

FASCICULE DES TRAVAUX DIRIGE

Ingénierie du Web

2016-2017

Version 2.2

A.ABDALLAH

TD1: HTML

TD2: CSS

TD3: JavaScript

TD4: (HTML+CSS+JS)

TD5: APACHE, PHP, MYSQL

TD 6: PHP+MYSQL +AJAX+JSON

TD7: Angular JS

TD8: ASP.NET

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 1 : Initiation HTML	

Vous allez découvrir dans ce TD comment fonctionne le web, créer un document HTML de base, apprendre à structurer votre contenu, ajouter des liens, des images et des tableaux. Enfin, vous vérifierez la validité de votre code.

SERVEUR WEB

Le rôle d'un serveur web est de *servir* (livrer) des documents web aux clients. Ceux-ci sont notamment : les pages web (HTML, PHP, ...), les images, les vidéos, mais également les scripts JavaScript, les feuilles de style CSS, les animations Flash, etc.

HTML

Comme nous l'avons déjà vu, **HTML** signifie **Hyper Text Markup Language** (langage de balises hyper texte). Concrètement, c'est du texte, avec des balises (éléments de syntaxe et de structuration) pour délimiter les différentes zones de texte et définir leur « catégorie ». HTML permet de créer des pages complètes, avec tout le contenu textuel (et sa structuration en titres, paragraphes, tableaux, ...), les images, les liens...

Par contre, idéalement, HTML ne s'occupe jamais de la mise en forme. C'est le rôle de CSS.

BALISES HTML

Mon objectif va être de vous faire découvrir les balises par vous-même, et qu'à la fin des exercices vous disposiez d'une page de référence contenant toutes les balises utiles et un exemple d'utilisation pour chacune. Pour cela, vous aurez sûrement besoin d'une référence (un manuel) en matière de balises HTML afin de vérifier la syntaxe et la signification des balises que vous allez devoir utiliser.

A toute fin utile, utilisez ceci : <https://developer.mozilla.org/fr/docs/Web/HTML/Element>.

Et n'oubliez pas de regarder le résultat de votre travail dans un navigateur, après chaque modification de votre fichier HTML !

EDITEUR DE TEXTE

Vous allez le voir très rapidement, mais le code informatique, de manière générale, se matérialise par du texte simple. Il est donc inutile d'avoir un logiciel de traitement spécifique pour faire du code, et tout programme capable de sauvegarder un document « .txt » fera l'affaire. Ceci étant dit, le meilleur outil, c'est celui que vous maîtrisez le mieux. Donc, si vous avez un éditeur de prédilection, utilisez-le ! Sinon, je vous recommande **Brackets.io**, **Atom.io**, **Gedit** ou **Notepad++**.

Exercice 1

1. A l'aide de votre outil d'édition favori, écrivez le code suivant :

```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8" />
5          <title>Hello World !</title>
6      </head>
7      <body>
8          Ceci est ma première page HTML !
9      </body>
10 </html>

```

2. Sauvegardez votre document avec le nom **01-hello_world.html**, sur votre espace de travail, dans le dossier TD1 que vous créerez pour l'occasion.
3. Ouvrez votre fichier avec votre navigateur.
4. Bravo :-)
5. Allez sur <http://validator.w3.org/> et validez que votre première page HTML est correctement structurée.

Exercice 2 : Mise en forme du texte

Commençons par le plus simple, mais aussi le plus riche en balises : la mise en forme du texte.

1. Vous allez créer un nouveau document vide, y placer correctement les balises spéciales requises (html, head, meta, body), et sauvegarder votre fichier avec le nom **02-refcard.html**.
2. Partant de cette page vide, et comme vu précédemment, vous allez commencer par ajouter un titre pour la page : « ma carte de référence sur les balises HTML ».
3. Vous allez utiliser les chapitres pour structurer votre contenu. Donc ajoutez une balise `<h1>` avec pour contenu « La structuration de données en HTML ».
4. Ajoutez une balise `<h2>` avec pour contenu « La mise en forme du texte ».
5. Essayez de saisir le texte suivant, sur 2 lignes : « nous allons découvrir ici quelles balises il faut utiliser pour mettre en forme du texte dans un document HTML. »
6. Découvrez la balise `
`, son utilité, et n'hésitez pas à l'utiliser à chaque fois que nécessaire!
 - a. Vous remarquez que `
` est une balise qui se ferme automatiquement. On peut voir `
` seul, mais cela signifie qu'il manquera une balise fermante. C'est accepté en HTML5, mais je préfère que vous fermiez toutes vos balises, donc utilisez `
`. On y reviendra.
 - b. Découvrez la balise `<p>` et écrivez un exemple de son utilisation : « ceci est mon premier paragraphe, j'utilise la balise spécialement prévue pour mettre en forme ce genre de contenu et automatiquement sauter les lignes adéquates en début et en fin de paragraphe. Une balise pratique ! Elle se distingue de la balise `br` par le fait qu'elle ne sert pas à aller à la ligne, mais bien à changer de paragraphe. C'est sémantiquement différent. ».
7. Découvrez la balise `<p>` et écrivez un exemple de son utilisation : « ceci est mon premier paragraphe, j'utilise la balise spécialement prévue pour mettre en forme ce genre de contenu et automatiquement sauter les lignes adéquates en début et en fin de paragraphe. Une balise pratique ! Elle se distingue de la balise `br` par le fait qu'elle ne sert pas à aller à la ligne, mais bien à changer de paragraphe. C'est sémantiquement différent. ».
8. Découvrez la balise `<abbr>` et servez-vous en pour définir le mot « HTML ».
9. Découvrez la balise `<dd>` et celles qui lui sont nécessaires, et définissez ce qu'est HTTP.
10. Découvrez la balise `` et saisissez le texte : « Internet et le web, c'est pareil ! ».

11. Découvrez la balise <ins> et saisissez le texte : « Internet c'est le réseau, le web c'est les sites ! ».
12. Découvrez les balises , et <mark>. Créez un nouveau paragraphe et insérez la phrase suivante : « En HTML5, de nouvelles balises sont apparues, comme mark par exemple. Elles ne servent qu'à définir la sémantique du contenu (mise en valeur, importance, surlignage). L'apparence change par défaut, mais on pourra la modifier en CSS. » et faites en sorte que « HTML5 » soit mis en valeur, « balises » soit important et « mark » soit surligné.
13. Découvrez la balise et celles qui lui sont associées, et créez une liste formatée ainsi :
 - a. HTML
 - c'est bien mais HTML5 c'est mieux
 - b. CSS
 - c'est encore mieux ! mais c'est beaucoup plus compliqué !
14. Découvrez les balises <sub> et <sup> et insérez le texte « c'est incroyable, H₂O et M^{lle} sont correctement affichés en HTML. »
15. Enfin, découvrez l'élément <cite> et insérez le texte suivant : « Parce qu'il n'y a pas d'apostrophe à Polytech, la brigade de l'apostrophe sanctionne ! ».
16. Modifiez votre page pour qu'avant l'apparition de chacun des exemples de balise que vous avez utilisé, vous ayez un titre correspondant, avec le bon niveau de hiérarchie (<h3> donc, si vous suivez !) et le nom de la balise utilisée en contenu. N'hésitez pas à reformater votre page, et supprimer des paragraphes (balise <p> si besoin !)
17. Sauvegardez votre document sur votre espace

Remarque importante : vous constatez que ces balises modifient la mise en forme du texte. Ce n'est pas leur objectif premier. En effet, cette mise en forme est complètement modifiable en CSS. Par contre, ces balises servent à enrichir, préciser le contexte et ajouter de la sémantique à votre contenu. C'est là, l'utilité fondamentale. Plus votre document est bien structuré, plus il sera lisible, et plus facilement il sera modifiable avec CSS.

LIENS

Sans les liens hypertextuels, le web n'est qu'une série de pages contenant du texte sans aucun moyen de faire référence de l'une à l'autre. La balise <a> pour *anchor* (ancree) fait partie de celles qui n'ont pas de signification sans attribut. Le plus important, le plus connu (et le seul vraiment utile d'ailleurs) est l'attribut « href », qui spécifie la destination du lien.

Il existe au moins 3 types de liens :

- Ceux qui envoient sur une autre page de votre site.
- Ceux qui envoient sur une page d'un autre site.
- Ceux qui envoient vers une autre section de votre site.

Les trois utilisent la même balise : <a>, mais avec un attribut href différent :

- le premier indique tout simplement le nom (et si besoin, le chemin) du fichier à ouvrir
- le deuxième indique l'adresse complète, en commençant impérativement par le protocole (ex : http://...)
- le troisième utilise le nom de l'ancree interne à la page, vers laquelle il doit pointer. Les noms d'ancres commencent par « # »

- pour définir une ancre dans le contenu d'une page, il faut utiliser la syntaxe suivante : ``. Puis on peut créer un lien qui permettra d'accéder à cette ancre rapidement, ainsi : `Retourner à mon ancre`.

Exercice 3 : Mise en place d'hyperliens

Dans le même fichier `02-refcard.html` :

1. Ajouter un titre de niveau 2 et indiquer « La gestion des liens »
2. Faites ce qu'il faut pour créer un lien qui renvoie en haut de la page avec pour texte « Retour au début ».
3. créez un lien vers votre première page `01-hello_world.html`, avec le texte « Redécouvrez ma première page ! »
4. créez un lien vers le validator w3c avec pour texte « Accès au W3C validator ».
5. ajouter les titres de niveau 3 adéquats pour ces différents exemples.
6. Sauvegardez votre document sur votre espace.

HYPERLIENS AVANCES

Lorsque qu'une page souhaite accéder à une autre page dans le même site, mais dans un autre dossier, il faut utiliser les chemins des dossiers dans les liens. Par exemple, une page qui souhaite accéder à la `page2.html` située dans le dossier `test` devra écrire un lien de la forme :

```
<a href="test/page2.html">voir la page 2</a>
```

Le système d'ancre interne à une page fonctionne également avec les liens vers d'autres pages. Il suffit d'utiliser une syntaxe du type :

- `voir mon ancre dans la page 2`.
- il est possible d'envoyer un e-mail avec un lien ! Pour cela, il faut utiliser la syntaxe : `envoyez moi un mail !`.

Exercice 4

1. Créez un nouveau dossier nommé « niveau_2 », et créez à l'intérieur un fichier nommé `03-test_liens.html`.
2. Dans ce fichier, qui sera syntaxiquement correct, mettez un titre au document : « Test des liens vers ancres et chemins ».
3. Ajoutez une ancre nommée « ancre_test » avec pour texte : « Ceci est une ancre pour faire un test ».
4. Toujours dans le dossier `niveau_2`, créez une nouvelle archive (un dossier compressé si vous préférez) nommée `04-test_archive.zip`.
5. Dans le fichier `02-refcard.html` :
 - a. Ajoutez un lien pour accéder à `ancre_test` dans la page `03-test_liens.html`, avec le texte « Lien vers une ancre dans une page d'un sous dossier ».
 - b. Ajoutez un lien permettant de vous envoyer un e-mail, avec pour sujet « yeah » et comme texte « Spammez-moi ! »
 - c. Ajoutez un lien pointant sur l'archive `04-test_archive.zip`, avec pour texte « Téléchargez mon .zip ! ». Testez ce lien ! Magique !
 - d. Ajoutez les titres de niveau 3 adéquats pour nos nouveaux liens.
6. Sauvegardez vos documents et dossiers sur votre espace

LES IMAGES

La balise `` fait également partie de celles qui n'ont pas de signification sans attribut. Le plus important est l'attribut « `src` », qui spécifie la *source* de l'image (le nom du fichier, et son éventuel chemin pour y accéder). L'attribut « `title` » est très utile pour afficher un texte lorsqu'on passe sur l'image avec la souris (« *tooltip* »). L'attribut « `alt` » est également très utile car il permet d'afficher un texte lorsque l'image n'a pas pu être chargée, ou lorsque le navigateur souhaite décrire l'image (description/accessibilité aux mal-voyants par exemple). Il existe 3 types d'images particulièrement adaptés et bien supportés par le web : PNG, GIF, JPEG

Exercice 5 : Ajout d'images

1. Créez un dossier nommé « images », insérez-y le logo de l'école, que vous nommerez o5-logo_insa avec l'extension correspondante à son format.
2. Dans le fichier o2-refcard.html :
 - a. Ajoutez un nouveau titre de niveau 2 et indiquez « Les images »
 - b. Insérez l'image o5-logo_insa et ajoutez un texte alternatif et un titre à l'image : « Le logo de INSA CVL »
3. Sauvegardez votre document sur votre espace.

LES TABLEAUX

Les tableaux sont assez pénibles à écrire en HTML, du fait de l'arborescence des balises. Le principe de fonctionnement est le suivant : on déclare une balise `<table>`, puis une (ou plusieurs) balises `<tr>` (table row : les lignes), qui elles-mêmes contiendront une (ou plusieurs) balises `<td>` représentant cellules (cases) du tableau. Ajoutez à cela des balises supplémentaires pour faire des entêtes de colonnes (`<th>`), des légendes (`<caption>`)

Exercice 6 : Ajout de tableau

Dans le fichier o2-refcard.html :

1. ajoutez un titre de niveau 2 nommé « Les tableaux ».
2. créez un tableau à 3 colonnes, avec pour entête « Nom », « Prénom » et « Age », et que vous remplirez avec des données de votre choix (au moins 2 lignes).
3. Ajoutez la légende : « L'âge de certaines personnes »
4. Validez votre document.
5. Sauvegardez votre document sur votre espace

LES FORMULAIRES

Les formulaires HTML sont utilisés pour collecter les entrées utilisateur. Un formulaire HTML est caractérisé par une balise `<form>` et possède un certain nombre d'éléments, de plusieurs types. Dans la suite on va étudier les deux types les plus courantes : `<input>` et `<select>`.

L'élément de formulaire la plus importante est `<input>`, utilisé pour entrer des données. Son attribut le plus important est type qui permet de définir sa fonctionnalité. Par exemple:

- `<input type="text">` crée un champ de texte (au plus 20 caractères);
- `<input type="radio">` crée un case d'option, qui permet à l'utilisateur de sélectionner l'un d'un nombre limité de choix (une seule);
- `<input type="checkbox">` crée une case à cocher, qui permet à l'utilisateur de sélectionner l'un d'un nombre limité de choix (plusieurs choix sont possibles);

- `<input type="button">` crée un bouton sur lequel on peut cliquer (par exemple, pour activer un script);
- `<input type="submit">` définit un bouton pour soumettre le formulaire à un gestionnaire des formulaires, l'adresse duquel est donné comme valeur à l'attribut action de `<form>` (voir ci-dessous);
- `<input type="image" src="monImage.png">` comme submit mais utilisant une image à la place du bouton;
- `<input type="reset">` définit un bouton pour réinitialiser toutes les valeurs du formulaire;

Exercice 7 : Ajout d'un formulaire

Créez un nouveau dossier nommé « niveau_3 », et créez à l'intérieur un fichier nommé o6-create_form.html.

