



# Skill Factory School

La scuola d'esperienza, che certifica le tue competenze...

Formazione Orientata al Lavoro

JQUERY



Sponsorizzato da [www.skillbook.it](http://www.skillbook.it) Job Learning community

## Introduzione a JQuery

### Che cos'è JQuery

JQuery è una libreria di funzioni Javascript (**FRAMEWORK**) che ha l'obiettivo di semplificare il codice per quanto riguarda la parte delle animazioni e quindi il CSS e HTML , degli eventi , della manipolazione (DOM), istruzioni di tipo AJAX e altre Utilità.

Con un'unica funzione jQuery si può rimpiazzare funzioni javascript che richiedono molte linee di codice .

Esistono molte librerie come jQuery ma essa tramite semplicità ed efficienza si è portata subito al primo posto come popolarità ed utilizzo, infatti il framework JQuery è utilizzato dalle più grandi compagnie del Web.

## Introduzione a JQuery

### Come utilizzare JQuery

Esistono vari modi per poter utilizzare il framework JQuery:

1 – Scaricare la libreria dal sito ufficiale <http://jquery.com> e il file js per gli stili su <https://jqueryui.com/download/> libreria per gestire le proprietà css. Integrarla nelle proprie pagine, nella sessione <head>.

```
<script src="jquery-1.11.2.min.js"> </script>  
<script src="jquery-ui.min.js" type="text/javascript"> </script>
```

2 - Includerlo tramite CDN ( Conten Delivery Network ossia Rete per la consegna di contenuti ) ed inserirla nella sessione <head> del documento. Tale libreria è fornita da google, ma esistono anche altri tipi rilasciate da altre aziende.

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"> </script>
```

## Introduzione a JQuery

### Come utilizzare JQuery

La sintassi di jQuery è stata progettata per selezionare e manipolare gli elementi HTML.

La sintassi base è la seguente :

`$(selettore).azione();`

Il simbolo \$ indica che si sta inizializzando una funzione in JQuery

Il Selettore è l'elemento HTML che si vuole manipolare

L'Azione è quella che si vuole far eseguire all'elemento selezionato

Nell'esempio l'istruzione `$document.ready(function()` permette di avviare il codice JQuery soltanto dopo il caricamento della pagina (per prevenire eventuali errori).

```
$( document ).ready(function() {  
  // Nasconde tutti i tag p della pagina  
  $("p").hide();  
  // Nasconde gli elementi con la classe prova  
  $(".prova").hide();  
  // Nasconde l'elemento con id=prova  
  $("#prova").hide();  
});
```

## Elementi costitutivi di JQuery

### I Selettori

La sintassi di jQuery è stata progettata per selezionare e manipolare gli elementi HTML.

Inizio jQuery

**\$** ("selettore").azione();

Elemento da Manipolare

Evento da applicare all'elemento

Il selettori servono a selezionare per poi manipolare gli elementi HTML della pagina Web.

Essi si possono trovare sia tramite classe CSS, tramite ID , tipo di elemento , valori e attributi.

A seconda dell'elemento che si vuole trovare , quindi , è importante utilizzare il selettore giusto, ad esempio se si vuole manipolare un solo elemento è più probabile che si usi come selettore un ID che, essendo univoco, corrisponde ad un solo elemento della pagina, mentre se si vuole lavorare su formattazioni di testo è consigliabile utilizzare i selettori di una classe creata in CSS oppure su tutti i tag <p>

## Elementi costitutivi di JQuery

### Altri tipi di selettori

Per selezionare determinati elementi all'interno del DOM jQuery utilizza anche altri tipi di selettori:

<code>\$("*")</code>	<- Seleziona tutti gli Elementi della Pagina
<code>\$("p:first")</code>	<- Seleziona il primo Elemento <p>
<code>\$("ul li:first-child")</code>	<- Seleziona il primo <li> di una lista non ordinata
<code>\$("table tr:first")</code>	<- Seleziona la prima riga della prima tabella
<code>\$("p","h1",this ...)</code>	<- per selezionare più elementi

### JQuery in un file esterno

Le funzioni che creeremo possono essere inserite in un file con estensione .js e successivamente caricate nella pagina tramite il riferimento al file settato nella sessione <head>

```
<script src="mie_funzioni.js"> </script>
```

## Elementi costitutivi di JQuery

### Eventi

Nelle istruzioni JQuery oltre ad indicare il selettore bisogna specificare l'azione da compiere sullo stesso.

L'evento quindi rappresenta il momento in cui l'utente compie una determinata azione.

Esistono vari tipi di eventi : azioni da Mouse, da Tastiera, tramite Form o documenti e finestre.



Click, dbclick, mouseenter, mouseleave



Keypress, Keydown, Keyup



Submit, Change, Focus



Load, scroll

## Elementi costitutivi di JQuery

### click()

L'evento click corrisponde al click del mouse da parte dell'utente sul selettore che abbiamo indicato, nell'esempio seguente al click sul bottone esso sparirà dalla pagina Web.

```
$("#button").click(function(){  
    $(this).hide();  
});
```

### focus()

L'evento focus si basa sulle form HTML. Esso avviene quando l'utente seleziona un campo della form. Nell'esempio seguente quando si selezionerà un campo di tipo input il suo colore di sfondo diventerà di colore giallo.

