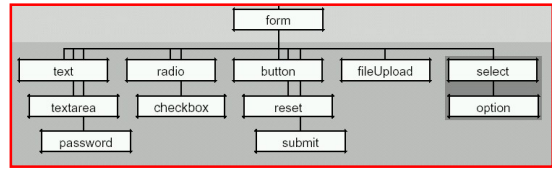


Programmazione lato client

JavaScript (4)

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form



Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form

- È uno degli oggetti più importanti del DOM
- Durante la lettura di un file HTML, viene creato un array con tante celle quanti sono i moduli all'interno del file

```
document.forms []
```

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form

Proprietà

```
document.forms [].length  
document.forms [].action  
document.forms [].method  
document.forms [].encoding  
document.forms [].name  
document.forms [].target  
document.forms [].elements []
```

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form

- `document.forms [].elements []` è a sua volta un array con tante celle quanti sono gli elementi del modulo

```
document.forms [0].elements [4]  
document.forms [2].elements [3]
```

[Esempio 1](#)

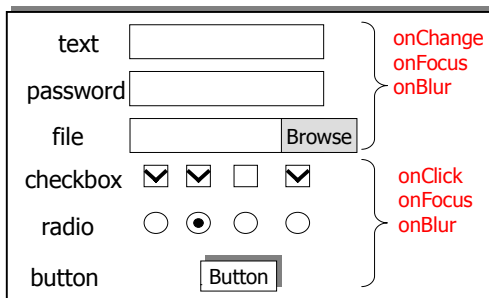
Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form

- È fondamentale dare dei **nomi** ai moduli e ai loro elementi
- I nomi permettono di distinguere tra i dati che vengono inviati al server
- Ogni volta che si introduce un nome, l'interprete JavaScript crea una **proprietà** corrispondente

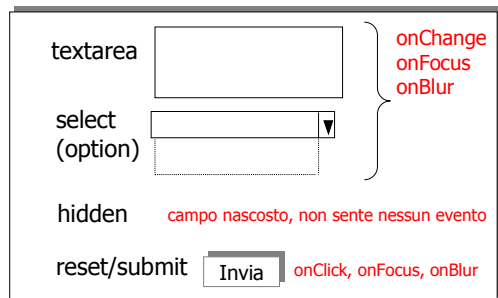
Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form: eventi



Applicazioni di Rete - M. Ribaudò - DISI

Oggetto form: eventi



Applicazioni di Rete - M. Ribaudò - DISI

Oggetto form: eventi

- Ogni volta che con il mouse si clicca su un elemento di un form viene generato un **evento**
- Se esiste un **gestore per l'evento**, vengono mandate in esecuzione le istruzioni JavaScript corrispondenti

```
<form name="frm1">  
<input type="button" name="b1" value="Cambia colore"  
onClick="document.bgColor='yellow';">  
</form>
```

Applicazioni di Rete - M. Ribaudò - DISI

Oggetto form: value

campo di testo di nome txt1 nel modulo frm1

```
<input type="text" name="txt1"  
onChange="alert(this.value);">  
onChange="alert(document.frm1.txt1.value);">
```

Esempio 2

Applicazioni di Rete - M. Ribaudò - DISI

Oggetto form: value

- Tutti gli elementi di un modulo hanno un valore che in JavaScript si legge grazie alla proprietà **value**
- Il valore può essere
 - ✓ L'**etichetta** dell'elemento specificata nell'attributo VALUE
 - ✓ Il **valore** dell'elemento specificato nell'attributo VALUE
 - ✓ Il valore fornito in **input dall'utente**

Esempio 3

Applicazioni di Rete - M. Ribaudò - DISI

Oggetto form: radio e checkbox

- I bottoni radio e checkbox hanno anche la proprietà **checked**

```
document.frm1.rd1.checked  
document.frm1.rd1.value
```

Applicazioni di Rete - M. Ribaudò - DISI

Oggetto form: select e option

- Gli oggetti `select` e `option` permettono di creare menu a discesa
- `Select` possiede le proprietà `selectedIndex` e `options[]`

```
i = document.frm1.sell.selectedIndex  
document.frm1.sell.options[i].value
```

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form: select e option

```
...  
<select name="citta"  
  onChange="i=this.selectedIndex;  
  alert(this.options[i].value);">  
  
  <option value="ge">Genova</option>  
  <option value="mi">Milano</option>  
  <option value="na">Napoli</option>  
  <option value="to">Torino</option>  
  <option value="ve">Venezia</option>  
</select>  
...
```

[Esempio 4](#)

[Esempio 5](#)

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form: invio

"... all input is evil
until proven otherwise ..."



"Data must be validated as it crosses
the boundary between untrusted and
trusted environment."

"Any data submitted by a user is
initially untrusted data."

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form: invio

- Al momento dell'invio di un modulo, oppure quando si vogliono cancellare i campi, si possono fare dei controlli mediante i gestori di eventi `onSubmit` e `onReset`

```
<form name="frm1" method="post" action="..."  
  onSubmit="return conferma_invio();"   
  onReset="return conferma_canc();">  
...  
...  
</form>
```

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form: invio

- Se le funzioni `conferma_invio()` e `conferma_canc()` restituiscono `false` le azioni corrispondenti vengono impedito
- Si può verificare
 - ✓ Che l'utente abbia inserito i dati nei campi obbligatori
 - ✓ Il formato dell'input

Applicazioni di Rete - M. Ribaldo - DISI

Oggetto form: invio

- [Esempio 6](#)
- [Esempio 7](#)
- [Esempio 8](#)

Applicazioni di Rete - M. Ribaldo - DISI

Espressioni regolari

- Le espressioni regolari permettono di descrivere pattern testuali
- Sono rappresentate da un oggetto `RegExp` ma possono essere create anche mediante assegnazione

```
var pattern = / exp.reg /  
var pattern = new RegExp("exp.reg")
```

Applicazioni di Rete - M. Ribaud - DISI

Espressioni regolari

- `/[a,b,c]/` "a", "b", "c"
- `/[^a,b,c]/` tutti tranne "a", "b", "c"
- `/[a-z]/`
- `/[A-Z]/`
- `/[0-9]/`
- `/[a-zA-Z0-9]/`
- `/[a-z]{n}/`
- `/[0-9]{n,m}/`
- `/[a,b,c]{n,}/`

Applicazioni di Rete - M. Ribaud - DISI

Espressioni regolari

- `/^Java/` "JavaScript" OK (inizio riga)
 - `/Java$/` "JavaScript" KO (fine riga)
 - `/a*/` "", "a", "aa", "aaa", ... (0,1,2,...)
 - `/b+/` "b", "bb", "bbb", ... (1,2,3,...)
 - `/c?/` "", "c" (0,1)
-
- `/exp.reg./i` confronto non case-sensitive
 - `/exp.reg./g` confronto globale

Applicazioni di Rete - M. Ribaud - DISI

Espressioni regolari

- `String.search(pattern)`
 - ✓ Restituisce la posizione del primo carattere della sottostringa che coincide con il pattern oppure -1
- `String.replace(pattern, str)`
- `String.match(pattern)`
 - ✓ Restituisce un array contenente i risultati del confronto. Supporta le ricerche globali

Applicazioni di Rete - M. Ribaud - DISI

Espressioni regolari

- Codice fiscale

```
/\b[A-Z]{6}[0-9]{2} .../
```

```
/^[A-Z]{6}\d{2}\[A-Z]{1}\d{2} .../
```

Esempio 9

Applicazioni di Rete - M. Ribaud - DISI