1. Découvrez la balise `<SELECT>`
2. Découvrez la balise `input` et ses différents types (text, password)
3. Découvrez la balise `<form>` et ses attributs (action, method)
4. Créer un formulaire dans la page HTML. Celui-ci devra permettre à l'utilisateur de saisir :
 - son nom ;
 - son prénom ;
 - son titre de civilité (un seul choix entre « M. », « Mme » et « Mlle ») ;
 - son adresse e-mail ;
 - sa date de naissance ;
 - son métier ;
 - sa nationalité (liste déroulante) ;
 - la façon dont il a entendu parler du site, lui proposant un certain nombre de réponses possibles (plusieurs choix possibles) ;

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 2 : Initiation CSS	

OUTILS :

Pour réaliser correctement ce TD (et les suivants), je vous recommande fortement d'utiliser la dernière version de Firefox pour tester vos réalisations, et de découvrir le raccourci clavier Shift-F7 et son onglet « éditeur de styles CSS » qui vous permet de visualiser votre feuille de style, de la modifier et de visualiser l'impact des changements en direct ! Concernant l'éditeur de texte, pas de changement par rapport à HTML. Si vous avez un éditeur de prédilection, utilisez-le ! Sinon, je vous recommande Brackets, Gedit ou Notepad++.

REFERENCES :

<https://developer.mozilla.org/fr/docs/CSS/Reference>

<http://www.w3schools.com/cssref/>

MES ALLIES POUR FAIRE CE TD :

<http://www.w3.org>

<http://validator.w3.org/>

<http://jigsaw.w3.org/css-validator/>

<http://fr.openclassrooms.com/informatique/html/cours>

<http://fr.openclassrooms.com/informatique/cours/apprenez-a-creeer-votre-site-web-avec-html5-et-css3>

<http://www.w3schools.com/html/default.asp>

<http://www.alsacreations.com/tutoriels/>

<http://www.html5rocks.com/en/resources>

<https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5>

<http://diveintohtml5.info/>

<http://html5doctor.com/>

Ex 1 : Premier doc CSS

1. Créez un nouveau dossier nommé TD2 dans votre espace de travail et copiez tous les fichiers et dossiers du TD précédent dedans. (Si vous n'avez pas fini le TD précédent, utilisez les éléments fournis pour ce TD)
2. Créez un sous dossier nommé css, et créez un fichier CSS à l'intérieur, nommé o6-style.css
3. Ouvrez votre fichier o6-style.css et écrivez ceci :

```

1  body {
2
3      background:black;
4      color:#FFFFFF;
5      text-align:center;
6  }
```

4. Dans le fichier **o1-hello_world.html**, faites ce qu'il faut pour que la feuille de style et la page HTML soient associées
5. Sauvegardez vos documents
6. Ouvrez votre fichier HTML avec votre navigateur et comparez avec le fichier du TD précédent.
7. Bravo :-)

Ex 2 : Propriétés générales

1. Dans votre dossier CSS, créez une nouvelle feuille de style que vous nommerez 07-style_refcard.css.
2. Ouvrez la page HTML 02-refcard.html et faites ce qu'il faut pour que la feuille de style qu'on vient de créer y soit liée correctement. Tous les exercices de ce TD utilisent ces deux fichiers.
3. Définissez un style pour la balise body, et à l'aide de la propriété background-color, choisissez la couleur de fond silver.
4. Définissez couleur #000 pour le texte, grâce à la propriété color.
5. Les documents sont beaucoup plus lisibles à l'écran avec une police sans-serif. Grâce à la propriété font-family, faites les modifications adéquates.
6. Sauvegardez vos documents dans votre espace de travail

Ex 3 : Formatage du texte

1. Faites un style pour le titre h1. Il devra être italique, d'une dimension de 3em et centré. Utilisez les propriétés font-size, font-style et **text-align**.
2. Faites un style pour les titres h2, qui seront en dimension 2em en vert, et souligné. Utilisez notamment la propriété **text-decoration**.
3. Faites un style pour les paragraphes, qui seront en alignement justifié.
4. Modifiez le style des citations pour qu'elles apparaissent en petites majuscules, utilisez la propriété font-variant. Améliorez la lisibilité mettant 0.1em dans la propriété letter-spacing.
5. Modifiez le style des termes de définition pour qu'ils soient en gras. Utilisez la propriété font-weight.
6. Faites en sorte que la première ligne des paragraphes soit indentée de 2em, avec text-indent.
7. Sauvegardez vos documents sur votre espace de travail.

Ex 4: Reformatage du texte

1. Modifiez le style des abréviations pour que lorsqu'on les survole, le curseur (cursor) soit un point d'interrogation.
2. Modifiez le style des textes « mis en valeur », pour qu'ils soient en gras en plus d'être en italique.
3. À l'aide de la propriété list-style-type (et de l'ordonnancement des sélecteurs !), faites en sorte que les éléments de liste non ordonnée (et uniquement ceux-ci) ait un cercle plein (disque).
4. Modifiez la couleur du surlignage pour qu'il soit rose (ne vous posez pas trop de questions sur la propriété, simplement, réfléchissez bien !)
5. Sauvegardez vos documents dans votre espace de travail.

Ex 5: Formatage des liens, id, classes et pseudo-classes

1. Il y a 6 liens dans la page HTML sur laquelle nous travaillons. Faites en sorte que les 2 premiers liens soient affichés en blanc (utilisez un sélecteur de type classe).
2. Le 3e lien doit être rouge, avec un soulignement qui n'apparaît que lorsqu'on survole le lien à la souris (utilisez un sélecteur de type id).
3. L'avant dernier lien sera d'une taille fixée à 5 pixels.
4. Les 4e et 6e liens seront en jaune, et passeront orange uniquement lorsqu'on clique dessus, tout en doublant de taille de police.
6. Sauvegardez vos documents dans votre espace de travail.

Ex 6: Formatage des images

1. Redimensionnez l'image pour qu'elle ne puisse pas déborder d'un rectangle de 150x100px.
2. Utilisez les propriétés max-width et max-height.
3. Ajoutez une bordure de 3 pixels en pointillés et de couleur orange. Regardez les propriétés border et associées (border-size, etc).
4. Sauvegardez vos documents

Ex 7: Formatage des tableaux

1. Faites en sorte que le tableau soit affiché ainsi :

Nom	Prénom	Age
Bob	L'éponge	8
McQueen	Flash	21

L'age de certaines personnes

Note : Voici les caractéristiques requises :

- le texte est calé à droite
- la bordure est un trait noir, d'un pixel de large.
- la marge intérieure (**padding**) des cellules est de 0,5em
- la légende est en bas, de taille 0,7em, en italique.

Aide : Pour y arriver, vous devrez notamment vous renseigner et utiliser les propriétés suivantes :

Padding
caption-side
border-collapse

2. Validez que votre feuille CSS est valide (avec l'outil du W3C), tout comme votre page HTML.
3. Sauvegarder vos documents

Ex 8: Pseudo-éléments

- 1 A l'aide du pseudo élément first-letter, faites en sorte que la première lettre de chaque titre de niveau 2 soit en blanc et avec une taille de 150%.
- 2 Sauvegardez vos documents.

Ex 9: Changer la police

1. Allez sur <http://www.google.com/fonts>, et cherchez la police nommée Gochi Hand, puis choisissez l'option « quick use ».
2. Appliquez les instructions pour faire en sorte que votre balise h1 soit affichée avec la police Gochi Hand.
3. Validez que votre feuille CSS est valide (avec l'outil du W3C), tout comme votre page HTML.
4. Sauvegardez vos documents.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 3 : JavaScript	

Objectifs (Lisez cette page !)

Ce TD illustre la partie du cours sur la programmation web et en particulier les concepts liés à la conception d'application riches coté client, c'est-à-dire dans le navigateur. Les concepts principaux abordés seront : Les principes de base du langage **JavaScript et la découverte du DOM**.

- Pensez à sauvegarder régulièrement votre travail et à commenter votre code.
- Présenter le langage JavaScript **en 2 heures de cours est une gageure**. Vous devez, donc, pendant ce TD, faire des efforts pour aller chercher les connaissances techniques nécessaires pour l'accomplir.
- **Comprendre** ce que vous lisez. Ne vous contentez pas, seulement, de recopier quelques instructions JavaScript du réseau Internet.

Références

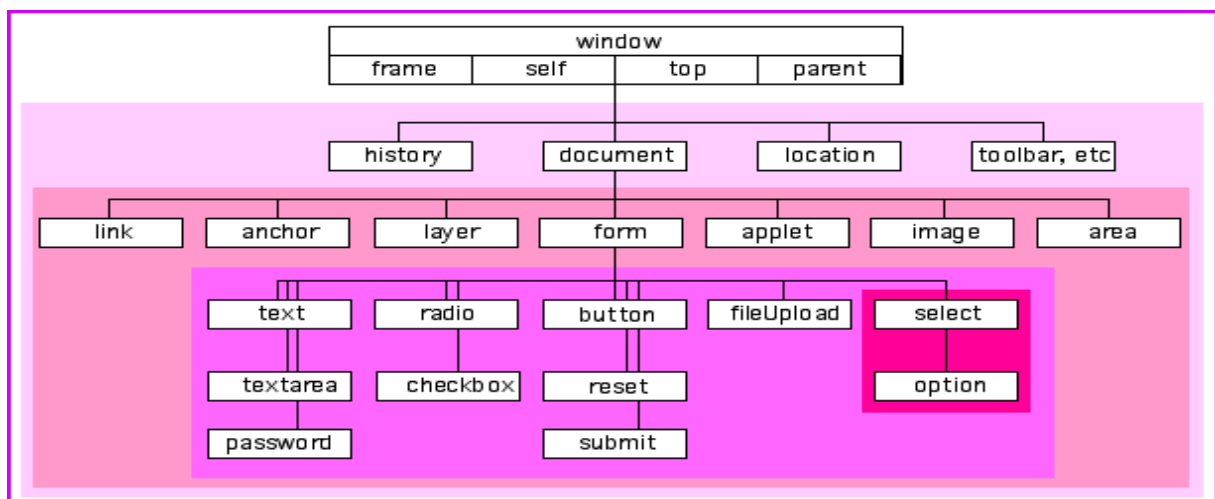
- https://www.w3schools.com/js/js_htmldom.asp

Rappel

JavaScript est un langage de programmation à part entière, permettant de réaliser des applications complexes dès que l'on a acquis une connaissance suffisante du langage et de ses diverses possibilités. Deux restrictions qu'il convient toutefois de souligner :

- JavaScript ne dispose d'aucune fonctionnalité graphique ;
- Pour des raisons de sécurité, JavaScript ne peut ni lire ni écrire un fichier.
- Les objets DOM (window, document,...) sont accessibles par JavaScript mais ne sont pas, proprement dit, des objets JavaScript

Hiérarchie des objets DOM du navigateur (browsers) :



Exercice 1 : Commençons doucement avec l'objet document

Nous allons commencer par écrire une simple page html qui devra contenir 1 balise h1, 2 balises h2 et 3 balises h3. Chacune de ces balises contiendra un court texte facilement identifiable et différent. Vous aurez donc au moins 6 textes différents. Libre à vous d'ajouter d'autres textes et d'autre balise à votre page. Le titre de votre page (défini par la balise title de la section head) devra être « exercice 1 ». Créer une feuille de style.css et donner quelques effets à vos balises.

Votre TD sera constitué de 3 fichiers : td3.html, js/app.js, css/style.css

1.1 La propriété document.title, textContent, innerHTML

- 1) Ajoutez un id= « titre » à votre h1.
- 2) Ecrivez une méthode **defTitre1()**, qui au chargement de la page, recherche dans celle-ci la balise ayant l'id « titre » et change le titre de la page avec le contenu de la balise

- Quel sera l'évènement qui déclenchera l'appelle de votre fonction ?
- Quelle méthode avez-vous utilisée pour récupérer l'objet représentant votre balise h1 ?
- Quelle propriété de l'objet représentant votre balise h1 avez-vous utilisée pour récupérer le texte de celui-ci ?

Maintenant on souhaite faire la même chose, mais en récupérant le texte de la première balise h2 du document. Attention, cette fois, il n'y a pas d'id dans la balise. Ecrivez une fonction **defTitre2()** qui fait cela.

- Quelle(s) méthode(s) avez-vous utilisée pour récupérer l'objet représentant votre balise h2 ?

Maintenant modifiez votre code pour prendre non pas le première mais la dernière balise h2. Attention, le code devra fonctionner quel que soit le nombre de balises h2. S'il n'y en a aucune, le titre de la page sera votre nom et prénom. Vous ferez cela dans la méthode **defTitre3()**.

- Comment faire pour connaître le nombre de balise h2 du document ?

Modifiez votre page en ajoutant le h1, le 2eme h2 et le premier h3 dans la classe CSS « firstOrLast ». Ecrivez une fonction **defTitre4()** qui sélectionne le premier élément de cette classe comme titre pour le document, si le nombre d'éléments de cette classe est paire. Si le nombre est impair, on utilisera le dernier. S'il n'y en a aucun, le titre de la page sera votre nom et prénom.

- Quelle méthode avez-vous utilisée pour récupérer l'objet de votre classe ?
- Quant est-il avec Internet Explorer ?
- Comment avez-vous déterminé si un nombre est pair ?

1.2 Les propriétés innerHTML, textContent

Ajoutez à votre page une balise div contenant une balise p contenant elle-même un petit texte (de 3 mots minimum) dont une partie est entourée par une balise span. Ajoutez à votre page un deuxième ensemble de balise identique au précédent mais dont seul le texte change. Ecrivez une fonction **inverseTexte()** qui inverse le contenu des deux balises p. On pourra faire l'hypothèse qu'il n'y a que nos 2 balises p dans toute la page.

1.3 La propriété document.lastModified

Une des choses importantes pour une page web est de savoir quand celle-ci a été mise à jour et par qui. Si on peut retrouver ces informations facilement aux travers des métadonnées associées à un fichier, cela n'est pas forcément facile à faire pour tout le monde. Voyons comment faire cela facilement avec une petite fonction JavaScript. Ajoutez à votre page les balises meta pour l'auteur, la description et les mots clés.

Ajoutez une balise div vide à la fin de votre page. Celle-ci aura l'id « date_modif ». Ecrivez une fonction **datemodif()**, qui automatiquement au chargement de la page, ajoute un texte du type « Dernière modification : le vendredi 9 janvier 2017 par John Doe » dans la div (en lieu et place du texte existant s'il y en a un). La date sera récupérée via une propriété de l'objet document. L'auteur lui sera identifié à l'aide de la balise meta. Pour cela vous aurez besoin d'un objet Date.

