APPUNTI DI JQUERY

INDICE:

Le pagine seguenti sono consigliate a chi ha una conoscenza (anche di base) di java script, html, css e soprattutto una grande passione e tanta tanta fantasia e voglia di

creare ed imparare

INTRODUZIONE

jQuery è una libreria di funzioni (framework) Javascript, cross-browser per le applicazioni web, che si propone come obiettivo quello di semplificare la programmazione lato client delle pagine HTML.
Tramite l'uso della libreria jQuery è possibile, con poche righe di codice, effettuare svariate operazioni, come ad esempio ottenere l'altezza di un elemento, o farlo scomparire con effetto dissolvenza.

Con Jquery è possibile usare il framework in tutti progetti senza paura di incappare in incompatibilità nel codice. Infatti, utilizzando la funzione **jQuery.noConflict()** verrà semplicemente rimosso l'alias $, e potremo usare, per esempio, Mootools richiamando questo framewok con $('mioId') e jQuery con jQuery("#mioId"), inoltre jQuery ha un semplice sistema di estensione che permette di aggiungere nuove funzionalità (plugin) oltre a quelle predefinite; la sua diffusione ha fatto in modo che attorno al team di sviluppo ufficiale crescesse una numerosa community che mette a disposizione plugin e supporto di ottimo livello. Infine, e potrebbe sembrare una cosa da poco, perché il motto di jQuery è "Write less, do more", ed effettivamente la sua sintassi sintetica ed efficiente è particolarmente apprezzabile quando si tratta di scrivere svariate linee di codice.

jQuery non ha intenzione di sostituire Javascript: il suo scopo è quello di semplificare la vita dei javascripters, consentendo di scrivere meno codice e, al contempo, ottenere risultati elevati difficilmente ottenibili programmando "a mano" i propri script.

INCLUDERE JQUERY NEI PROPRI PROGETTI
Vi sono due modi per includere JQUERY nei propri progetti:
il primo metodo consiste nello scaricare dal sito jquery.com la librerie per poi includerla in una cartella nel nostro progetto per poi richiamarla.

<script language="javascript" type="text/javascript" src="jquery-1.3.2.min.js"></script>

Il second metodo consiste è quello di utilizzare una copia del framework presente sul network di Google

<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>

Questa soluzione, che lega il funzionamento dei nostri script alla raggiungibilità del servizio, ha il vantaggio che l'utente potrebbe già aver scaricato il file in questione nella cache del browser con conseguente diminuzione del tempo di caricamento della pagina.

JQUERY

La sintassi jQuery prevede sempre l'utilizzo della funzione **jQuery()**, sostituibile utilizzando il simbolo del dollaro **$()**.
Come già detto è un frameworks completamente cross-browser, ma ha una particolarità: gli script che vanno eseguiti vanno eseguiti al caricamento della pagina vanno inseriti dopo il codice HTML con cui jQuery stesso deve interagire, oppure è necessario utilizzare l'evento **load()** oppure **ready()**

ESEMPIO INTERNO DI PAGINA	**ESEMPIO READY**
<script type="text/javascript" src="jquery.js"></script> Codice HTML con cui jQuery deve interagire... <script type="text/javascript"> // Qui lo script... </script>	<script type="text/javascript" src="jquery.js"></script> <script type="text/javascript"> $(document).ready(function(){ $("#testo").css("color", "#FF0000"); }); </script> W jQuery!

N.B. : *Per chi non abbia dimestichezza con gli script Java ricordiamo che la differenza tra load e ready consiste nel fatto che la prima esegue lo script al completo caricamento della pagina, mentre la seconda lo esegue a prescindere dal caricamento e torna quindi molto più utile e frequentemente utilizzata. L'utilizzo dell'una o dell'altra opzione dipende da quando voglio che il mio script venga eseguito e/o sia pronto ad essere attivo* .

