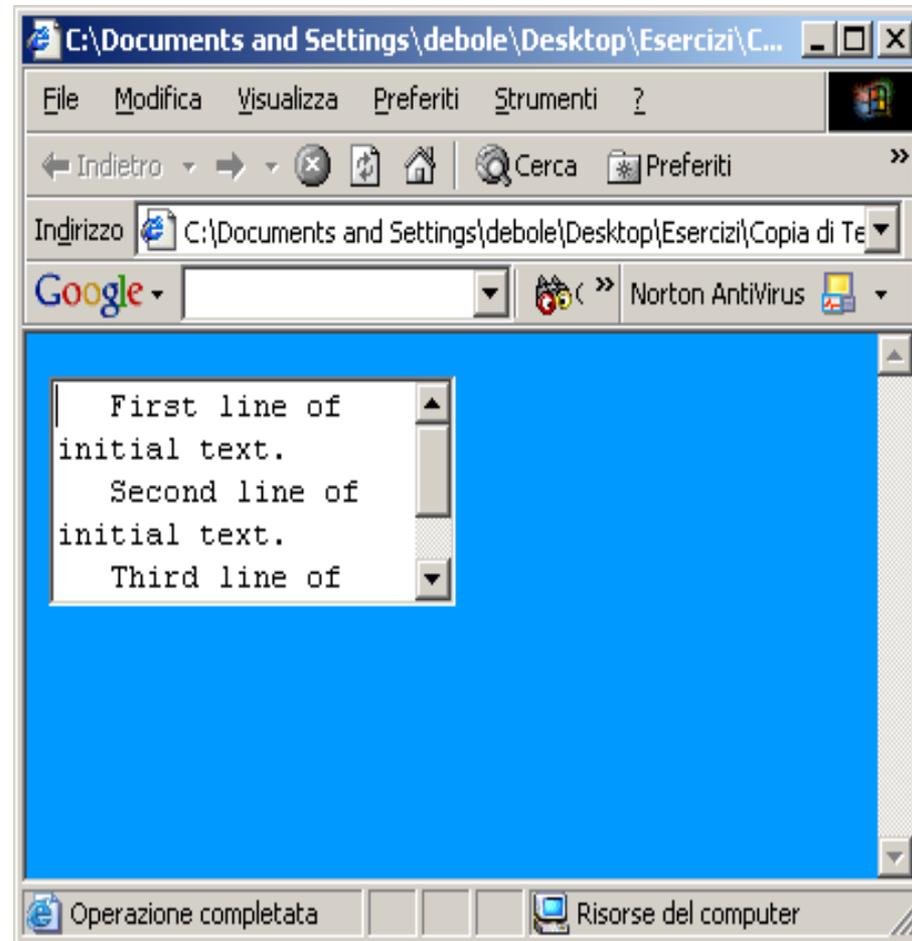
Second line of initial text.