Exercice 2 : l'objet Date

Comme nous avons déjà commencé à utiliser l'objet Date, continuons un peu à étudier son fonctionnement.

2.1 Date.getTime(), Math.round()

Ajoutez une balise paragraphe dans votre page. Celle-ci contiendra le texte suivant : « il reste xxx jours avant le 19 juillet 2015 ». Ajoutez une fonction **majNbJours()** qui calcule le nombre de jour restant quand on passe le curseur au-dessus de la balise paragraphe et qui remplace les xxx par la valeur. On pensera également à supprimer le s de jour quand cela sera nécessaire.

- Comment obtenez-vous le nombre de jours ?
- Comment faites-vous la mise à jour du texte ?

2.2 setInterval et setTimeout

Ajoutez une nouvelle balise paragraphe avec l'id « horloge » à votre page. Dès le chargement de votre page, cette balise devra contenir l'heure au format (hh:mm:ss). Pour cela vous créerez deux fonctions majHorloge1() et majHorloge2(). La première devra utiliser setInterval, la deuxième setTimeout. Pour cela vous aurez besoin d'un peu de documentation sur Internet.

Exercice 3 : HTML, CSS et JavaScript

3.1 Champ Texte et Couleur d'arrière-plan , isNaN, addEventListener()

Ajoutez un champ texte de saisie (input avec type="text" dans un **formulaire**) avec un fond de couleur blanc. On rappelle que la mise en forme doit être gérée par des instructions CSS qui seront, de préférence, contenues dans un fichier différent du fichier de la page HTML. Pour cela préparer trois classes de styles CSS comme dans l'exemple ci-dessous :

```
.blanc { background-color: rgb(255,255,255);}  
.vert {background-color: rgb(150,255,150);}  
.rouge {background-color: rgb(255,150,150);}
```

Faites-en sorte que la zone de texte devienne rouge si le texte entré n'est pas un nombre et si l'utilisateur tape un nombre, alors le fond doit devenir vert. Attention, si la zone de texte est vide, elle doit être de couleur blanche.

- Quel évènement avez-vous utilisé ?
- Comment avez-vous fait changer la couleur du champ texte ?

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 4: JS+HTML+CSS	

Contexte

JavaScript arrête de s'exécuter à la première erreur rencontrée :

- Pour faciliter le débogage et éviter de passer des heures pour trouver une erreur de syntaxe (**UTILISER un IDE** en ligne).
- JS est un langage non typé.
- Votre TD sera structuré, comme d'hab, de 3 composants : xx.html, js/app.js, css/style.css. Créer un dossier TD4 et y stocker les 3 composants.

Comment chercher efficacement sur Google !

JS est un langage objet avec un style de notation objet il est écrit par des anglophones comme beaucoup d'autres langages. La notation **objet** se veut naturelle et suit généralement la structure de la langue anglaise:

subject.verb(noun) ;

you eat an apple : devient en notation objet => *you.eat(apple) ;*

Do you own (have) a car => *you.hasCar() ;*

Are you eating an apple => *you.isEating(apple) ;*

Naturellement le mot « has » et « is » sont les préfixes de beaucoup de méthodes objet.

Pour chercher une information sur un objet, reformuler la question dans votre tête en anglais !

Je veux savoir comment dupliquer un nœud (élément HTML) ?

Google Javascript DOM HTML copy

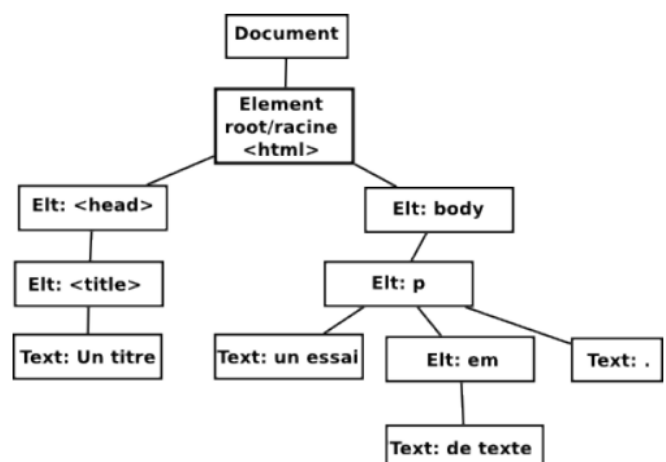
Je veux savoir si une méthode permet de questionner un objet sur ses attributs

Google Javascript DOM HTML has attributes

DOM

Un document HTML, aux yeux du DOM, est un arbre composé de nœuds, qui représentent les balises (appelées *éléments*), le *texte*, et les *attributs*. En pratique, on traitera les attributs à part. Soit donc le document (simplifié):

```
<html>
  <head>
    <title>Un titre</title>
  </head>
  <body>
    <p> Un essai <em>de texte</em>.</p>
  </body>
</html>
```



On commence avec l'objet document. Celui-ci contient un élément racine, qui représente la balise html. Par la suite, on mélange des éléments (des balises) et du texte. Chaque élément a des enfants (childNodes) qui représentent son contenu. Ces enfants sont, soit du texte, soit des éléments.

Exo 1 : premier pas (document, getElementByName, innerHTML, onclick)

Recopier la page html (ex1.html) comportant 5 check-box permettant de sélectionner des nombres entre 1 et 5, et un bouton permettant d'appeler une fonction JavaScript permettant de calculer leur somme et de l'afficher dans une fenêtre d'alerte ; si aucune case n'est cochée cette valeur sera 0. Pour accéder au code source de cette exercice, j'ai mis un weave en ligne à cette adresse : <http://liveweave.com/QZyCdI>

```
<!DOCTYPE html>
<html>
<head>
<title>exo js</title>
<script type="text/javascript">
function somme(){
  var s=0;
  var nombres=document.getElementsByName('nombre' );
  for (var i=0; i<nombres . length ; i++){
    if (nombres[i].checked ) s=s + parseInt (nombres[i].value ) ;
  }
  alert("la somme vaut" +s) ;
}
</script>
</head>
<body>
<p>
  <input type="checkbox" name="nombre" id="1" value="1" />
    <label for="1">1</ label>
  <input type="checkbox" name="nombre" id="1" value="2" />
    <label for="1">2</ label>
  <input type="checkbox" name="nombre" id="1" value="3" />
    <label for="1">3</ label>
  <input type="checkbox" name="nombre" id="4" value="4" />
    <label for="1">4</ label>
  <input type="button" value="faire la somme" onclick="somme();" />
</p>
</body>
</html>
```

1.1

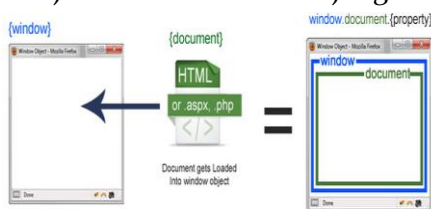
Mettez le code JS dans le module app.js. n'oubliez pas de l'inclure avec la balise <script> au document HTML. Tester la page.

1.2

Ajouter une fonction JS qui calcule la moyenne. Afficher le résultat dans une zone, de la page, prévue à cet effet. Utiliser si nécessaire la fonction globale isNaN().

Notions

L'objet **window** est un objet global représentant un onglet ouvert dans le navigateur. Il est le parent de chaque objet qui compose la page web (e.g. document=HTML). L'objet document représente votre document HTML. Vous pouvez accéder aux méthodes de l'objet document de cette manière : window.document.getElementById () ou document.getElementById()

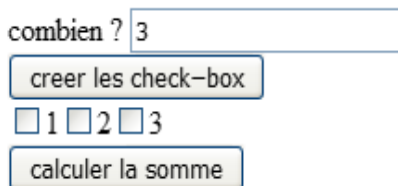


1.3

Nous allons maintenant générer les check-box. Pour cela on modifiera la page et on y ajoutera une zone de saisie permettant d'entrer un nombre entre 1 et 100 et écrira une fonction « créer » qui génère autant de check-box que le nombre entré. Cela générera aussi un bouton permettant de calculer la somme des nombres associés aux check-box cochées.

- Utiliser la fonction globale **window.parseInt()** pour convertir la valeur lue en un entier
- Si, toto est le « id » d'un élément de type « input », lire sa valeur se fait simplement :
 - **document.getElementById(toto).value**

La page **ex1.html** contiendra, dans sa balise **body**, le code suivant:



```
<body>
  <div id = "c">
    <label for = "combien">combien ?</label>
    <input type = "text" name = "combien" id = "combien" value = "" />
    <input type = "button" value = "créer les check-box" onclick = "creer();" />
  </div>
</body>
```

<http://liveweave.com/muQrpK>

Exo 2 : Arbre DOM

Un document HTML est représenté comme un arbre DOM en mémoire du navigateur.

2.1

Créer un nouveau document HTML **exo2.html**. Écrivez le code suivant :

```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Affichage des caractéristiques élément par le DOM niveau 2.0</title>
  <script type="text/javascript" src="js/app.js"></script>
</head>
<body >
  <h1>Affichage des caractéristiques des éléments par le DOM niveau 2.0.</h1>
  <div class="demo" id="ma_div">
    <h1> titre 1 </h1>
    <h2> titre 2 </h2>
    <h3> titre 3 </h3>
  </div>

  <p> Liste de courses :</p>
  <input type="text" id="ajout">
  <input type="button" value="Ajouter" onClick="ajouter()">
  <ul id="courses">
```

```

    <li>Côte de bœuf</li>
    <li>Pommes de terre</li>
</ul>

</body>
</html>

```

Dans le module app.js, Ecrivez le code JavaScript qui suit :

```

function inf() {
    var res = "",
        at, el, nd, all_attr;
    nd = document.getElementById('ma_div'); //node

    all_attr = nd.attributes;
    for (i = 0; i != all_attr.length; i++) {
        at = all_attr[i];
        res += 'ATTR: ' + at.nodeName + ' = ' + at.nodeValue + "\n";
    }

    var enfants = nd.childNodes;
    for (i = 0; i != enfants.length; i++) {
        el = enfants[i];
        res += el.nodeName;
        if (el.nodeType == 3)
            res += ' = ' + el.nodeValue + "\n";
        else
            res += " --\n";
    }
    resnode = document.createElement('div');
    body = document.getElementsByTagName('body')[0];
    body.appendChild(resnode);
    titlenode = document.createElement('h1');
    titlenode.appendChild(document.createTextNode('Ceci et le contenu de la div'));
    contentnode = document.createElement('pre');
    contentnode.appendChild(document.createTextNode(res));
    resnode.appendChild(titlenode);
    resnode.appendChild(contentnode);
}

```

2.2

Ajouter un bouton à la page HTML. En JavaScript on peut décider d'exécuter un script quand un certain événement (causé, par exemple, par l'utilisateur) se produit. Ainsi, programmez le bouton pour que lorsqu'on clique dessus, la fonction JS inf() du module app.js sera appelée.

2.3

Écrivez la fonction Ajouter() qui s'assure que le champ texte est non vide (id=ajout), et rajoute l'élément qui a été saisi à la liste de courses. Écrivez ensuite la fonction Retirer() (et créez un bouton associé) qui supprime le dernier élément de la liste – vous penserez à gérer le cas «liste vide».

Indice : getElementById("..."), createElement("li"), appendChild(), removeChild()

Exo 3 : déplacer un bloc

Créer un nouveau document HTML **exo3.html**. Ecrivez le code suivant :

Le fichier **style.css**

```
#container{
  height: 400px;
  width: 600px;
  outline: 1px solid black;
  position: relative;
}
#carre {
  position: absolute;
  height: 50px;
  width: 50px;
  outline: 1px solid black;
  background-color: red;
}
```

Le fichier **exo3.html**:

```
<div id='container'>
  <div id='bloc' />
</div>
```

Ecrire une fonction `animer(e)` qui reçoit le paramètre `e` lorsque je presse une touche de mon clavier. Lorsque la touche est l'une des 4 flèches, le bloc se déplace dans le sens de la flèche choisie (droite, gauche, haut, bas). Evidemment il faut associer l'événement **onkeydown** à la fonction `animer` lors du chargement du document.

Indice : `e.KeyCode`, `document.onkeydown`.

Exo 4 : Première approches des objets JS

Notre ami (entre autre) durant ce TD est la console NodeJS en ligne : <http://www.node-console.com/script/code>

Déclarer un objet (instance) JS littéral :

```
var animal = {
  init: function (type) {
    this.type = type;
  },
  eat: function () {
    console.log("manger :" + this.type);
  }
};
```

Comment créer un objet ayant les mêmes caractéristiques que l'objet `animal` :

```
var duck = Object.create(animal); //
duck.init("carrot");
duck.eat();
```

`animal` est le prototype de `duck`, faisons le test :
`console.log(animal.isPrototypeOf(duck)); // true`

4.1

Ajouter à l'objet **animal** la propriété sound initialisée à « the voice ». Imprimer ensuite la propriété sound de l'objet duck : `console.log(duck.sound)`. !!?

4.2

Ajouter à l'objet animal la propriété talk (comme la propriété eat) qui affichera la valeur sound de l'instance. !!??

4.3

Les animaux sont maintenant équipés de la méthode talk. Tester l'effet des instructions suivantes :

```
duck.talk(); // imprime the voice
duck.sound="coing coing";
duck.talk(); // imprime coing coing
dog =Object.create(animal);
dog.talk(); // imprime the voice
```

4.4

Changer la définition de la méthode talk de l'objet animal comme ça :

```
animal.talk = function() {
  console.log("sound :" + this.sound.toUpperCase());
}
```

Que donnera-t-elle l'instruction suivante : `duck.talk()` ?

Au lieu de créer un objet JS de manière littérale, nous allons apprendre à définir un constructeur (un objet fonction) permettant de créer des instances.

4.5

Créer un objet fonction (constructeur d'objets) :

```
var animal = function(type) {
  this.type = type;
  this.eat = function () {
    console.log("manger :" + this.type);
  }
};

terrier = new animal("os");
terrier.eat();
```

plugger la fonction talk sur l'objet animal directement

```
animal.talk= function() {
  console.log("wouf wouf");
};
```

Faites parler l'objet terrier :

```
terrier.talk(); // un pb ?
```

terrier ne peut « parler ». Pour l'aider à le faire, il faut « plugger » sa définition au prototype du constructeur de l'objet animal :

```
animal.prototype.talk = function() {  
  console.log("wouf wouf");  
};  
terrier.talk(); // alors
```

4.6

Recopier l'objet fonction suivant dans un votre module JS(constructeur d'objets) :

```
var compte = function () {  
  var transactions=[{  
    idt: 123,  
    amount: 1000  
  },  
  
  {  
    idt: 124,  
    amount: -500  
  }  
];  
  
  var amount = 0;  
  this.ga = function getAmount() {  
    transactions.forEach(function (record) {  
      amount = amount + record.amount;  
    });  
    return amount;  
  };  
};
```

Proposer une « petite application » composée d'une page HTML et le module JS exploitant cet objet. Enrichir cet objet avec de nouvelles propriétés et méthodes. <http://liveweave.com>

Exo 5 : Callback

JavaScript propose la mise en place de callback, car le langage ne possède pas de **support pour des exécutions multithreadées**, et que le mécanisme de callback est l'unique solution pour ne pas bloquer une exécution.