```
$("#input").focus(function()  
{  
    $(this).css("background-color", "yellow");  
});
```



## Elementi costitutivi di JQuery

### Effetti

Attraverso l'utilizzo di metodi è possibile associare degli effetti grafici a determinati elementi della pagina web. Di seguito analizzeremo i metodi più utilizzati.

### hide() e show()

Questi metodi corrispondono rispettivamente alla scomparsa o alla visualizzazione di un determinato elemento selezionato.

```
$("#button").click(function()
{
    $(this).show();
});
```

```
$("#button").click(function()
{
    $(this).hide();
});
```

## Elementi costitutivi di JQuery

### toggle()

Questo metodo permette all'elemento di passare dallo stato nascosto allo stato visibile e viceversa in modo automatico creando quindi una sorta di fusione tra **hide** e **show**.

Nell'esempio p si visualizzerà se è nascosto o si nasconderà se è già visualizzato, quel valore (500) equivale al tempo di transizione dello stato espresso in millisecondi.

```
$("#button").click(function()  
{  
    $(p).toggle(500);  
});
```

## Elementi costitutivi di JQuery

### fadeIn(), fadeOut(), fadeToggle()

Questi metodi compiono la stessa azione dei metodi visti in precedenza (hide e show) con la differenza che la comparsa e scomparsa avviene attraverso una dissolvenza.

```
$("#button").click(function(){  
    $("#p").fadeIn();  
});
```

```
$("#button").click(function(){  
    $("#p").fadeOut();  
});
```

### slideDown(), slideUp(), slideToggle()

Questi metodi invece della dissolvenza utilizzano un'animazione a tendina. Con il Down mostra e con l'Up nasconde.

```
$("#button").click(function(){  
    $("#p").slideDown();  
});
```

```
$("#button").click(function(){  
    $("#p").slideUp();  
});
```

## Elementi costitutivi di JQuery

### Manipolare il css

Attraverso l'utilizzo di JQuery è possibile manipolare lo stile di un elemento per creare animazioni personalizzate.

**`$(selettore).animate({parametri}, velocità);`**

`animete()` è il metodo che mi permette di creare animazioni personalizzate.

`parametri()` definisce le proprietà css che verranno manipolate.

`velocità()` definisce il tempo di durata della transizione (può essere espressa sia in millisecondi oppure in due modalità predefinite quali "slow" e "fast" ).

Per il funzionamento di `animate` e del codice css bisogna importare il plugin jQuery Color scaricabile dal seguente link:

<https://github.com/jquery/jquery-color>.

L'esempio a lato esegue una manipolazione del css dei tag `<div>` al click sul tag `<button>` applicando uno sfondo di colore rosso, il testo di colore bianco e trasformando il testo in grassetto. L'animazione verrà eseguita in modalità lenta.

P.S. E' possibile indicare le proprietà anche con la sintassi css:

E.S. **`'font-size': "20px";`**

```
$("#button").click(function(){
    $("#div").animate({
        background-color:'red',
        color:'white',
        font-weight:'bold'
    }, "slow"); });
```

## Elementi costitutivi di JQuery

### Callback e concatenazione

Con il **callback** è possibile effettuare un'altra azione al compimento dell'azione precedente. Questo può prevenire alcuni errori di scripting. La sintassi classica è la seguente

**`$(selettore).azione(opzioni,callback);`**

L>alert verrà visualizzato solo quando i <p> all'interno del documento vengono nascosti (quindi gli effetti di tempo influiscono sul callback).

E' possibile concatenare più metodi applicati ad un singolo Selettore

Nell'esempio a lato, i tag <p> diverranno rossi poi scompariranno e riappariranno con tempistiche differenti. Quindi il **callback** è utilizzabile quando ci sono diversi elementi mentre la **concatenazione** è consigliata quando tutti i metodi si applicano ad un singolo selettore.

```
$("#button").click(function(){  
    $("#p").hide("fast", function(){  
        alert("Il testo è scomparso");  
    });  
});
```

```
$("#p").css("color",  
"red").hide(300).show("slow");
```

## DOM e JQuery

### Ricevere contenuti

Esistono tre metodi per ricevere i contenuti di un elemento selezionato :

- **text()** Serve per inserire o ricevere il contenuto testuale dell'elemento selezionato
- **html()** Serve per inserire o ricevere il contenuto compreso dei tag HTML dell'elemento selezionato
- **val()** Serve per inserire o ricevere il contenuto di un campo input

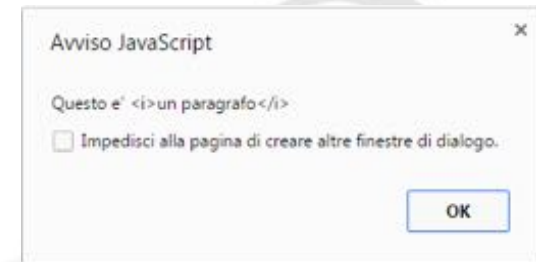
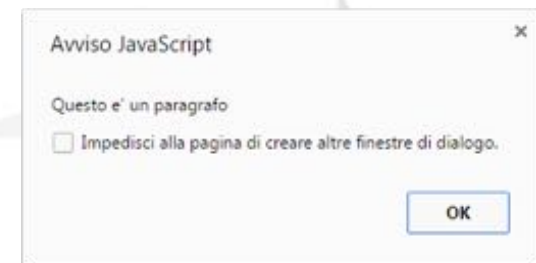
## DOM e JQuery

### Esempi

