2

2 La llibreria jQuery.

En aquesta lliçó aprofundirem en l'estudi de la interactivitat que l'usuari efectua a través de la interfície gràfica del programa. Utilitzarem una llibreria javascript anomenada jQuery per dotar a les pàgines web d'animacions i interactivitat augmentada utilitzant el paradigma de la programació en resposta a events.

2.1 Introducció a la llibreria jQuery.

jQuery és una llibreria JavaScript que facilita l'ús del llenguatge JavaScript en les pàgines web fent més fàcil l'escriptura del programa. La principal avantatge és que esdevé una capa d'abstracció damunt el Javascript que permet amagar les diferències en la interpretació del codi que en fa el Javascript de cada navegador. És a dir, tot el que fem amb jQuery és universal, serveix per a qualsevol navegador

jQuery facilita moltes tasques comuns, que requeririen moltes línies de codi JavaScript per aconseguir-ho, a través de funcions predissenyades que es poden invocar amb una única línia de codi.

La llibreria jQuery té les següents capacitats:

- manipulació i modificació del HTML/DOM de la pàgina (informació i maquetació).
- manipulació i modificació del CSS de la pàgina (disseny).
- programació dels events del HTML (interacció de l'usuari).
- efectes i animacions preprogramats.

Al tractar-se d'una llibreria de codi obert, es disposen de molts plugins per fer la majoria de les tasques comuns necessàries en el disseny d'una web.

Podeu trobar molta documentació oficial a http://learn.jquery.com .

2.1.1 Com Afegir jQuery a la pàgina Web?

Hi ha dues maneres per començar a utilitzar jQuery en el nostre lloc web:

- descarregar la llibreria jQuery des del web http://jquery.com/download/
- vinculant jQuery des d'una altra web, com Google

Descarregant jQuery

La llibreria jQuery és un únic fitxer JavaScript, jquery-3.6.0.min.js, que descarregarem de http://jquery.com/download/ i endreçarem a una carpeta javascript dins el nostre lloc web.

Per poder-lo usar dins la nostra pàgina HTML cal incloure la llibreria amb:

```
<script src="javascript/jquery-3.6.0.min.js"></script>
```

i sempre es farà abans d'incloure el nostre programa o altres llibreries.

Vinculant jQuery

Si no volem descarregar i allotjar la llibreria en el nostre servidor web, aquesta es pot incloure des d'un CDN (Content Delivery Network). D'aquesta manera, si hem visitat una web que la feia servir, s'haurà descarregat a la memòria cau del nostre ordinador i, per tant, el navegador no la tornarà a descarregar.

Tant Google com Microsoft allotgen la llibreria jQuery.

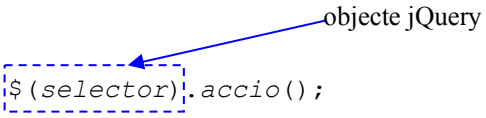
Google CDN:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js">
</script>
```

2.1.2 Sintaxi jQuery

La sintaxi de jQuery es basa en fer la **selecció** d'elements HTML per fer-hi alguna **acció**.

La sintaxi bàsica és:



- un signe `$` per definir/accedir a jQuery
- un *(selector)* per seleccionar (query) uns elements HTML determinats. *$(selector)* és un objecte jQuery, amb propietats i mètodes.
- una *accio()* jQuery que cal fer als elements HTML seleccionats. Aquesta acció pot estar parametritzada. Si cal fer quelcom després que acabi l'acció sobre els elements seleccionats, podem passar una funció a l'acció:

```
$(selector).accio(function(){
 // escrivim què cal fer després d'aplicar l'acció als elements seleccionats
});
```

Exemples,

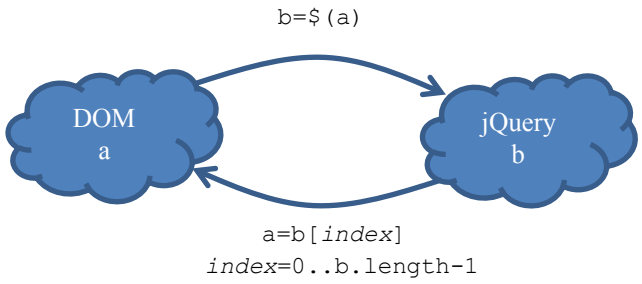
```
$("p").hide();          // amaga tots els elements paràgraf, <p>.
$(".test").show();      // mostra tots els elements amb class="test".
$("#test").fadeOut();// fa desaparèixer, gradualment, l' element amb id="test".
$(this).slideDown(); // desplega (canvia la propietat height, gradualment) l'element actual.
```

Es pot guardar l'objecte jQuery (un conjunt d'elements HTML seleccionats) en una variable i així podem demanar-li diferents accions al mateix objecte jQuery sense haver de tornar a fer la selecció:

