



# Table des matières

- Introduction
- Rappels
- L'objet JQuery et les Sélecteurs
- Méthodes et Propriétés de base de l'objet JQuery
- JQuery et les évènements
- JQuery et le DOM
- JQuery et le CSS
- JQuery et le Scrolling
- Exercices intégrés

# Introduction

# Introduction



Bibliothèque JavaScript multi-navigateur conçue  
afin de simplifier l'écriture de script côté client

# Alternatives à jQuery

- UmbrellaJS
- Prototype,
- AngularJS,
- Mootools,
- (Scriptaculous,)
- (Yahoo UI, )
- (Mochikit,)
- (ExtJS,)
- (...)

# Avantages et Inconvénients de jQuery

- Gratuit
- Simplifie la manipulation du DOM
- Simplifie l'AJAX
- Permet de faire des animations
- Syntaxe courte
- Communauté active
- ...



# Example

```
function avoirDefilementVertical(){
    if(typeof(window.pageYOffset) == 'number'){
        // Netscape
        return window.pageYOffset;
    } else if( document.body &&
(document.body.scrollTop) ||
navigator.appName == "Microsoft Internet
Explorer") {
        // DOM
        return document.body.scrollTop;
    } else if( document.documentElement &&
(document.documentElement.scrollTop) ) {
        // Internet Explorer 6
        return
document.documentElement.scrollTop;
    }
}
```

```
function avoirDefilementVertical(){
    return $(document).scrollTop();
}
```

# Version de jQuery

<http://code.jquery.com/>



# Intégration de la bibliothèque

Même procédure qu'un script JS classique:

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

Via les serveurs de Google:

```
<script type="text/javascript"  
src="http://ajax.googleapis.com/ajax/libs/jquery/1/jquery.  
min.js"></script>
```

- Améliore la performance du client (cache)
- Améliore la performance du serveur

# **L'objet jQuery et Les Sélecteurs**

# La fonction jQuery

- `jQuery()` est la fonction principale de la bibliothèque
- Permet de sélectionner des éléments dans la page web
- Abrégée par la fonction `$()`
- Retourne un *objet jQuery* via un *sélecteur CSS*

# L'objet jQuery

- Contient une collection de *références* à des éléments du DOM
  - On peut accéder à chaque élément via un index:  
`$('span')[i]`  
**//Retourne un Élément DOM**
  - L'objet n'est pas un tableau

# L'objet jQuery

- Contient une série de méthodes que l'on pourra appliquer **sur l'ensemble** de ses éléments
    - Attention: On ne peut appliquer ces méthodes **que sur un objet jQuery**, pas sur un élément
- Correct:**      `$ ( ' span ' ) .html ( ) ;`
- Faux:**        `$ ( ' span ' ) [1] .html ( ) ;`
- Correct:**      `$ ( ' span ' ) [1] .innerHTML ;`



# Sélecteurs CSS Classiques

Selecteur	Selection
*	Toutes les balises.
elem	Les balises elem.
#id	Balise ayant l'id "id".
.class	Balises ayant la classe "class".
elem[attr]	Balises elem dont l'attribut "attr" est spécifié.
elem[attr="val"]	Balises elem dont l'attribut "attr" est à la valeur val.
elem bal	Balises bal contenues dans une balise elem.
elem > bal	Balises bal directement descendantes de balises elem.
elem + bal	Balises bal immédiatement précédées d'une balise elem.
elem ~ bal	Balises bal précédées d'une balise elem.