```
<-- JQuery -->
$(document).ready(function(){
  $("#text").click(function(){
    alert($("#text").text());
  });
  $("#html").click(function(){
    alert($("#html").html());
  });
});
<-- HTML -->
<p id="test">Questo e' <i>un paragrafo</i> </p>
<button id="text">Mostra Testo</button>
<button id="html">Mostra HTML</button>
```

text()

html()

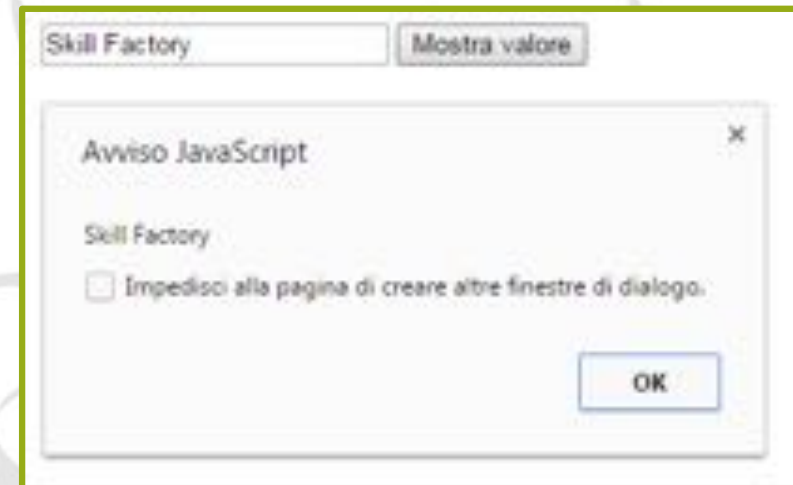


## DOM e JQuery

### Esempi

```
<-- Script -->
$(document).ready(function(){
  $("#val").click(function(){
    alert($("#valore").val());
  });
});

<-- HTML -->
<input type="text" id="valore"/>
<button id="val">Mostra valore</button>
```



N.B. Quindi per i campi della form viene applicato un metodo differente rispetto agli altri elementi



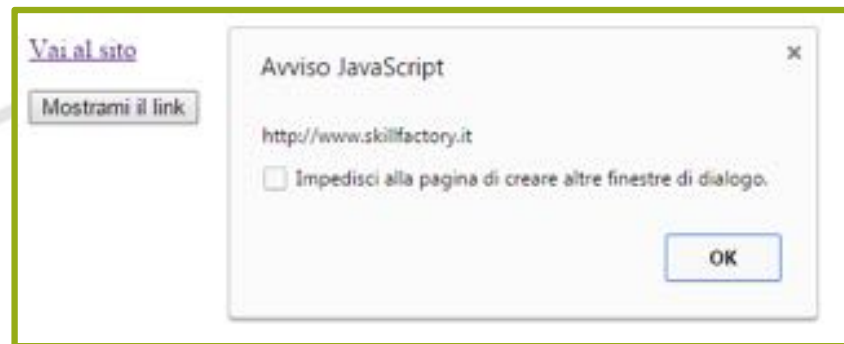
## DOM e JQuery

### Ricevere attributi

Con JQuery è possibile inserire o ricevere un attributo di un elemento utilizzando il metodo `attr()`.  
Nell'esempio riportato l'alert mostrerà il contenuto della proprietà href dell'elemento che ha come id valore url.

```
<-- Script -->
$(document).ready(function(){
    $("#mostraurl").click(function(){
        alert($("#url").attr("href"));
    });
});

<-- HTML -->
<p><a href="http://www.skillfactory.it"
id="url">Vai al sito</a></p>
<button id="mostraurl">Mostrami il
link</button>
```



## DOM e JQuery

### Inserire contenuti e attributi

Abbiamo visto in precedenza come ricevere contenuti e attributi di un elemento selezionato. E' possibile anche inserire tramite JQuery valori a nostro piacimento utilizzando gli analoghi metodi.

Per inserire valori nei nostri selettori basta inserire all'interno delle parentesi () dei metodi ciò che vogliamo inserire in quel elemento. E' possibile inserire sia stringhe fisse come nell'esempio sia variabili.

Per gli attributi la procedura è analoga. E' possibile settare più attributi di un elemento in un'unica istruzione, vedi esempio. Gli attributi tra loro vanno separati dalla virgola mentre i valori sono preceduti da i due punti.

```
$("#p").text("Testo da inserire")  
$("#p").html("<i>Testo da inserire</i>")  
$("#input").val("Skill Factory");
```

```
$("#skill").attr({  
  "href" : "http://www.skillfactory.it", "title" :  
  "Vai al sito Skill Factory"  
});
```

## DOM e JQuery

### Aggiungere contenuti o elementi HTML

Con il Framework JQuery è possibile non solo inserire ma anche aggiungere Contenuti o creare nuovi elementi HTML all'interno delle nostre pagine Web. Per fare ciò abbiamo a disposizione 4 Metodi che lavorano in modo differente sul DOM: `append()` `prepend()` `after()` `before()`

### `append()`

Con l'utilizzo del metodo `append()` è possibile inserire contenuto alla fine dell'elemento selezionato.

```
<--Script-->
$("#aggiungi").click(function()
{
  $("p").append("Skill Factory");
});
<--HTML-->
<p>Vai al sito </p>
<button id="aggiungi">Mostrami il testo</button>
```



Vai al sito Skill Factory

Mostrami il testo

## DOM e JQuery

### prepend()

Il metodo prepend() inserisce il contenuto all'inizio del selettore utilizzato.

