مكتب التكوين المهني وإنعاش الشغل
Office de la Formation Professionnelle
et de la Promotion du Travail

Module 5 : PROGRAMMER EN JAVASCRIPT

Chapitre 5 : Manipuler Jquery

Réalisé par :
Mme ZROURI Amira

Plan du chapitre

1. Découvrir jQuery.
2. Découvrir AJAX .

1. Découvrir jQuery

Fonctions essentielles et chaînage

JQUERY : introduction

JQuery est une bibliothèque JavaScript open-source inventée par John Resig en 2006. Cette bibliothèque est compatible avec les différents navigateurs web.
Le rôle de JQuery est de simplifier l'utilisation de JavaScript et la manipulation du DOM sur les site Web.
En effet, les traitements nécessitant l'écriture de nombreuses
lignes de code JavaScript sont encapsulés dans des méthodes appelées dans une seule ligne de code.

La bibliothèque jQuery contient les fonctionnalités suivantes :

- Manipulation du HTML et DOM
- Manipulation du CSS
- Méthodes d'événement HTML
- Effets et animations
- AJAX

1. Découvrir jQuery

Fonctions essentielles et chaînage

JQUERY : Installation

On peut utiliser deux manières pour utiliser JQuery dans les pages html :
Méthode 1 : Téléchargement de JQuery
Il existe deux versions de JQuery téléchargées depuis le site jQuery.com.
• **Version de production :** version minifiée et compressée pour la phase de l'hébergement.
• **Version de développement :** version non compressée et lisible, pour la phase de développement et de tests.
La bibliothèque jQuery téléchargée correspond à un fichier JavaScript. Pour l'utiliser il faut le référencer avec la balise <script> dans la section <head> :

```
<head>
    <script src="jquery-3.5.1.min.js"></script>
</head>
```

1. Découvrir jQuery

Fonctions essentielles et chaînage

JQUERY : Installation

On peut utiliser deux manières pour utiliser JQuery dans les pages html :
Méthode 1 : Téléchargement de JQuery
Il existe deux versions de JQuery téléchargées depuis le site jQuery.com.
• **Version de production :** version minifiée et compressée pour la phase de l'hébergement.
• **Version de développement :** version non compressée et lisible, pour la phase de développement et de tests.
La bibliothèque jQuery téléchargée correspond à un fichier JavaScript. Pour l'utiliser il faut le référencer avec la balise <script> dans la section <head> :

```
<head>
    <script src="jquery-3.5.1.min.js"></script>
</head>
```

1. Découvrir jQuery

Fonctions essentielles et chaînage

JQUERY : Installation

Méthode 2 : JQuery via CDN (Content Delivery Network)
On peut inclure JQuery à partir d'un CDN (Content Delivery Network) sans besoin de télécharger les fichiers.
Exemple d'utilisation de JQuery hébergé par Google :

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

1. Découvrir jQuery

Fonctions essentielles et chaînage

Syntaxe de JQUERY

La syntaxe de JQuery est basée sur les sélecteurs : `$( sélecteur ). action ()`

- **$()** est un raccourci vers la fonction **jQuery()** qui trouve des éléments dans une page et crée des objets jQuery qui référencent ces éléments.

Par exemple : $('p') et jQuery('p') : sélectionne tous les éléments p (paragraphe).

- **Sélecteur** correspond à sélecteur CSS pour interroger (ou rechercher) des éléments HTML.
- **Action** correspond à une action à effectuer sur le(s) élément(s) sélectionnés.

Exemples :

- **$(this).hide() :** masque l'élément courant.
- **$("p").hide() :** masque tous les éléments <p>.
- **$(".test").hide() :** masque tous les éléments avec class="test".
- **$("#test").hide() :** masque l'élément avec id="test".

1. Découvrir jQuery

Fonctions essentielles et chaînage

L'événement ready pour le document

Les méthodes jQuery se trouvent dans l'événement **ready** de l'objet document qui permet d'empêcher l'exécution du code jQuery avant la fin du chargement du document.

