

Licence STIC – IUT de Marne-la-Vallée  
02/04/2015  
Cours de jQuery

## *Cours 2*

# *Événements et effets avec jQuery*

# Sources

---

- Cours de Jean-Loup Guillaume

[http://jlguillaume.free.fr/www/documents/teaching/ntw1213/LI385\\_C5\\_Jquery.pdf](http://jlguillaume.free.fr/www/documents/teaching/ntw1213/LI385_C5_Jquery.pdf)

- Cours de programmation web avancée de Thierry Hamon

<https://perso.limsi.fr/hamon/PWA-20122013/Cours/JQuery.pdf>

- *jQuery : écrivez moins pour faire plus !*, de tit\_toinou

<http://openclassrooms.com/courses/jquery-ecrivez-moins-pour-faire-plus>

- *jQuery, Le guide complet*, de Guillaume Allain et Timothy Stubbs

- *Javascript & Ajax pour les nuls*, d'Andy Harris

# ÉVÉNEMENTS

<https://api.jquery.com/category/events/>

# Événements

## Premier événement :

`$(document).ready(...)` : quand le DOM est prêt

`≠ onLoad` en Javascript : quand tous les éléments sont chargés (images...)

## Autres événements :

`blur, focus, load, resize, scroll, unload, beforeunload, click, dblclick, mousedown, mouseup, mousemove, mouseover, mouseout, mouseenter, mouseleave, change, select, submit, keydown, keypress, keyup, error`

# Associer des événements (jQuery≥1.7)

```
// associer une fonction à un événement
$("div").on("click",
    function() {
        $(this).text("code HTML : "+$(this).html())
    });

// arrêter d'exécuter l'événement
$("div").on("click",
    function() {
        $(this).text("code HTML : "+$(this).html());
        $("div").off("click")
    });

// exécuter une seule fois (pour chaque objet)
$("div").one("click",
    function() {
        $(this).text("code HTML : "+$(this).html())
    });
```

# Associer des événements (jQuery<1.7)

```
// associer une fonction à un événement
$("div").bind("click",
  function() {
    $(this).text($(this).html()) + "test"
  });

// arrêter d'exécuter l'événement
$("div").bind("click",
  function() {
    $(this).text("test" + $(this).html());
    $("div").unbind("click")
  });
```

# Associer des événements dans le futur (jQuery<1.7)

Un événement n'est appliqué qu'aux éléments qui existent :

- Si un `<div>` est créé après l'ajout de l'événement : pas cliquable.
- Avec `live()`, même si le `<div>` est créé plus tard, il sera cliquable.
- `die()` détruit tous les événements `live()`.

```
// attacher un événement même dans le futur
$("div").live("click", fn);

// détacher les événements créés avec live
$("div").die("click", fn);
```

# Associer des événements dans le futur (jQuery≥1.7)

```
// attacher un événement même dans le futur
$("document").on("click", "div",
    function() {
        $(this).text("test" + $(this).html());
    }
)
```

→ On met trois paramètres en entrée de la fonction `on`

# Répétition d'événements

Le même événement peut être créé plusieurs fois :

**tous les événements seront exécutés.**

```
<a href="">clic</a>
```

```
<script>
```

```
  $("a").click(function(event) {  
    alert(event.type);  
  });
```

```
  $("a").click(function(event) {  
    alert(event.pageX + ", " + event.pageY);  
  });
```

```
</script>
```

# Attributs de l'objet event

`type` : nom de l'événement exécuté

`target` : objet qui a exécuté l'événement

`currentTarget` : = `this`

`pageX` et `pageY` : position de la souris

Autres : <https://api.jquery.com/category/events/event-object/>

`altKey`, `bubbles`, `button`, `cancelable`, `charCode`,  
`clientX`, `clientY`, `ctrlKey`, `currentTarget`, `data`,  
`detail`, `eventPhase`, `metaKey`, `offsetX`, `offsetY`,  
`originalTarget`, `pageX`, `pageY`, `relatedTarget`,  
`screenX`, `screenY`, `shiftKey`, `target`, `view`, `which`

# Attributs de l'objet event

Exemple avec événement lié au déplacement de la souris, actif sur tout le document :

```
<div id="log"></div>
```

```
<script>
```

```
$(document).on('mousemove',function(e) {  
    $("#log").text(e.pageX + ", " + e.pageY);  
});
```

```
</script>
```

## Mise en pratique :

- 1) ajouter un élément d'id «log» dans la page
- 2) insérer le code ci-dessus
- 3) où se trouve le point (0,0) ?
- 4) selon le même principe, affichez les codes clavier correspondant aux touches pressées