```
<--Script-->
$("#aggiungi").click(function(){
    $("p").prepend("Skill");
});

<--HTML-->
<p> Factory Praticamente </p>
<button id="aggiungi">Aggiungi Testo</button>
```



Skill Factory Praticamente

Mostrami il testo

## DOM e JQuery

### Aggiungere elementi HTML con append() e prepend()

Con l'utilizzo dei metodi append() e prepend() è possibile inoltre aggiungere elementi HTML aggiuntivi alla pagina inserendoli come stringa all'interno delle parentesi tonde.

```
<--Script-->
$("#aggiungi").click(function(){
    $("#paragrafo").prepend("<p>Skill
    Factory</p>");
});

<--HTML-->
<div id="paragrafo"> </div>
<button id="aggiungi">Mostrami il Testo</button>
```

Skill Factory

Skill Factory

Mostrami il testo

```
<--Script-->
$("#aggiungi").click(function(){
    $("#ol").append("<li>Skill Factory</li>");
    <--HTML-->
<ol>
<li>Elemento</li> <li>Elemento</li>
</ol>
<button id="aggiungi">Aggiungi </button>
```

1. Elemento
2. Elemento
3. Skill Factory
4. Skill Factory

Aggiungi

## DOM e JQuery

### after() before()

I metodi after() e before() sono analoghi a quelli visti precedentemente.

```
<--Script-->
$("#aggiungi").click(function(){
    $("#span").after("<b> Skill Factory</b>");
});
<--HTML-->
<p><span>Benvenuti in</span></p>
<button id="aggiungi">Aggiungi</button>
```

Benvenuti in

Aggiungi



Benvenuti in **Skill Factory**

Aggiungi

```
<--Script-->
$("#aggiungi").click(function(){
    $("#span").before("<b>Titolo: </b>");
    <--HTML-->
    <p><span>Javascript</span></p>
    <p><span>JQuery</span></p>
    <button id="aggiungi">Aggiungi</button>
```

Javascript

JQuery

Aggiungi



**Titolo:** Javascript

**Titolo:** JQuery

Aggiungi

## DOM e JQuery

### Rimuovere elementi HTML

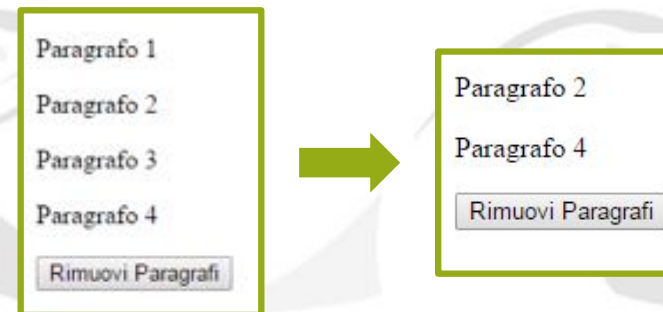
Con JQuery è possibile di eliminare elementi all'interno della pagina Web, questo grazie l'utilizzo dei metodi **remove()** e **empty()**. E' possibile filtrare gli elementi aggiungendo all'interno delle parentesi del metodo qualsiasi tipo di selettore JQuery.

**\$(selettore).remove()** Con questo metodo viene rimosso l'elemento selezionato con i corrispettivi figli

**\$(selettore).empty()** Con questo metodo vengono rimossi i figli dell'elemento selezionato

```
<--Script-->
$("#rimuovi").click(function(){
    $("p").remove(".cancellabile")
});

<--HTML-->
<p class="cancellabile">Paragrafo 1</p>
<p>Paragrafo 2</p>
<p class="cancellabile">Paragrafo 3</p>
<p>Paragrafo 4</p>
<button id="rimuovi">Rimuovi Paragrafi</button>
```



## DOM e Jquery

### Metodo on()

L'event delegation si basa su una caratteristica comune a tutti i browser: un evento innescato su un elemento.

Con il metodo on in base al numero e alla tipologia di argomenti passati, definisce un evento tradizionale o delegato:

```
$('#contenitore').on('click', function () {  
    //evento lanciato al click sul contenitore  
    // uguale a .bind('click', ...)  
});  
  
$('#contenitore').on('click', 'a', function () {  
    //evento lanciato al click su un link interno  
    // uguale a .delegate('a', 'click', ...)  
});  
//equivalente  
$('#contenitore ').click(function() {  
});
```



## DOM e JQuery

### Metodo `has()`

Il metodo `has()` serve a verificare, in modo booleano, se un elemento contiene un determinato elemento tra i suoi discendenti specificato come selettore.

Al click sul bottone solo l'elemento che contiene altri elementi al suo interno verrà aggiunta una classe css.

l'elemento padre con i figli sarà di colore di sfondo giallo.

```
--CSS--
.test {
  background: yellow;
}

--HTML--
<p> <button id="has">has() </button> </p>
<ul>
  <li>list item 1
    <ul>
      <li>list item 1-a</li>
      <li>list item 1-b</li>
    </ul>
  </li>
  <li>list item 2</li>
</ul>

--SCRIPT--
$('#has').click(function() {
  $('li').has('ul').addClass('test');
});
```

## Jquery e CSS

### Stili css

Oltre a ricevere attributi, contenuti e valori all'interno delle nostre pagine HTML è possibile inoltre applicare stili o ricevere i valori dei nostri selettori. Per ricevere il valore di una determinata proprietà CSS si usa il `css()` con questa sintassi:

```
$(selettore).css(proprietà);
```

Se invece vogliamo aggiungere o modificare una proprietà CSS la sintassi è leggermente differente:

```
$(selettore).css(proprietà, valore);
```

E' possibile inserire più proprietà contemporaneamente come l'esempio riportato di seguito. Tutti i valori e le proprietà devono essere racchiuse tra apici e tutto il contenuto deve essere racchiuso tra le parentesi {}

```
$("#p").css({"background-color": "black", "font-size": "20px", "color": "white"});
```

## Jquery e CSS

### Classi css

Ai nostri elementi HTML possiamo aggiungere o rimuovere classi CSS per la formattazione di essi ( le regole vanno create nel foglio di stile interno o esterno ). Per fare questo esistono tre metodi contrapposti :

<code>\$(selettore).addClass(Nome Classe)</code>	Aggiunge una classe all'elemento esistente
<code>\$(selettore).removeClass(Nome Classe)</code>	Rimuove una classe dall'elemento esistente
<code>\$(selettore).toggleClass(Nome Classe)</code>	Aggiunge o rimuove una classe all'elemento esistente

Il metodo `toggleClass` analogamente agli altri `toggle` (es. `hide` e `show`) cambia in modo automatico lo stato della classe controllando se è già esistente sull'elemento o meno. E' possibile aggiungere o rimuovere più di una Classe con un'unica istruzione semplicemente inserendo più Nomi delle classi separati da uno spazio:

`$(selettore).addClass(NomeClasse NomeClasse)`

## Jquery e CSS

### Esempio

```
<-- CSS -->
.testorosso{color:red;}
.grassetto{font-weight:bold}

<--Script-->

$("#aggiungi").click(function(){
    $("p").addClass("testorosso grassetto")
});

<--HTML-->

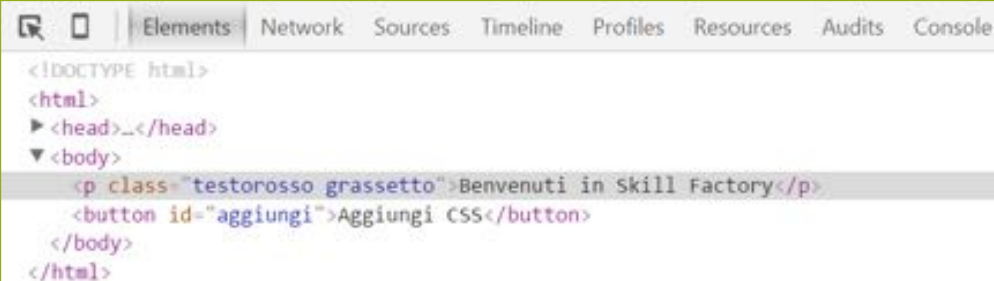
<p>Benvenuti in Skill Factory</p>
<button id="aggiungi">Aggiungi CSS</button>
```

Benvenuti in Skill Factory

Aggiungi CSS

**Benvenuti in Skill Factory**

Aggiungi CSS



```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <p class="testorosso grassetto">Benvenuti in Skill Factory</p>
    <button id="aggiungi">Aggiungi CSS</button>
  </body>
</html>
```

## Jquery e CSS

### Muoversi all'interno del DOM con jQuery

Con JQuery attraverso l'utilizzo di alcuni metodi specifici, possiamo manipolare gli elementi all'interno del DOM.

Metodo **find()**

Esso ci permette di selezionare alcuni elementi contenuti all'interno del nostro selettore.

```
$("#intro").find("p").hide();
```

L'istruzione permette di nascondere tutti i paragrafi presenti all'interno dell'elemento con ID "intro".

## Jquery e CSS

### Muoversi all'interno del DOM con jQuery

Metodo **children()**

Tale metodo, restituisce i figli della selezione su cui è applicato. I figli considerati sono solo quelli immediati; non vengono selezionati gli elementi ulteriormente annidati.

Nell'esempio a lato, viene colorato di rosso lo sfondo dei capitoli, ma non del sottocapitolo perchè non è un figlio diretto dell'elemento individuato nel selettore.

```
<--Script-->
$("#indice").children().css("background-color", "red");

<--HTML-->

<ul id="indice">
<li>Primo capitolo</li>
<li>Secondo capitolo
    <ul>
        <li>Sottocapitolo</li>
    </ul>
</li>
<li>Terzo capitolo</li>
</ul>
```

## Jquery e CSS

### Muoversi all'interno del DOM con jQuery

Metodo **parent()**

Il metodo `.parent()` assolve alla funzione contraria di `.children()`, cioè seleziona il genitore dell'elemento considerato.

Nell'esempio l'istruzione produrrà uno sfondo rosso per l'elenco livello-2.  
Poiché non forniamo un'espressione di selettore, l'elemento principale viene incluso in modo inequivocabile come parte dell'oggetto.

```
<ul class="level-1">
  <li class="item-i">I</li>
  <li class="item-ii">II
    <ul class="level-2">
      <li class="item-a">A</li>
      <li class="item-b">B
        <ul class="level-3">
          <li class="item-1">1</li>
          <li class="item-2">2</li>
          <li class="item-3">3</li>
        </ul>
      </li>
    </ul>
  <li class="item-c">C</li>
</ul>
<li class="item-iii">III</li>
</ul>
```

