307 - Réaliser des pages Web interactives

Rapport personnel

Date de création : 15.05.2023

Version 1 du : 05.06.2023

|  |  |
| --- | --- |
| Une image contenant texte  Description générée automatiquement | Module du 15.05.2023 au 13.06.2023 |

Table des matières

[1 Introduction 5](#_Toc136863126)

[2 Exercice 1 (Découverte des écouteurs) 5](#_Toc136863127)

[3 Exercice 2 5](#_Toc136863128)

[3.1 Objectif de l’exercice 5](#_Toc136863129)

[3.2 Explications / Description 6](#_Toc136863130)

[3.3 Extrait de code 6](#_Toc136863131)

[3.4 Capture 7](#_Toc136863132)

[4 Exercice 3 (PHP) 7](#_Toc136863133)

[4.1 Objectif de l’exercice 7](#_Toc136863134)

[4.2 Explications / Description 7](#_Toc136863135)

[4.3 Extrait de code 8](#_Toc136863136)

[4.4 Capture 9](#_Toc136863137)

[5 Exercice 4 (JSFiddle) 9](#_Toc136863138)

[5.1 Objectif de l’exercice 9](#_Toc136863139)

[5.2 Explications / Description 9](#_Toc136863140)

[5.3 Extrait de code 9](#_Toc136863141)

[5.4 Capture 9](#_Toc136863142)

[6 Exercice 5 10](#_Toc136863143)

[6.1 Objectif de l’exercice 10](#_Toc136863144)

[6.2 Explications / Description 10](#_Toc136863145)

[6.3 Extrait de code 10](#_Toc136863146)

[6.4 Capture 12](#_Toc136863147)

[7 Exercice 6 13](#_Toc136863148)

[7.1 Objectif de l’exercice 13](#_Toc136863149)

[7.2 Explications / Description 13](#_Toc136863150)

[7.3 Extrait de code 13](#_Toc136863151)

[7.4 Capture 14](#_Toc136863152)

[8 Exercice 7 15](#_Toc136863153)

[8.1 Objectif de l’exercice 15](#_Toc136863154)

[8.2 Explications / Description 15](#_Toc136863155)

[8.3 Extrait de code 15](#_Toc136863156)

[8.4 Capture 15](#_Toc136863157)

[9 Exercice 8 15](#_Toc136863158)

[9.1 Objectif de l’exercice 15](#_Toc136863159)

[9.2 Explications / Description 15](#_Toc136863160)

[9.3 Extrait de code 15](#_Toc136863161)

[9.4 Capture 15](#_Toc136863162)

[10 Exercice 9 15](#_Toc136863163)

[10.1 Objectif de l’exercice 15](#_Toc136863164)

[10.2 Explications / Description 15](#_Toc136863165)

[10.3 Extrait de code 15](#_Toc136863166)

[10.4 Capture 15](#_Toc136863167)

[11 Exercice 10 15](#_Toc136863168)

[11.1 Objectif de l’exercice 15](#_Toc136863169)

[11.2 Explications / Description 15](#_Toc136863170)

[11.2.1 Document.onreadystatechange 15](#_Toc136863171)

[11.3 Extrait de code 15](#_Toc136863172)

[11.4 Capture 15](#_Toc136863173)

[12 Trois égals 15](#_Toc136863174)

[13 Exercice 11 15](#_Toc136863175)

[13.1 Objectif de l’exercice 15](#_Toc136863176)

[13.2 Explications / Description 15](#_Toc136863177)

[13.3 Extrait de code 16](#_Toc136863178)

[13.4 Capture 16](#_Toc136863179)

[14 Exercice 12 16](#_Toc136863180)

[14.1 Objectif de l’exercice 16](#_Toc136863181)

[14.2 Explications / Description 16](#_Toc136863182)

[14.3 Extrait de code 16](#_Toc136863183)

[14.4 Capture 16](#_Toc136863184)

[15 Exercice 13 16](#_Toc136863185)