# Pseudo-Classes CSS Classiques

Selecteur	Example	Example description
<a href="#"><u>:link</u></a>	a:link	Selects all unvisited links
<a href="#"><u>:visited</u></a>	a:visited	Selects all visited links
<a href="#"><u>:active</u></a>	a:active	Selects the active link
<a href="#"><u>:hover</u></a>	a:hover	Selects links on mouse over
<a href="#"><u>:focus</u></a>	input:focus	Selects the input element which has focus
<a href="#"><u>:first-child</u></a>	p:first-child	Selects every <p> elements that is the first child of its parent
<a href="#"><u>:last-child</u></a>	p:last-child	Selects every <p> elements that is the last child of its parent
:nth-child	p:nth-child(5)	Selects every <p> elements that is the 5th child of its parent
:nth-child(even)	p:nth-child(even)	Selects every <p> elements that is an even child of its parent

# Sélecteurs spécifiques à jQuery

Expression	Retour
:hidden	Éléments invisibles, cachés.
:visible	Éléments visibles.
:parent	Éléments qui ont des éléments enfants.
:header	Balises de titres : h1, h2, h3, h4, h5 et h6.
:not(s, t, ...)	Éléments qui ne sont pas sélectionnés par le sélecteur s, t, ...
:has(s, t, ...)	Éléments qui contiennent des éléments sélectionnés par le sélecteur s, t, ...
:contains(t)	Éléments qui contiennent du texte t.
:empty	Éléments dont le contenu est vide.
:eq(n) et :nth(n)	Le n-ième élément, en partant de zéro.
:gt(n) (greater than, signifiant plus grand que)	Éléments dont le numéro (on dit l'« index ») est plus grand que n.
:lt(n) (less than, signifiant plus petit que)	Éléments dont l'index est plus petit que n.
:first	Le premier élément (équivalent à :eq(0)). (déprécié : .first())
:last	Le dernier élément. (déprécié : .last())
:even (pair)	Éléments dont l'index est pair.
:odd (impair)	Éléments dont l'index est impair.





# Affiner la sélection

## `$.find(selector)`

Sélectionne les descendants d'un objet JQuery qui répondent au sélecteur

```
let $conteneur = $(' .conteneur:eq(0) ');  
$conteneur.find(' .a-trouver');
```

Autre syntaxe:

```
let $conteneur = $(' .conteneur:eq(0) ');  
$(' .a-trouver', $conteneur);
```



# Parcourir les éléments

## `$.each(function)`

Permet de spécifier une fonction à exécuter pour chacun des éléments de la sélection

L'élément est accessible via **`$(this)`**

## Exemple

```
$('.to-hide').each(function() {  
    $(this).hide();  
});
```

# Contraintes

`$().has(selector);`

Sélectionne un sous-ensemble des éléments qui répondent à la contrainte

Exemple:

```
let $ulNonVide = $('ul').has('li');
```

```
//Récupère tous les <ul>
```

```
//qui ont au moins un <li> à l'intérieur.
```

# Comparaisons

`$.is(selector);`

Compare une sélection avec un sélecteur et retourne true si au moins un de ses éléments "match" le selector

Exemple:

```
let isLi = $('ul li:first').is('li');  
let test = $('ul li:first').is('li, .red, :odd');  
//Les virgules sont des OU-logiques
```

# Méthodes et propriétés de base de l'objet JQuery

## Quelques méthodes

`$().html()`



- **Sans argument**: accède au contenu du premier élément de l'objet jQuery
  - Retourne le contenu, balises comprises, sous forme de string
- **Avec une chaîne de caractères comme premier argument**: remplace le contenu de tous les éléments par cette chaîne
- Equivalent à **innerHTML** en JS

## Quelques méthodes

`$ ( ) . text ( )`



- Manipule le contenu d'un élément comme du texte et non comme des balises
  - Changer le contenu avec `text()` convertit les caractères en entités HTML
    - ex: les chevrons `<` et `>` sont convertit comme `&lt;` et `&gt;`;
  - Récupérer le contenu avec `text()` renvoie le contenu textuel (avec les espaces, retour à la ligne...) sans les balises (renvoie le premier contenu textuel trouvé)
- Equivalent de `textContent` en JS