Une fonction qui prend une autre fonction comme paramètre et/ou retourne une fonction est appelée fonction de premier ordre. La fonction passée en paramètre est appelée callback.

5.1

Recopier le code suivant dans l'espace JS de <http://jsbin.com/?js,console,output>

Exécuter le code avec (Run with JS) et observer la console

```
var panier = [];  
function display(info) {  
  if (typeof info === "string") {  
    console.log(info);  
  } else if (typeof info === "object") {  
    for (var item in info) {  
      console.log(item + ": " + info[item]);  
    }  
  }  
}
```

```
function getInput(options, callback) {
    panier.push(options);
    callback(options);
}
```

```
getInput({
    fruit: "Orange",
    prix: 10
}, display);
```

La fonction display passée en paramètre est la fonction de rappel (callback).

5.2

Ecrire une nouvelle fonction de rappel (callback) qui sera appelée par getInput. La fonction de rappel sera appelée changerPrix (info), qui multiplie le prix par 10 et afficher un message spécifique.

Exo 5 : Jeu

Pour cet exercice vous devez partir d'une page HTML5 ci-dessous et du fichier CSS donné. Aucun de ces deux éléments ne pourra être modifié. La page HTML :

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>Jeu</title>
<link rel="stylesheet" type="text/css" href="css/style.css" />
<script type="text/javascript" src="js/app.js"> </script>
</head>
<body onload="init()">
<div id="jeu">
<div style="top:0px;left:0px;" class="case"><h1>6</h1></div>
<div style="top:0px;left:102px;" class="case"><h1>1</h1></div>
<div style="top:0px;left:204px;" class="case"><h1>7</h1></div>
<div style="top:102px;left:0px;" class="case"><h1>3</h1></div>
<div style="top:102px;left:102px;" class="case"><h1>4</h1></div>
<div style="top:102px;left:204px;" class="case"><h1>8</h1></div>
<div style="top:204px;left:0px;" class="case"><h1>5</h1></div>
<div style="top:204px;left:102px;" class="case"><h1>2</h1></div>
<div style="top:204px;left:204px;" class="case vide"><h1> </h1></div>
</div>
</body>
</html>
```

Le fichier CSS :

```
html { font-size: 62.5%; }
body { margin: 0px; padding: 0px; }
#jeu {
border-style:solid;
border-width:1px;
background-color:#00008B;
margin:0px;
```

```

width: 306px;
height: 306px;
position: absolute;
}
.case {
border-style: solid;
border-width: 1px;
background-color: #1E90FF;
margin: 0px;
padding: 0px;
width: 100px;
height: 100px;
position: absolute;
z-index: 1;
transition-property : top, left;
transition-duration : 1s;
}
h1{ text-align: center; font-size: 4rem; }
.vide { background-color: #00008B; z-index: 0; border: none}

```

6.1

Ecrivez la fonction *init()* qui ajoute (de manière propre) un écouteur sur l'évènement « click » pour chaque élément HTML de type DIV étant dans la classe CSS « case ».

6.2

Ecrivez une fonction *selection()* qui échangera la position de la case vide avec la case que laquelle on vient de cliquer si et seulement si, ces deux cases ont un coté commun.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 5: APACHE-PHP-MYSQL	

Contexte

- Pour ce TD je pars d'une installation de Debian 8 avec les composants PHP, MYSQL, Apache déjà installés.
- Afin de mettre en musique les différents éléments qui ont été présentés dans les 4 thèmes du cours (**HTML**, **CSS**, **JAVASCRIPT**, **PHP**), nous développerons une petite application web en insistant sur la séparation entre couches IHM, Logique métier et Accès aux Données.
- Dans le répertoire /var/www de la machine VM, créer l'arborescence suivante :
 /var/www/site, /var/www/site/modele, /var/www/site/vue, /var/www/site/vue/css, /var/www/site/vue/js, /var/www/site/controleur
- L'image **turnkey LAMP Stack**
 - Télécharger la version VMDK au format OVF
 - Dézipper l'image ensuite l'ouvrir, sans la démarrer, avec VMWare Workstation
 - Allez dans l'onglet settings et cocher l'option NAT

Network connection

☐ Bridged: Connected directly to the physical network

☐ Replicate physical network connection state

☒ NAT: Used to share the host's IP address

☐ Host-only: A private network shared with the host

☐ Custom: Specific virtual network

- Démarrer la machine. Une fois logué :
 - apt-get update
 - apt-get install console-setup // choisir clavier français
- Vous pouvez accéder à la machine VM à travers le navigateur web de la machine réelle.

| <https://192.168.127.168>

vos favoris en les ajoutant à la barre de favoris. [Importer mes favoris maintenant...](#)

TurnKey LAMP



Exo 1 : Premier pas avec PHP

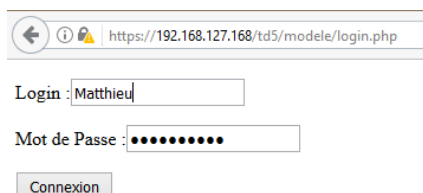
1. Recopier le code php suivant dans un fichier (login.php) et le placer dans le répertoire **modele** du site :

```
<?php
session_start();
$_SESSION['login'] = "";
$_SESSION['password'] = "";
if (isset($_POST['submit']))
{
    // bouton submit pressé, je traite le formulaire
    $login = (isset($_POST['login'])) ? $_POST['login'] : "";
    $pass = (isset($_POST['pass'])) ? $_POST['pass'] : "";
    if (($login == "Matthieu") && ($pass == "Tux"))
    {
        $_SESSION['login'] = "Matthieu";
        $_SESSION['password'] = "Tux";
        echo '<a href="accueil.php" title="Accueil de la section membre">Accueil</a>';
    }
    else
    {
        // une erreur de saisie ...?
        echo '<p style="color:#FF0000; font-weight:bold;">Erreur de connexion.</p>';
    }
}; // fin if (isset($_POST['submit']))
if (!isset($_POST['submit']))
{
    // Bouton submit non pressé j'affiche le formulaire
    echo '
<form id="conn" method="post" action="">
    <p><label for="login">Login :</label><input type="text" id="login" name="login" /></p>
    <p><label for="pass">Mot de Passe :</label><input type="password" id="pass" name="pass" /></p>
    <p><input type="submit" id="submit" name="submit" value="Connexion" /></p>
</form>';
}; // fin if (!isset($_POST['submit']))
?>
```

Recopier aussi le code de la page d'accueil (accueil.php). Au lieu de vous donner le code sur les pages de ce TD, allez sur le site célène pour récupérer le code **site1-base.zip**. ce fichier contient :

```
root@lamp /var/www# tree site1-v1/
site1-v1/
├── controleur
├── modele
│   ├── login.php
├── vue
│   ├── accueil.php
│   ├── css
│   │   └── intra.css
│   └── js
```

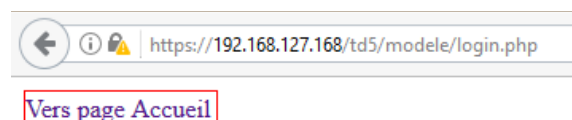
2. Depuis la machine réelle, connectez-vous à la page login.php :



https://192.168.127.168/td5/modele/login.php

Login :

Mot de Passe :



3. Si l'internaute Matthieu est bien authentifié, il sera dirigé vers un lien à la page d'accueil des membres. Si vous êtes un petit malin, vous pouvez venir à cette page, accueil.php, sans passer par la procédure d'authentification. Comment ? Modifier la page accueil.php, pour prévenir les curieux de venir à cet endroit sans authentification et les rediriger vers la page de login.

Solution : Le code à modifier dans la page **accueil.php** :

```
<?php
session_start();
if ((isset($_SESSION['login'])) || (empty($_SESSION['login'])))
{
    // la variable 'login' de session est non déclaré ou vide
    echo '<p>Petit curieux... <a href="../modele/login.php" title="Connexion">Connexion d\'abord !</a></p>';
    exit();
}
print '<div style="background-color:green;color:#FFD700;"> Bonjour, Monsieur ' . $_SESSION['login'] . '</div>';
print '<hr/> Bienvenue sur notre site p\'tit chef réservé aux membres!';
?>
```

Dans ce code :

- a) J'initialise la session (session récupérée si déjà existante)
 - b) Je teste si une des variables de session (dans mon exemple, login) est définie et remplie ... Si non définie ou non remplie, dehors :-) (Notez la présence de **exit()**; qui coupe net l'exécution de PHP, ainsi la suite du code source de **accueil.php** ne sera pas exécutée.)
 - c) Je mets ce test tout en haut de page, pour que ça soit le premier élément vérifié par le serveur.
 - d) Si je mets ces quelques lignes dans toutes les pages que je veux protéger, alors ma "section membre" est presque finie
4. Ajouter à la page **accueil.php** les instructions nécessaires permettant d'obtenir à l'exécution :



Solution : Ma page **accueil.php** modifiée est :

```
<body>
<?php
session_start();
if ((isset($_SESSION['login'])) || (empty($_SESSION['login'])))
{
    // la variable 'login' de session est non déclaré ou vide
    echo ' <p>Petit curieux... <a href="../modele/login.php" title="Connexion">Connexion d\'abord !</a></p>';
    exit();
}

print '<div> Bonjour, Monsieur ' . $_SESSION['login'] . '</div>';
print '<form id="logout" method="post" action="../modele/logout.php">
    <p><input type="submit" id="logout" name="logout" value="Déconnexion" /></p>
</form>';
print '<hr/> Bienvenue sur notre site p\'tit chef réservé aux membres!';
?>
<h1>Plat du Jour: Poulet rapide</h1>
```

5. Ecrire la page **logout.php** qui sert à détruire la session de l'utilisateur et le renverra à la page d'authentification. Mettre cette page dans le dossier « modele »

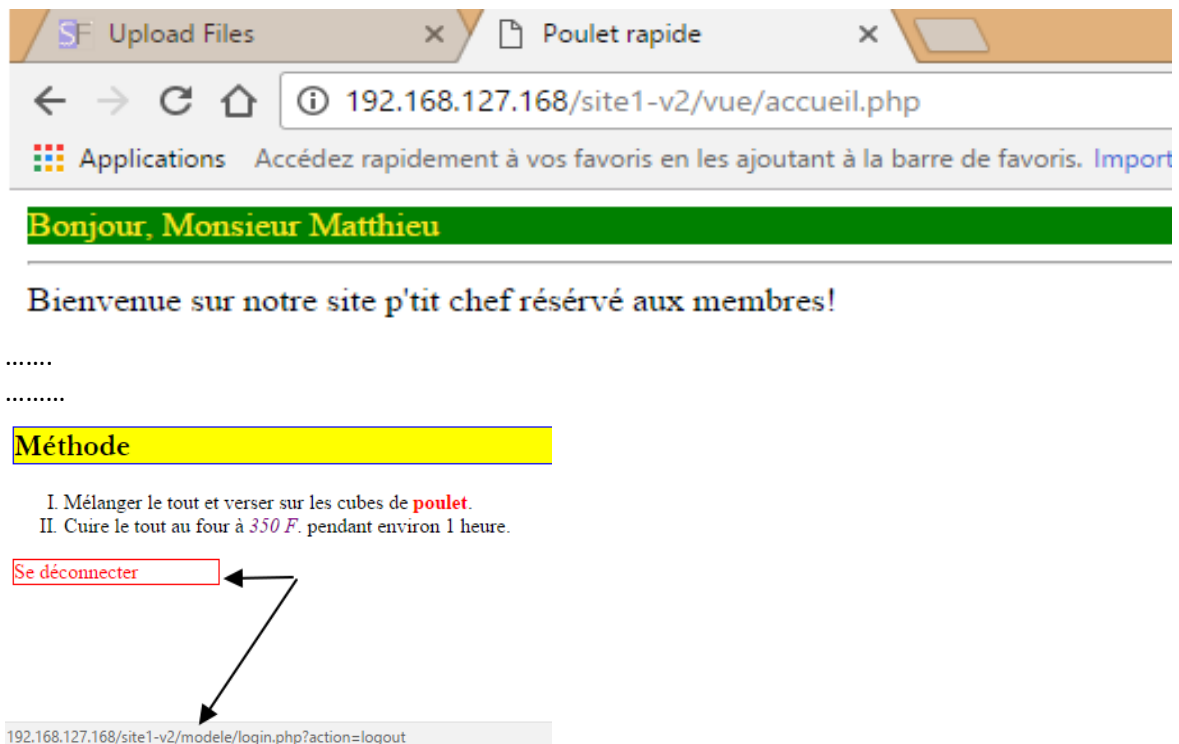
Solution :

```
<?php
session_start();
$_SESSION = array();
session_destroy();
header('location: login.php');
?>
```

6. Ecrire une nouvelle page HTML, **deconnexion.html**, contenant la balise suivante :
<p>Se déconnecter</p>

Modifier la page **login.php** pour détruire la session quand je clique sur le lien « Se déconnecter ».

Vous pouvez aussi, si vous voulez, ajouter la balise, <p>Se déconnecter</p>, à la page accueil.php. Je vous montre sur l'image suivante, ce qui affiche la barre d'état de Chrome, lorsque le curseur de la souris passe au-dessus du lien hypertexte :



Exo 2 : Session http

Le protocole http est un protocole **sans état**. Il traite chaque requête comme une transaction indépendante et sans relation avec une quelconque précédente requête. Car il n'y a pas de connexion ou de session entre le client et le serveur. Pour pallier cette absence d'état, nous avons introduit la notion de **session**, une zone mémoire créée et allouée sur le serveur et qui identifie de manière unique un utilisateur. En plus, un développeur Web ne doit jamais « **faire confiance** » aux internautes. Ils peuvent « atterrir » sur n'importe quelle page web de l'application sans respecter un scénario préétabli. **BIENVENU DANS LE PARADIS DU PROCTOLE HTTP**. Avec ce protocole vous ne pouvez pas imposer aux internautes de venir d'abord à la page une, ensuite la page deux, ainsi de suite ! Il faut donc « armer » chaque page web pour la protéger.

Créer une nouvelle structure dans le répertoire du serveur web (/var/www) :