[15.1 Objectif de l’exercice 16](#_Toc136863186)

[15.2 Explications / Description 16](#_Toc136863187)

[15.3 Extrait de code 16](#_Toc136863188)

[15.4 Capture 16](#_Toc136863189)

[16 Exercice 14 16](#_Toc136863190)

[16.1 Objectif de l’exercice 16](#_Toc136863191)

[16.2 Explications / Description 16](#_Toc136863192)

[16.3 Extrait de code 19](#_Toc136863193)

[16.4 Capture 19](#_Toc136863194)

[17 Exercice 15 19](#_Toc136863195)

[17.1 Objectif de l’exercice 19](#_Toc136863196)

[17.2 Explications / Description 19](#_Toc136863197)

[17.3 Extrait de code 20](#_Toc136863198)

[17.4 Capture 20](#_Toc136863199)

[18 Exercice 16 20](#_Toc136863200)

[18.1 Objectif de l’exercice 20](#_Toc136863201)

[18.2 Explications / Description 20](#_Toc136863202)

[18.3 Extrait de code 20](#_Toc136863203)

[18.4 Capture 20](#_Toc136863204)

[19 Exercice 17 20](#_Toc136863205)

[19.1 Objectif de l’exercice 20](#_Toc136863206)

[19.2 Explications / Description 20](#_Toc136863207)

[19.2.1 REST 20](#_Toc136863208)

[19.2.2 SOAP 21](#_Toc136863209)

[20 Exercice 18 21](#_Toc136863210)

[20.1 Objectif de l’exercice 21](#_Toc136863211)

[20.2 Explications / Description 21](#_Toc136863212)

[20.3 Extrait de code 21](#_Toc136863213)

[20.4 Capture 21](#_Toc136863214)

[21 Exercice 20 21](#_Toc136863215)

[21.1 Objectif de l’exercice 21](#_Toc136863216)

[21.2 Explications / Description 21](#_Toc136863217)

[21.3 Extrait de code 21](#_Toc136863218)

[21.4 Capture 21](#_Toc136863219)

[22 Conclusion 21](#_Toc136863220)

# Introduction

# Exercice 1 (Découverte des écouteurs)

Comment ajouter un écouteur "click" via Javascript.

Lorsque l’on lance l’application, on va appeler une méthode pour ajouter un écouteur. Voici la ligne de code que l’on va mettre dans la méthode dans le javascript :

document.getElementById("testez").addEventListener("click", testez);

Le premier « testez » correspond à l’élément sur lequel appliquer l’écouteur, ensuite le « click » correspond au type d’action auxquels l’écouteur va réagir. Et le dernier testez est la méthode que l’on va faire lorsque l’écouteur s’activera.

Et maintenant sur notre élément où l’on veut ajouter l’écouteur, on va ajouter un id avec le nom testez :

<button id="testez">

Comment ajouter directement un écouteur "click" sur un bouton.

Pour ajouter l’écouteur directement sur le bouton, on va ajouter un attribut onclick avec le nom de la méthode à exécuter lorsque l’écouteur sera activé. Voici un exemple :

<button onclick="testez()">

Expliquez comment exécuter du code Javascript lorsque la page HTML a fini d'être chargée.

Pour pouvoir exécuter une méthode lorsque la page aura fini de se charger, on va modifier le body et mettre cela :

<body onload="initCtrl()">

Le initCtrl est le nom de la méthode à lancer.

Expliquez ce que signifie DOM.

Il signifie Document Object Model et permet de créer un modèle pour javascript qui converti tous les éléments en object pour que l’on puisse par la suite effectuer des requêtes avec javascript sur notre document html.

Où le script Javascript peuvent être chargés et pourquoi.

Il est chargé dans le body et dans le onload. Il est exécuté à ce moment étant donné que si on le fait avant, le script sera exécuté alors que les éléments ne sont pas encore créés.

# Exercice 2

## Objectif de l’exercice

Dans cet exercice, le but sera de découvrir un peu les débuts de javascript. Le but sera de contrôler si un username et un password écrit par un utilisateur correspond à des valeurs en dur dans le programme.