## Quelques méthodes

`$().val()`



- Manipule le contenu d'un champ input
  - Changer le contenu avec `val("Nouvelle valeur")`
  - Récupérer le contenu avec `val()` renvoie le contenu textuel d'un champ input
- Equivalent de **value** en JS



# Quelques propriétés

- `$().length`

- Renvoie le nombre d'éléments contenus dans l'objet jQuery

# Chaînage des méthodes

Uniquement avec les méthodes renvoyant un objet jQuery

```
$('a')  
  .before('Ceci est un lien :')  
  .after('<br/>')  
  .css('color', '#FF3388');
```

# **jQuery et les évènements**



# Les évènements

- Exemple JavaScript:  
`element.onevenement = function() {...};`
- Affecter une fonction anonyme à un évènement:  
`$().evenement(function(e) {  
 alert('bonjour');  
});`
- Déclencher un évènement:  
`$().evenement();`  
`$().trigger('evenement');`
- [Liste évènements](#)

# Lier un comportement à un ensemble d'éléments créé dynamiquement

`$( 'sélecteur' ).evenement(function(){} )`

Ne fonctionne qu'avec les éléments sélectionnés au moment de l'appel

Exemple:

```
$( '#tab01 td' ).click(function() {  
    alert( $(this).text() );  
});
```



# Lier un comportement

```
$('#sélecteur').on("évènement", function(){  
    //  
});
```

- **Permet de lier un comportement à un évènement**

# **jQuery et le DOM**

# Créer un élément

`$ ('<div>')`

Identique à :

`document.createElement('balise')`

`$ ('<div class="left">Text</div>')`

Identique à :

`innerHTML ('<div class="left">Text</div>')`





# Manipuler la balise et le contenu

## `$().replaceWith(newContent)`

Remplace la balise et le contenu de l'élément avec lequel on appelle la méthode

### Paramètres:

*newContent*: String, Element, jQuery

### Exemple

```
$('div').replaceWith('<span>Salut!</span>');  
$('div').replaceWith($('span'));
```

# Manipuler la balise et le contenu

## `$.replaceAll(oldContent)`

Va remplacer les balises des arguments par la balise appliquant la méthode

### Paramètres:

*oldContent*: Selector, Element, jQuery

### Exemple

```
$('div').replaceAll($('span')) ;  
$('div').replaceAll('span') ;
```



# Insertion à l'intérieur d'un élément

## `$().prepend(contenu) - $().append(contenu)`

- Accepte en argument:
  - Un objet jQuery
  - Une chaîne de caractères

## `$().prependTo(cible) - $().appendTo(cible)`

- Accepte en argument:
  - Une chaîne de caractères
  - Un sélecteur

# Insertion à l'extérieur d'un élément



## `$().before(contenu) - $().after(contenu)`

Ajoute du contenu ou un élément avant ou après l'élément sélectionné

- Accepte en argument:
  - Un objet jQuery
  - Une chaîne de caractères

## `$().insertBefore(cible) - $().insertAfter(cible)`

- Accepte en argument:
  - Une chaîne de caractères
  - Un sélecteur
  - Un objet jQuery



# Envelopper et désenvelopper

- **`$().wrap()`**
  - permet d'envelopper chaque élément entre deux balises
- **`$().wrapAll()`**
  - permet d'envelopper tous les éléments entre deux balises
- **`$().unwrap()`**
  - permet de « désenvelopper » (ou déballer) un élément