```
var x= $("p");     // x és un objecte jQuery que conté tots els elements <p> de la pàgina
x.length           // és un enter que indica quants elements conté la selecció
x[index]           // objecte DOM (element HTML) que ocupa la posició index en la col·lecció
```

Sempre que vulguem tractar un objecte del DOM+BOM com a objecte jQuery farem `$(objDOM)`, per exemple:

```
var x=document.getElementById("identificador"); // x és un objecte DOM
var y=$(x);       // l'hem convertit en un objecte jQuery
y.fadeOut();      // el fem desaparèixer
x=y[0];           // tornem a recuperar l'objecte DOM original
```



Es pot iterar tots els elements de la col·lecció amb l'acció `each()`, per exemple

```
x.each(function (index){
    console.log($(this).text());          // mostra el text del paràgraf actual
    console.log(x.eq(index).text());      // el mateix.
});
```

és a dir, en cada iteració, **`this`** és l'objecte DOM que ocupa la posició `index` dins l'objecte jQuery.

2.1.3 Quan es pot utilitzar el jQuery?

Només es pot utilitzar jQuery quan el navegador ha descarregat totalment la pàgina HTML que estem visitant. Per això, cal posar el jQuery dins la funció que dóna resposta a l'event "*document ready*":

```
$(document).ready( function(){
   // el codi del programa va aquí...

} );
```

funció que s'executa quan el document està preparat

on `document` és un objecte BOM

2.2 Accés i modificació del DOM amb jQuery.

2.2.1 Selectors.

Amb els selectors jQuery podem seleccionar elements HTML,