```
site1-v2
├── controleur
├── modele
│   ├── login.php
│   └── logout.php
├── modle
├── vue
│   ├── accueil.php
│   ├── css
│   │   └── intra.css
│   └── informations.php
```

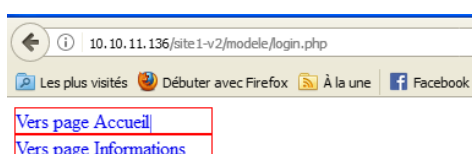
Recopier les fichiers de la version **v1**.

1. Développer une nouvelle page PHP **informations.php** et mettez-là avec la page accueil.php. Dans le code de la page **informations.php**, **ajouter** les lignes suivantes :

```
<html>
<body>
<?php
    session_start();
    if((!isset($_SESSION['login']))||(empty($_SESSION['login']))) {
        // la variable 'login' de session est non déclaré ou vide
        echo ' <p>Petit curieux... <a href="login.php" title="Connexion">Connexion d\'abord
!</a></p>';
    } else {
        print '<div> Bonjour, ' . $_SESSION['login'] . '</div>';
        print '<div style="color:yellow;background-color: magenta; border:1px red solid;">';
        print 'Les entêtes de votre requête HTTP.</div>';
        foreach (getallheaders() as $name => $value) {
            echo "<p>$name: $value </p>";
        }
        print '<hr/> session name: ' . session_name() . '<br>';
        print '<hr/> session ID: ' . $_COOKIE['PHPSESSID'] . ' <br>';
        print '<hr/>'. ' cliquer sur le bouton pour quitter la session';

    ?>
    <p><a href=" ../modele/logout.php?action=logout" title="Déconnexion">Se
déconnecter</a></p>;
    <?php
    }
    ?>
</body>
</html><?php
}
?>
</body>
</html>
```

2. Modifier la page **login.php** pour obtenir, une fois l'internaute authentifié, le résultat suivant :



Et en cliquant sur le lien « vers page informations », j'aurai :

The screenshot shows a web browser window with the address bar displaying `192.168.127.168/site1-v2/vue/informations.php`. Below the address bar, there's a green bar with the text "Bonjour, Matthieu" and a magenta bar with the text "Les entêtes de votre requête HTTP". The main content area displays various HTTP headers in yellow bars: "Host: 192.168.127.168", "Connection: keep-alive", "Cache-Control: max-age=0", "Upgrade-Insecure-Requests: 1", "User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.133 Safari/537.36", "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8", "Accept-Encoding: gzip, deflate, sdch", "Accept-Language: fr-FR;q=0.8,en-US;q=0.6,en;q=0.4", and "Cookie: PHPSESSID=a29p0bmnbnd7t19ta7d39e8o2". Below the headers, there's a section for session information: "session name: PHPSESSID", "session ID: a29p0bmnbnd7t19ta7d39e8o2", and a link "cliquer sur le bouton pour quitter la session". At the bottom, there's a button labeled "Se déconnecter".

Vous remarquerez que l'identifiant de la session est une donnée numérique cryptographique créée par le serveur. Elle est repérée par la variable **PHPSESSID** et stockée dans le tableau associatif `$_COOKIE`. Ce cookie de session sera renvoyé systématiquement par le navigateur et récupéré par le serveur afin d'identifier l'utilisateur. Vous pouvez à tout moment détruire la session et rendre, ainsi, le cookie (l'identifiant) invalide.

Ce cookie sera systématiquement renvoyé par le navigateur au serveur et par le serveur au navigateur comme une balle de **ping pong** :

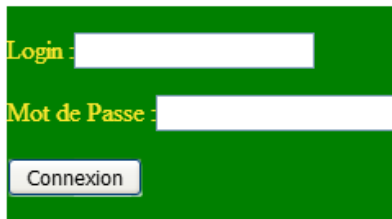


L'inspecteur de Chrome me donne l'aperçu suivant :

The screenshot shows the Chrome DevTools Elements panel. The top bar includes tabs for "Elements", "Console", "Sources", "Network", "Timeline", and "Profiles". The "Elements" tab is selected, showing the DOM tree. The root element is `<html>`, which contains `<head>` and `<body>`. The `<body>` element contains several nested `<div>` and `<p>` elements. The `<div>` elements contain the text "Bonjour, Matthieu" and "Les entêtes de votre requête HTTP". The `<p>` elements contain the HTTP headers: "Host: 10.10.11.136", "Connection: keep-alive", "Upgrade-Insecure-Requests: 1", "Referer: http://10.10.11.136/site2/modele/login.php", "Accept-Encoding: gzip, deflate, sdch", "Accept-Language: fr-FR;q=0.8,en-US;q=0.6,en;q=0.4", and "Cookie: PHPSESSID=b7toa1m84emmnuijgh8snt145". Below the headers, there's a section for session information: "session name: PHPSESSID", "session ID: b7toa1m84emmnuijgh8snt145", and a link "cliquer sur le bouton pour quitter la session". At the bottom, there's a `<form>` element with the id "logout" and method "post", which contains a "Se déconnecter" button.

- 3 Modifier le code de la page **login.php** afin de court-circuiter la procédure d'authentification si l'utilisateur s'est déjà correctement authentifié. Acheminer, ensuite, l'utilisateur vers la page d'accueil.
- 4 Si vous regarder la page accueil.php (récupérée de cèle) vous noterez que j'ai ajouté des balises html pour associer la feuille de style intra.css au flux produit par le script PHP. Faites de même avec les autres pages PHP.

Par exemple, ma page d'authentification est maintenant :



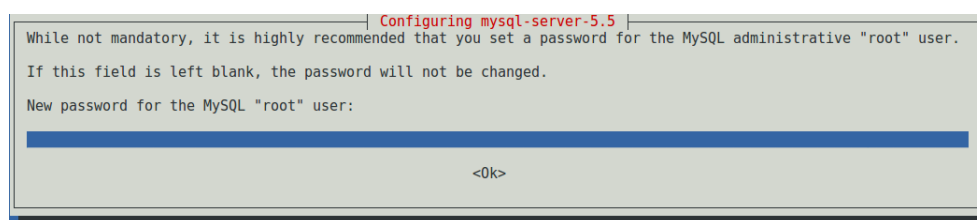
Note : si vous ajoutez l'instruction suivante

setcookie("PHPSESSID","",time()-3600,"/"); dans la page login.php, après l'instruction **session_start()** , par exemple, vous remarquerez que même après une authentification réussie de Matthieu, les pages accueil.php et informations.php afficheront le message « petit curieux....connexion d'abord ». En effet l'instruction **setcookie("PHPSESSID","",time()-3600,"/");** a désactivé le cookie de session (PHPSESSID), et a rendu impossible aux pages php de retrouver la bonne session de l'internaute (sachant que cette session existe belle et bien !).

NOTE CONCERNANT LE SERVEUR MYSQL ET PHPMYADMIN

a) Avant d'installer phpmyadmin, commencer par « resetter » le mot de passe du serveur mysql avec la commande dpkg-reconfigure :

```
File Edit View Search Terminal Help
root@lamp /var/www# dpkg-reconfigure mysql-server-5.5
```



Choisir un nouveau mot de passe (« orange ») pour root et confirmer.

b) procéder à l'installation de phpmyadmin

```
root@lamp /var/www# apt-get install phpmyadmin
```

Pendant l'installation, l'installateur vous réclame le mot de passe de **root**, ensuite un nouvel utilisateur sera créé (**phpmyadmin**) et il vous demande de lui choisir un mot de passe. L'installateur créera aussi une base de données appelée **phpmyadmin** d'administration de **mysql**.

c) les différents composant web de **phpmyadmin** seront installés dans le répertoire **/usr/share/phpmyadmin**. Pour rendre l'appli disponible et accessible sur le web, créer un lien symbolique, dans **www**, à ce répertoire :

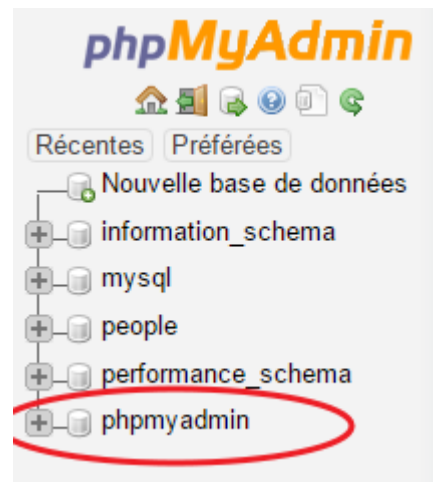
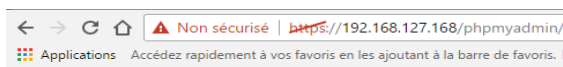
```
root@lamp /var/www# ln -s /usr/share/phpmyadmin/
```

Vérifier :

```
root@lamp /var/www# ls -l ph*
-rw-r--r-- 1 root root 20 Apr 8 2016 phpinfo.php
lrwxrwxrwx 1 root root 22 Apr 15 11:13 phpmyadmin -> /usr/share/phpmyadmin/

php:
total 0
root@lamp /var/www#
```

Vous pouvez maintenant accéder à mysql à partir de votre navigateur préféré de la machine « réelle » :



d) pour redémarrer le service **mysqld** (démon) et le service **httpd**, procéder ainsi :

```
root@lamp /var/www# service mysql restart
root@lamp /var/www# service apache2 restart
```

Il faut toujours redémarrer le service **httpd** après le service **mysqld**. Par contre, il faut toujours arrêter le service **httpd** avant le service **mysqld** ;).

e) pour connaître l'état d'un service :

```
root@lamp /var/www# systemctl status mysql.service
● mysql.service - LSB: Start and stop the mysql database server daemon
   Loaded: loaded (/etc/init.d/mysql)
   Active: active (running) since Sat 2017-04-15 14:57:20 UTC; 6min ago
     Process: 20745 ExecStop=/etc/init.d/mysql stop (code=exited, status=0/SUCCESS)
     Process: 20782 ExecStart=/etc/init.d/mysql start (code=exited, status=0/SUCCESS)
    CGroup: /system.slice/mysql.service
            └─20809 /bin/sh /usr/bin/mysqld_safe
              └─21333 /usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql -...
```

Exo 3 : premiers pas avec php et mysql

Dans cet exercice les produits, d'une appli web de vente en ligne, seront stockés dans une base de données relationnelle.

Créer dans le répertoire du serveur web, www, un dossier « **vitrine** » et la structure suivante :

```
root@lamp /var/www# tree vitrine
vitrine
├── controleur
├── modele
└── vue

3 directories, 0 files
root@lamp /var/www#
```

A l'aide de l'outil **phpMyAdmin**, créer une base « **Produits** » contenant une table « **Produit** » avec les données indiquées ci-dessous.

Produit (code , nom , description, prix)			
code (Int PRIMARY KEY Auto-Incrément)	nom (Varchar 25)	description (Varchar 50 NULL)	prix (Float)
1	Autocollant	Autocollant au symbole du club	15,00
2	T-shirt Official	T-shirt officiel du club	35,00
3	T-shirt baby-look	T-shirt au symbole du club, modèle féminin	25,00

Toujours à l'aide de l'outil phpMyAdmin, modifier les privilèges de la base de données «Produits » afin de lui ajouter un nouvel utilisateur nommé « **vitrine** » avec, pour mot de passe « orange ».

Ajouter un utilisateur



Donner à l'utilisateur « vitrine » les privilèges nécessaires à la base « produits »

Survol des utilisateurs

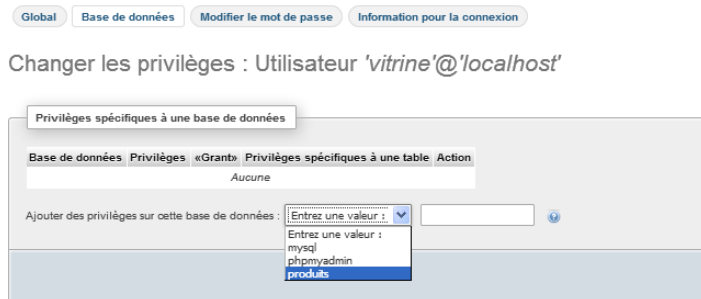
Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grants»	Action
<input type="checkbox"/> debian-sys-maint	localhost	Oui	ALL PRIVILEGES		Oui	Changer les privilèges Exporter
<input type="checkbox"/> php	localhost	Oui	USAGE		Non	Changer les privilèges Exporter
<input type="checkbox"/> root	127.0.0.1	Oui	ALL PRIVILEGES		Oui	Changer les privilèges Exporter
<input type="checkbox"/> root	:::1	Oui	ALL PRIVILEGES		Oui	Changer les privilèges Exporter
<input type="checkbox"/> root	localhost	Oui	ALL PRIVILEGES		Oui	Changer les privilèges Exporter
<input type="checkbox"/> vitrine	localhost	Oui	USAGE		Non	Changer les privilèges Exporter

☐ Tout cacher Pour la sélection : [Exporter](#)

Ensuite :

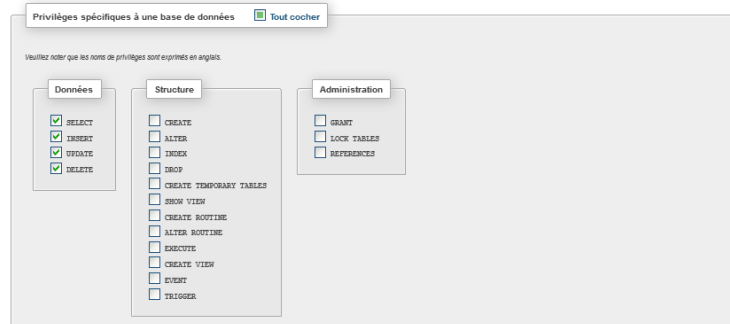


Choisir la base « produits » :



Accorder les privilèges et exécuter:

Changer les privilèges : Utilisateur 'vitrine'@'localhost' - Base de données produits



Pour la suite, je vous conseille d'utiliser un client FTP sur la machine réelle (fileZilla par exemple) pour mettre à jour votre site sur la machine « virtuelle ».

3.1 Créer une page PHP, *connect.php*, stocké dans le répertoire *modele*, capable de se connecter à la base de données « Produits », utilisant l'utilisateur et le mot de passe que vous venez de créer, et de lister l'ensemble de produits présents dans la base de données. Vous pouvez utiliser pour cela la requête SQL « SELECT * FROM Produit » :

La première étape consiste donc à se **connecter à la base de données** utilisant l'utilisateur que nous venons de créer (user « **vitrine** », mot de passe « **orange** »).

```
$server="localhost";
$db="produits";
$user="vitrine";
$password="orange";

// Create connection
$mysqli = new mysqli($server, $user, $password, $db);
```

Il est important de tester si la connexion s'est correctement effectuée.