## Explications / Description

Le code pour le fichier html était donné et pour le css et le javascript, nous avons dû le faire nous-même.

Tout d’abord, il y aura les fonction alert(), confirm() et prompt() qui permettent de faire des popup. Alert() donne une information, confirm() une popup de confirmation et prompt() demande du texte.

L’attribut placeholder dans les textes d’inputs permettent de mettre une description de ce que l’on veut dans le champ. Et l’attribut autofocus permet de spécifier que ce champ sera mis en avant au lancement de la page.

La différence entre une balise button et un bouton input est que pour un input, on ne pourra mettre que du texte tandis que dans un button, on peut mettre une image.

Ensuite pour récupérer du texte dans notre input de texte par exemple, on va d’abord récupérer l’élément grâce à document.getElementById et ensuite on va mettre un point et value. Voici un exemple :

document.getElementById("username").value

Comme dans netbeans, on peut écrire dans la console. Pour cela, on va faire :

Console.log("texte à écrire")

Cela permet par exemple de voir une valeur d’un élément à un moment donné. Le message sera donc affiché dans la console et pour le voir, on va faire « inspecter la page » et ensuite on va sous la console. Cela nous permet aussi de voir les différentes erreurs.

Pour créer une fonction, on va juste mettre function et le nom de la fonction. Voici un exemple :

function validerUtilisateur() {}

Et pour créer une variable, on va faire comme dans java mais on ne met pas le type au début mais « let » voici un exemple :

let username = “admin”

Les variables n’ont donc pas de type.

Pour convertir une chaîne de caractère en minuscule, on va faire la chaîne de caractère point toLowerCase(). Voici un exemple :

username = username.toLowerCase()

## Extrait de code

Voici la fonction qui va être effectuée lorsque l’on va appuyer sur le bouton valider :

function validerUtilisateur() {  
 let username = document.getElementById("username").value  
 let password = document.getElementById("password").value  
 username = username.toLowerCase()  
 console.log(username)  
 console.log(password)  
 if (username == "admin" && password == "emf123") {  
 window.alert("Validation OK.")  
 }else{  
 window.alert("Utilisateur ou mot de passe incorrect !!!")  
 }  
}

## Capture

Une image contenant texte, logiciel, Logiciel multimédia, capture d’écran

Description générée automatiquementVoici tout d’abord une capture du site au début quand l’utilisateur n’a rien écrit :

Et maintenant lorsque l’on fait le login juste :

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquementUne image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquementEt pour finir lorsque le login n’est pas juste :

# Exercice 3 (PHP)

## Objectif de l’exercice

Dans cet exercice, le but sera de découvrir les liens avec le backend et frontend.

## Explications / Description

Dans cet exercice, on va donc reprendre le même exercice que le 2 mais à la place de faire le code dans javascript, on va le faire dans PHP.

Voici la liste des boutons disponible avec une petite explication :

* Button : Simple bouton cliquable
* Submit : Bouton qui envoie un formulaire vers un serveur
* Reset : Réinitialiser les valeurs d’un formulaire
* Image : Bouton représenté par une image
* Checkbox : Bouton qui peut être coché ou décoché
* Radio : Bouton sélectionné dans un groupe de bouton
* File : Bouton où l’on peut sélectionner un fichier par le client

Un fichier PHP va être de cette manière :

<?PHP

?>

Ensuite, on devra choisir si on veut faire un GET ou un POST étant donné que de base pour récupérer des informations, on va utiliser le GET mais pour un login, on va plus souvent utiliser un POST parce que comme ça les paramètres ne seront pas affichés dans l’URL. Mais le type de requête que l’on va définir dans l’html devra être le même que celui utilisé dans PHP.

Pour spécifier les éléments que l’on va envoyer au fichier PHP, dans les input, on va ajouter un attribut name avec le nom qu’on veut qu’il sois envoyé.

