LE WEB DEVIENT DYNAMIQUE

AJAX, LES FONDAMENTAUX

LE WEB DEVIENT DYNAMIQUE

FONCTION LOAD() POUR CHARGER DU CONTENU

Avant d'étudier les requêtes Ajax à proprement parler, il est utile de s'intéresser à un raccourci qui permet d'éviter l'écriture d'un code trop long pour simplement afficher du contenu dans une page Web. La fonction load() est un raccourci très utile lorsqu'il s'agit de charger du contenu dans une page Web avec une requête Ajax.

Fonction load() sans option

La fonction load() charge le document upFirst.html dans la balise #first au click sur la balise #upFirst.

Il est possible d'ajouter une fonction de callBack.

```
$('#upFirst').click(function() {
    $('#first').load('html/upFirst.html', function() {
        console.log('La première zone a été mise à jour');
    });
});
```

Fonction load() avec options

La première fonction load() charge la balise #firstTag du document upElements.html dans la balise #first et la deuxième la balise #secondeTag dans la balise #seconde.

L'option est ajoutée à la suite de l'adresse du document HTML à charger, il suffit de spécifier la balise à charger.

```
$('#upFirst').click(function() {
    $('#first').load('html/upElements.html #firstTag', function() {
        console.log('La première zone a été mise à jour');
    });
});

$('#upSeconde').click(function() {
    $('#seconde').load('html/upElements.html #secondeTag', function() {
        console.log('La deuxième zone a été mise à jour');
    });
});
```

Cette fonction n'est pas à utiliser pour gérer l'envoi de données vers un serveur

FONCTION GET() POUR PRECISER LE COMPORTEMENT

Lors d'une requête Ajax nous demandons au serveur de charger du contenu dans une page Web. Si la requête fonctionne, le chargement et presque automatique mais il est possible qu'un problème soit rencontré lors de la requête. Pour pouvoir adopter un comportement différent si la requête à fonctionné ou bien échoué, il est recommandé d'utiliser la requête get() suivie des fonctions done(), fail() et always().

Fonction get()
Proche de la fonction laod(), la fonction get() permet de spécifier des comportements selon le résultat de la requête.

Fonction done()
Cette fonction est déclenchée si la requête a abouti. Le paramètre data correspond au contenu à charger.

Fonction fail()
Cette fonction est déclenchée si la requête a échoué.

Fonction always()
Cette fonction est déclenchée quand la requête est terminée.

```
$('#upFirst').click(function() {
    $.get('html/upElements.html', function() {
        console.log('Requète get() envoyée');
    })

    .done(function(data){
        $('#seconde').append(data);
    })

    .fail(function(){
        console.log('Erreur lors de la requète');
    })

    .always(function(){
        console.log('Requète get() terminée');
    });
});
```

Cette fonction est à préférer à la fonction load() car elle est plus précise.

FONCTION POST() POUR ENVOYER DES DONNEES

Ajax est l'art d'échanger des données avec un serveur, et la mise à jour des parties d'une page Web, sans recharger la page entière. La fonction post() est spécifiquement définie pour envoyer des données au serveur et attendre en retour une réponse adaptée aux données envoyées. La fonction post() est également un racoursi de requête Ajax qui permet une gestion plus simple de la requête.

Paramètre récupéré
Récupération de la valeur d'un input.

Fonction post()
La fonction post() permet d'envoyer des données au fichier pour récupérer une réponse adaptée.

Fonction done()
Cette fonction est déclenchée si la requête a abouti. Le paramètre data correspond au contenu à charger.

Fonction fail()
Cette fonction est déclenchée si la requête a échoué.

Fonction always()
Cette fonction est déclenchée quand la requête est terminée.

```
var param = $('#userChoice').val();

$('#upFirst').click(function() {
    $('#first').post('php/citations.php', {choice: param}, function() {
        console.log('Requête post() envoyée');
    })
    .done(function(data){
        $('#seconde').append(data);
    })
    .fail(function(){
        console.log('Erreur lors de la requête');
    })
    .always(function(){
        console.log('Requête post() terminée');
    });
});
```

Pour envoyer plusieurs données il est conseillé d'utiliser la fonction serialize()

REQUETE AJAX SUR UN FORMULAIRE DE CONNEXION

Nous allons à présent réaliser une requête Ajax complète qui permet à un utilisateur de se connecter à un site par le biais d'un formulaire HTML. Ce cas concret nous permet d'analyser un programme complet et de savoir précisément à quel moment la requête Ajax doit être lancée pour réaliser un système de connexion basique.

Evènement submit
Un input de type="submit" est intégré au formulaire.

Vérification des champs
Si les champs sont vide le code est bloqué.

Requète ajax()
TYPE : peut être POST ou GET
URL : le chemin vers le fichier de traitement
DATA : permet d'envoyer les données du formulaire
SUCCESS : fonction de callBack

msg == "Success"
Si la requète a été correctement réalisée, le fichier de traitement PHP renvoi echo "Succes".

Condition else
En cas d'erreur lors du traitement PHP un message est affiché dans la console

```
$('form').submit(function(event){
    event.preventDefault();
    if ($("#pseudo").val() == 0 || $("#pass").val() == 0) {
       console.log("Tous les champs sont obligatoire");
    }
    else{
        $.ajax({
            type: "POST",
            url: "php/connexion.php",
            data: $('form').serialize(),
            success: function(msg){
                if(msg == "Succes"){
                    console.log("Vous êtes à présent connecté");
                }
                else{
                    console.log("Erreur PHP");
                }
            }
        });
    };
};
```

LES PARAMETRES SUCCESS, ERROR ET COMPLETE

L'exemple précédent présent des conditions if est else pour vérifier les retours du fichier de traitement PHP mais pas pour vérifier la requête Ajax elle-même comme pour les fonctions load(), get() et post(). Une requête fonctionne différemment, il faut associer des fonctions aux paramètres succès, error et complète.

Requère de type GET
Dans une requête du type GET, il est possible de préciser le type de donnée à charger avec le paramètre dataType

La requère à réussi :success
Le paramètre success prend en paramètre le contenu du fichier chargé et l'état de la requête

La requère à echoué :error
Le paramètre error prend en paramètre un objet résultat jQuery, l'état de la requête et le type d'erreur

La requère est terminée :complete
Le paramètre complete prend en paramètre l'objet résultat jQuery et l'état de la requête

```
$.ajax({
    type: "GET",
    url: "html/comments.html",
    dataType: "html",

    success: function(data, status){
        $("#sectionComments").append(data);
        console.log("Etat de la requète : " + status);
    },

    error: function(result, status, error){
        console.log("Réponse jQuery : " + result);
        console.log("Statut de la requète : " + status);
        console.log("Type d'erreur : " + error);
    },

    complete: function(result, status){
        console.log("Réponse jQuery : " + result);
        console.log("Etat de la requète : " + status);
    }
});
```