```
$(selector)
```

on `selector` pot ser:

- un `String` javascript contenint un selector CSS,
- objectes del DOM o BOM
- **`this`**.

Veiem alguns exemples de `selector`

Exemple de selector	Descripció
`$("#nom")`	Selecciona l'element amb `id="nom"`
`$(".intro")`	Selecciona tots els elements amb la `class="intro"` aplicada
`$("p.intro strong")`	Selecciona tots els elements `strong` que estan dins un paràgraf amb `class="intro"` aplicada.
`$(document)`	Selecciona l'element BOM document
`$(this)`	Selecciona l'element DOM actual
`$("[href]")`	Selecciona tots els elements amb un atribut `href`
`$("a[target='_blank']")`	Selecciona tots els elements `<a>` amb un atribut `target` amb valor igual a `"_blank"`

selector amb filtres	Descripció
`$("p:first")`	Selecciona el primer element `<p>`
`$("p:last")`	Selecciona el darrer element `<p>`
`$("ul li:first")`	Selecciona el primer element `<li>` del primer element `<ul>`
`$("ul li:first-child")`	Selecciona el primer element `<li>` de tots els elements `<ul>`
`$(":button")`	Selecciona tots els elements `<input>` amb `type="button"`
`$(":checked")`	Selecciona tots els element `<input> checked`
`$(":selected")`	Selecciona tots els element `<option> selected`
`$("tr:even")`	Selecciona tots els elements `<tr>` parells: 0, 2, 4, ...
`$("tr:odd")`	Selecciona tots els elements `<tr>` imparells: 1, 3, 5, ...

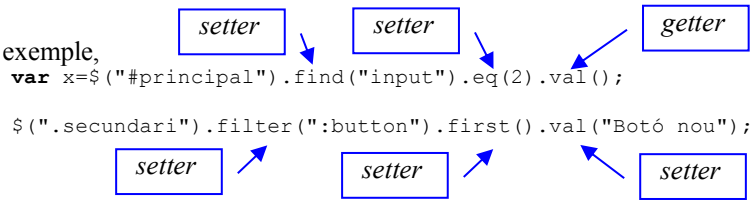
També es poden filtrar (reduir el conjunt d'elements seleccionats) amb accions que aplicarem posteriorment a la selecció. També podrem tornar a seleccionar dins de la selecció actual:

acció filtre	**setter o getter**	**Descripció**
`filter("selector")`	*setter*	redueix el conjunt als que compleixen amb "selector" Exemple: `$("li").filter(":even")==$("li:even")`
`has("selector")`	*setter*	redueix el conjunt als que tenen un descendent que compleixen amb "selector" Exemple: `$("li").has("ul")` `//` tots els li que tenen un `ul`
`not("selector")`	*setter*	esborra del conjunt els element que no compleixen amb "selector"
`eq(n)`	*setter*	redueix el conjunt a només l'element que està a la posició n
`first()`	*setter*	primer element
`last()`	*setter*	darrer element
acció reselecció		**Descripció**
`find("selector")`	*setter*	agafa els descendent de cada element de la selecció actual que compleixen amb "selector"
`children("selector")`	*setter*	agafa els fills directes (primer descendent) de cada element de la selecció actual que compleixen amb "selector"
`next()`	*setter*	agafa el següent germà de cada element de la selecció
`prev()`	*setter*	agafa el germà anterior de cada element de la selecció
`parent()`	*setter*	agafa l'element pare de cada element de la selecció

Totes aquestes accions són *setters*.

Els *setters* sempre retornen un objecte jQuery, per tant, es poden encadenar. Els *getters* retornen el valor que se'ls demana, per tant no es poden encadenar, és a dir, el *getter* , si n'hi ha, ha de ser el darrer de la cadena.

Per exemple,



Podeu consultar la llista complerta a http://api.jquery.com/category/selectors/

2.2.2 Afegir, canviar i esborrar elements HTML.

Hi ha accions per llegir o canviar l'HTML del document:

Acció	setter o getter	Descripció
`html()`	*getter*	retorna el contingut HTML del primer element seleccionat. Inclou les marques HTML

`html("contingut")`	*setter*	Dóna (passant el codi HTML com argument) el contingut HTML als elements seleccionats. Inclou les marques HTML. Així, aquestes dues sentències són equivalents:
		`$("#resultat").html("resultat = "+resultat);`
		`document.getElementById("resultat").innerHTML="resultat = "+resultat;`

`text()`	*getter*	retorna el contingut text de tots els elements HTML seleccionats. No inclou les marques HTML. És l'únic *getter* que s'aplica a tots els elements de la col·lecció.
`text("text")`	*setter*	Dóna (passant el text com argument) el contingut text als elements HTML seleccionats. No inclou les marques HTML

`val()`	*getter*	retorna el `value` del primer element `input` seleccionat. Així,aquestes dues sentències són equivalents:
		`var nom = $("#nom").val();`
		`var nom = document.getElementById("nom").value;`

`val("valor")`	*setter*	Dóna (passant el valor com argument) el `value` als `input` seleccionats.
`attr("atribut")`	*getter*	retorna el valor de l'atribut, passat com argument, del primer element seleccionat.
`attr("atribut", "valor")`	*setter*	Dóna (passant el valor com segon argument) el valor a l'atribut per a tots els elements seleccionats.
`is(":selected")`	*getter*	retorna si el primer element de l'objecte jQuery està seleccionat per l'usuari.
`width()`	*getter*	retorna l'amplada, en píxels, directament en format número.
`width(numero)`	*setter*	Dóna l'amplada, en píxels, donat el número de píxels.
`height()`	*getter*	retorna l'alçada, en píxels, directament en format número.
`height (numero)`	*setter*	Dóna l'alçada, en píxels, donat el número de píxels.
`position()`	*getter*	retorna la posició (objecte amb les propietats, `left` i `top`) respecte el primer ascendent posicionat diferent de `static`.
`data("clau")`	*getter*	retorna el valor associat a la clau del primer element seleccionat.
`data("clau",valor)`	*setter*	Dóna el valor (qualsevol objecte javascript) associat a la clau (un string) a tots els elements seleccionats.
`append("contingut")`	*setter*	Insereix contingut HTML al final del contingut afectat dels elements seleccionats (últim fill)
`prepend("contingut")`	*setter*	Insereix contingut HTML al principi del contingut afectat dels elements seleccionats (primer fill)
`after("contingut")`	*setter*	Insereix contingut HTML just després dels elements seleccionats
`before("contingut")`	*setter*	Insereix contingut HTML just abans dels elements seleccionats
`remove()`	*setter*	elimina els elements seleccionats i tots els seus elements fills

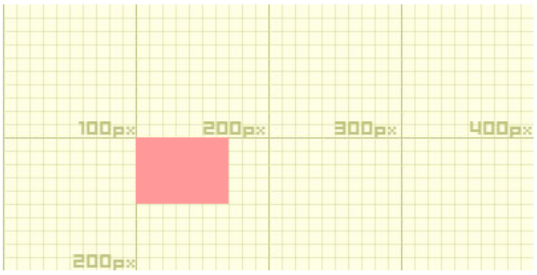
Exemple 1. Com agafar (*get*) o donar (*set*) el contingut d'elements HTML.

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>Propietat position</title>
<style type="text/css">
#capa1 {
       position:relative;
       width:400px;  height:200px;
       background: url(imatges/pixel_grid.jpg);
}
#capa2 {
       position:absolute;
       left:100px; top:100px;
       width:70px; height:50px;
       background-color: #FF9999; /* vermellós  */
}

</style>
</head>

<body>
<div id="capa1">
  <div id="capa2"></div>
</div>
<p>position.Left = <span></span>, position.Top = <span></span></p>

<script src="javascript/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function(){
  var posicio=$("p").first().find("span");
  var s='<p>width = <span id="amplada"></span>, height = <span id="alçada"></span></p>';
  posicio.first().text($("#capa2").position().left);
  posicio.last().text($("#capa2").position().top);
  $("body").append(s);
  $("#amplada").text($("#capa2").width());
  $("#alçada").text($("#capa2").height());
});
</script>
</body>
</html>
```