```
if ($mysqli->connect_errno) {  
    die ("Echec lors de la connexion "  
        . $mysqli->connect_error);  
}
```

Maintenant que nous avons la connexion avec la base de données établie, nous allons pouvoir écrire une nouvelle page php, **afficherProduits.php**, stocké dans le répertoire **modele**, pour lui soumettre une requête afin de récupérer les informations sur les produits. La requête SQL « **select * from Produits** » nous permet ceci. Pour la soumettre à la base de données, nous allons utiliser l'opération « **query** » de l'objet « **mysqli** » (« **\$mysqli->query** »). **N'oubliez pas d'inclure la page connect.php dans la page afficherProduits.php ;).**

```
$sql = "SELECT * FROM Produit" ;  
  
$resultat = $mysqli->query($sql);
```

Une fois en possession des résultats de la requête, nous allons pouvoir les afficher dans un tableau (un « **echo** " <table> " ; » nous aidera pour cela. Ces résultats incluent non seulement les données (nos produits), mais aussi les noms des colonnes (attributs) de la table « **Produit** ». Nous allons donc pouvoir récupérer les noms des colonnes pour les afficher dans notre tableau. Ceci est possible à travers l'opération « **fetch_fields** » de l'objet **\$resultat** (« **\$resultat -> fetch_fields()** ; »). Cette opération nous retournera un tableau associatif que nous allons parcourir à l'aide d'une boucle « **foreach** ».

```
$titres = $resultat -> fetch_fields() ;  
foreach ($titres as $colonne) {  
    echo "<th> "  
        . $colonne->name . "</th>";  
}
```

Après avoir affiché les noms de colonnes, nous pouvons nous concentrer sur les données proprement parlées. Nous allons récupérer ligne à ligne (c'est-à-dire, produit à produit), à l'aide de l'opération « **fetch_object** » de l'objet **\$resultat** dans une boucle (« **while (\$ligne = \$resultat->fetch_object())** »). Cette opération nous retournera un objet dont les attributs correspondent aux colonnes de la table **Produit**. Nous pouvons ainsi utiliser une boucle « **foreach** » pour récupérer la valeur de chacun de ces attributs (« **foreach (\$ligne as \$colonne=>\$val)** »). Une fois les données récupérées et affichées, nous pouvons fermer la connexion que nous avons ouvert tout au début (« **\$mysqli->close();** »).

```
while ($ligne = $resultat->fetch_object()) {  
    echo "<tr>";  
    //on récupère chaque attribut de la ligne  
    foreach ($ligne as $colonne=>$val) {  
        echo "<td> " . $val . "</td>";  
    }  
    echo "</tr>";  
}
```

Tester votre page PHP!

3.2 Si le test est concluant, commencer par construire un formulaire HTML *insérerProduit.html*, dans le répertoire *vue*, et une page PHP, *insérerProduit.php*, stocké dans le répertoire *modele*, permettant à un utilisateur d'insérer un nouveau produit dans la base de données.

Dans cet exercice, nous aurons besoin de construire une page HTML contenant un formulaire qui nous permettra de renseigner les données du nouveau produit. Nous devons avoir ainsi un champ pour chaque attribut de la table « **Produit** ». Ces informations seront récupérées par la page PHP, toujours à l'aide du tableau associatif « **\$_POST** ». Puisque nous allons utiliser celles-ci pour créer un nouveau produit, nous devons vérifier si toutes les informations nécessaires (et notamment le nom et le prix) ont bien été remplies. Ceci se fera à l'aide d'un test « **if** » et de la fonction « **empty** ». Nous pouvons aussi tester si la valeur renseignée pour le prix est bel et bien un numéro (et pas un texte) à l'aide de la fonction « **is_numeric** », ce qui nous donne le test suivant « **if (empty(\$nom) OR empty(\$prix) OR !is_numeric(\$prix))** ».

Exo 4 : Ajout d'un mécanisme d'authentification à l'appli vitrine

Jusqu'à maintenant notre application permet à tout internaute de consulter et ajouter des produits dans la base de données.

4.1 créer une table *users* dans la base « produits » :

+ Options							
<div><div></div><div></div><div></div></div>				code	nom	password	email
<input type="checkbox"/>		Modifier	Copier Effacer	1	matthieu	81dc	matthieu@tata.com
<input type="checkbox"/>		Modifier	Copier Effacer	3	Ahmad	81dc	titi@titi.com
<input type="checkbox"/>		Modifier	Copier Effacer	5	mich	202c	mich@free.com

Le champ « code » est de type *entier auto-increment* (A.I)

Je vous fournis en ligne la nouvelle mouture de l'appli web vitrine.

4.2 Téléchargez-la l'appli web vitrine2. Et installer là dans le dossier *www* du serveur apache.

4.3 Revoir le code de l'appli fournie et essayer de comprendre son fonctionnement

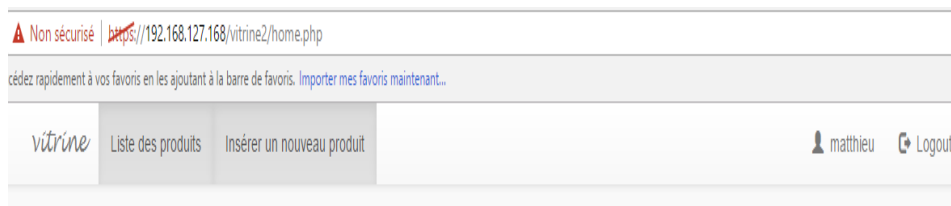
La page index.php

<https://192.168.127.168/vitrine2/index.php>

05 favoris en les ajoutant à la barre de favoris. [Importer mes favoris maintenant...](#)

Log In.

Après authentification nous atterrissons sur la page d'accueil `home.php`:



Liste des produits :

code	nom	description	prix
1	autocollant	autocollant au symbole du club	15
2	T-shirt officiel	T-shirt officiel du club	35
3	t-shirt baby-look	t-shirt au symbole du club, modèle féminin	25
7	chaussette	chaussette au club	10
8	TV	tv samsung	150
9	basket	basket ball, club	0
10	smartphone	smartphone samsung galaxy s8	800

Insérer un produit avec la page `insererproduit.php` (exemple qui passe mal):

Ajout produit.

assiette

assiette à 2 sous

jhhf

Ajouter produit Afficher produits

Ajout produit.

❗ prix saisie non numérique

nom produit

description

prix produit

Ajouter produit Afficher produits

M'inscrire :

M'inscrire

matthieu

matthieu@tata.com

1234

Créer votre compte Log In

Le mot de passe de l'internaute est stocké à l'aide d'une fonction cryptographique (md5).

4.4 l'appli web mise en ligne lui manque la page `insererproduit.php`. En s'inspirant fortement des pages comme `home.php` et `index.php`, développez la page manquante en suivant le

même principe ergonomique. Vous avez bien remarqué que j'ai utilisé le Framework **twitter bootstrap** auquel j'ai emprunté quelques classes prédéfinies et éléments de décoration css. L'avantage de bootstrap est double : en matière ergonomique, il offre des très belles classes CSS et de vous éviter de coder votre propre code JS pour valider le formulaire avant sa soumission au serveur.

Ajout produit.

Ajout produit.

Produit Ajouté

- 4.5 télécharger la solution qui est en ligne maintenant (vitrine4.5). En plus, j'ai amélioré la page « liste de produits » ajoutant deux fonctionnalités « edit » et « delete » des produits. La fonctionnalité « edit » est implémentée. Vous devez implémenter l'autre fonctionnalité selon le même principe.

Non sécurisé | <https://192.168.127.168/vitrine2/afficherproduits.php>

ccédez rapidement à vos favoris en les ajoutant à la barre de favoris. [Importer mes favoris maintenant...](#)

vitrine	Liste des produits	Insérer un nouveau produit
---------	--------------------	----------------------------

code	nom	description	prix	Modifier
1	autocollant	autocollant au symbole du club	1500	Edit Delete
2	T-shirt officiel	T-shirt officiel du club	35	Edit Delete
3	t-shirt baby-look	t-shirt au symbole du club, modèle féminin	25	Edit Delete
7	chaussette	chaussette au club	10	Edit Delete
8	TV	tv samsung	150	Edit Delete
9	basket	basket ball, club	0	Edit Delete
10	smartphone	smartphone samsung galaxy s8	800	Edit Delete
11	cassette	k7 vidéo	13	Edit Delete

Quand vous passerez la souris au-dessus du lien « Edit », la barre d'état affichera l'adresse suivante :

<https://192.168.127.168/vitrine2/editproduit.php?code=1>

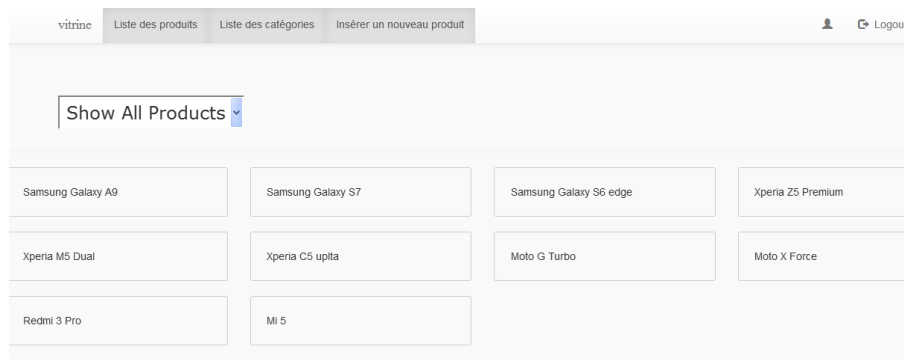
Exo 5 : AJAX

Si vous avez réalisé le TD6 (!), vous vous souvenez que la base données contiendra maintenant les trois tables suivantes :

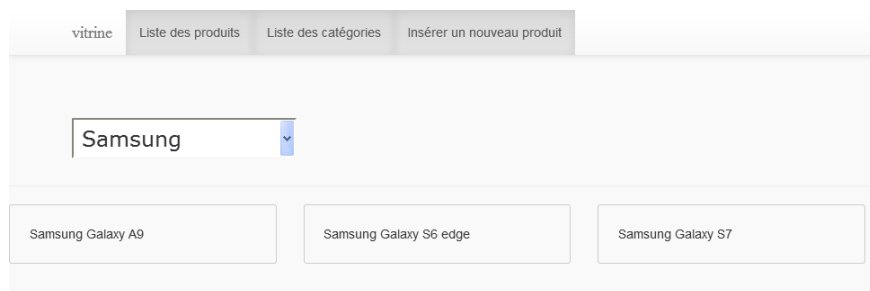
<input type="checkbox"/> Table	Moteur	Interclassement	Longueur
<input type="checkbox"/> categorie	InnoDB	latin1_swedish_ci	
<input type="checkbox"/> produit	InnoDB	latin1_swedish_ci	
<input type="checkbox"/> users	InnoDB	utf8_general_ci	
3 au total	InnoDB	utf8_general_ci	

Télécharger la version 5 de l'application vitrine.

L'application web dispose d'un button « liste des catégories » :



Et d'une liste déroulante pour choisir la catégorie des produits à afficher :



La page `affichercategories.php` utilise le mécanisme AJAXien pour remplir les champs « display » :

```
<div class="" id="display">
    <!-- Records will be displayed here -->
</div>
```

Elle appelle la page `getproducts.php` de deux manières différentes

Renvoyer tous les produits :

```
<script type="text/javascript">
    $(document).ready(function() {
        // function to get all records from table

        function getAll() {

            $.ajax({
                url: 'getproducts.php',
                data: 'action=showAll',
                cache: false,
                success: function(r) {

                    $("#display").html(r);

                }
            });
        }
    });
}
```

Renvoyer les produits de la catégorie spécifiée :

```
getAll();  
// code to get all records from table via select box  
$("#getProducts").change(function() {  
    var id = $(this).find(":selected").val();  
  
    var dataString = 'action=' + id;  
  
    $.ajax({  
        url: 'getproducts.php',  
        data: dataString,  
        cache: false,  
        success: function(r) {  
            $("#display").html(r);  
        }  
    });  
});
```

Développer la page getproducts.php

Exo 6 : Expiration de la Session http

1. Ouvrir le fichier php.ini se trouvant dans le répertoire /etc/php5/apache2. Repérer les propriétés : **session.cookie_lifetime** et **session.gc_maxlifetime**.

Vous remarquerez que la valeur de **session.cookie_lifetime** mise à 0. Cette valeur, par défaut, indique que le cookie de session restera actif jusqu'au prochain démarrage de votre navigateur.

La valeur indiquée par défaut de **session.gc_maxlifetime** est **1440 secondes (24 minutes)**. Après 24 minutes d'inactivité le processus de ramasse-miettes nettoiera les valeurs stockées dans la session.

Exo en cours ;).

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 6: PHP+MYSQL +AJAX+JSON	

Contexte

Durant ce TD nous allons apprendre à développer une petite application web monopage 'Single Page Application =SPA'.

« Une **application web monopage** (en anglais *single-page application* ou **SPA**) est une [application web](#) accessible via une [page web](#) unique. Le but est d'éviter le chargement d'une nouvelle page à chaque action demandée, et de fluidifier ainsi l'expérience utilisateur. Deux méthodes existent pour ce faire : l'ensemble des éléments de l'application est chargé (contenu, images, [CSS](#) et [JavaScript](#)) dans un unique fichier [HTML](#), soit les ressources nécessaires sont récupérées et affichées dynamiquement en fonction des actions de l'utilisateur. Le terme a été introduit par [Steve Yen](#) en 2005 ».

Vous pouvez trouver sur Internet, des informations détaillées sur les raisons de cette nouvelle technique, de ma part je me contente d'en citer deux :

- si vous souhaitez effectuer pas mal de traitement offline (sur le client), SPA est une bonne approche
- la gestion de sessions et cookies est facilitée.

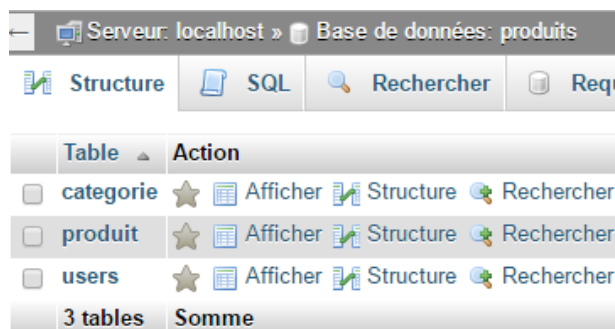
On parle aujourd'hui de migration des applications web classiques vers le modèle SPA. Ce mouvement est appelé **AJAXification**.

Ce TD sera l'occasion de découvrir les technos qui se cachent derrière AJAX : **XMLHttpRequest+JS+CSS+HTML+XML+JSON**. La seule nouveauté est l'objet XMLHttpRequest. Aussi, nous allons utiliser la librairie JQuery ; elle offre une implémentation « sympathique » des appels AJAX en cachant l'objet XMLHttpRequest .

Commençons le développement sans tarder !

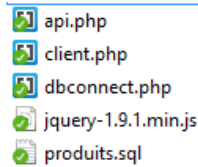
Exercice 1

Vous allez modifier la structure de la base « produits ».



Les étapes de création de la base décrites ultérieurement peuvent être exécutées simplement avec phpmyadmin, mais j'en profite de l'occasion pour vous montrer comment les faire avec une console de commande en ligne.

- 1.1 Télécharger le code ajax.zip (sur le site cèleène) et le stocker/dézipper sur le serveur dans le répertoire de l'application vitrine. Dans ajax.zip vous trouverez les fichiers suivants :



api.php
client.php
dbconnect.php
jquery-1.9.1.min.js
produits.sql

- Connectez-vous au serveur mysql