# Déclenchement automatique d'un événement

## Fonction trigger

```
<button>#1</button>
<button>#2</button>
<div>
  <span>0</span> clics.
</div>
<div>
  <span>0</span> clics.
</div>
```

```
$("#button:first").click(
  function () {
    update($("#span:first"));
  });

$("#button:last").click(
  function () {
    $("#button:first")
      .trigger('click');
    update($("#span:last"));
  });

function update(j) {
  var n = parseInt(j.text(), 10);
  j.text(n + 1);
}
```

# Déclenchement automatique d'un événement

Fonction `triggerHandler` pour ne pas exécuter le comportement par défaut

```
<button id="old">
  trigger
</button>

<button id="new">
  triggerH
</button>

<input type="text"
value="Focus"/>
```

```
$("#old").click(function() {
  $("input").trigger("focus");
});

$("#new").click(function() {
  $("input").
    triggerHandler("focus");
});

$("input").focus(function() {
  $("<p>Focus</p>").
    appendTo("body");
});
```

# Blocage du comportement par défaut

```
// modifier le comportement par défaut  
$("div").triggerHandler("click");
```

```
// empêcher le comportement par défaut  
function clickHandler(e) {  
e.preventDefault();  
}
```

```
// similaire à :  
function clickHandler(e) {  
return false;  
}
```

# Propagation des événements

Exemple : menu déroulant multi-niveaux.

Un clic se propage sur tous les objets associés :

- si on clique sur `<li> Niveau 3 : item 2</li>` alors on clique aussi sur le `<li>` du niveau 2 et celui du niveau 1.
- on a donc trois alertes.
- la propagation est « ascendante ».

```
$(document).ready(function() {  
    $("li").click(function () {  
        alert($(this).html());  
    });  
});
```

```
<ul>  
<li> Niveau 1 : item 1</li>  
<li> Niveau 1 : item 2  
    <ul><li> Niveau 2 : item 1</li>  
        <li> Niveau 2 : item 2  
            <ul><li> Niveau 3 : item 1</li>  
                <li> Niveau 3 : item 2</li></ul>  
        </li></ul>  
</li>  
</ul>
```

# Blocage de la propagation des événements

On peut stopper la propagation des événements :

- `stopPropagation()` ;
- ou faire `return false;` (attention cela peut bloquer d'autres choses)

```
$(document).ready(function () {  
    $("li").click(function (e) {  
        alert($(this).html());  
        e.stopPropagation();  
    });  
});
```

Voir aussi :

- `isPropagationStopped`
- `stopImmediatePropagation`
- `isImmediatePropagationStopped`

# EFFETS

<https://api.jquery.com/category/effects/>

# Apparition et disparition

```
// montrer un élément
$("div").show();

// montrer un élément lentement (slow=600ms)
$("div").show("slow");

// cacher un élément rapidement (fast=200ms)
$("div").hide("fast");

// inverser (montrer ou cacher) en une durée fixée
$("div").toggle(100);
```

# Fin d'un effet

```
// fait disparaître l'objet lentement
// une fois la disparition terminée, on réaffiche l'objet
$("div")
  .hide("slow", function() {
    $("div").show("slow");});

$("div").hide();

$("a").click(function() {
  $("div").show("fast", function() {
    $(this).html("show div");});});
});
```

# Effet personnalisé

`.animate(options, durée, transition, complete, ...)` :

- Options : ensemble de propriétés CSS.
- Transition : comment se déroule l'animation (linéaire ou pas).
- Complete : callback exécuté après la fin de l'animation.
- ...

```
// réduction de la largeur à 90%,  
// ajout d'une bordure bleue de largeur 5px et  
// changement d'opacité. Le tout en 1s.
```

```
$("#div").animate({  
  width: "10%",  
  opacity: 0.5,  
  borderWidth: "5px"},  
  1000);
```

# Enchaînement d'animations

Par défaut les animations sont effectuées l'une à la suite de l'autre.

Modifiable en utilisant `queue: false`.

```
// enchaînement des animations

// modification du style, puis de la largeur
// et enfin de l'opacité
$("div")
.animate({border: "5px solid blue"},2000)
.animate({width: "20%"},2000)
.animate({opacity: 0.5},2000);

// animations simultanées

$("div")
.animate({border: "5px solid blue"},{queue:false,
    duration:100})
.animate({width: "20%"}, {queue:false, duration:2000})
.animate({opacity: 0.5},2000);
```