I SELETTORI

I selettori permettono di chiamare quale elemento della pagina sia quello interessato alla nostra funzione, sostituendo in effetti la sintassi del DOM document.getElementById("testo").
La sintassi del selettore è: $("[nome elemento]")
Il selettore permette anche di accedere ad una classe $(".[nome classe]") o a tutte le occorrenze di uno stesso tag $("[nome occorrenze]")
Nel caso di tag nidificati possiamo accedervi specificando la sequenza degli elementi.
Esempi:

SELETTORE	**CLASSE**	**OCCORRENZA (tutti i div)**	**NIDIFICATO (solo lo span nel div)**
$("#testo")	$("div.testo")	$("div")	$("div span")

GLI EVENTI

La gestione degli eventi con jQuery e uno degli aspetti piu complessi per chi non conosce i meccanismi di creazione, propagazione e cattura degli eventi.
Un evento è una notifica creata dal browser in risposta a qualche cambiamento nel DOM. In molti casi gli eventi sono creati in seguito ad azioni dell'utente: ridimensionamento della finestra del browser, cliccare su un pulsante o muovereil mouse, per fare alcuni esempi. Quando l'evento viene creato, si dice che viene lanciato o sparato (in inglese fired).
Un esempio (come abbiamo visto parlando della sintassi) può essere:

```
$(document).ready( function() {     // Il mio codice } ) ;
```


In questo caso l'evento intercetta caricamento, anche parziale, della pagina.
I principali eventi sono:

ready - caricamento anche parziale della pagina.
load - caricamento completo della pagina.
click - click su di un elemento.
dblclick- doppioclick su di un elemento.
mouseover - passaggio del mouse di un elemento.
mouseout - uscita del mouse da un elemento.
keypress- Pressione di un tasto sulla tastiera.
error - gestione di un eventuale errore.
scroll - scrolling della pagina.
resize - ridimensionamento della pagina.
focus - focus su di un elemento di un form.
submit - invio di un form.

I vantaggi di usare jQuery, anziché i tradizionali eventi JavaScript, sono molteplici. Innanzitutto jQuery gestisce molti più eventi, permettendo così una gestione più granulare e specifica di quanto accade nel browser. In secondo luogo le API offerte da jQuery sono in generale più portabili di quelle JavaScript.

GLI ATTRIBUTI

Uno dei gruppi di metodi più potenti in jQuery è sicuramente quello relativo agli attributi.
Il metodo per ricavare ed impostare gli attributi degli elementi è **.attr()**
Tramite questo metodo possiamo, ad esempio leggere o impostare l'attributo di un link (href), capire se un checkbox è selezionato, ottenere il valore di una select, accedere alla classe css di un tag html, ecc.
La sintassi è:
$('SELETTORE').attr('NOME_ATTRIBUTO'); per ottenere il valore dell'elemento
$('SELETTORE').attr('NOME_ATTRIBUTO', 'NUOVO_VALORE'); per sovrascrivere il valore dell'elemento HTML selezionato

COMANDI JQUERY

Gli elementi di una pagina restituiti dalla funzione $() possono essere elaborati mediante apposite funzioni, dette comandi o metodi, che possono comporsi sequenzialmente, separando funzioni consecutive con un singolo punto.
Vediamo un esempio concreto di quello che intendo dire: apriamo il nostro editor HTML (vedere il corso per costruire un sito web) e scrivere il seguente codive:

```
<html>
<head>
<title>Esempio 1</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
<style type="text/css">
#contenuto {
        width:300px;
        height:300px;
        border:#000 2px solid;
        background-color:#F00;
        color:#FFF;
}
</style>
</head>
<body>
<div id="contenuto"> test </div>
<script type="text/javascript">
 $("#contenuto").fadeOut(6000).fadeIn(1000).fadeOut("slow").fadeIn(“fast”);
</script>
</body>
</html>
```


Salviamo il nostro documento e apriamolo nel browser. Vediamo che il quadrato rosso (**#contenuto**) prima scompare impiegando 6000ms **(*.fadeOut(6000)*)** , poi riappare impiegando 1000ms (***fadeIn(1000)*)** per sparire nuovamente lentamente (***fadeOut("slow")*)**e ricomparire velocemente (***fadeIn(“fast”)*)**
Se avessimo scritto solamente $("#contenuto").fadeOut(6000). Il nostro quadrato si sarebbe limitato a sparire