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

    // Méthodes jQuery...
});
```

Exemples d'actions qui peuvent échouer si les méthodes sont exécutées avant le chargement complet du document :

- Masquer un élément qui n'est pas encore crée.
- Obtenir la taille d'une image qui n'est pas encore chargée.

1. Découvrir jQuery

Fonctions essentielles et chaînage

Notion de chaînage

Les instructions **JQuery** peuvent être écrites soit l'une après l'autre (dans des lignes différentes) ou en utilisant la technique de chaînage.
Le chaînage permet d'exécuter plusieurs actions JQuery l'une après l'autre (dans la même ligne), sur le même élément.
Exemple : Les méthodes css(), slideUp() et slideDown() sont appelées sur le paragraphe identifié par l'ID « p1 ». Ainsi, l'élément "p1" devient d'abord rouge, puis il glisse
vers le haut, puis vers le bas :

1. Découvrir jQuery

Fonctions essentielles et chaînage

Notion de chaînage

Syntaxe 1 :

```
$("#p1").css("color", "red").slideUp(2000).slideDown(2000);
```

Syntaxe 2 :

```
$("#p1").css("color", "red")
   .slideUp(2000)
   .slideDown(2000);
```

1. Découvrir jQuery

Comportement des liens

Désactiver un lien href en JQUERY

La méthode **removeAttr()** de JQuery permet de supprimer l'attribut « href » du lien, qui devient non cliquable.

```
<head>
   <meta charset="utf-8">
   <title>Désactiver un lien</title>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
   <script type="text/javascript">
     $(document).ready(function(){
       $(".menu a").each(function(){
         if($(this).hasClass("disabled")){
           $(this).removeAttr("href");
         }
       });
     });
   </script>
 </head>
```

```
<body>
  <ul class="menu">
    <li><a href="https://www.ofppt.ma/">Lien1</a></li>
    <li><a href="https://www.ofppt.ma/">Lien2</a></li>
    <li><a href="https://www.ofppt.ma/" class="disabled">Lien3</a></li>
  </ul>
</body>
```

1. Découvrir jQuery

Comportement des liens

Activer un lien href en JQUERY

La méthode **attr()** de JQuery permet d'jouter l'attribut « href » à un lien.

```
<head>
   <meta charset="utf-8">
   <title>Désactiver un lien</title>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
   <script type="text/javascript">
     $(document).ready(function(){
       $(".menu a").each(function(){
         if($(this).hasClass("disabled")){
           $(this).attr("href","https://www.google.com/");
         }
       });
     });
   </script>
 </head>
```

```
<body>
  <ul class="menu">
    <li><a href="https://www.ofppt.ma/">Lien1</a></li>
    <li><a href="https://www.ofppt.ma/">Lien2</a></li>
    <li><a class="disabled">Lien3</a></li>
  </ul>
</body>
```

1. Découvrir jQuery

Association d'évènements et déclenchement

Les événements en JQUERY

Les événements DOM ont une méthode jQuery équivalente.

Exemple 1 : La méthode **click()**
Attribuer un événement de clic et une fonction à tous les paragraphes d'une page. La fonction masque l'élément cliqué.

```
$("p").click(function(){
    // actions de l'évènement
        $(this).hide();
  });
```

1. Découvrir jQuery

Association d'évènements et déclenchement

Les événements en JQUERY

Exemple 2 : La méthode **dbclick()**
Attribuer un événement de double clic et une fonction à tous les paragraphes d'une page. La fonction est exécutée lorsque l'utilisateur double-clique sur le paragraphe.

