JQUERY

LI288 – web et développement web

jQuery

- Librairie Javascript qui permet de :
  - Simplifier les taches de base en Javascript.
  - Accéder à des partie d'une page HTML.
  - Modifier l'apparence et le contenu de la page.
  - Créer et modifier des événements.
  - Animer une page.
  - Faire de l'AJAX.
  - Eviter les problèmes de compatibilité entre navigateurs.

jQuery

- Projet open-source, bien supporté et documenté.
  - http://jQuery.com/
- Ne se substitue pas à la connaissance de Javascript
  - Tout ne se fait pas naturellement en jQuery.
  - Eviter de tout "Jqueriser".
- Beaucoup d'autres librairies similaires :
  - Mootools, Prototype, Dojo, script.aculo.us, …
  - Jquery est efficace, légère, bien documentée

Philosophie jQuery

- Sélectionner une partie du document.
- Agir dessus

- Objet jQuery = ensemble de nœuds du DOM
  - C'est-à-dire un ensemble de balises du document.
- Les objets jQuery se créent avec la fonction $().
  - $("div") retourne tous les "div" du document.
  - $(" ") crée une nouvelle balise div.

Philosophie jQuery

- Sélectionner une partie du document.
- Agir dessus

```
// récupérer tous les div du DOM puis les cacher
$("div").hide();

// Simuler un clic sur tous les boutons
$(":button").click();

// Créer un <span> et l'ajouter à la fin du body
$("<span/>").appendTo("body");
```

Inclure du jQuery

- Télécharger le fichier jQuery.js
  - Il existe un version jQuery-min.js plus légère
  - Gain de bande passante mais fichier non lisible.

- Inclusion du fichier (comme tout javascript) :
  - En tant que script externe :
    - <script type="text/javascript" src="jQuery.js"></script>
  - Depuis le site de jQuery :
    - src="http://code.jquery.com/jquery-1.8.2.min.js"

Un exemple simple

- On veut ajouter une classe c1 à tous les div en javascript.

```
<script type="text/javascript">
function addClassC1() {
   var divs = document.getElementsByTagName("div");
   for (var i = 0; i < divs.length; i++) {
     var div = divs[i];
     div.setAttribute("class", "c1");
   }
}
</script>

<body onLoad='addClassC1();'>
```

Un exemple simple

- En jQuery :
  - $("div") permet de sélectionner tous les div.
  - La méthode .addClass() permet d'ajouter une classe à un objet.

```
/* Solution avec fonction */
function addClassC1() {
  $("div").addClass("c1");
}
<body onLoad='addClassC1();'>

/* Solution directe */
<body onLoad='$("div").addClass("c1");'>
```

Un exemple simple

- En jQuery :
  - <body onLoad='$("div").addClass("c1");'>

- Problèmes :
  - Il faut modifier la partie HTML
  - Tout doit être chargé (y compris les images) :
    - L'affichage va changer à l'exécution de la fonction.

Un exemple simple

- Événement disponible après la création du DOM et avant le chargement des images
  - $(document).ready(f)
  - $(f) : identique mais plus court/moins lisible
  - jQuery(f) : pas de confusion avec une d'autres librairies
- Attention à ne rien exécuter avant…

```
<script type="text/javascript">
$(document).ready(addClassC1);
// $(addClassC1);
// jQuery(addClassC1);
</script>
```

Un exemple simple

- Utilisation de fonction anonyme

```
/* version simple */
$(document).ready(addClassC1);

/* avec une fonction anonyme */
$(document).ready(
  function(){
    $("div").addClass("c1");
  }
);
```

Retour des fonctions

- Les fonctions retournent des objets jQuery :
  - Possibilité d'enchaîner les appels de fonction.
  - Attention à la lisibilité !

```
$("div")                // sélection de toutes les balises div
   .show()              // rendre tous les objets visibles
   .addClass("main")    // ajouter la classe main
   .html("Hello jQuery"); // modifier le contenu
```

Besoin d'aide

- http://jQuery.com/
  - Tout y est :
    - Le code sources
    - La doc sur tous les paramètres et toutes les fonctions.
    - Des exemples.
    - Plein de plugins.
    - …

Dans la suite

- Des exemples pour :
  - Sélectionner des objets.
  - Ajouter des effets d'animations.
  - Utiliser les événements.
  - Faire des requêtes "à la AJAX".
  - La librairie jQuery user interface.

JQUERY
COMMENT SÉLECTIONNER

LI288 – web et développement web

Sélecteur

- Principe de base :
  - Un sélecteur retourne un tableau d'objets jQuery :
    - $("*") retourne toutes les balises
    - $("div") est un tableau contenant tous les de la page.
    - $("div").length permet de connaître le nombre de div dans la page.