position.Left = 100, position.Top = 100

width = 70, height = 50

http://ssh.eupmt.tecnocampus.cat/~jou/LM/exemples/exemple3e.html

2.2.3 Llegir i canviar propietats CSS.

Hi ha accions per llegir o canviar el CSS del document:

Acció	setter o getter	Descripció
`addClass("classe")`	*setter*	Afegeix una o més classes (separades per un espai en blanc) als elements seleccionats
`removeClass("classe")`	*setter*	Esborrar una o més classes als elements seleccionats
`toggleClass("classe")`	*setter*	Alterna entra afegir i esborrar classes als elements seleccionats

`css("nomPropietatCSS")`	*getter*	retorna el valor de la propietat CSS del primer element seleccionat. El nom de la propietat es pot donar com a string o com a identificador amb notació *camelCase*:
		`var colorFons=$("p").css("background-color");`
		`colorFons==$("p").css(backgroundColor);`

`css("nomPropietatCSS","valor")`	*setter*	Dóna el valor de la propietat CSS a tots els elements seleccionats. El nom de la propietat es pot donar com a string o com a identificador amb notació *camelCase*.
`css({"nomPropietat_1":"valor_1", "nomPropietat_2":"valor_2" })`	*setter*	Es poden activar més d'una propietat en la mateixa instrucció posant-les entre claus, com si fos un objecte. Una alternativa millor es tenir uns `class` preparada per afegir-la (`addClass`) o treure-la (`removeClass`).
`hasClass("classe")`	*getter*	Retorna `true` o `false` si l'element té assignada la classe

Exemple 2. Quan l'usuari fa clic en el botó, els paràgrafs canviaran d'estil CSS:

```
<!DOCTYPE html>
<html>
<head>
<title>Exemple 5.</title>
<meta charset="utf-8" />
<script src="javascript/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function(){

  $("#boto").on("click",function(){
    $("p").css({"background-color":"yellow","font-size":"200%"});
  });

});
</script>
</head>

<body>
<p>Això és un paràgraf.</p>
<input type="button" id="boto" value="Canviar l'estil del praràgraf" />
</body>
</html
```

http://ssh.eupmt.tecnocampus.cat/~jou/FP/capitol_3/exemples/ex05.html

2.2.4 Efectes jQuery: mostrar/amagar, animacions.

La següent taula llista les accions jQuery més usades per crear efectes d'animació en els elements HTML seleccionats. Totes aquestes accions són *setters*, per tant, es poden encadenar.