```
$("p").dblclick(function(){
    $(this).hide();
  });
```

1. Découvrir jQuery

Association d'évènements et déclenchement

Les événements en JQUERY

Exemple 3 : La méthode **mouseenter()**

```
$("#p1").mouseenter(function(){
    alert("Vous êtes sur le paragraphe p1!");
  });
```

Exemple 4 : La méthode **mouseleave()**

```
$("#p1").mouseleave(function(){
    alert("vous avez quitté le paragraphe p1!");
  });
```

1. Découvrir jQuery

Intégration de plugins existants

Intérêt des plugins

• Un plugin est un code écrit dans un fichier JavaScript standard. Il fournit des méthodes jQuery utiles qui peuvent être utilisées avec les méthodes de la bibliothèque
jQuery.
• L'installation des modules JQuery additionnels, appelés **JQuery plugins** permet d'étendre les fonctionnalités offertes par JQuery et gagner en termes de rapidité du
développement en réutilisant des codes existants.

1. Découvrir jQuery

Intégration de plugins existants

Intégrer des plugins existants

• Il existe de nombreux sites proposant des plugins jQuery. Parmi ces sites, on peut utiliser le site officiel https://jquery.com/plugins.
• Pour intégrer un plugin dans une page HTML, il faut télécharger le plugin à partir du site dédié puis le référencer dans la page HTML.

1. Découvrir jQuery

Intégration de plugins existants

Intégrer des plugins existants

Liste des plugins

Figure 26 : téléchargement des plugins Jquery

Intégrer le plugin jquery.plug-in.js téléchargé dans le fichier HTML :

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

1. Découvrir jQuery

Utilisation de plugins existants

Utilisation de plugins existants

Exemple : https://github.com/loebi-ch/jquery-clock-timepicker/blob/master/README.md

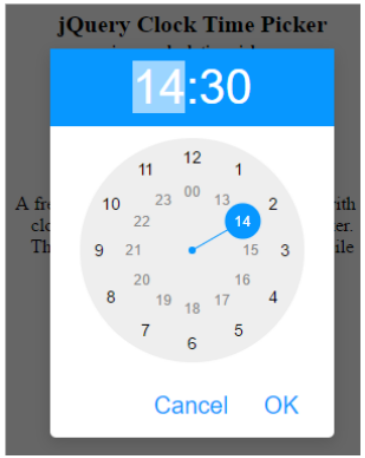

Figure 27: JQuery clock Time Picker **[https://github.com/loebi-ch/jquery-clock-timepicker]**

2. Découvrir Ajax

Introduction à AJAX

Qu'est-ce qu'AJAX ?

AJAX est acronyme de « Asynchronous JavaScript And XML ».
AJAX est une technologie basée sur :
• Un objet **XMLHttpRequest** intégré au navigateur (pour demander des données à un serveur Web).
• **JavaScript** et **DOM HTML** (pour afficher les données).
AJAX permet de :
• Lire les données d'un serveur web (après le chargement d'une page web) ;
• Mettre à jour une page web sans la recharger ;
• Envoyer les données à un serveur web (en arrière-plan).

2. Découvrir Ajax

Introduction à AJAX