Pour concaténer deux chaînes de caractères, nous allons utiliser le point. Voici un exemple :

echo "username: ".$username."</br>";

Ensuite pour mettre en minuscule, on va utiliser la fonction strtolower. Voici un exemple :

$username = strtolower($\_POST['username']);

Pour renvoyer des éléments au client, on va utiliser le echo. Voici un exemple où l’on renvoie quelque chose.

echo "<script>alert('Validation OK');</script>";

Dans notre fichier PHP, on va aussi retrouver du javascript.

## Extrait de code

Pour les extraits de code, il n’y en a pas beaucoup. Mais voici quand même le code pour le bouton :

<div class="button">

<input type="submit" value="Valider" id="valider">

</div>

Ensuite voici un exemple de l’input pour le username :

<input type="text" size="30" id="username"

placeholder="un nom svp" autofocus name="username"/>

Et pour finir, voici la ligne pour le form :

<form class="user-form" action="https://carrardr.emf-informatique.ch/307/Exercices/Exercice\_3/login.php" method="post">

## Capture

Une image contenant texte, logiciel, capture d’écran

Description générée automatiquementPour les captures d’écran, cela sera les mêmes que pour l’exercice d’avant sauf que le design a un peu changé donc voici le nouveau design :

# Exercice 4 (JSFiddle)

## Objectif de l’exercice

Dans cet exercice, l’objectif est de découvrir le site JSFiddle et de voir à quoi il sert.

## Explications / Description

JSFiddle est un site disponible sur internet qui permet de tester du code HTML, CSS et javascript. Cela nous permet donc de voir si notre code fonctionne et si tout est comme on le veut. L’avantage est que l’on n’a pas besoin de faire toute la structure de base pour html. On peut directement commencer par une balise avec un bouton par exemple.

## Extrait de code

Pour cet exercice, il n’y aura pas d’extrait de code étant donné que le code que l’on doit faire est le même que celui dans les exercices précédents.

## Capture

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquementEt pour la capture, voici la capture d’écran du site en question avec le résultat de l’exercice :

# Exercice 5

## Objectif de l’exercice

Dans cet exercice 5, on va principalement travailler sur les variables. On va devoir en concaténer, en additionner etc.

## Explications / Description

Pour déclarer une variable dans javascript, il existe 4 manières. La première est celle-ci :

maVar;

Les fonctions sont identiques à celle-ci :

var maVar;

Les deux ne sont plus utilisés étant donné que l’on peut déclarer deux fois la même variable et on a donc trop de libertés.

Maintenant, on utilise le mot-clé let :

let maVar;

C’est celle qui est le plus utilisés étant donné que comparé aux deux autres, elle ne peut pas être déclarée une deuxième fois.

Et pour finir il y a le mot-clé const :

const maVar="Valeur";

C’est une sorte de constante étant donné que l’on ne peut pas réaffecter la variable mais on peut modifier par exemple si c’est un tableau on pourra modifier les cellules.

Pour la convention de nommage, comme en java, les variables basiques s’écriront en dos de chameau et les constantes en majuscules.

Pour le premier type de déclaration, c’est celle que l’on va utiliser pour déclarer des variables globales.

Dans javascript, il existe que 4 types de base : « number » pour un nombre, « string » pour une chaîne de caractères, « boolean » pour un booléen et « object » pour n’importe quel objet.

Pour connaitre le type d’une variable, on peut utiliser typeof. Voici un exemple :

console.log(typeof(maVar));

Et voici le résultat dans la console pour un objet :

## Extrait de code

Pour l’extrait de code, voici le code en entier commenté par partie :

// Vider la console  
console.clear();

// Assigner et afficher la variable a  
let a;  
console.log(a);  
a = 15;  
console.log("Ma variable a = " + a);

// Identique pour la b  
let b = 9;  
console.log("Ma variable b = " + b);

// Calcul entre les deux variables  
console.log(`${a} + ${b} = ${a + b}`);  
console.log(`${a} - ${b} = ${a - b}`);  
console.log(`${a} \* ${b} = ${a \* b}`);  
console.log(`${a} / ${b} = ${a / b}`);