Acció	Descripció
`show()`	Mostra els elements. Es pot subministrar un primer argument que seria el temps, en milisegons, que ha de trigar en aparèixer els elements. Per defecte, l'acció és instantània. L'acció consisteix en animar les propietats `width` i `height` dels elements seleccionats.
`hide()`	Amaga els elements
`toggle()`	Alterna entre les accions `hide()` i `show()`.
`fadeIn()`	Fa aparèixer, gradualment, els elements. Actua sobre la propietat `opacity` dels elements. Per defecte, ho fa a una velocitat "`slow`".
`fadeOut()`	Fa desaparèixer, gradualment, els elements. Primer el fa transparent mantenint les mides, però, quan ja és totalment transparent (`opacity`==0), també fa que `width` i `height` sigui zero
`fadeTo()`	Canvia el grau d'opacitat, gradualment, fins al valor donat. Cal posar sempre la velocitat (primer paràmetre): "`slow`", "fast" o milisegons, i el grau d'opacitat a assolir (0 = transparent,... , 1 = opac)
`fadeToggle()`	Alterna entre les accions `fadeIn()` i `fadeOut()`
`slideDown()`	Desplega com una persiana (mostra) els elements. Actua sobre la propietat `height`.
`slideUp()`	Plega com una persiana (amaga) els elements
`slideToggle()`	Alterna entre les accions `slideUp()` i `slideDown()`
`delay(milisegons)`	S'espera el nombre de milisegons donats per continuar amb la cadena d'animacions aplicada a la selecció
`animate()`	Executa una animació personalitzada en els elements. Es tracta de canviar, al mateix temps, un conjunt arbitrari de propietats CSS dels elements seleccionats. Només es poden animar propietats de valor numèric (px, em, %, etc.), per tant, no es pot animar el color ni les transformacions. El nom de la propietat es pot escriure com un string (entre "" o ' ') o com un identificador usant la notació camelCase El nou valor de la propietat que volem animar es pot donar de forma absoluta o relativa: "`+=10px`" o "`-=1px`".

Podeu consultar la llista complerta a http://api.jquery.com/category/effects/

Totes les accions tenen dos paràmetres opcionals, la velocitat i una funció que es cridarà després que s'hagi aplicat l'efecte a tots els elements seleccionats. Per exemple:

```
$(selector).hide(velocitat,callback);
```

els elements seleccionats són amagats amb una *`velocitat`* determinada, que pot ser els valors: "`slow`", "`fast`" o un valor en milisegons.

Un cop a acabat l'efecte, es crida a la funció *`callback`*.

Exemple 3. Apareixen i desapareixen elements, simultàniament i seqüencialment.

```
<!DOCTYPE html>
<html>
<head>
<title>Apareixen i desapareixen elements, gradualment: fadeToggle().</title>
<meta charset="utf-8" />
<style>
.capa{
       width:80px; height:80px; text-align:center; font-size: 3em;
}
</style>
</head>
<body>
<p>Demostració de l'efecte fadeToggle(). Totes les capes desapareixen al mateix temps.<br />
<input type="button" id="boto1" value="animació simultànea"></p>
<p>Primer farem desaparèixer i tornar aparèixer la capa 1. Ens esperarem 2 segons i farem que
la capa 2 es plegui. Simultàneament, la capa 3 s'anirà plegant durant 5 segons <br />
<input type="button" id="boto2" value="animació seqüencial"></p>
<div id="div1" class="capa" style="background-color:red;">1</div>
<div id="div2" class="capa" style="background-color:green;">2</div>
<div id="div3" class="capa" style="background-color:blue;">3</div>
<script src="javascript/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function(){
  $("#boto1").on("click",function(){
    $(".capa").fadeToggle(3000);
  });
  $("#boto2").on("click",function(){
    $("#div1").fadeOut(1000)    // primer desapareix, triga 1 segon
              .delay(2000)      // s'espera 2 segons durant els quals width=height=opacity=0
              .fadeIn(1000,     // primer recupera instantàneament l'amplada i alçada i després
                                // es va fent opac en 1 segon
              function(){
                $("#div2").slideUp(1000); // quan acaba el fadeIn de la capa 1,
              });                         // comença a plegar-se la capa 2
    $("#div3").slideUp(5000); // animació simultànea a #div1
  });
});
</script>
</body></html>
```