Sélection restreinte

- Possibilité de sélectionner (comme en CSS) :
  - Par type de bloc.
  - Par identifiant (attribut id d'une balise).
  - Par classe (attribut class d'une balise).

```
// <div>Test</div>
$("div")

// <span id="test">JL</span>
$("#test")

// <ul class="test">JL</ul>
$(".test")
```

Sélection restreinte

- Possibilité de combiner les sélecteurs.

```
// tous les divs de classe main
$("div.main")

// le tableau d'identifiant data
$("table#data")

// objets d'identifiant "content" ou de classe "menu"
// attention à la position des guillemets
$("#content, .menu")
```

Sélection dans un contexte

- Sélection dans un contexte :
  - $(*recherche*, *contexte*)
  - Sélectionne d'abord le *contexte* (une partie du DOM) puis effectue la *recherche* à l'intérieur.

```
// comme en CSS :
$("div .header").hide();

// ou chercher tous les div
// chercher des balises de class header à l'intérieur
$(".header", $("div"))
   .hide(); // cacher ces balises
```

Sélection dans un contexte

- Dans certaines situations on a besoin du contexte :
  - On peut attribuer une action sur le contexte
  - Puis agir sur une partie de celui-ci.

```
// si on clique sur un div
// tous les p à l'intérieur vont disparaitre ou apparaitre
$('div')
   .click(function() {
        $('p', this).toggle();
   });
```

Sélecteurs d'ordre

```
// premier paragraphe
p:first

// dernier élément de liste
li:last

// quatrième lien
a:eq(3)

// paragraphes pairs ou impairs
p:even ou p:odd

// Tous les liens à partir (greater than) du quatrième ou avant
   (lower than)
a:gt(3) ou a:lt(4)
```

Sélecteurs de hiérarchie

- Possibilité de sélectionner de manière plus précise :
  - Frère, enfants, parents
  - Utilisation de fonctions

```
// frère suivant
.next(expr)
// frère précédent
.prev(expr)
// frères
.siblings(expr)
// enfants
.children(expr)
// père
.parent(expr)
```

Sélecteurs d'attributs

```
// les éléments <div /> ayant un identifiant
$("div[id]")

// les éléments <div /> avec id "test"
$("div[id='test']")

// les éléments <div /> dont l'id commence par test
$("div[id^='test']")

// les éléments <div /> dont l'id termine par test
$("div[id$='test']")

// les éléments <div /> dont l'id contient test
$("a[href*='test']")
```

Sélecteurs de formulaires

```
// sélectionner les checkbox
$("input:checkbox")

// sélectionner les boutons radio
$("input:radio")

// sélectionner les bouton
$(":button")

// sélectionner les champs texte
$(":text")
```

Sélecteurs de formulaires

```
$("input:checked")
$("input:selected")
$("input:enabled")
$("input:disabled")
```

```
<select name="valeur">
   <option value="1">1</option>
   <option value="2" selected="selected">2</option>
   <option value="3">3</option>
</select>

$("select option:selected").val()
```

Sélecteurs supplémentaires

- Utilisation de is()

```
$("table td")
  .each(function() {
    if ($(this).is(":first-child")) {
      $(this).addClass("firstCol");
    }
  });
```

La fonction each

- appelle une fonction pour chaque élément :
  - $(this) = élément courant
  - i - index de l'élément courant
  - Possibilité de récupérer l'élément sous forme DOM

```
$("table tr")
   .each(function(i){
        if (i % 2==0)
               $(this).addClass("odd");
   });
```

JQUERY COMMENT MODIFIER

LI288 – web et développement web

Modifier le contenu HTML

```
// modifier le contenu de tous les p du document
$("p").html("<div>Bonjour</div>");

// modifier le contenu de div.a en celui de div.c
$("div.a").html($("div.c").html());

// idem pour div.b avec échappement du contenu de div.c
$("div.b").text($("div.c").html());
```

```
<p></p>


<div id="a">Bonjour</div>
<div id="b"><a href="#">Au
   revoir</a></div>
<div id="c"><a href="#">Au
   revoir</a></div>
```

```
<p><div>Bonjour</div></p>


<div id="a"><a href="#">Au
   revoir</a></div>
<div id="b">&lt;a href="#"&gt;Au
   revoir&lt;/a&gt;</div>
<div id="c"><a href="#">Au
   revoir</a></div>
```

Modifier des valeurs

- val() : permet d'obtenir la valeur des objets
- val(valeur) permet de modifier la valeur des objets

```
// obtenir la valeur de la première checkbox cochée
$("input:checkbox:checked").val();

// modifier la valeur d'un champ text de nom txt
$(":text[name='txt']").val("Hello");

// sélectionne une valeur d'un select d'identifiant lst
$("#lst").val("NS");
```