Qu'est-ce qu'AJAX ?

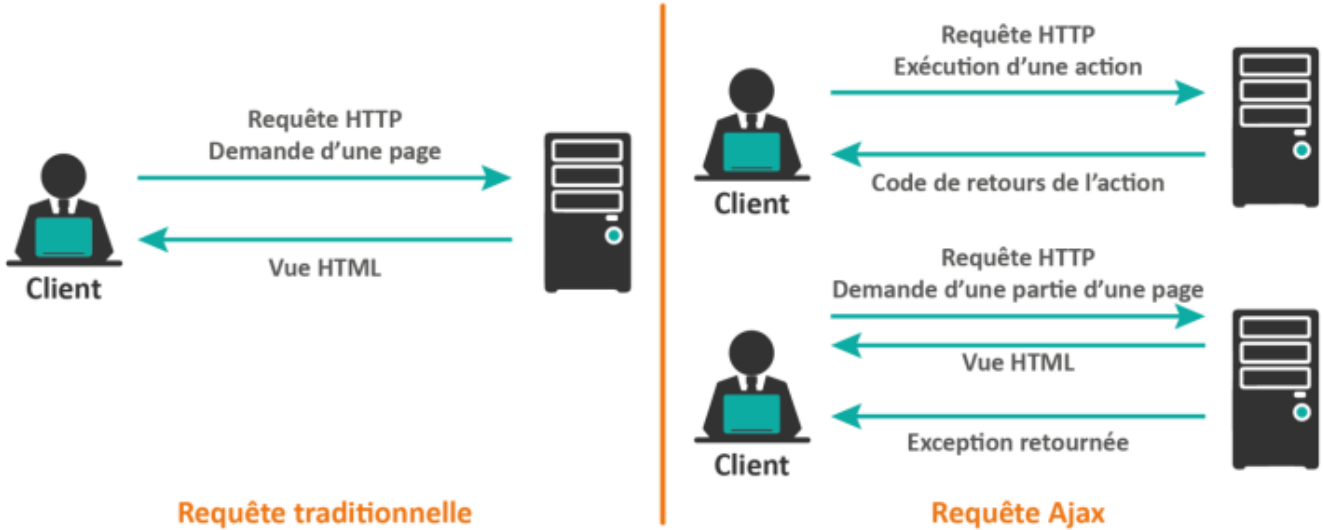

Figure 28 : Fonctionnement AJAX

2. Découvrir Ajax

Introduction à AJAX

L'objet XMLHttpRequest

• L'objet **XMLHttpRequest** de la technologie AJAX est un objet qui permet d'envoyer des requêtes HTTP au serveur, de recevoir des réponses et de mettre à jour la page Web.
• En mode asynchrone, cette mise à jour se réalise sans devoir recharger la page et donc de façon totalement transparente pour l'utilisateur.
• L'objet **XMLHttpRequest** est basé sur le principe d'échange de données entre le client (la page web) et le serveur (sur lequel se trouve la page ou la source de données à laquelle la page Web veut accéder).

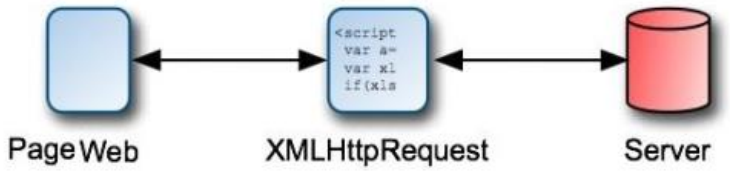

Figure 29 : l'objet XMLHttpRequest de AJAX

2. Découvrir Ajax

Fonctionnement d'AJAX

Fonctionnement de l'objet XMLHttpRequest

L'objet **XMLHttpRequest (XHR)** est créé par le moteur JavaScript du navigateur pour initier des requêtes (demandes) HTTP à partir de l'application vers le serveur, qui à son tour renvoie la réponse au navigateur. La propriété **XMLHttpRequest.readyState** renvoie l'état d'un client XMLHttpRequest. Les états possible du "**readyState**" sont :

- **0 (Requête non initialisée) :** le client XMLHttpRequest a été créé, mais la méthode open() n'a pas encore été appelée.
- **1 (Connexion au serveur établie) :** la méthode **open()** a été invoquée, les en-têtes de demande peuvent être définies à l'aide de la méthode **setRequestHeader()** et la méthode **send()** peut être appelée, ce qui lancera la récupération.
- **2 (Requête reçue).**
- **3 (Requête en cours de traitement).**
- **4 (Requête terminée et réponse prête).**

2. Découvrir Ajax

Fonctionnement d’AJAX

Fonctionnement de l’objet XMLHttpRequest

Les requêtes du serveur doivent être envoyées de manière **asynchrone:**
JavaScript n'a pas à attendre la réponse du serveur, et peut exécuter d'autres scripts en attendant la réponse du serveur ou bien traiter la réponse une fois que la réponse est prête.

2. Découvrir Ajax

Fonctionnement d'AJAX

Fonctionnement de l'objet XMLHttpRequest

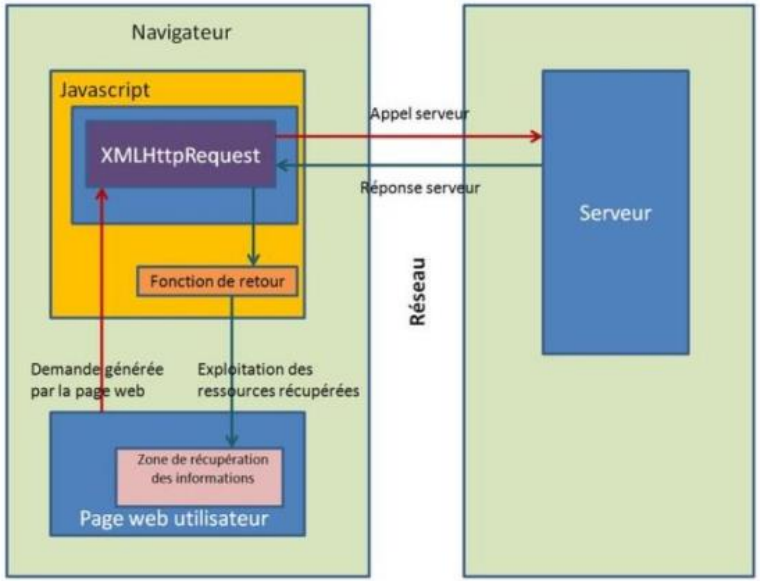

Figure 30 : [Source : https://apcpedagogie.com/lobjet-xmlhttprequest/]

2. Découvrir Ajax

Fonctionnement d'AJAX

Créer un objet XMLHttpRequest

Syntaxe de création d'un objet **XMLHttpRequest** (reconnu par la plupart des navigateurs).