// Concaténation de String  
a = "Bonjour";  
b = " les amis";  
console.log(a + b);  
console.log(`${a}${b}`);

// Vérifications entre des booléen  
a = true;  
b = false;  
console.log(`${a} AND ${b} = ${a && b}`);  
console.log(`${a} OR ${b} = ${a || b}`);

// Calcul avec les Dates  
a = new Date(Date.now());  
b = new Date();  
b.setDate(a.getDate() - 61);  
console.log(a.toLocaleDateString() + " " + a.toLocaleTimeString());  
console.log(b.toLocaleDateString() + " " + a.toLocaleTimeString());  
console.log(a.toLocaleDateString());  
console.log(a.toLocaleTimeString());  
console.log(b.toLocaleDateString());  
console.log(b.toLocaleTimeString());

// Observation des différents types  
a = Math.PI;  
b = "bonjour";  
let c = true;  
let d = new Date(Date.now());  
let e;  
console.log(typeof(a));  
console.log(typeof(b));  
console.log(typeof(c));  
console.log(typeof(d));  
console.log(typeof(e));

## Capture

Une image contenant texte, capture d’écran, Police

Description générée automatiquementEt dans les captures, voici une capture avec une partie des résultats :

Et maintenant les résultats complet :

Undefined  
"Ma variable a = 15"  
"Ma variable b = 9"  
"15 + 9 = 24"  
"15 - 9 = 6"  
"15 \* 9 = 135"  
"15 / 9 = 1.6666666666666667"  
"Bonjour les amis"  
"Bonjour les amis"  
"true AND false = false"  
"true OR false = true"  
"16.05.2023 09:31:28"  
"16.03.2023 09:31:28"  
"16.05.2023"  
"09:31:28"  
"16.03.2023"  
"09:31:28"  
"number"  
"string"  
"boolean"  
"object"  
"undefined"

# Exercice 6

## Objectif de l’exercice

Dans l’exercice, nous allons utiliser un if ou un switch et on va aussi de nouveau utiliser les dates.

## Explications / Description

Dans cet exercice, on devra afficher le jour de la semaine lorsque l’on appuie sur le bouton. Donc d’abord il faudra récupérer le numéro du jour actuel. Pour cela on va initialiser une date et on va récupérer le jour de la semaine grâce à getDay().

Pour insérer du texte dans une balise html existante, on va utiliser la balise document.getElementByID avec l’ID de l’élément. Et ensuite on va utiliser le innerHTML pour écrire dans l’élément ce que l’on va mettre après le innerHTML. Voici un exemple :

document.getElementById("info").innerHTML = s + "mardi";

Pour ajouter du code c’est de la même façon. Maintenant pour créer un tableau, on va créer une variable et on va mettre les crochets. Dans els crochets on pourra mettre les valeurs que l’on veut. Voici un exemple de création :

const tab = ["dimanche", "lundi", "mardi","mercredi","jeudi","vendredi","samedi"];

Ensuite on va pouvoir effectuer plein d’action différentes en utilisant le nom du tableau un point et le nom de l’action. Pour la première action, c’est pour ajouter un élément au début d’un tableau. Pour cela on va utiliser le unshift. Pour ajouter à la fin, on va utiliser le push. Pour supprimer le premier, on utilise le shift et pour supprimer le dernier le pop. Voici les commandes :

tab.push('Eric'); // ajoute Eric à la fin du tableau  
tab.unshift('AA'); // ajoute AA au début du tableau  
tab.pop(); // efface le dernier élément du tableau  
tab.shift(); // efface le premier élément du tableau

Pour récupérer la valeur d’un case précise d’un tableau, on va mettre le tableau et entre crochets le numéro de la case. Voici un exemple :

tab[2]

Et pour finir pour afficher toutes les valeurs d’un tableau, on va faire une boucle foreach comme la suivante :

noms.forEach(nom => console.log (nom));