Manipulation de CSS

- Il est possible de modifier les classes des objets :
  - addClass, removeClass, toggleClass
  - hasClass

```
// ajouter et supprimer une classe
$("p").removeClass("blue").addClass("red");

// ajouter si absent ou l'inverse
$("div").toggleClass("main");

// vérifier l'existence d'une classe
if ($("div").hasClass("main")) { //… }
```

Manipulation de CSS

```
// récupérer le style (un argument)
$("div").css("background-color");

// modifier le style (deux arguments)
$("div").css("float", "left");

// modifier le style, version rapide
$("div")
   .css({
        "color":"blue",
        "padding": "1em",
        "margin-right": "0",
        "margin-left": "10px"
   });
```

Insérer des éléments

```
// Ajouter à la fin de l'objet
// sélection des ul et ajout à la fin
$("ul").append("<li>Test</li>");

// création d'un objet <li>
// modification du contenu
// ajout à la fin des ul
$("<li />").html("Test").appendTo("ul");

// Ajouter au début
$("ul").prepend("<li>Test</li>");
$("<li />").html("Test").prependTo("ul");
```

Remplacer des éléments

```
// remplace tous les <h1> par des <div>Test</div>
$("h1").replaceWith("<div>Test</div>");

// replaceAll fonctionne à l'envers
$("<div>Test</div>").replaceAll("h1");

// Sans modifier le contenu :
$("h1").each(function(){
   $(this)
        .replaceWith("<div>"+ $(this).html()+"</div>");
   });
```

Supprimer des éléments

```
// supprimer tous les span de classe names
$("span.names").remove();

// vider tout le contenu de l'objet d'identifiant mainContent
$("#mainContent").empty();

// déplacer un objet par suppression + création
// supprime tous les p et les ajoute à la fin du document
$("p")
   .remove()
   .appendTo("body");
```

Gestion des attributs

```
// ajoute l'attribut href
$("a").attr("href","home.htm");
// <a href="home.htm">...</a>

// sélection de tous les boutons sauf le premier
// changement de l'attribut disabled en le mettant comme celui
   du premier bouton
$("button:gt(0)")
   .attr("disabled",
        $("button:eq(0)").attr("disabled));

// supprime l'attribut disabled de tous les boutons
$("button").removeAttr("disabled")
```

JQUERY
LES EFFETS

LI288 – web et développement web

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);
```

Translation et fading

```
// Translation
$("div").slideUp();
$("div").slideDown("fast");
$("div").slideToggle(1000);

// Fading
$("div").fadeIn("fast");
$("div").fadeOut("normal");

// Fading avec opacité fixée
$("div").fadeTo("fast", 0.5);
```

Fin d'un effet

```
// fait disparaitre 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 50% et changement d'opacité.
// Le tout en 1s.
$("div")
  .animate(
    {width: "50%", opacity: 0.5,},
    1000);

<div style="border:1px solid red">contenu du div</div>
```

Enchainement d'animations

- Par défaut les animations sont effectuées l'une à la suite de l'autre.
- Modifiable en utilisant "queue:false".

```
// enchainement des animations
// modification du style, puis de la largeur et enfin de l'opacité
$("div")
   .animate({width: "20%"},2000)
   .animate({opacity: 0.5},2000);

// animations simultanées
$("div")
   .animate({width: "20%"}, {queue:false, duration:2000})
   .animate({opacity: 0.5},2000);
```

JQUERY
LES ÉVÉNEMENTS

LI288 – web et développement web

Le premier événement

- Chargement de la page
  - $(document).ready(…)
  - Assez similaire à onLoad() :
    - onLoad : quand tout est chargé
    - ready : quand le DOM est prêt (avant chargement des images)

```
$(document).ready(function(){
   ...
});

// similaire à
$(function() {...});
```

Gestionnaire d'événements

- Événements disponibles :
  - 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

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

// exécuter une seule fois (pour chaque objet)
$("div").one("click",
  function() {
    $(this).text("test")
  });

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

Trigger

- Force l'appel aux événements liés à un objet

```
<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);
}
```

TriggerHandler

- triggerHandler = trigger mais le comportement par défaut n'est pas exécuté.
  - Le deuxième bouton ne donne pas le focus à l'input

```
<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");});
```

L'objet Event - attributs

- type :
  - nom de l'événement exécuté
- target :
  - objet qui a exécuté l'événement
  - cf propagation des événements
- currentTarget :
  - = this
- pageX et pageY :
  - position de la souris
- …

Un exemple : position de la souris

- Exemple avec événement lié au déplacement de la souris.
  - Via un événement lié à tout le document.

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

<script>
$(document)
   .bind('mousemove',function(e){
        $("#log")
                .text(e.pageX + ", " + e.pageY);
   });
</script>
```

Remarque

- Un événement (objet, type) 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>
```

JQUERY
FONCTIONNALITÉS "AJAX"

LI288 – web et développement web

Charger du contenu

- Pour mettre du contenu dans un objet :
  - C'est un peu plus simple qu'en "Ajax".