```
variable = new XMLHttpRequest();
```

2. Découvrir Ajax

Fonctionnement d'AJAX

Méthodes d'objet XMLHttpRequest

Méthode	Description
new XMLHttpRequest()	Créer un nouvel objet XMLHttpRequest
abort()	Quitter la requête courante
getAllResponseHeaders()	Retourner toutes les informations du header
getResponseHeader()	Retourner une information spécifique du header
open(method,url,async,user,psw)	Specifier la requête : • **method**: GET ou POST • **url**: emplacement du fichier • **async**: true (asynchrone, c'est à dire JavaScript n'a pas à attendre la réponse du serveur) ou false (synchrone) • **user**: nom d'utilisateur (optionel) • **psw**: mot de passe (optionnel)
send()	Envoyer la requête au serveur (utilisé dans les requêtes GET)
send(string)	Envoyer la requête au serveur (utilisé dans les requêtes POST)

2. Découvrir Ajax

Fonctionnement d'AJAX

Propriétés de l'objet XMLHttpRequest

Propriété	Description
onreadystatechange	Définit une fonction à appeler lorsque la propriété **readyState** change
readyState	Contient le statut de XMLHttpRequest
responseText	Renvoie les données de réponse sous forme de chaîne
responseXML	Renvoie les données de réponse sous forme de données XML
status	Renvoie le numéro d'état d'une requête 200: "OK" 403: "Forbidden" 404: "Not Found"
statusText	Renvoie le texte d'état (par exemple "OK" ou "Not Found")

2. Découvrir Ajax

Fonctionnement d'AJAX

Envoyer une demande à un serveur

Les méthodes **open()** et **send()** de l'objet XMLHttpRequest permettent d'envoyer une requête à un serveur.

Syntaxe :

```
open(method, url, async)
```

Exemple 1 :

```
let xhttp = new XMLHttpRequest();
xhttp.open("GET", "code.php", true);
xhttp.send();
```

2. Découvrir Ajax

Fonctionnement d'AJAX

Envoyer une demande à un serveur

Exemple 2 : Envoi des données dans la requête GET

```
let xhttp = new XMLHttpRequest();
xhttp.open("GET", "code.php?prenom=Hassan&nom=FILALI", true);
xhttp.send();
```

Exemple 3 :