```
$( "li.item-a" ).parent().css( "background-color", "red" );
```

## Jquery e CSS

### Muoversi all'interno del DOM con jQuery

Altri metodi utili:

**next()** seleziona solamente l'elemento immediatamente successivo all'interno dello stesso livello gerarchico;

**nextAll()** invece seleziona tutti i successivi elementi, sempre all'interno dello stesso livello.

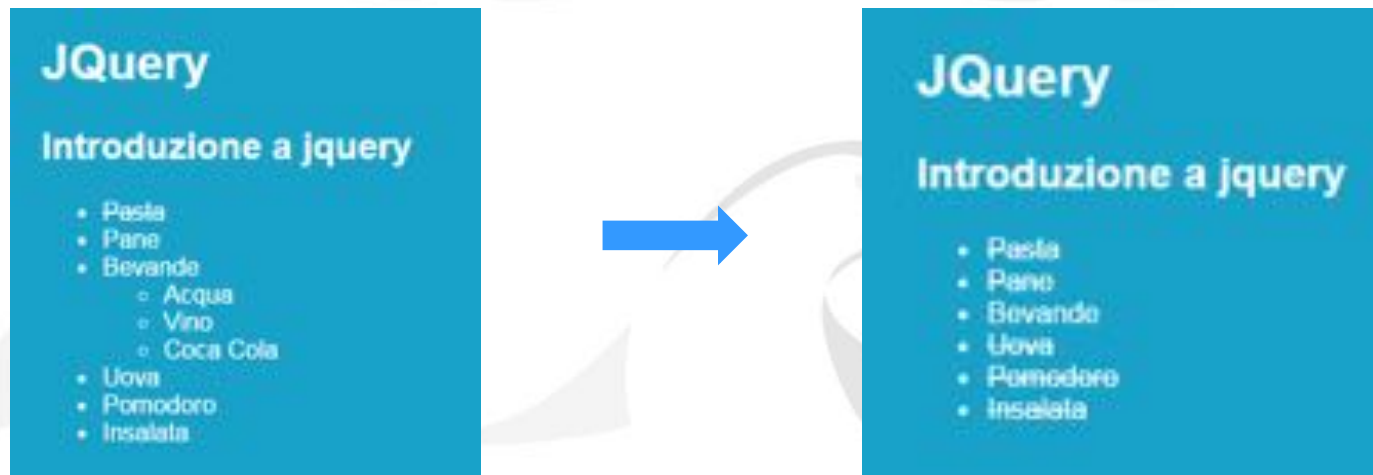
**prev()** e **prevAll()** svolgono rispettivamente il compito di selezionare l'elemento precedente e tutti gli elementi precedenti.



## Esempi

### Modifica grafica di una lista

Si vuole realizzare una lista, che attraverso l'utilizzo del framework JQuery è possibile modificare graficamente. Quando ogni elemento della lista viene cliccato verrà associata un'azione che porterà ad una determinata visualizzazione grafica della stessa.



# HTML



## Esempi

### Codice HTML e CSS

```
<!doctype html>
<!-- questa è la sintassi abbreviata indica al browser il tipo di documento -->
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <!--indica il tipo di codifica dei caratteri è la forma abbreviata usata in html5
  e permette a tutte le lettere accentate della pagine HTML di essere visualizzate correttamente
  -->
  <style>
    body{font-family: arial;padding:40px; margin:0; background: #19a2c9; color:#fff; }
    a{color:#fff210; text-decoration: none; display: inline-block; width:33%;text-align: center}
    hr{ border: 0; height: 1px; background: #ccc; margin-bottom: 40px; }
    .lined{text-decoration: line-through}
  </style>
</head>
<body>
  <h1>jQuery</h1>
  <h2>Introduzione a jquery</h2>

  <ul class="lista">
    <li class="lined">Pasta</li>
    <li>Pane</li>
    <li> Bevande
      <ul class="sotto-lista">
        <li>Acqua</li>
        <li>Vino</li>
        <li>Coca Cola</li>
      </ul>
    </li>
    <li>Uova</li>
    <li>Pomodoro</li>
    <li>Insalata</li>
  </ul>
</body>
</html>
```

## Esempi

### Script

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.4/jquery.min.js"></script>
<script>
    $(document).ready(function(){
        $("li").on('click', function(){
            if($(this).has('ul').length == 0){
                $(this).addClass('lined');
            }
            if($(this).parent().is('.sotto-lista')){
                $(this).remove();
            }
        });
    });
</script>
```

E' possibile far riferimento al framework JQuery utilizzando il tag <script> sia nella sessione <body> sia nella sessione <head> (buona norma) del documento. Di seguito è riportato lo script jquery.

## Esempi

### Validazione di una form

Per poter validare i campi di un form ci interessa estrarre dall'archivio il file jquery.validate.js.

```
<script type="text/javascript" src="jquery.js"></script>  
<script type="text/javascript" src="jquery.validate.js"></script>
```

**jqueryvalidation** dispone di numerose proprietà di controllo che consentono di impostare condizioni specifiche per la validazione di un campo.

**accept** - verifica che in un campo in cui si inserisce il nome ed il percorso di un file, l'estensione di un file sia presente tra quelle specificate, separandole col carattere pipe (|).

**creditcard** - espressione regolare per il controllo di un numero di carta di credito.

**date** - espressione per la verifica di una stringa in formato data.

**dateDE** - verifica che una data sia in formato coerente con la data tedesca (il plug-in "proviene" dalla Germania).

**dateISO** - verifica che il formato di una data rispetti gli standard ISO.

**digits** - verifica che il contenuto di un campo sia un numero intero.

## Jquery e CSS

### Proprietà di jqueryvalidation

**email** - espressione regolare per il controllo di un indirizzo email.

**equalTo** - verifica che il valore di un campo sia uguale al valore specificato in un altro campo.

**max** - verifica che un valore numerico immesso in un campo non sia maggiore di quello specificato.

**maxLength** - accetta un valore numerico che specifica il massimo numero di caratteri che devono essere inseriti.

**min** - verifica che un valore numerico immesso in un campo non sia minore di quello specificato.

**minLength** - accetta un valore numerico che specifica il minimo numero di caratteri che devono essere inseriti.

**number** - booleano, verifica che il contenuto di un campo sia numerico; accetta il punto (.) come separatore dei decimali.