# Enfants et Parents

**`$().parents()`** Renvoie un objet JQuery comprenant l'ensemble des éléments parents des éléments en question

**`$().parent()`** Renvoie un objet JQuery comprenant l'élément *parent direct* des éléments en question

**`$().children()`** Renvoie un objet JQuery comprenant les éléments enfants **directs** des éléments en question

**Remarque:** Un sélecteur peut être passé en argument afin d'affiner la sélection

# Siblings

## `$().prev()`

récupère l'élément juste avant l'élément courant

## `$().prevAll()`

récupère tous les éléments avant l'élément courant

## `$().next()`

l'élément juste après

## `$().nextAll()`

tous les éléments après

# Manipuler les attributs

**`$().attr('attribut')`**

Récupère la valeur d'un attribut du premier élément de l'objet jQuery

**`$().attr('attribut',valeur)`**

Définit un attribut

**`$().attr({attribut1: 'valeur1', attribut2: 'valeur2'})`**

Définir plusieurs attributs



# Supprimer les éléments

**`$().remove()`**

Supprime les éléments

**`$().empty()`**

Vide les éléments

**`$().clone()`**

Clone les éléments

# Affecter des fonctions aux méthodes d'un objet jQuery

```
$('p').prepend(function(i) {  
    return i + 'ème : '  
});
```

# **jQuery et le CSS**



# CSS

**`$().css('attribut')`**

Récupère la valeur d'un attribut CSS

**`$().css('attribut',valeur)`**

Définit un attribut CSS

**`$().css({  
attribut1: 'valeur1',  
attribut2: 'valeur2'  
})`**

Définit plusieurs attributs CSS

# CSS



**`$().addClass('classe1 classe2')`**

Ajoute les classes spécifiées (séparées par un espace)

**`$().hasClass('classe')`**

Détermine si l'un des éléments sélectionnés possède la classe spécifiée

**`$().removeClass('classe1 classe2')`**

Supprime les classes spécifiées (séparées par un espace)

**`$().toggleClass('classe1 classe2')`**

Ajoute/Supprime les classes spécifiées (séparées par un espace) en fonction de leur présence

# Animations

`$().animate( properties [, duration ] [, easing ] [, complete ] )`

Permet d'animer un élément

## Paramètres

- *properties*: Object (avec propriétés CSS)
- *duration* (default: 400): Number ou String (ms)
- *easing* (default: swing): String
- *complete*: Function()

# Animations

## Description objet *Properties*

```
{  
    "nomProprieteCss1" : "valeur1",  
    "nomProprieteCss2" : "valeur2"  
}
```

## Propriétés:

- scrollTop, scrollLeft peuvent être utilisés en plus des propriétés CSS
- Pour animer des couleurs, JQuery UI est nécessaire

## Valeurs:

- show, hide et toggle peuvent être utilisés comme valeur
- += et -= . Ils peuvent être placés devant des valeurs numériques afin d'incrémenter ladite valeur



# Animations

```
$().animate({  
    'width': '200px',  
    'height': '+=50px',  
    'background-color': '#FF00FF'  
}, 2000, 'easeOutBounce' )
```



# Animations

**`$().show([duration], [easing], [complete])`**

Affiche les éléments

**`$().hide([duration], [easing], [complete])`**

Cache les éléments

## Paramètres

- *duration*: Number ou String
- *easing* (default: swing): String
- *complete*: Function()

# Animations

**`$().fadeIn([duration], [easing], [complete])`**

Fait apparaître un élément en augmentant son opacité

**`$().fadeOut([duration], [easing], [complete])`**

Rend un élément transparent

## Paramètres

- *duration*: Number ou String
- *easing* (default: swing): String
- *complete*: Function()



# JQuery UI

`$.addClass('classes' [, durée] [, easing] [,complete])`

`$.removeClass('classes' [, durée] [, easing] [,complete])`

`$.toggleClass('classes' [, durée] [, easing] [,complete])`

# Animations

## `$.stop()`

Permet de stopper l'animation d'un élément

Exemple :

```
// Start animation
$( "#start" ).click(function() {
    $( ".block" ).animate({ left: "+=100px" }, 2000 );
});
// Stop animation when button is clicked
$( "#stop" ).click(function() {
    $( ".block" ).stop();
});
```

# **jQuery et le Scrolling**

# Scrolling

## `$().scrollTop()`

Récupère la position de la scroll barre d'un élément (en nombre de pixels) par rapport au sommet de l'élément