## Extrait de code

Comme extrait de code, je vais mettre la façon la plus simple pour afficher le jour de la semaine. Dans cette façon on va utiliser un tableau et on va récupérer la valeur du tableau qui correspond à la position du jour correspondant :

function afficherJourSemaineTableau() {  
 let date = new Date();  
 let s = "On est ";  
 const tab = ["dimanche", "lundi", "mardi","mercredi","jeudi","vendredi","samedi"];  
 const jour = tab[date.getDay()]  
 document.getElementById("info").innerHTML = s + jour  
}

## Capture

Voici la capture du site avec un bouton pour chaque façon de faire : Une image contenant texte, capture d’écran, affichage, Police

Description générée automatiquement

# Exercice 7

## Objectif de l’exercice

Dans cet exercice, on va principalement utiliser les boucles. On aura donc la boucle for, la boucle while et la boucle do-while.

## Explications / Description

Pour toutes les boucles, c’est comme dans java. Mais voici quand même pour la boucle for (on connaît le nombre d’itération) :

for (let i = 0; i < 5; i++) {  
 s += " i = " + i + "<br/>";  
}

Maintenant pour la boucle while (exécuter tant que…) :

let i = 0;  
while (i<5) {  
 s += " i = " + i + "<br/>";  
 i++;  
}

Et pour finir la boucle do-while (exécuter tant que… mais au moins une fois) :

do {  
 s += " i = " + i + "<br/>";  
 i++;  
 } while (i<5);

## Extrait de code

Pour cet exercice, il n’y aura pas d’extrait de code étant donné que le code est déjà dans le point précédent pour l’explication.

## Capture

Et pour les capture d’écrans, voici l’exemple lorsque l’on appuie sur un bouton :Une image contenant texte, affichage, logiciel, Icône d’ordinateur

Description générée automatiquement

# Exercice 8

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Exercice 9

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Exercice 10

## Objectif de l’exercice

## Explications / Description

### Document.onreadystatechange

## Extrait de code

## Capture

# Trois égals

# Exercice 11

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Exercice 12

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Exercice 13

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Exercice 14

## Objectif de l’exercice

Dans cet exercice, on ne va pas vraiment faire du code étant donné que l’on aura un site internet avec des slides pour expliquer les principes de base de jquery

## Explications / Description

Jquery est une librairie disponible pour javascript. Elle permet de faciliter énormément les échanges entre du code javascript et du code HTML. Elle simplifie énormément la manipulation du DOM.

Dans jquery, on va tout le temps utiliser le $ avec des parenthèses. Cela nous permet de sélectionner un élément dans notre HTML. C’est donc grâce à cela qu’on peut reconnaitre du code jquery.

Les deux tâches principales d’un code jquery est de retrouver un ou plusieurs éléments dans un fichier HTML et ensuite de faire une action avec ce ou ces éléments.

Pour retrouver un élément de notre HTML, on va mettre le nom de l’élément entre parenthèses dans le $(""). Voici un exemple où l’on va récupérer l’ensemble des div du document :

$("div")

Ensuite si on veut sélectionner un ID, on va mettre un # juste avant le nom. Et pour une classe CSS, on va rajouter un point avant.

Lorsque l’on lance la page, la fonction ready va être lancée. Voici la structure que l’on aura.

$(document).ready(function(){

// Code jQuery ici

});

Dans jquery, on pourra sélectionner des éléments par rapport à un autre. Par exemple on peut sélectionner l’élément parent d’un élément. Il existe 5 sélections comme celle-ci. On aura donc parent() qui sélectionne le parent, next() qui sélectionne le prochain élément, prev() pour celui d’avant, children() pour les éléments enfants et siblings() pour les éléments frères. Voici un exemple d’utilisation de ces sélections :

$("button").parent().css("border", "3px solid red");

Ensuite si on veut modifier du css pour un élément, on va mettre un point et css. Ensuite on va mettre entre parenthèses des crochets et ensuite on pourra mettre du code css comme si on était dans le fichier. Voici un exemple :