**numberDE** - booleano, verifica che il contenuto di un campo sia numerico; accetta la virgola (,) come separatore dei decimali.

**range** - controlla che il valore .

**rangeLength** - verifica che il numero di caratteri inseriti in un campo siano compresi tra quello minimo e quello massimo specificato.

**remote** - accetta la URL di uno script lato server esterno (ASP, PHP etc...) che effettui il controllo.

**required** - booleano, accetta "true" o "false" a seconda che il campo sia richiesto o meno.

**url** - espressione regolare per il controllo di una URL.

## Jquery e CSS

### rules e message

Jqueryvalidation prevede l'impiego del comando **rules** in cui vengono elencate, campo per campo, tutte le regole.

Con il comando **message** invece indicheremo i messaggi di errore, in cui elencheremo nuovamente i nomi dei campi ed in corrispondenza del tipo di controllo il relativo messaggio di errore, qualora il controllo stesso non sia stato soddisfatto.

```
$(document).ready(function()
{
    $("#modulo").validate(
    {
        rules:
        {
            nome: "required",
            cognome: "required",
            nickname: "required",
            password: "required",
            conferma:
            {
                required: true,
                equalTo: "#password"
            },
            email:
            {
                required: true,
                email: true
            },
            accetto: "required"
        },
        messages:
        {
            nome: "Inserisci il tuo nome!",
            cognome: "Inserisci il tuo cognome!",
            nickname: "Scegli un nickname!",
            password: "Scegli una password!",
            conferma: "La conferma non corrisponde alla scelta della password!",
            email: "Inserisci un indirizzo email valido!",
            accetto: "Non hai accettato i termini del servizio!"
        }
    }
});
```

## Jquery e CSS

### .addMethod

Per poter creare un metodo di validazione personalizzato utilizzeremo:

`jQuery.validator.addMethod (nome, il metodo [, message])`

Esso è costituito da un nome (principalmente un identificativo), una funzione javascript e un messaggio di errore predefinito.

<--Script-->

```
jQuery.validator.addMethod("codfiscale",  
function(value) {  
var regex = /[A-Z]{6}[\d]{2}[A-Z][\d]{2}[A-Z][\d]{3}[A-  
Z]/; return value. match(regex); }, "Inserisci un codice  
fiscale valido");
```

<--HTML-->

```
<p>  
<label>Codice fiscale:</label>  
<input type="text" name="codfis" id="codfiscale"/>  
</p>
```

## Jquery e CSS

### Esempio Form validata

#### Registrazione

Login

Inserisci la login

Email

Inserisci la tua email

Password

Inserisci una password password

Registrati



## Jquery e CSS

### File html sessione head

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Esempio di validazione con JQuery</title>
  <!--Ultima versione di jQuery (minified) -->
  <script src="https://code.jquery.com/jquery-2.1.4.min.js"></script>
  <!-- Ultima versione di jquery.validate (minified) -->
  <script src="http://ajax.aspnetcdn.com/ajax/jquery.validate/1.14.0/jquery.validate.min.js"></script>
  <!-- Ultima versione di bootstrap (minified) -->
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css">
  <!-- codice jQuery -->
  <script type="text/javascript" src="prova_form.js"></script>
  <style>
    #form label.error {
      color: red;
      font-weight: bold;
    }

    .main {
      width: 600px;
      margin: 0 auto;
    }
  </style>
</head>
```

## Jquery e CSS

### File html sessione body

Nell'esempio sono stati utilizzati tag html5 (<label>) e riferimenti a classi bootstrap (form-group).

```
<body>
<!-- Form container -->
<div class="main">
  <h1>Registrazione</h1>

  <!-- form da validare -->
  <form action="#" id="form">
    <div class="form-group">
      <label for="login">Login</label>
      <input type="text" name="login" class="form-control">
    </div>

    <div class="form-group">
      <label for="email">Email</label>
      <input type="text" name="email" class="form-control">
    </div>

    <div class="form-group">
      <label for="password">Password</label>
      <input type="password" name="password" class="form-control">
    </div>

    <div class="form-group">
      <input type="submit" value="Registrati" class="submit" class="form-control">
    </div>
  </form>
</div>
</body>
</html>
```

## Jquery e CSS

File html sessione body

```
$(document).ready(function() {  
    // Selezione form e definizione dei metodi di validazione  
    $("#form").validate({  
        // Definiamo le nostre regole di validazione  
        rules : {  
            // login - nome del campo di input da validare  
            login : {  
                // Definiamo il campo login come obbligatorio  
                required : true  
            },  
            email : {  
                required : true,  
                // Definiamo il campo email come un campo di tipo email  
                email : true  
            },  
            password : {  
                required : true,  
                // Settiamo la lunghezza minima e massima per il campo password  
                minlength : 3,  
                maxlength : 8  
            }  
        },  
        // Personalizziamo i messaggi di errore  
        messages : {  
            login : "Inserisci la login",  
            password : {  
                required : "Inserisci una password password",  
                minlength : "La password deve essere lunga minimo 3 caratteri",  
                maxlength : "La password deve essere lunga al massimo 8 caratteri"  
            },  
            email : "Inserisci la tua email"  
        },  
        // Settiamo il submit handler per la form  
        submitHandler : function(form) {  
            form.submit();  
        }  
    });  
});
```

## Jquery e CSS

### Altri Esempi

Si vuole realizzare una pagina che gestisca la visualizzazione di 3 div attraverso il framework JQuery.

Di seguito è riportato il codice HTML e CSS.