Note: *L’attributo slow per default è impostato a 400ms.*

EVENTI E COMANDI INSIEME

Vediamo ora come possiamo far interagire gli eventi ed i comandi insieme.
Riprendiamo il nostro esempio di prima e cambiamo lo script con il seguente:

```
<script type="text/javascript">
 $("#contenuto").hover(function() { $("#contenuto").fadeOut(600); });
 </script>
```


Vediamo che quando apriamo la pagina, se clicchiamo con il mouse sopra il nostro quadrato questo scomparirà.
Possiamo anche fare in modo che un evento possa operare su un oggetto che non sia se stesso. Scriviamo il codice:

```
<html >
<head>
<title>Esempio 3</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
<style type="text/css">
#contenuto {width:300px; height:300px; border:#000 2px solid;  background-color:#F00; color:#FFF; }
.spostami { width:100px; border:#000 2px solid; }
</style>
<script>
$(document).ready(function(){
  $(".spostami").click(function(){
  $("#contenuto").animate({marginLeft:"200px",marginTop:"100px"});
  });
});
</script>


</head>
<body>
<input type="submit" name="spostami" id="spostami" value="SPOSTAMI" class="spostami" />
<div id="contenuto"> test </div>
</body>
</html>
```

Adesso se noi clicchiamo sul nostro tasto “SPOSTAMI” il quadrato si sposterà di 200px dal margine sinistro e di 100px dal margine superiore

UN PO' DI PRATICA

Come abbiamo visto , usare jquery è molto semplice e non richiedo molto sforzo per poter ottenere dei risultati interessanti. Adesso un esempio per impratichirci ed imparare nuovi metodi.
Nei due esempi precedenti abbiamo visto come far scomparire un elemento sul quale clicchiamo sopra e come spostare un elemento cliccando su di un altro.
Ma se volessimo alternare la scomparsa o la ricomparsa di un elemento?
Nulla di più facile.: jquery possiede tre distinti metodi per gestire lo sliding degli elementi: **slideDown()**, **slideUp()** e **slideToggle()**.

- **slideDown()** mostra un elemento nascosto facendolo scivolare dall'alto verso il basso
- **slideUp()** nasconde un elemento visibile facendolo scivolare dal basso verso l'alto
- **slideToggle()** varia in base alla visibilità dell'elemento.

Quest' ultimo è quello che useremo per fare in modo che il nostro box compaia o scompaia a seconda che sia visibile o meno il relativo comando.
La sintassi, comune per tutti gli elementi è

$(elemento).slideNome(durata, callback)

dove durata è la durata dell'animazione e callback: la funzione da eseguire quando l'animazione è completa. Scriviamo il nostro codice:

```
<html>
<head>
<title>Esempio 3</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
<style type="text/css">

#box {width:400px; height:200px; background-color:#CCCCCC; border:1px solid #666666; }
</style>
<script type="text/javascript">
    function alterna () { $("#box").slideToggle("slow"); }
</script>
</head>
<body>
<a href="javascript:alterna();">ALTERNA </a>
<div id="contenitore">
        <div id="box">Box con testo</div>
</div>
</body>
</html>
```

NOTE. *Se voglio inserire un callback, ad esempio un allert alla fine dell'animazione la funzione sarà:*

function alterna () { $("#box").slideToggle("slow",function() {alert('slideToggle completato');}); }

JQUERY E HTML

jQuery consente di interagire dinamicamente sul codice HTML di uno o più elementi presenti una pagina attraverso diverse funzioni.
La prima che analizziamo è la funzione HTML(): il metodo .html() non solo permette di interagire dinamicamente sul codice HTML della pagina ma anche di leggere e prelevare il contenuto html di un elemento.
La duplice funzione che svolge ne fa un ottimo alleato in tutti quei casi in cui lo sviluppatore ha necessità di modificare "a volo" la pagina web, in relazione ad esempio ad un azione dell'utente o ad una chiamata Ajax.
La sintassi è:

$(elemento).html();

Facciamo un esempio nel quale scriveremo un testo all'interno di un div che originariamente, nel codice html risulta vuoto. Scriviamo il seguente codice:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Esempio 5</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>

<style type="text/css">
    #testo {width:400px;  height:100px; border:1px solid #666666; font-size:15px; }
</style>
</head>
<body>

<script type="text/javascript">
$(document).ready(function(){
  $("#testo").html("CI SONO!");
});
</script>

<div id="testo"></div>

</body>
</html>
```

Quando apriremo la nostra pagina all'interno del browser vedremo che il div testo, che nel codice HTML non contiene nulla, conterrà la scritta CI SONO!
Possiamo modificare il codice facendo in modo che la scritta compaia alla pressione di un link (come nell'esempio precedente della scomparsa del div)
Sostituiamo **$(document).ready(function(){ $("#testo").html("CI SONO!"); });** con **function scrivi () { $("#testo").html("CI SONO!"); }**
Ed aggiungiamo nella pagina il codice **SCRIVI ** prima del tag .

Come detto in precedenza il metodo HTML() permette anche di leggere il contenuto di un elemento. Vediamo subito un esempio:

```
<html >
<head>
<title>Esempio 5</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
<style type="text/css">
        #testo {  width:400px; height:100px; border:1px solid #666666; font-size:15px; }
</style>

</head>
<body>
        <script type="text/javascript">
        function leggi(id){
           var contenuto = $('#'+id).html();
           alert('Il contenuto è:\n\n'+contenuto);
        }
        </script>

<div id="testo">
Io ne ho viste cose che voi umani non potreste immaginarvi,
navi da combattimento in fiamme al largo dei bastioni di Orione,
e ho visto i raggi B balenare nel buio vicino alle porte di Tannhäuser.
E tutti quei momenti andranno perduti nel tempo
come lacrime nella pioggia.
È tempo di morire.

</div>
<a href="javascript:leggi('testo');">LEGGI </a>

</body>
</html>
```


Come si vede, se premo al link LEGGI viene richiamata la funzione leggi e come riferimento l'elemento della pagina testo. Il contenuto del div in oggetto viene memorizzato nella variabile contenuto. Quindi chiamiamo un allert che mostra il contenuto

NOTE: *posso anche scrivere var contenuto = $("testo").html(); ma perdo la possibilità di applicare la stessa funzione ad altri elementi.*

Un altro metodo che ci interessa, per la gestione del codice HTML è append() che, come il suo nome lascia ad intendere, aggiunge del testo ad un testo già esistente.
Scriviamo il codice:

```
<html >
<head>
<title>Esempio6</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
<style type="text/css">
        #testo { width:400px; border:1px solid #666666; font-size:15px; }
</style>

</head>
<body>

        <script type="text/javascript">
        function aggiungi () {
        $("#testo").append(" AGGIUNTO!<br/>");
         }
        </script>

<a href="javascript:aggiungi();">AGGIUNGI </a>
<div id="testo"></div>

</body>
</html>
```

Vediamo che quando premo AGGIUNGI viene aggiunta all'interno del div #testo la parola AGGIUNTO!

NOTE: *Posso anche inserire dei tag HTML come in questo caso che ho aggiunto, dopo la parola AGGIUNTO! Il tag
 che va accapo*
*Alre funzioni interessanti sono .**prepend()** che inserisce del testo prima del testo già presente nel layer HTML, .**remove()** che rimuove un elemento e .**clone()** copia l'elemento e lo ripropone; il loro utilizzo è simile a quello di **.append()**.*

JQUERY E CSS

jQuery consente anche di lavorare con i fogli di stile in modo diretto.
Per fare ciò si utilizza il metodo .**css()**. Come nel caso .html() anche .css() permette sia di cambiare il valore di un contenuto che di leggerlo.
Il metodo è molto semplice e la sua sintassi è:

$(elemento).css(comando CSS, valore);

Apriamo il nostro editor e scriviamo:

```
<html >
<head>
<title>Esempio 7</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
<style type="text/css">
        #testo { width:400px; border:1px solid #666666; font-size:15px; }
</style>

</head>
<body>
            <script type="text/javascript">
            function colorami () {
            $("#testo").css("background-color","#FF0000");
             }
            </script>
<a href="javascript:colorami();">colora </a>
<div id="testo">COLORAMI</div>

</body>
</html>
```

Vediamo che quando premiamo su colora lo sfondo del nostro div #testo diventa rosso.
Se volessimo cambiare il bordo, basta scrivere: **$("#testo").css("border","8px solid #112266");** nel metodo .css().

NOTE: *per usare il metodo .css() è necessario conoscere i CSS… ovviamente!!!*

EPPUR SI MUOVE

Una delle caratteristiche più apprezzate di Jquery è la possibilità di muovere elementi della pagina. Per fare questo si usa il metodo .animate().
La sintassi di .animate() è:

$(elemento).animate(stile, durata, easing, callback)

- **stile:** un valore di testo contenente i nomi e i valori delle proprietà CSS da animare (larghezza,altezza,posizionamento,margini,padding, visibilità, ecc.)
- **durata:** la durata dell'animazione, espressa in millisecondi
- **easing:** l'algoritmo di easing che regola la progressione dell'animazione. jQuery supporta nativamente i valori swing (predefinito) e linear,
- **callback:** la funzione che viene eseguita quando l'animazione è completa.

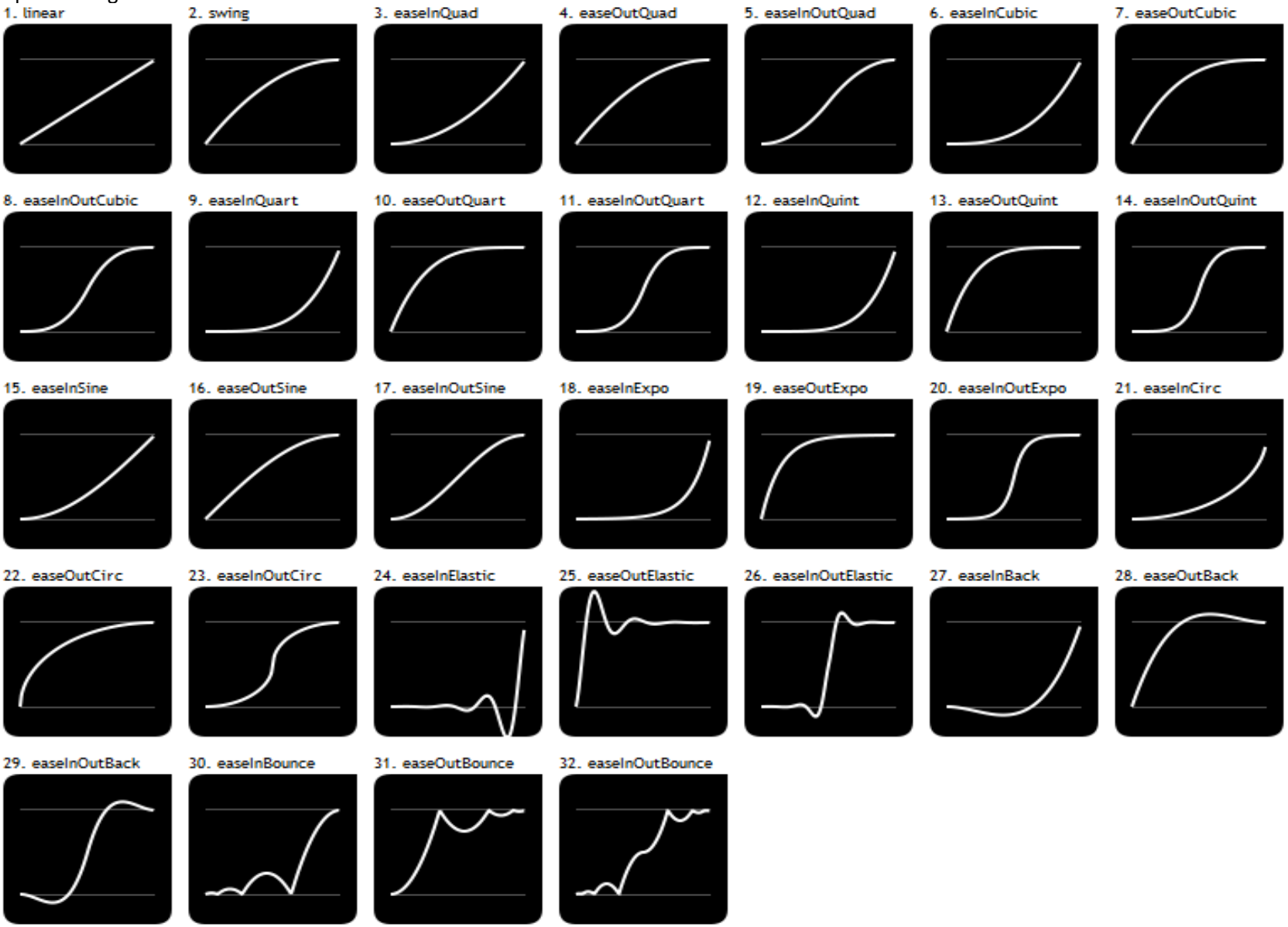
NOTE: *Nei valori CSS, questo metodo supporta una sintassi incrementale, ad esempio +=numero per aumentare il valore corrente e -=numero per diminuirlo. I numeri senza unità di misura vengono interpretati come pixel in quelle proprietà CSS che supportano le unità di misura.*

Facciamo un sempio:

```
<html>
<head>
<title>Esempio 8</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
<style type="text/css">
        #box1 {   width:50px; height:50px; border:1px solid #666666;font-size:15px; position:absolute; margin-top:51px; }
        #box2 {   width:50px; height:50px; border:1px solid #666666;font-size:15px; float:left; }
</style>
</head>
<body>
        <script type="text/javascript">
        $(document).ready(function(){
          $(".btn1").click(function(){
          $("#box1").animate({marginLeft:"200px",marginTop:"100px"});
          $("#box2").hide(6000);
          });
        });
        </script>
<button class="btn1">PREMIMI</button>
<div id="box2">io sparisco</div>
<div id="box1">io mi muovo</div>
</body>
</html>
```

Come vediamo da questo esempio il nostro quadrato si muove cambiando le proprietà di **marginleft** e **margintop**.

[1] tipi di easing



PAROLA CHIAVE "THIS"

In JavaScript, come nella maggior parte dei linguaggi di programmazione orientati agli oggetti, "this" è una parola chiave speciale che viene utilizzato nei metodi per fare riferimento all'oggetto sul quale viene richiamato un metodo.
La parola chiave "this" può far riferimento a due contesti differenti:

- un elemento del DOM
- un oggetto jQuery

Normalmente la parola chiave "this" si riferisce a un elemento DOM e viene utilizzata all'interno di una funzione come ad esempio durante l'associazione a un evento. La parola chiave THIS ha la sintassi $(this).


```
<html >
<head>
<title>Esempio 9</title>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>

<script type="text/javascript">
$(document).ready(function(){
 $(".menu a").hover(
   function() {$(this).next("em").animate({opacity: "show"}, "slow");},
   function() {$(this).next("em").animate({opacity: "hide"},"slow");}
 );
});
</script>