```
let xhttp = new XMLHttpRequest();
xhttp.open("POST", "code.php", true);
xhttp.setRequestHeader("Content-type", "application/x-www-form-
urlencoded");
xhttp.send("prenom=Hassan&nom=FILALI");
```

Pour utiliser la méthode dans un formulaire, il faut ajouter un en-tête HTTP avec **setRequestHeader().**

2. Découvrir Ajax

Fonctionnement d'AJAX

La propriété onreadystatechange

• La propriété **onreadystatechange** de l'objet **XMLHttpResponse** permet de définir une fonction à exécuter quand une réponse est reçue.
• La propriété **onreadystatechange** exécute la fonction chaque fois que **readyState** change de valeur : Lorsque **readyState** est 4 et status est 200, la réponse est prête.

Exemple 1 : Source : https://www.w3schools.com

```
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("demo").innerHTML = this.responseText;
    }
  };
  xhttp.open("GET", "ajax_info.txt", true);
  xhttp.send();
```

2. Découvrir Ajax

Fonctionnement d'AJAX

La propriété onreadystatechange

Exemple 2 : Source : https://www.w3schools.com

```
function loadDoc() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            document.getElementById("demo").innerHTML =
            this.responseText;
        }
    };
    xhttp.open("GET", "ajax_info.txt", true);
    xhttp.send();
}
```

2. Découvrir Ajax

Fonctionnement d’AJAX

Utilisation d'une fonction Callback

• Une fonction de rappel est une fonction passée en paramètre à une autre fonction.
• Les fonctions de rappel sont utilisées quand plusieurs tâches AJAX doivent être exécutées dans un site web : il faut créer une fonction pour exécuter l'objet **XMLHttpRequest** et une fonction de rappel pour chaque tâche AJAX.
• L'appel de la fonction doit contenir l'URL et la fonction à appeler lorsque la réponse est prête.

2. Découvrir Ajax

Fonctionnement d'AJAX

Utilisation d'une fonction Callback

Exemple 1 :

Source : https://www.w3schools.com

```
<div id="demo">

<h1>Objet XMLHttpRequest</h1>

<button type="button"
onclick="loadDoc('ajax_info.txt', myFunction)">
Change Content
</button>
</div>

<script>

</script>
```

```
function loadDoc(url, cFunction) {
  var xhttp;
  xhttp=new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      cFunction(this);
    }
  };
  xhttp.open("GET", url, true);
  xhttp.send();
}
function myFunction(xhttp) {
  document.getElementById("demo").innerHTML =
  xhttp.responseText;
}
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

La méthode jQuery ajax()

La méthode jQuery **ajax()** fournit les fonctionnalités de base d'Ajax dans jQuery. Il envoie des requêtes HTTP asynchrones au serveur.

Syntaxe :

```
$.ajax(url,[options]);
```

URL : chaîne de l'URL vers laquelle les données sont soumises (ou récupérées).
Options : options de configuration pour la requête Ajax. Un paramètre d'options peut être spécifié à l'aide du format JSON. Ce paramètre est facultatif.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Les options de la méthode jQuery ajax()

Options	Description
contentType	Une chaîne contenant un type de contenu lors de l'envoi de contenu MIME au serveur. La valeur par défaut est "application/x-www-form-urlencoded; charset=UTF-8"
data	Une donnée à envoyer au serveur. Il peut s'agir d'un objet JSON, d'une chaîne ou d'un tableau.
dataType	Le type de données attendues du serveur.
error	Une fonction de rappel à exécuter lorsque la requête échoue.
global	Un booléen indiquant s'il faut ou non déclencher un gestionnaire de requêtes Ajax global. La valeur par défaut est true.
headers	Un objet de paires clé/valeur d'en-tête supplémentaires à envoyer avec la demande.
statusCode	Un objet JSON contenant des codes HTTP numériques et des fonctions à appeler lorsque la réponse a le code correspondant.
success	Une fonction de rappel à exécuter lorsque la requête Ajax réussit.
timeout	Une valeur numérique en millisecondes pour le délai d'expiration de la demande.
type	Un type de requête http : POST, PUT et GET. La valeur par défaut est GET.
url	Une chaîne contenant l'URL à laquelle la demande est envoyée.
username	Un nom d'utilisateur à utiliser avec XMLHttpRequest en réponse à une demande d'authentification d'accès HTTP.
password	Un mot de passe à utiliser avec XMLHttpRequest en réponse à une demande d'authentification d'accès HTTP.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Envoyer une demande Ajax

La méthode ajax() effectue exécute une requête HTTP asynchrone et récupère les données du serveur.