http://ssh.eupmt.tecnocampus.cat/~jou/LM/exemples/exemple3a.html

Exemple 4. Plegar i desplegar una "persiana": `slideToggle()`

```
<!DOCTYPE html>
<html>
<head>
<title> Plegar i desplegar.</title>
<meta charset="utf-8" />
<script src="javascript/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function(){
  $("#flip").on("click",function(){
    $("#panell").slideToggle("slow");
  });
});
</script>
<style type="text/css">
#panell, #flip {
       padding:5px; text-align:center;
       background-color:#eeeecc; border:solid 1px #c3c3c3;
}
#panell {
       padding:50px;  display:none;
}
</style>
</head>
<body>
<div id="flip">Fes clic per desplegar i plegar la persiana (panell)</div>
<div id="panell">Hola!</div>
</body></html
```

http://ssh.eupmt.tecnocampus.cat/~jou/LM/exemples/exemple3b.html

Exemple 5. Animació.

```
<!DOCTYPE html>
<html>
<head>
<title>Animació.</title>
<meta charset="utf-8" />
<style>
.capa{
       position:absolute;  // per poder utilitzar les propietats top i left
       height:100px;width:100px;
}
</style>
</head>

<body>
<input type="button" id="boto1" value="animació simultànea">
<input type="button" id="boto3" value="animació seqüencial">

<pre>
Per defecte, tots els elements HTML tenen una posició estàtica i, per tant, no es poden moure.
Per canviar la posició cal canviar la propietat CSS position a un valor relative, fixed, o
absolute
</pre>
<div class="capa" style="background:#bfb;">Hola</div>

<script src="javascript/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function(){
  $("#boto1").on("click",function(){
    $("div").animate({  // animació simultànea de totes les propietats (com identificadors)
      left:"+=250px",   // valor relatiu
      opacity:"0.5",    // valor absolut
      height:"+=150px",
      width:"250px",
      fontSize:"3em"    // el nom de la propietat com a identificador amb notació camelCase
    },2000);
  });
  $("#boto3").on("click",function(){
    $("div").animate({  // animació simultànea de totes les propietats (com strings)
        "left":"+=250px",
        "opacity":"0.5",
        "height":"+=150px",
        "width":"250px"
    }, 2000);
    $("div").animate({    // Segona animació. Començarà quan acabi l'anterior perquè
        "font-size":"3em" // afecta al mateix element
    },2000);
  });
});
</script>
</body>
</html>
```

http://ssh.eupmt.tecnocampus.cat/~jou/LM/exemples/exemple3c.html

Hi ha una llibreria jQuery especialitzada en enriquir la Interfície Gràfica d'Usuari de les pàgines web. És la llibreria http://jqueryui.com/

2.3 Programació en resposta a events de jQuery.

Totes les interaccions dels usuaris de la nostra web es correspon amb un event. Les interaccions de l'usuari, normalment, es fan amb el ratolí o el teclat.

Veurem els events més importants:

event	descripció	a quins elements pot afectar
`click`	el ratolí és al damunt de l'element i el botó del ratolí es prem i es deixa de prémer. Cada `click` equival a un `mousedown` més un `mouseup`.	qualsevol element HTML
`dblclick`	el ratolí és al damunt de l'element i el botó del ratolí es prem i es deixa de prémer dues vegades seguides amb poc espai de temps.	
`mousedown`	el ratolí és al damunt de l'element i el botó del ratolí es prem.	
`mouseup`	el ratolí és al damunt de l'element i el botó del ratolí es deixa de prémer.	
`mousemove`	el ratolí és al damunt de l'element i el ratolí es mou dins de l'element.	
`mouseenter`	el ratolí ha entrat (dins les vores) de l'element	
`mouseleave`	el ratolí ha sortit (fora dels vores) de l'element	
`keydown`	s'ha premut una tecla qualsevol del teclat	l'element que té el focus, típicament el document o qualsevol camp editable d'un formulari.
`keyup`	s'ha deixat de prémer una tecla qualsevol del teclat	
`focus`	quan l'element pren el focus	camps de formularis: `<input>` `<select>` `<textarea>`, etc. i els enllaços `<a>`
`blur`	quan l'element perd el focus	
`change`	quan l'element canvia el valor de la propietat `value`	`<input>`, `<textarea>` i `<select>`
`load`	quan un element (i tots els elements que contingui) s'han carregat a memòria.	elements amb una URL associada: `<img>`, `<script>`, `<iframe>` i l'objecte `window`
`ready`	quan el DOM del document s'ha carregat	`document`