</head>
<body>
<ul class="menu">
<li><a href="#">Passami il mouse sopra</a><em>Quarda cosa mi succede...</em></li>
<li><a href="#">Adesso il mouse sopra a me</a><em>...e quarda cosa mi succede... </em></li>
<li><a href="#">Adesso che sei da me torna su</a><em>...ed ammira l'effetto</em></li>
</ul>
</html>
```

NOTE: *Il DOM (**D**ocument **O**bject **M**odel) è un'API, indipendente dalla piattaforma, che descrive la struttura di un documento HTML (e XML), con il quale i costruttori di pagine per il Web possono accedere e manipolare tutti gli elementi della pagina stessa. Rifacendoci alla sua sigla, il DOM è un modello che descrive come i diversi oggetti di una pagina sono collegati tra loro. È importante sottolineare che il DOM è indipendente dalla piattaforma, ovvero è un'interfaccia definita dal W3C per essere lo strumento universale per tutti i creatori di pagine Web. Questo significa che il DOM definito dal W3C è indipendente dal tipo di browser, di versione e di sistema operativo.*

CLONARE, COPIARE, ELIMINARE

Ci sono dei metodi che più di altri possono risultare utili per lavorare su DOM; questi metodi sono:

- .clone()
- .replaceWith()
- .replaceAll()
- .remove()
- .empty()

Il metodo .**clone**, come dice la parola stessa, clona un'intera struttura DOM senza la necessità di un ulteriore caricamento della pagina; questo metodo permette di aggiungere l'oggetto clonato ad un altro oggetto (**.appendTo**), oppure di inserirlo dopo (**.insertAfter**).

Il metodo **.replaceWith(),** sostituisce un elemento con un altro oppure con del semplice codice HTML.

Il metodo **.replaceAll()** sostituisce gli elementi selezionati con nuovi elementi HTML. La sintassi è $(content).replaceAll(selettore), dove content specifica il contenuto da inserire (deve contenere tag HTML) e selettore specifica gli elementi da sostituire.

Il metodo **.remove()** rimuove un oggetto dalla pagina.

Il metodo **.empty()** serve per svuotare un elemento di tutto il suo contenuto.

NOTE:
*.**clone** non copia gli eventi associati agli elementi della collezione se non viene indicato esplicitamente passandogli **true** come argomento: **.clone(true)***
***.remove:** posso passare, eventualmente, un selettore per filtrare gli elementi da eliminare: ad esempio **.remove("#oggetto_filtrato")**.*
*La differenza tra **.remove()** ed **.empty()** è che, mentre la prima funzione rimuove l'elemento selezionato dalla pagina e quindi non è più presente, la seconda funzione semplicemente lo svuota lasciandolo però all'interno della pagina stessa.*

Esempio:

```
<html>
<head>
<title>Esempio 10</title>
<style type="text/css">
#oggetto {font-family:"Arial Black", Gadget, sans-serif; font-size:12px; width:150px; border:#000 1px solid; margin:10px; float:left }
</style>
<script language="javascript" type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>

<script type="text/javascript">
  function clonamenu () { $("#oggetto").clone().appendTo("#contenitore"); }
  function clonamenu2 () { $("#contenitore").clone().insertAfter("#contenitore"); }
  function sostituisci () { $("#oggetto").replaceWith("<p> SOSTITUITO </p>") }
  function svuota () {  $("#contenitore").empty(); }
  function rimuovi () { $("#oggetto").remove(); }
</script>

</head>
<body>
  <a href="javascript:clonamenu();">Clona con .appendTO</a> <br/>
  <a href="javascript:clonamenu2();">Clona il contenitore .insertAfter</a> <br/>
  <a href="javascript:sostituisci();">Sostituisci</a> <br/>
  <a href="javascript:svuota();">Svuota</a> <br/>
  <a href="javascript:rimuovi();">Rimuovi menu</a><br/>
<hr>
<div id="contenitore">
 <div id="oggetto">IO SONO UN OGGETTO </div>
</div>
</html>
```

NOTE FINALI

Questa è solo una breve introduzione a JQuery, e non vuole ne essere un corso ne, tantomeno, un manuale; ci siamo limitati ad una breve introduzione ed a spiegare alcuni metodi principali in modo da aiutare chi si avvicina per la prima volta al mondo jquery.
Nel' introduzione è stato dato per scontata la conoscenza dei principi base della programmazione, dell' html e dei CSS.
Per chi fosse interessato ad approfondire l'argomento alcuni link di seguito:

http://jquery.com/
https://api.jquery.com/
http://www.html.it/guide/guida-jquery/
http://www.filoweb.it/tutorials/muove.html
http://www.jqueryitalia.org/