```
root@lamp www/vitrine2# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1334
Server version: 5.5.54-0+deb8u1 (Debian)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

- Supprimer la base « produits »

```
mysql> drop database produits;
```

- Recréer la nouvelle base « produits »

```
mysql> create database produits;
```

- Exécuter le script SQL de création des tables **produits.sql** (users, produit, categorie) :

```
mysql> use produits;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
```

```
mysql> source produits.sql;
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

- Afficher les tables ainsi créées :

```
mysql> show tables;
+-----+
| Tables_in_monitor |
+-----+
| categorie          |
| produit            |
| users              |
+-----+
3 rows in set (0.00 sec)
```

- Vérifier le contenu des tables

```
mysql> select * from produit;
```

code	nom	description	prix	cat_id
1	Samsung Galaxy A9	Samsung Galaxy A9	2000	1
2	Samsung Galaxy S7	Samsung Galaxy S7	250	1
3	Samsung Galaxy S6 edge	Samsung Galaxy S6 edge	300	1
4	Xperia Z5 Premium	Xperia Z5 Premium	400	2
5	Xperia M5 Dual	Xperia M5 Dual	450	2
6	Xperia C5 uplta	Xperia C5 uplta	500	2
7	Moto G Turbo	Moto G Turbo	600	3
8	Moto X Force	Moto X Force	700	3
9	Redmi 3 Pro	Redmi 3 Pro	800	4
10	Mi 5	Mi 5	900	4

```
10 rows in set (0.00 sec)
```

Structure de la table catégorie :

```
CREATE TABLE IF NOT EXISTS `categorie` (
  `cat_id` int(11) NOT NULL,
  `cat_name` varchar(20) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=latin1;
```

```
-- Contenu de la table `categorie`
INSERT INTO `categorie` (`cat_id`, `cat_name`) VALUES
(1, 'Samsung'),
(2, 'Sony'),
(3, 'Motorola'),
(4, 'Xiaomi');
```

Structure de la table produit :

```
CREATE TABLE IF NOT EXISTS `produit` (
  `code` int(11) NOT NULL,
  `nom` varchar(50) NOT NULL,
  `description` varchar(50) NOT NULL,
  `prix` int(11) NOT NULL,
  `cat_id` int(11) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=latin1;
```

```
--Contenu de la table `produit`
INSERT INTO `produit` (`code`, `nom`, `description`, `prix`, `cat_id`) VALUES
(1, 'Samsung Galaxy A9', 'Samsung Galaxy A9', 2000, 1),
(2, 'Samsung Galaxy S7', 'Samsung Galaxy S7', 250, 1),
(3, 'Samsung Galaxy S6 edge', 'Samsung Galaxy S6 edge', 300, 1),
(4, 'Xperia Z5 Premium', 'Xperia Z5 Premium', 400, 2),
(5, 'Xperia M5 Dual', 'Xperia M5 Dual', 450, 2),
(6, 'Xperia C5 uplta', 'Xperia C5 uplta', 500, 2),
(7, 'Moto G Turbo', 'Moto G Turbo', 600, 3),
(8, 'Moto X Force', 'Moto X Force', 700, 3),
(9, 'Redmi 3 Pro', 'Redmi 3 Pro', 800, 4),
(10, 'Mi 5', 'Mi 5', 900, 4);
```

Structure de la table users :

```
CREATE TABLE IF NOT EXISTS `users` (
  `code` int(11) NOT NULL,
  `nom` varchar(10) NOT NULL,
```

```
`password` varchar(4) NOT NULL,
`email` varchar(60) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8;
```

-- Contenu de la table `users`

```
INSERT INTO `users` (`code`, `nom`, `password`, `email`) VALUES
(1, 'matthieu', '81dc', 'matthieu@tata.com'),
(3, 'ahmad', '81dc', 'titi@titi.com'),
(5, 'mich', '202c', 'mich@free.com');
```

1.2 l'application SPA est constituée de deux pages php :

- **api.php**, une page php simple qui récupère tous les produits de la base.
- **client.php**, la page SPA, qui représente l'interface unique disponible à l'internaute. Cette page instancie, en arrière-plan, l'objet AJAX et émet la requête asynchrone à destination de api.php.
- n'oublions pas :
 - la librairie jQuery.
 - dbconnect.php

code de la page client.php :

```
<html>
<head>
  <script language="javascript" type="text/javascript" src="jquery-1.9.1.min.js"></script>
</head>
<body>
  <h2> Client </h2>
  <h3>Output: </h3>
  <div id="output"> dans cet élément le résultat sera affiché !</div>

  <script id="source" language="javascript" type="text/javascript">

$(document).ready(function () // JQuery
{

$.ajax({
  url: 'api.php',           //appel de api.php
  data: "",                 //you can insert url arguments here to pass to api.php
                             //for example "id=5&parent=6"
  type: "GET",              // méthode d'envoi GET/POST

  dataType: 'json',         // format attendu des données retournées
  success: function(rows)  //on recieve of reply
  {

    for (var i in rows)
    {
      var row = rows[i];
      var id = row["code"];
      var vname = row["nom"];
      $('#output').append("<br/><b>id: </b>" + id + "<b> name: </b>" + vname);
    }
  }
}
```

```
});  
});
```

```
</script>  
</body>  
</html>
```

Déployer l'application, et Testez-là.

Le point central de cette architecture SPA : vous pouvez découper votre page unique en plusieurs zones (<div id="output">), chacune de ces zones se voient modifier par une requête AJAX asynchrone.

1.3 Ajouter une nouvelle zone d'affichage à votre page **client.php**. Cette zone récupérera les utilisateurs de votre site (table users). Cette zone sera, par exemple, une zone de type liste de sélection.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 7 : Angular	

Pour développer votre application avec AngularJS, vous pouvez télécharger AngularJS à partir du site [http:// angularjs.org/](http://angularjs.org/).

Pour utiliser AngularJS il suffit de spécifier dans le code source:

```
<script src="lib/angular.min.js"></script>
```

Ceci signifie que le fichier AngularJS sera chargé depuis un répertoire local. Vous pouvez aussi utiliser AngularJS de la manière suivante :

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js"></script>
```

1. Premiers pas avec ng

Pour commencer il faut créer avec un éditeur de texte un document html que nous allons nommer **demo.html**.

Ecrivez le script suivant dans **demo.html**

```

1  <!DOCTYPE html>
2  <html ng-app>
3  <head>
4      <title>Getting Started</title>
5      <script src="lib/angular.min.js"></script>
6  </head>
7  <body>
8      <h1>Demo simple AngularJS</h1>
9      <br>
10     <p>
11         2 + 3 = {{ 2+3 }}
12     </p>
13     <p>
14         2 - 3 = {{ 2-3 }}
15     </p>
16     <p>
17         2 * 3 = {{ 2*3 }}
18     </p>
19 </body>
20 </html>

```

ngApp qui se trouve dans la balise HTML est une directive pour **injecter** directement des comportements dynamiques dans le code HTML. On peut le placer dans les balises HTML ou BODY.

Enregistrez ce code et exécutez-le.

2. Hello world

Dans un deuxième exemple on va développer une application AngularJS qui utilise un contrôleur. Vous devez créer deux fichiers, un fichier HTML intitulé **hello.html** un fichier contrôleur intitulé **app.js**.

Pour commencer, il faut créer le fichier **hello.html** contenant le script suivant :

```
1 <!DOCTYPE html>
2 <html ng-app='monApp'>
3
4 <head>
5     <title>Hello AngularJS</title>
6     <script src="lib/angular.min.js"></script>
7     <script src="controllers/app.js"></script>
8 </head>
9
10 <body>
11     <div ng-controller='sayHello'>
12         <label for="name">Your name:</label>
13         <input type="text" id="name" ng-model='name.text'>
14         <p>{{name.text}}, how are you?</p>
15     </div>
16 </body>
17
18 </html>
```

Explication:

On définit in contrôleur **AngularJS** auquel on donne le nom *sayHello*, ng-controller='sayHello'

- On définit un modèle AngularJS concernant le test en entrée, ng-model = 'name.text'
- Déclarer le texte {{name.text}} dans la balise <p>

Ensuite on crée un contrôleur AngularJS, contenant le code suivant :

```
function sayHello($scope) {
    $scope.name = {
        text: 'Your Name'
    };
}
```

Maintenons vous allez transformer ce contrôleur en un module en modifiant le code comme suit :

```
1 var app = angular.module('monApp', []);
2 app.controller('sayHello', ['$scope', function ($scope) {
3     $scope.name = {
4         text: "Your Name"
5     };
6 }]);
```

Enregistrez et testez.

3 Contrôleur avec plusieurs attributs

Dans cette section on crée un contrôleur avec plusieurs attributs, par exemple fonction et valeur.

Commencez par créer un fichier intitulé **simplemodule.js** et contenant le code suivant :

```
1 var myApp = angular.module('SimpleModule', []);
2 myApp.controller('MyController', ['$scope', function ($scope) {
3     $scope.title = 'Module Simple';
4     $scope.result = 0;
5     $scope.add = function (a, b) {
6         $scope.result = a + b;
7     };
8     $scope.multiple = function (a, b) {
9         $scope.result = a * b;
10    };
11 }]);
12
```

Vous pouvez voir que les résultats de la fonction sont enregistrés dans `$scope.result`. Maintenant nous pouvons utiliser ce module dans le fichier HTML. Créez un fichier HTML intitulé **simplemodule.html** contenant le code suivant :

```
1 <!DOCTYPE html>
2 <html ng-app="SimpleModule">
3
4 <head>
5     <title> Module Simple</title>
6     <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js"></script>
7     <script src="controllers/simplemodule.js"></script>
8 </head>
9
10 <body ng-controller='MyController'>
11     <br>
12     <h1>Demo {{title}}</h1>
13     <div>
14         <p>
15             <label for="valA">Valeur de A:</label>
16             <input type="number" id="valA" ng-model='valA'>
17         </p>
18         <p>
19             <label for="valB">Valeur de B:</label>
20             <input type="number" id="valB" ng-model='valB'>
21         </p>
22         <p>
23             Result: {{ result }}
24         </p>
25         <p>
26             <button ng-click="add(valA,valB)">Additionner</button>
27             <button ng-click="multiple(valA,valB)">Multiplier</button>
28         </p>
29     </div>
30 </body>
31
32 </html>
```

Demo Module Simple

Valeur de A:

Valeur de B:

Result: 0

4 Travailler avec plusieurs Contrôleur

Parfois vous pouvez avoir besoin de créer plusieurs contrôleurs dans un module Angular. Pour illustrer cela vous allez créer un fichier intitulé **multicontrollers.js** et contenant le code suivant :

Vous avez créé deux contrôleurs, Math1 and Math2.

```
1  var myApp = angular.module('MultiController', []);
2  myApp.controller('Math1', ['$scope', function($scope) {
3      $scope.result = 0;
4      $scope.add = function(a,b) {
5          $scope.result = a+b;
6      };
7  }]);
8  myApp.controller('Math2', ['$scope', function($scope) {
9      $scope.result = 0;
10     $scope.multiple = function(a,b) {
11         $scope.result = a*b;
12     };
13 }]);
```

Dans un fichier HTML, créez un fichier **multicontroller.html** contenant le code suivant.

```
1  <!DOCTYPE html>
2  <html ng-app="MultiController">
3  <head>
4      <title>Simple Module</title>
5      <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js"></script>
6      <script src="controllers/multicontrollers.js"></script>
7  </head>
8  <body>
9      <br/>
10     <div>
11         <p>
12             <label for="valA">Valeur de A:</label>
13             <input type="number" id="valA" ng-model='valA'> {{}}
14         </p>
15         <p>
16             <label for="valB">Valeur de B:</label>
17             <input type="number" id="valB" ng-model='valB'>
18         </p>
19         <p ng-controller='Math1'>
20             <button ng-click="add(valA,valB)">Additionner</button> Result: {{result}}
21         </p>
22         <p ng-controller='Math2'>
23             <button ng-click="multiple(valA,valB)">Multiplier</button> Result: {{result}}
24         </p>
25     </div>
26 </body>
27 </html>
```

Chaque contrôleur est défini dans une balise `<p>`. Ensuite on utilise **ng-click** pour appeler la fonction contrôleur et montrer le résultat avec `{{ result }}`. Enregistrez les codes et exécutez le fichier **multicontroller.html**. Réfléchissez à l'utilité d'utilisation de deux contrôleurs dans le cas de cet exemple ?

5 Appeler les fonctions et contrôleur

Supposons que vous avez besoin d'appeler un contrôleur NG depuis JS.

Créer le fichier **contactmodule.js** contenant le script suivant :

```
1  var myApp = angular.module('ContactModule', []);
2  myApp.controller('ContactController', ['$scope', function($scope) {
3      $scope.contact = {
4          name: "Ahmad Abdallah",
5          email: "ahmad.abdallah@insa-cvl.fr",
6          phone: "064000000"
7      };
8      $scope.calculate = function(a,b) {
9          return (a+b)*3;
10     }
11 }]);
```

On appelle la fonction `calculate()` et utiliser les données de contact depuis le JavaScript de la page HTML.