## `$().scrollTop(pixel )`

Positionne la scroll barre à la distance déterminée en *pixel* par rapport au sommet de l'élément

# Scrolling

## `$.offset().top`

Récupère la distance entre un élément et le dessus du document

## `$.offset().left`

Récupère la distance entre un élément et la gauche du document

# Exercices intégrés



# Exercice 1

Ajouter

- item

Ajouter

- item
- item

Ajouter

- Départ :
  - `<ul></ul>`
  - `<button></button>`
- Action :
  - au clic du bouton un li est créé avec le mot item et est placé dans le ul

# Exercice 2

Veuillez remplir le champ

- exemple1

- exemple1
- exemple2

- Départ :
  - `<ul></ul>`
  - `<input ...>`
  - `<button></button>`
- Action :
  - au clic du bouton :
    - vérifier que le champ n'est pas vide et pas composé d'espace
    - Si ok ajouter le champ dans la liste
- /!\ Vider le champ une fois ajouté et retirer le message d'erreur si plus besoin

# Exercice 3

**Descriptif produit**

Chaise	25	Ajouter
Table	150	Ajouter
Meuble TV	250	Ajouter

**Mon panier**

Prix total : 0

**Descriptif produit**

Chaise	25	Ajouter
Table	150	Ajouter
Meuble TV	250	Ajouter

**Mon panier**

chaise	25
chaise	25
meuble tv	250
table	150

Prix total : 450

- Départ :
  - Une liste de produit
- Action :
  - au clic du bouton "Ajouter", le produit est ajouté au panier.
  - Le prix total est calculé au fur et à mesure

# Exercice 4

<b>Descriptif produit</b>			
Chaise	25	Ajouter	Retirer
Table	150	Ajouter	Retirer
Meuble tv	250	Ajouter	Retirer
<b>Mon panier</b>			
Prix total : 0			

<b>Descriptif produit</b>			
Chaise	25	Ajouter	Retirer
Table	150	Ajouter	Retirer
Meuble tv	250	Ajouter	Retirer
<b>Mon panier</b>			
Chaise	25	2	
Meuble tv	250	1	
Prix total : 300			

- **Départ :**
  - la liste des produits :
    - 1 bouton Ajouter
    - 1 bouton Retirer
  - le panier qui est vide
  - le prix total à 0
- **Action :**
  - Ajouter :
    - Vérifier si produit est déjà là
      - oui : incrémenter la quantité de 1
      - non : créer la ligne
  - Retirer :
    - Vérifier si produit est déjà là
      - oui : décrémenter la quantité de 1
      - non : (disabled le bouton)

# Exercise 5

New Row	New Col		

- Départ :
  - Un tableau de 2x2
- Action :
  - New Row : ajoute une nouvelle ligne
  - New Col : ajoute une nouvelle colonne
- /!\ Doit ajouter autant de colonnes qu'il y en a au fur et à mesure de l'ajout.

# Exercice 6

**Calculatrice**

4  \*  3,3  = 13.2

**Calculatrice**

4  /  3,3  = 1.21212121212

**Calculatrice**

4  /  0  = Infinity

- **Départ :**
  - champ
    - nombre1
    - operateur
    - nombre2
  - bouton pour calculer
- **Action :**
  - Appui du bouton : calcul

# Jeu : juste prix

- Réaliser un petit jeu

**Bienvenue sur le jeu du juste prix**

Modifier le niveau

Le juste prix est un nombre compris entre 0 et 10

Montrer l'indice

Entrez votre proposition :

Vérifier

# Jeu : juste prix

- Au clic de modifier le niveau

**Bienvenue sur le jeu du juste prix**

**Modifier le niveau**

Facile : entre 0 et 10   Moyen : entre 0 et 100   Difficile : entre 0 et 1000

Le juste prix est un nombre compris entre 0 et 10

**Montrer l'indice**

Entrez votre proposition :

**Vérifier**



# Jeu : juste prix

- Mise à jour en fonction du menu sélectionné:

**Bienvenue sur le jeu du juste prix**

Modifier le niveau

Le juste prix est un nombre compris entre 0 et 100

Montrer l'indice

Entrez votre proposition :

Vérifier

# Jeu : juste prix

- Au clic de montrer l'indice ;

**Bienvenue sur le jeu du juste prix**

Modifier le niveau

Le juste prix est un nombre compris entre 0 et 100

Le juste prix est 54

Entrez votre proposition :

Vérifier

# Jeu : juste prix

- Au clic de la zone de proposition, elle disparaît

**Bienvenue sur le jeu du juste prix**

Modifier le niveau

Le juste prix est un nombre compris entre 0 et 100

Le juste prix est 54

Entrez votre proposition :

Vérifier

# Jeu : juste prix

- Au clic de vérifier

**Bienvenue sur le jeu du juste prix**

Modifier le niveau

Le juste prix est un nombre compris entre 0 et 100

Le juste prix est 99

Entrez votre proposition :

C'est moins

Vérifier

# Jeu : juste prix

**Bienvenue sur le jeu du juste prix**

Modifier le niveau

Le juste prix est un nombre compris entre 0 et 100

Le juste prix est 99

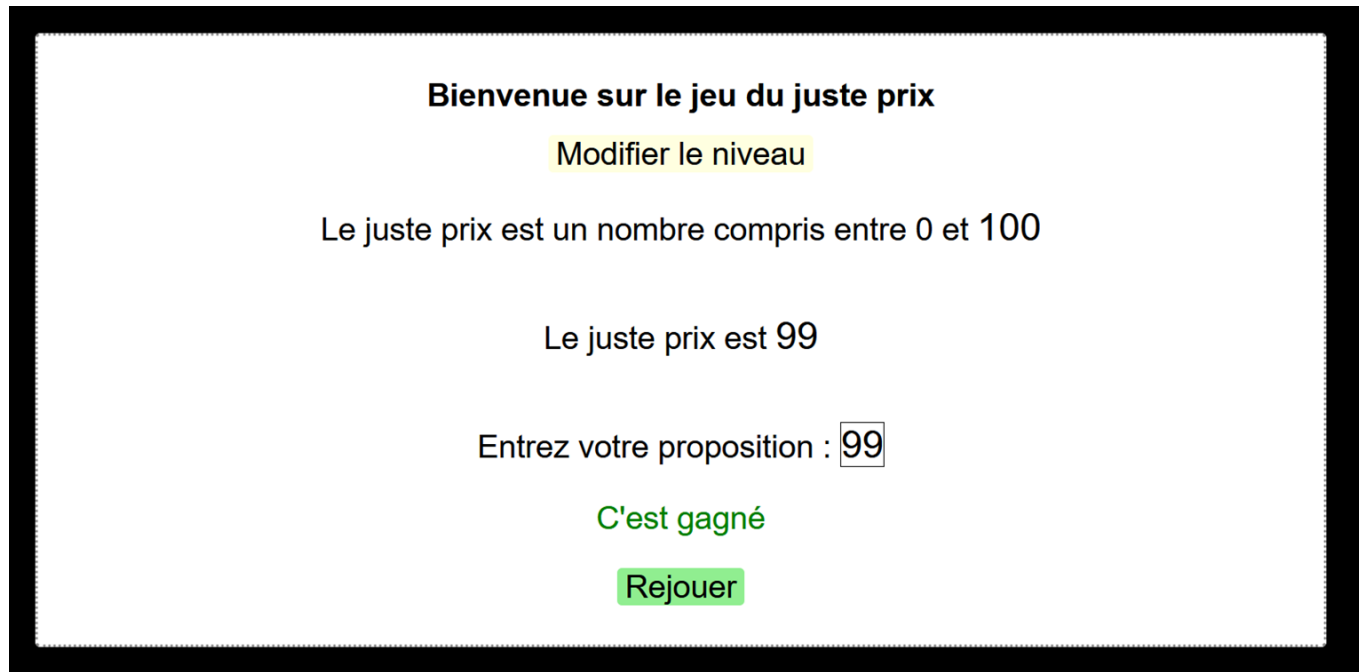
Entrez votre proposition :