```
<html>
<head>
  <meta charset="UTF-8" />
  <title>Document</title>
  <style>
    body{font-family: arial;padding:40px; margin:0; background: #19a2c9; color:#fff; }
    .panel{float:left; width: 33%; display: inline-block;min-height: 1px}
    .panel p {background: #1482a1; margin:20px;padding:20px;}
    a{color:#fff210; text-decoration: none; display: inline-block; width:33%;text-align: center}
    hr{ border: 0; height: 1px; background: #ccc; margin-bottom: 40px; }
  </style>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.4/jquery.min.js"></script>
</head>
<body>
  <h1>JQuery</h1>
  <h2>Esempi</h2>

  <hr/>

  <p> <a href="#" id="trigger1">Trigger 1</a> <a href="#" id="trigger2">Trigger 2</a>
  <a href="#" id="trigger3">Trigger 3</a><p>

  <div class="panel" id="info1"><p>Info 1</p></div>
  <div class="panel" id="info2"><p>Info 2</p></div>
  <div class="panel" id="info3"><p>Info 3</p></div>

</body>
</html>
```

## Jquery e CSS

### Altri Esempi

Lo script JQuery in tal caso è inserito all'interno della pagina HTML nella sessione <BODY>

```
<script>
$(document).ready(function(){
    $("#trigger1").on('click', function(){
        $("#info1 p").slideToggle();
    });

    $("#trigger2").on('mouseover', function(){
        $("#info2 p").fadeToggle();
    });

    $("#trigger3").on('click', function(){
        $("#info3 p").fadeOut();
    });

    $("#trigger3").on('mouseover', function(){
        $("#info3 p").slideDown();
    });

});
</script>
```

## Jquery e CSS

### Altri Esempi

Lo script JQuery in tal caso è inserito all'interno della pagina HTML nella sessione <BODY>

## JQuery

### Introduzione a Jquery

**Titolo 1** Titolo 2 Titolo 3

#### Pannello 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute inure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# HTML



## Jquery e CSS

### Altri Esempi

#### HTML e CSS

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8" />
  <style>
    body{font-family: arial;padding:40px; margin:0; background: #19a2c9; color:#fff; }
    a{color:#fff210; text-decoration: none; display: inline-block; width:33%;text-align: center}
    hr{ border: 0; height: 1px; background: #ccc; margin-bottom: 40px; }
    ul{padding: 0;margin:0;}
    li{display: inline-block;padding: 5px}
    .panel{display: none; padding:0; margin: 0;}
    .panel.active{display: block}
    li.active{background: #333}
  </style>
</head>
<body>
  <h1>jQuery</h1>
  <h2>Introduzione a jquery</h2>
  <div class="tab-container">
    <ul class="tab-menu">
      <li class="active" data-panel="panel1">Titolo 1</li>
      <li data-panel="panel2">Titolo 2</li>
      <li data-panel="panel3">Titolo 3</li>
    </ul>
    <div class="panel active" id="panel1">
      <h3>Pannello 1</h3>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod ...
    </div>
    <div class="panel" id="panel2">
      <h3>Pannello 2</h3>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod ...
    </div>
    <div class="panel" id="panel3">
      <h3>Pannello 3</h3>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod ...
    </div>
  </div>
</body>
</html>
```



## Jquery e CSS

### Altri Esempi

#### SCRIPT

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.4/jquery.min.js"></script>
<script>
$(document).ready(function() {

    // evento che ascolta ogni click su gli elementi li
    $(".tab-menu li").on('click', function() {

        //stabilisco qual'è il container
        var $currentContainer = $(this).closest(".tab-container");

        //trovo l'elemento corrente nel menu e rimuovo la classe active
        $currentContainer.find(".tab-menu li.active").removeClass("active");

        //aggiungo la classe al nuovo elemento corrente
        $(this).addClass("active");

        //stabilisco qual'è il pannello da mostrare
        // 1) this si riferisce a .tab-menu li
        var panelToShow = $(this).attr("data-panel");

        //trovo il pannello attivo lo nascondo e mostro quello nuovo
        $currentContainer.find(".panel.active").slideUp(400, showPanel);

        //funzione che viene passata dopo lo slideUp come "callback"
        function showPanel() {

            // 2) this si riferisce a .panel.active
            $(this).removeClass('active');

            //creo il selettore per mostrare il nuovo pannello corrente
            $(".tab-menu li[data-panel='"+panelToShow+"']").slideDown(400, function () {

                // 3) this si riferisce al panelToShow
                $(this).addClass('active');
            });
        }
    });
});
</script>
```





# Skill Factory School

La scuola d'esperienza, che certifica le tue competenze...

## Formazione Orientata al Lavoro



### Licenza d'uso

Materiale didattico di proprietà della **Skill Factory**, può essere divulgato gratuitamente per fini non commerciali e deve essere destinato esclusivamente a scopi didattici. Attenzione, in nessun caso, i contenuti di questo materiale didattico, potranno essere modificati senza autorizzazione della **Skill Factory**.

Sponsorizzato da [www.skillbook.it](http://www.skillbook.it) Job Learning community