Exemple :

```
$.ajax('/jquery/getdata',
    {
        success: function (data, status, xhr) {
            $('p').append(data);
    }
});

<p></p>
```

2. Découvrir Ajax

Implémentation d’AJAX via jQuery

Envoyer une demande Ajax

Dans l'exemple ci-dessus, le premier paramètre '/getData' de la méthode ajax() est une URL à partir de laquelle on veut récupérer les données.
Par défaut, la méthode ajax() exécute la requête http GET si le paramètre d'option n'inclut pas l' option de méthode .
Le deuxième paramètre est le paramètre options au format JSON qui spécifie la fonction de rappel qui sera exécutée.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Envoyer une demande Ajax

L'exemple suivant montre comment obtenir les données JSON à l'aide de la méthode ajax().

```
$.ajax('/jquery/getjsondata',
{
    dataType: 'json', // type des données de la réponse
    timeout: 500,     // délai d'expiration en milliseconds
    success: function (data,status,xhr) {   // fonction callback en cas de succès
        $('p').append(data.firstName + ' ' + data.middleName + ' ' + data.lastName);
    },
    error: function (jqXhr, textStatus, errorMessage) { // cas d'erreur
        $('p').append('Error: ' + errorMessage);
    }
});

<p></p>
```

2. Découvrir Ajax

Implémentation d’AJAX via jQuery

Envoyer une demande Ajax

Le premier paramètre est une URL de la requête qui revoie des données au format JSON. Dans le paramètre options, l’option **dataType** spécifie le type de données de réponse (JSON dans ce cas). L’option **timeout** spécifie le délai d'expiration de la demande en millisecondes.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Envoyer une demande Ajax

Syntaxe équivalente à l'exemple précédent :

```
var ajaxReq = $.ajax('GetJsonData', {
    dataType: 'json',
    timeout: 500
});

ajaxReq.success(function (data, status, jqXhr) {
$('p').append(data.firstName + ' ' + data.middleName + ' ' + data.lastName);
})

ajaxReq.error(function (jqXhr, textStatus, errorMessage) {
$('p').append('Error: ' + errorMessage);
})

<p></p>
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Envoyer une requête HTTP POST en utilisant ajax()

La méthode **ajax()** peut envoyer tout type de requêtes HTTP (GET, PSOT, PUT).
Exemple : Envoie d'une requête HTTP POST au serveur.

```
$.ajax('/jquery/submitData', {
    type: 'POST',  // Méthode POST
    data: { myData: 'Mes données.' },  // données à envoyer
    success: function (data, status, xhr) {
        $('p').append('status: ' + status + ', data: ' + data);
    },
    error: function (jqXhr, textStatus, errorMessage) {
            $('p').append('Error' + errorMessage);
    }
});
<p></p>
```

2. Découvrir Ajax

Implémentation d’AJAX via jQuery

Envoyer une requête HTTP POST en utilisant ajax()

Le premier paramètre est une URL de la requête qui revoie des données au format JSON. L’option **type** désigne le type de la requête (POST dans ce cas). L'option **data** spécifie que les données qui seront soumises au serveur le seront en tant qu'objet JSON.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery get()

La méthode jQuery **get()** envoie une requête http GET asynchrone au serveur et récupère les données.

Syntaxe :

```
$.get(url, [données],[rappel]);
```

URL : url de la requête à partir de laquelle on récupère les données.
Data : données à envoyer au serveur.
Callback : fonction à exécuter lorsque la requête aboutit.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery get()

Exemple : Récupération des données à partir d'un fichier texte.

```
$.get('/data.txt',  // url
      function (data, textStatus, jqXHR) {  // success callback
          alert('status: ' + textStatus + ', data:' + data);
    });
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery getJSON()

La méthode jQuery **getJSON()** envoie une requête http GET asynchrone au serveur et récupère les données au format JSON uniquement.
Rappel : La méthode **get()** récupère tout type de données.