C'est plus

Vérifier

# Jeu : juste prix

- Lorsque l'on a gagné, le bouton Vérifier, devient Rejouer



Bienvenue sur le jeu du juste prix

Modifier le niveau

Le juste prix est un nombre compris entre 0 et 100

Le juste prix est 99

Entrez votre proposition :

C'est gagné

Rejouer

# Liste d'exercices et corrigés supplémentaires

- [Grille/Curseur](#)
- [Panier d'achat](#)
- [Panier d'achat avec Formulaire](#)
- [Panier + Drag/Drop](#)
- [BarChart Dynamique](#)
- [Spreadsheet](#)
- [Spreadsheet \(simple\)](#)
- [Button To Top](#)
- [Sticky Menu](#)
- [Sticky Menu + Scroll Effect](#)
- [Drag&Drop \(AJAX + JQuery UI\)](#)

# Exercices supplémentaires

- <http://jsbin.com/napogosepi/edit?js,output>
- <http://jsbin.com/qafojufana/edit?output>
- <http://jsbin.com/yuwaxepi/15/edit?output>
- <http://jsbin.com/yuwaxepi/48/edit?output>
- <http://jsbin.com/vexiruta/30/edit?output>
- <http://jsbin.com/nezilixulo/9/edit?output>
- <http://jsbin.com/vubetatifu/2/edit?output>
- <http://jsbin.com/xavurivu/5/edit?output>



# Exercices supplémentaires

- <http://jsbin.com/yukesoci/7/edit?output>
- <http://jsbin.com/yukesoci/10/edit?output>
- <http://jsbin.com/zeroyana/96/edit?output>

# Ajax et JQuery

Exercice

<http://jsbin.com/qapokumugi/>

# Ajax et JQuery

# Les paramètres de \$.ajax()

```
$("#btn").click(function(){
    $.ajax({
        url : 'script.php',
        type : 'GET',
        dataType : 'html',
        success : function(rep, statut){
            $(code_html).appendTo("#div");
// On passe code_html à jQuery() qui va nous créer l'arbre DOM !
        },
        error : function(resultat, statut, erreur){
        },
        complete : function(resultat, statut){
        }
    });
});
```

# Les différents paramètres

- url : L'URL du fichier côté serveur à exécuter
- Type : la méthode de communication
  - GET : recevoir des données du serveur sans les changer.
  - POST : envoyer des données au serveur, qui vont potentiellement modifier l'état de ce dernier (par exemple, ajouter une ligne dans une table de la base de données)
- dataType : le format de retour du serveur
  - text, json, xml, script, html

# Les différents paramètres

- success : la fonction exécutée lorsque l'appel l'appel AJAX réussit.
- error : la fonction exécutée lorsque l'appel AJAX échoue
- complete : la fonction exécutée après l'exécution de success ou error

# La fonction `success()`

- `data` : donnée renvoyée par le serveur, dans le format spécifié par `dataType`
- `textStatus` : une chaîne décrivant le statut
- `xhr` : l'objet `XMLHttpRequest` (ou `ActiveXObject` pour Internet Explorer)

# La fonction `error()`

- `xhr` : idem que dans `success()`
- `textStatus` : le statut décrivant l'erreur
  - `timeout`, `error`, `notmodified`, `parsererror`
- `errorThrown` : l'exception s'il y en a une.



# La fonction complete()

- xhr : idem que précédemment
- textStatus : idem que précédemment

# Exercices

- Refaire l'exercice du code postal
- A l'aide l'api Last-fm :  
<https://www.last.fm/api/?lang=fr&>
- Faites une application qui permet de rechercher un artiste et de voir son album