```
// version sans arguments :
// récupère fichier.html et met le contenu dans le div#content
$("div#content").load(
  "fichier.html"
);

// version avec arguments :
// appelle la page fichier.php?nom=guillaume
$("div#content").load(
  "fichier.php",
  {"nom":"guillaume"}
);
```

Charger du contenu

- Avec GET / POST

```
// récupère le fichier fichier.html puis exécute la fonction
$.get(
  "fichier.html",
  {id:1},
  function(data){
    alert(data);
  });

$.post(
  " fichier.html",
  {id:1},
  function(data){
    alert(data);
  });
```

Les fonctions AJAX

- .ajaxComplete()
  - Fonction à appeler à la fin de la requête.
- .ajaxError()
  - Fonction à appeler en cas de fin sur erreur
- .ajaxStart()
  - Fonction à appeler au lancement de la requête.
- .ajaxSend()
  - Fonction à appeler avant l'envoi.
- .ajaxStop()
  - Fcontion à appeler quand toutes les requêtes Ajax sont terminées.
- .ajaxSuccess()
  - Fonction à appeler quand la requête termine avec succès.

Les fonctions AJAX

```
$('.log').ajaxStart(function() {$(this).append('Start.');});
$('.log').ajaxSend(function() {$(this).append('Send.');});
$('.log').ajaxComplete(function() {$(this).append('Complete.');});
$('.log').ajaxStop(function() {$(this).append('Stop.');});
$('.log').ajaxSuccess(function() {$(this).append('Success.');});

$('.trigger').click(function() {
    $('.result').load('fichier.json');
});
```

```
<div class="trigger">Trigger</div>
<div class="result"></div>
<div class="log"></div>
```

```
<div class="trigger">trigger</div>
<div class="result">[0]</div>
<div class="log">
    Start.Send.Success.Complete.Sto
    p.
</div>
```

Récupérer du JSON - Javascript

```
$.getJSON(
   "fichier.json",
   {id:1},
   function(users) {
        alert(users[0].name);
   });

$.getScript(
   "script.js",
   function() {
        ...;
   });
```

De manière générale

- Il existe la fonction ajax générique :
  - Toutes les autres fonctions sont un cas particulier de celle-ci.

```
$.ajax({
   async: false,
   type: "POST",
   url: "test.html",
   data: "nom=JL",
   success: function(msg){
        alert( "Data Saved: " + msg );}
});
```

Attention

- Les événements AJAX ne sont pas liés à un appel en particulier :
  - il faut se souvenir des appels et retrouver d'où est venu l'appel

```
$('.log')
  .ajaxComplete(function(e, xhr, settings) {
    if (settings.url == 'ajax/test.html') {
      $(this).text('ok.');
    }
  });
```

CONCLUSIONS

LI288 – web et développement web

jQuery

- Possibilité de récupérer des objets du DOM.
- Evénements et animations.
- AJAX.

- Tout ça de manière indépendante des navigateurs.

Mais encore

- En plus des fonctions, on a accès à de nombreuses fonctions d'interface utilisateur (jQuery-UI) :
  - Drag and drop
  - Tableaux triables
  - Accordéons
  - Eléments triables
  - Barres de progression
  - Onglets
  - …
- Plugins
  - plus de 4000 plugins actuellement

Sortable – éléments triables

```
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript" src="jquery-ui.js"></script>
<script type="text/javascript">
  $(function() {
    $( "#sortable" ).sortable();
  });
</script>
</head>
<body>
<ul id="sortable">
   <li>Item 1</li>
   <li>Item 2</li>
   <li>Item 3</li>
   <li>Item 4</li>
</ul>
</body>
```

Draggable / droppable

```
<script type="text/javascript">
  $(function() {
    $( "#draggable" ).draggable();
    $( "#droppable" ).droppable({drop: function( event, ui ) {
      $( this ).html( "laché!" );}});
  });
</script>
</head>

<body>
<div id="draggable" >move</div>
<div id="droppable" style="border:1px solid red">on peut
   dropper ici</div>
```

Diaporama

```
/* code jquery */
function slideShowStart() {
  $('#gallery img').hide();
  $('#gallery img:first').addClass('show').show();
  $('#gallery img').click(nextImage);
  setInterval('nextImage()',3000);
}
function nextImage() {
  var current = $('#gallery img.show');
  var next = (current.next().length)?current.next():$('#gallery img:first');
  next.addClass('show').show();
  current.removeClass('show').hide();
}
$(document).ready(function() {slideShowStart();});

/* code html */
<div id="gallery">
  <img src="image1.jpg" alt="Image 1" />
  <img src="image2.jpg" alt="Image 2" />
  <img src="image3.jpg" alt="Image 3" />
</div>
```