$("li a").css({   
 color: "red",   
 fontWeight: "bold"   
});

Pour ajouter un élément après celui sélectionner, on va utiliser le after. Et si on veut mettre un élément avant ce sera le before :

$("a[target=\_blank]")  
 .after("<img src='images/open.png'/>");

Sinon on aura aussi le append pour ajouter directement du texte dans la balise de l’élément. Voici un exemple :

$("a[target=\_blank]")  
 .append("(mon texte)");

Ensuite si on veut supprimer un élément, on va utiliser le remove. Pour cela ce sera comme pour ajouter mais on ne va rien mettre entre parenthèses. Voici l’exemple :

$("a[target=\_blank]").remove()

Pour les formulaires, on peut utiliser l’évènement submit(). Sinon on aura aussi l’événement click() qui va nous permettre de faire une action dès que l’on va cliquer sur l’élément sélectionné. Et dans le même style on a hover mais ici l’action sera faite uniquement lorsque l’on aura la souris sur l’élément.

L’évènement toogle() permet d’activer ou de désactiver certain éléments. Par exemple dans l’exemple suivant, on va activer ou désactiver les éléments lorsque l’on va cliquer sur le menu :

$("a.menu").click(function(){  
 $(this).next().toggle();  
 return false;  
});

On peut aussi utiliser le animate pour faire une petite animation durant l’action :

$("li").hover(function(){  
 $(this).animate({marginLeft: 38, marginRight: 0});  
}, function(){  
 $(this).animate({marginLeft: 18, marginRight: 18});  
});

On va utiliser aussi l’élément live() qui va remplacer le hover. Cela nous permet de faire ces actions aussi sur les nouveaux éléments et pas seulement sur ceux de base du fichier. Voici un exemple :

$("a.menu").live("hover", function(){  
 $(this).next().toggle(200);  
 return false;  
});

Ensuite pour cacher et afficher un élément, on va utiliser hide() et show(). Le premier argument est la vitesse de l’action et le deuxième est ce que l’on va faire l’orsque l’on aura fini l’action. Dans l’exemple suivant on va d’abord cacher l’élément et ensuite on va le réafficher :

$("div.block").hide("slow", function(){  
 $(this).show("slow");  
});

Pour lire un fichier avec ajax, on va faire $.ajax({}). Entre les accolades, on va mettre tout d’abord l’url du fichier XML et ensuite on va faire une fonction. Voici la déclaration avec un exemple de fonction :

$.ajax({  
 url: "file.xml",  
 success: function( xml ) {  
 $(xml).find("tab").each(function(){  
 $("ul").append(  
 "<li>" + $(this).text() + "</li>");  
 });  
 }  
});

Ensuite pour un fichier JSON, on va utiliser la méthode $.getJSON(). Dans cette méthode, on va d’abord mettre en paramètre le fichier json et la fonction que l’on va faire. Dans la fonction on peut par exemple lire tout les éléments dans le json et les ajouter à une liste. Voici l’exemple en code :

$.getJSON("file.json", function( obj ) {  
 for ( var prop in obj ) {  
 $("ul").append(  
 "<li>" + prop + ": " + obj[prop] + "</li>");  
 }  
});

Pour décider de charger un fichier HTML autre part, on va utiliser le load(). On va donc d’abord mettre l’élément dans lequel stocker et ensuite le fichier HTML. Voici un exemple :

$("div.load").load("file.html");

Et pour finir pour intégrer jquery dans notre projet, on va mettre dans le head du HTML (comme pour le js) cette ligne :

<script src='http://code.jquery.com/jquery.js'></script>

## Extrait de code

Dans cet exercice, il n’y aura pas d’extrait de code étant donné que l’on n’a pas fait de code nous-même.

## Capture

Et pour les captures, c’est pareil il n’y en aura pas pour cet exercice.