A totes aquestes accions se'ls hi passa una funció de callback que esdevé el manegador de l'event,

```
$("p").on("click",function(){
    $(this).hide();
});
```

o bé,

```
$("p").on("click",manegadorClic);

function manegadorClic(){
    $(this).hide();
});
```

Podeu consultar la llista complerta a http://api.jquery.com/category/events/

Exemple 6. Quan l'usuari fa clic en un camp `input` d'un formulari (li dóna el focus), aquest canvia el color de `background`.

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

http://ssh.eupmt.tecnocampus.cat/~jou/FP/capitol_3/exemples/ex11.html

Tenim diferents possibilitats a l'hora de gestionar els events:

- un sol event,

```
$("p").on("click", function(){
    console.log( "s'ha clicat <p>" );
});
```

- varis events i un sol manegador,

```
$("div").on( "mouseenter mouseleave", function() {
    console.log( "el ratolí ha entrat o sortit de la div" );
});
```

- varis events i varis manegadors,

```
$("div").on({
    mouseenter: function() {
        console.log( "el ratolí ha entrat a la div" );
    },
    mouseleave: function() {
        console.log( " el ratolí ha sortit de la div " );
    },
    click: function() {
        console.log( "s'ha clicat la div" );
    }
});
```

- passant dades al manegador de l'event,

```
$("p").on("click",{dada: "Hola"}, function(e){
    console.log( "dada de e: " + e.data.dada); // dada de e: "Hola"
});
```

- deixar d'escoltar l'event,

```
$("p").off("click");
```

2.3.1 Obtenció d'informació de l'event (objecte `Event`)

Per exemple, per poder donar resposta amb una acció als events clic a tots els paràgrafs d'una pàgina cal fer el següent:

```
$("p").on("click",function(e){
   // aquí escrivim l'acció que s'ha d'executar quan succeeixi l'event
});
```

El paràmetre `e` és un objecte `event` que porta informació sobre el mateix:

objecte `event`	Descripció
`e.pageX`	és la coordenada X del ratolí respecta al document quan ha succeït l'event del ratolí
`e.pageY`	és la coordenada Y del ratolí respecta al document quan ha succeït l'event del ratolí
`e.which`	serveix per a events de ratolí i teclat i indica la tecla concreta (número associat a la tecla igual que el `keyCode`) o el botó del ratolí concret que s'ha premut (1 botó esquerra, 2 botó del mig i 3 botó de la dreta)
`e.keyCode`	és el codi numèric de la tecla premuda
`e.data()`	`data` és l'objecte javascript passat al manegador de l'event.
`e.target()`	és l'element HTML on ha succeït l'event del teclat
`e.preventDefault()`	perquè no es propagui l'event per defecte en l'element HTML
`e.stopPropagation()`	avorta la propagació de l'event cap als elements HTML ascendents.

Exemple 7. Veure la posició del ratolí i veure el codi de la tecla premuda.

```
<!DOCTYPE html>
<html>
<head>
<title> Posició del ratolí i codi de la tecla.</title>
<meta charset="utf-8" />
</head>

<body>
<p>Passa el ratolí pel damunt del rectangle i veuràs la posició del ratolí</p>
<p>Prem qualsevol tecla i veuràs el seu codi numèric</p>
<div id="posicioRatoli" style="width:200px; height:100px;
                               border:thin solid #000;
                              position:relative; top:0; left:0;"></div>

<script src="javascript/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function(){
  $("#posicioRatoli").on("mousemove",function(e){
        $("#posicioRatoli").text("x = "+e.pageX+" , y = "+e.pageY);
  });
  $(document).on("keyup",function(e){
    $("#posicioRatoli").text("key = "+e.which);
  });
});
</script>

</body>
</html>
```

http://ssh.eupmt.tecnocampus.cat/~jou/LM/exemples/exemple3d.html