Syntaxe :

```
$.getJSON(url, [données],[rappel]);
```

URL : url de la requête à partir de laquelle on récupère les données.
Data : données à envoyer au serveur.
Callback : fonction à exécuter lorsque la requête aboutit.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery getJSON()

Exemple 1 : Récupérer des données JSON

```
$.getJSON('/jquery/getjsondata', {name:'Ali'},
function (data, textStatus, jqXHR){
    $('p').append(data.firstName);
});


<p></p>
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery getJSON()

Exemple 2 : Utiliser les méthodes de rappel fail() et done()

```
$.getJSON('/jquery/getjsondata', { name:'Ali'},
function(data, textStatus, jqXHR){
    alert(data.firstName);
})
.done(function () { alert('Request done!'); })
.fail(function (jqxhr,settings,ex) {
alert('failed, '+ ex); });
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery post()

La méthode jQuery **post()** envoie une requête HTTP POST asynchrone au serveur.

Syntaxe :

```
$.post(url,[données],[rappel],[type]);
```

URL : url de la requête à partir de laquelle on récupère / soumet les données.
Data : données JSON à envoyer au serveur.
Callback : fonction à exécuter lorsque la requête aboutit.
Type : type de données du contenu de la réponse.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery post()

Exemple 1 :

```
$.post('/jquery/submitData',    // url
       { myData: 'This is my data.' }, // données à soumettre
       function(data, status, jqXHR) {// succès
                $('p').append('status: ' + status + ', data: ' + data);
        })

<p></p>
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery post()

Exemple 2 : Les données JSON sont obtenues en tant que réponse du serveur. Ainsi, la méthode post() analysera automatiquement la réponse dans l'objet JSON.

```
$.post('/submitJSONData',  // url
       { myData: 'This is my data.' }, // data to be submit
       function(data, status, xhr) {   // succès
                alert('status: ' + status + ', data: ' + data.responseData);
            },
       'json'); // format des données de réponse
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery post()

Exemple 3: Attacher les méthodes de rappel fail() et done() à la méthode post().

```
$.post('/jquery/submitData',
        { myData: 'This is my data.' },
        function(data, status, xhr) {
            $('p').append('status: ' + status + ', data: ' + data);

        }).done(function() { alert('Request done!'); })
        .fail(function(jqxhr, settings, ex) { alert('failed, ' + ex); });

<p></p>
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery load()

La méthode jQuery **load()** permet de charger du contenu HTML ou texte à partir d'un serveur et de l'ajouter dans un élément DOM.

Syntaxe :

```
$.load(url,[données],[rappel]);
```

URL : url de la requête à partir de laquelle on récupère le contenu.
Data : données JSON à envoyer au serveur.
Callback : fonction à exécuter lorsque la requête aboutit.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery load()

Exemple : charger du contenu html depuis le serveur et l'ajouter à l'élément div.

```
$('#msgDiv').load('/demo.html');

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

Remarque

- Si aucun élément n'est trouvé par le sélecteur alors la requête Ajax ne sera pas envoyée.

2. Découvrir Ajax

Implémentation d’AJAX via jQuery

Méthode jQuery load()

La méthode **load()** permet de spécifier une partie du document de réponse à insérer dans l'élément DOM.
Ceci peut être réalisé à l'aide du paramètre **url**, en spécifiant un sélecteur avec une URL séparée par un ou plusieurs caractères d'espacement.

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery load()

Exemple : Le contenu de l'élément dont l'ID est **myHtmlContent** sera ajouté à l'élément **msgDiv.**

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
</head>
<body>
    <h1>Page html.</h1>
    <div id="myHtmlContent">This is my
html content.</div>
</body>
</html>
```

2. Découvrir Ajax

Implémentation d'AJAX via jQuery

Méthode jQuery load()

Exemple : Le contenu de l'élément dont l'ID est **myHtmlContent** sera ajouté à l'élément **msgDiv.**

```
$('#msgDiv').load('/demo.html #myHtmlContent');

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