# Exercice 15

## Objectif de l’exercice

Dans l’exercice 15, on va commencer à utiliser jquery dans un vrai programme. Le principe est de changer la couleur de fond de la page avec jquery et avec une animation.

## Explications / Description

Déjà au tout début il faudra récupérer la valeur de la couleur que l’on a choisi. Pour cela on va récupérer notre liste avec l’id couleurs et ensuite on va obtenir sa valeur avec « .val() ».

$("#couleurs").val()

Ensuite pour changer la couleur de fond, on va utiliser le « .css » et on va modifier l’attribut background-color. On va utiliser aussi color pour changer la couleur de l’écriture. Voici un exemple du css :

$(#container).css({

"background-color": "green",

"color": "white"

})

Et pour finir pour mettre une animation de disparition et de réapparition, onva utiliser le fadeOut durant une seconde pour qu’il disparaisse, après on fait le css et pour finir on fait un slideToogle pour qu’il réapparaisse en glissant depuis le haut.

## Extrait de code

## Capture

# Exercice 16

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Exercice 17

## Objectif de l’exercice

Dans cet exercice, on va faire des présentation pour découvrir les deux webservices et les possibilités.

## Explications / Description

On a d’abord deux webservices possibles. Le premier est REST et le deuxième SOAP.

### REST

Pour REST, on aura une architecture client-serveur avec des identifiants pour les ressources (URI). On va pouvoir faire 5 types de requêtes que l’on va faire avec http. Elles sont expliquées dans les points suivant sauf pour le PATCH que l’on ne va pas utiliser. Le plus souvent, on va représenter les données avec du JSON mais c’est aussi possible avec du XML et du HTML.

REST est stateless ce qui veut dire qu’il ne prend pas en compte les requêtes précédentes. Il répond également avec du code.

#### GET

#### POST

#### PUT

#### DELETE

### SOAP

# Exercice 18

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Exercice 20

## Objectif de l’exercice

## Explications / Description

## Extrait de code

## Capture

# Introduction du projet

## Définition: but, objectifs et fonctionnement désiré de votre projet

## Explications API

### Site de référence

### Fonctionnement: requête, réponse, format

### Exemple d’utilisation

# Analyse

## Diagramme de use cases / Explications des cas

Une image contenant texte, diagramme, ligne, capture d’écran

Description générée automatiquementPour débuter cette analyse, voici le use case de mon site :

On pourra donc récupérer la liste des compositeurs et la filtrer pour l’afficher ensuite. Après on pourra récupérer les détail d’un compositeur. Et pour finir on peut récupérer les détails d’un morceau.

## Maquette

Pour la maquette, voici la maquette pour la page avec la liste de tous les compositeurs :

On aura donc la liste des composteurs à droite avec leur photo et à gauche un filtre pour les compositeurs.

Une image contenant texte, capture d’écran, Police

Description générée automatiquementUne image contenant texte, capture d’écran, Police, nombre

Description générée automatiquementVoici maintenant la page pour les détails du compositeur :

On aura quelques informations sur le compositeurs et aussi tout les morceau qu’il a écrit.

Une image contenant texte, capture d’écran, Police

Description générée automatiquementEt pour les détails d’un morceau, voici la page :

On aura aussi quelques détails sur le morceau et le compositeur à la fin.

# Conception

## Diagramme de navigation

Une image contenant texte, diagramme, ligne, Plan

Description générée automatiquementPour la conception, voici le diagramme de navigation :

Dans ce diagramme la partie la plus intéressant est la partie en bas à droite. On aura donc le indexCtrl.js qui va créer le service.js qui va lui communiquer avec le serveur qui contient l’API. Les trois autres fichier js utiliseront le service.js. et pour finir les fichiers html utiliseront le fichier js correspondant.

# Implémentation

## HTML / CSS (buts des fichiers, éventuellement des extraits de code avec explications)

## Javascript général (buts des fichiers, extraits de code avec explications)

## Javascript spécifiques (buts des fichiers, extraits de code avec explications)

## Descente de code complète pour une action qui fait appel à un service Web

# Conclusion