Third line of initial text.

```
</ textarea>
```

```
</p>
```

```
</form>
```



# Tag <Form>

<form attr<sub>1</sub>="val1" attr<sub>n</sub>="valn" >

Testo, tag di formattazione,

<input></input>

<button></button>

<textarea></textarea>

**<select></select>**

</form>

# Tag <Select>

```
<form>
```

Sai usare le form?

```
<select multiple="multiple"  
  size="3">
```

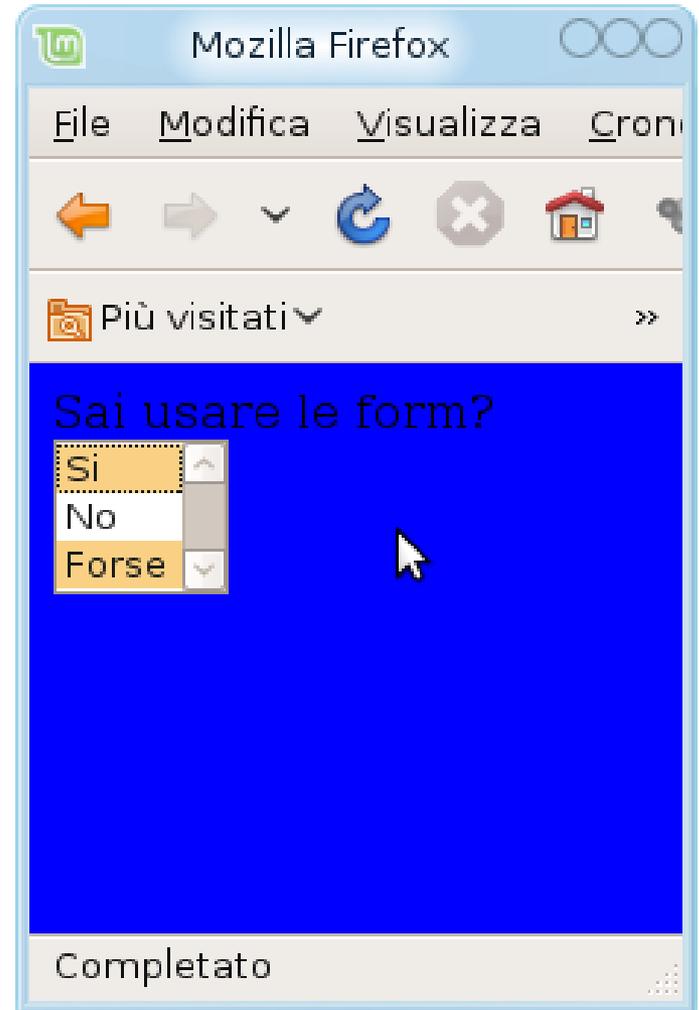
```
<option value="yes">Si</option>
```

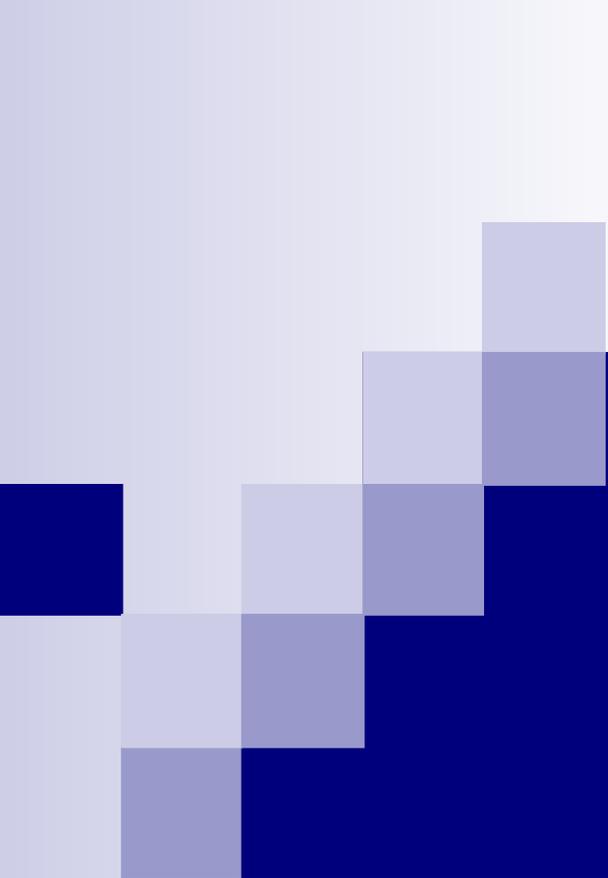
```
<option value="no">No</option>
```

```
<option value="maybe">Forse  
</option>
```

```
</select>
```

```
</form>
```





EVENTI

# Eventi

- L'interazione tra i controlli (bottoni, immagini, ...) è gestita tramite **eventi**
- Un evento è quel segnale istantaneo che viene lanciato da un **controllo** che viene “**stimolato**”
- Esempio:
  - Un **bottone** che viene **premuta** (click)
  - Un'**area di testo** su cui si **muove** il puntatore
  - ...

# Eventi più comuni

<b>Evento</b>	<b>Definizione (Quando avviene)</b>
onLoad	Al caricamento della pagina
onUnload	All'uscita dalla pagina
onClick	Click sinistro su un elemento
onDbClick	Doppio click sinistro
onMouseOver	Transito del mouse su un elemento
onMouseMove	Movimento del mouse
onMouseOut	Uscita del mouse da un elemento
onMouseDown	Pressione di un tasto del mouse
onChange	Cambiamento di un tag
onSubmit	Pressione del tasto invio/return
...	...

# Gestione degli Eventi

- Quando un evento viene “**lanciato**” tutti **coloro** che hanno richiesto di essere avvertiti vengono allertati dal sistema (eventualmente nessuno)
- Le entità che possono fare richiesta (si dice che si **abbonano** all'evento) sono, in genere, piccoli programmi (**gestori**) che eseguono qualche compito in risposta
- I gestori vengono creati con linguaggi di script, tra cui il più usato è **Javascript**

# Abbonarsi ad un Evento

- Per creare l'associazione evento/gestore si usa un attributo che si chiama esattamente come l'evento da gestire
- Esempio:  
`<input type="button" onClick="chi_gestisce_l'evento">`
- Se il gestore è la funzione JS `saluta()`  
`<input type="button" onClick="saluta()">`
- Se la funzione `saluta()` mostra il messaggio "Ciao, Mondo!", cliccando sul bottone si vedrà tale comportamento