Créer un fichier **contactdemo.html**

```
1  <html ng-app="ContactModule">
2  <head>
3      <title>Calling AngularJS Controller</title>
4      <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js"></script>
5      <script src="controllers/contactmodule.js"></script>
6      <script language="JavaScript">
7          function getContact() {
8              var mydiv = document.getElementById('mydiv');
9              var scope = angular.element(mydiv).scope();
10             var contact = scope.contact;
11             scope.$apply();
12             alert("Read contact." + JSON.stringify(contact));
13         }
14
15         function calculate() {
16             var mydiv = document.getElementById('mydiv');
17             var scope = angular.element(mydiv).scope();
18             var result = scope.calculate(2,3);
19             scope.$apply();
20             alert("call AngularJS controller function. Result: " + result);
21         }
22     </script>
23 </head>
24 <body>
25     <p>Demo calling AngularJS controller function from External JavaScript</p>
26     <div id="mydiv" ng-controller="ContactController">
27         <button onclick="getContact()" >Get Contact</button>
28         <button onclick="calculate()" >Calculate</button>
29     </div>
30 </body>
31 </html>
```

Enregistrez vos codes. Maintenant exécutez le fichier **contactdemo.html** dans votre navigateur. En cliquant sur **GetContact** Vous devriez obtenir le contact grâce à `alert()`. Remplacer les appels natifs javascript par des requêtes jQuery.

6 Mise en relation des données

Dans cette exo, nous essayons de faire correspondre des éléments d'entrée HTML dans un Angular *model*. Nous commençons par créer un module appelé `simplebinding.js`

```
var myApp = angular.module('SimpleBinding', []);
myApp.controller('MyControlller', ['$scope', function($scope){
    $scope.customer = {}
}
]);
```

customer est défini, dans le scope, en tant qu'objet JSON.

L'étape suivante consiste à créer un fichier HTML, appelé **simplebinding.html** contenant le code suivant (le début) :

```
<!DOCTYPE html>
<html ng-app="SimpleBinding">
<head>
<title>Simple Module</title>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js"></script>
<script src="controllers/simplebinding.js"></script>
</head>
<body ng-controller="MyControlller">
<form>
```

Les premières données d'entrée sont *name* et *email*.

On définit un *model* Angular `ng-model="customer.name"` et `ng-model="customer.email"` comme suit:

```
<!DOCTYPE html>
<html ng-app="SimpleBinding">
<head>
<title>Simple Module</title>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js"></script>
<script src="controllers/simplebinding.js"></script>
</head>
<body ng-controller="MyControlller">
<form>

<div>
<label for="name">Name: </label>
<input type="text" id="name" ng-model="customer.name">
</div>
<div>
<label for="email">Email: </label>
<input type="text" id="email" ng-model="customer.email">
</div>
</form>
<body>
</html>
```

Pour voir quel utilisateur a renseigné les données requises, nous pouvons utiliser l'expression angular `{{customer}}`.

```
<div>
  value:{{customer}}
</div>
```

Enregistrez vos pages, exécutez et observez le résultat.

Ajouter dans le formulaire une liste déroulante et un `ng-model = 'customer.country'` :

```
<div>
<label for="selectcountry">Country:</label>
<select name="selectcountry" id="selectcountry" ng-model="customer.country">
<option value="AU">Australia</option>
<option value="CA">Canada</option>
<option value="FR">France</option>
<option value="DE">Germany</option>
<option value="NL">Netherlands</option>
<option value="CH">Switzerland</option>
<option value="GB">United Kingdom</option>
<option value="US">United States</option>
</select>
</div>
```

Enregistrez vos pages, exécutez et observez le résultat.

Ajouter dans le formulaire une fourchette :

```
<div>
<label for="level">Skill level:</label>
<input type="range" name="level" id="level" min="1" max="100" alt="Level 1-100" ng-
model="customer.skilllevel" >
</div>
```

Enregistrez vos pages, exécutez et observez le résultat.

Compléter l'interface (la vue) pour avoir quelque chose proche de :

Name:

Email:

Country:

Preference color: ☐ Red ☐ Green ☐ Blue

Birth Date:

Skill level:

☒ I agree to the terms & conditions

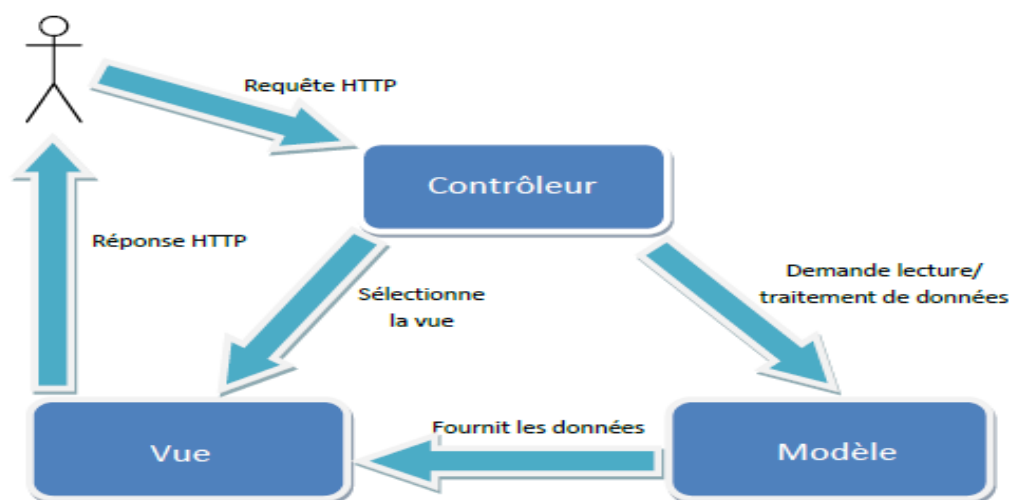
Value:{"skilllevel":51,"agree":true}

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE	5 ^{ème} Année STI Ingénierie du Web	2016-2017
	TD 8 : ASP.NET	

Le but de ce TD est de vous montrer comment construire une application ASP.NET MVC. Dans ce TD, nous allons commencer par la construction d'une application ASP.NET MVC du début à la fin. L'application construite est application orientée base de données (database-driven).

Le modèle MVC est constitué des éléments suivants :

- Le **Modèle** : représente la couche métier d'une application, présentant des classes permettant de créer les objets contenant des données métier manipulées par l'application au travers de traitements, constituant les services métiers.
- La **Vue** : elle constitue les éléments d'interface utilisateurs : pages web, contrôles Web...
- Le **Contrôleur** : permettant de piloter l'application, il interprète les actions à réaliser et ordonne leur exécution (lecture, traitement de données et mises à jour).



Nous aurons besoin de l'image virtuelle Visual Studio 2010. Retrouvez-la dans mon dossier.

MVC DANS LE FRAMEWORK .NET

Pour mettre en œuvre le modèle MVC, Microsoft a ajouté un nouvel espace de noms dans le Framework .NET, nommé System.Web.Mvc. Il contient toutes les classes et interfaces permettant de mettre en œuvre le modèle MVC : classes de base pour les contrôleurs, classes pour les vues, classes désignant les actions, permettant de créer des liaisons de données entre la vue et le modèle...

On trouvera aussi l'espace de noms System.Web.Mvc.Ajax, permettant de mettre en œuvre des mécanismes Ajax dans les pages ASP .NET MVC.

PAGES WEB VS PAGES MVC

La structure des applications ASP .NET « classiques » et les applications ASP .NET MVC sont radicalement différentes, tout en ayant le même but : construire des applications Web.

APERÇU DE L'APPLICATION SIS (STUDENT INFORMATION SYSTEM)

Parce que notre objectif est de garder les choses simples, nous allons construire une application SIS simple.

Notre application SIS va nous permettre de faire trois choses:

- Liste des étudiants
- Enregistrer un étudiant
- Modifier un étudiant

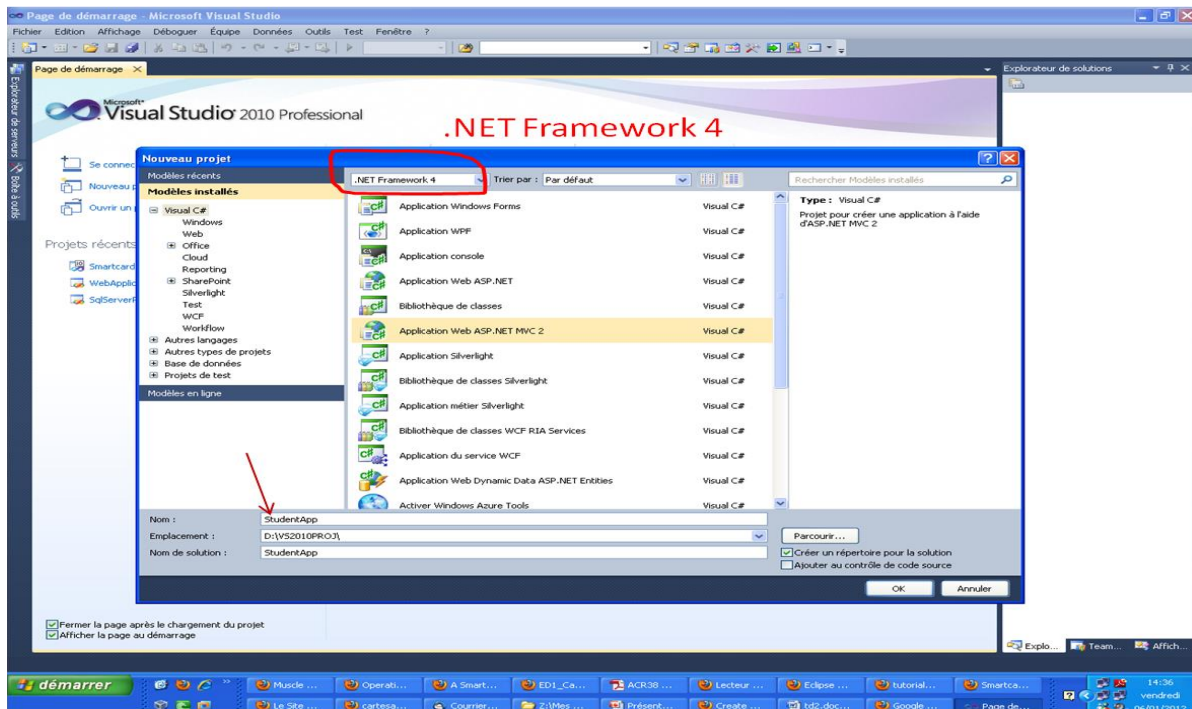
CREATION D'ASP.NET MVC WEB APPLICATION PROJECT

Commençons par créer un nouveau projet ASP.NET MVC Web Application dans Visual Studio. Sélectionnez l'option du menu Fichier, Nouveau projet, et vous verrez la boîte de dialogue Nouveau projet dans la figure 1.

Sélectionnez Visual C# comme langage de programmation et sélectionnez le modèle Web ASP.NET MVC2 de projet d'application. Donnez à votre projet le nom **StudentApp** et cliquez sur le bouton OK.

Lorsque vous créez un nouveau projet d'application Web MVC, Visual Studio vous invite à créer une unité séparée de projet de test. Sélectionnez l'option Non et cliquez sur le bouton OK.

Lorsque vous créez un nouveau projet d'application Web MVC, Visual Studio vous invite à créer une unité séparée de projet de test. Sélectionnez l'option Non et cliquez sur le bouton OK.



Une application ASP.NET MVC est un ensemble standard de dossiers: Modèles, Vues, et le dossier des contrôleurs. Vous pouvez voir cela dans la fenêtre Explorateur de solutions. Parce que nous voulons commencer à partir de zéro, nous avons besoin de supprimer le contenu de cet exemple d'application. Vous devez supprimer le fichier suivant et le dossier suivant:

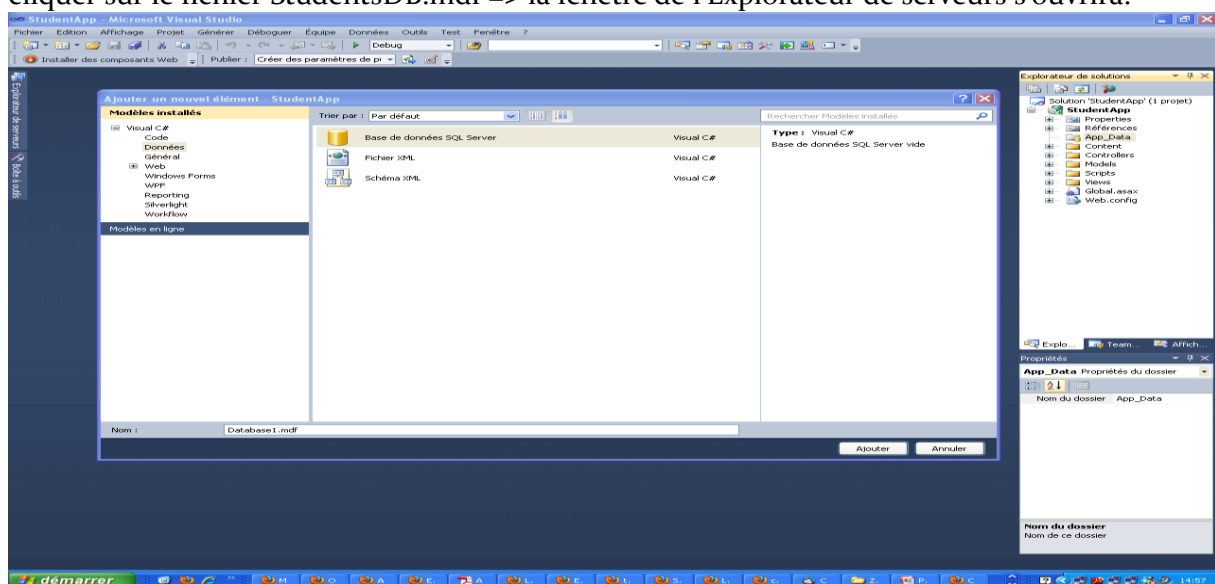
- Controllers\HomeController.cs (fichier)
- Views\Home (rep.)

CREATION DE LA BASE DE DONNEES

Nous avons besoin de créer une base de données pour tenir nos données. Heureusement, Visual Studio inclut une base de données gratuite nommée SQL Server Express. Suivez ces étapes pour créer la base de données:

- Cliquez-droit sur le dossier App_Data dans la fenêtre de l'Explorateur de solutions et sélectionnez l'option Ajouter, Nouvel Elément.
- Sélectionnez la catégorie de données et sélectionnez le modèle de données SQL Server (voir figure).
- Nom de votre nouvelle base de données : StudentsDB.mdf et cliquez sur le bouton Ajouter.

Après avoir créé votre base de données, vous pouvez vous connecter à la base de données en double-cliquant sur le fichier StudentsDB.mdf trouvé dans le dossier App_Data. Double-cliquer sur le fichier StudentsDB.mdf => la fenêtre de l'Explorateur de serveurs s'ouvrira.



Ensuite, nous avons besoin pour créer une table nouvelle base de données. De l'intérieur de la fenêtre Explorateur de Sever, cliquez-droit sur le dossier Tables et sélectionnez l'option de menu Ajouter une nouvelle table. La sélection de cette option de menu ouvre le concepteur de table de la base. Créer des colonnes de bases de données suivantes:

Column Name	Data Type	Allow Nulls
id	Int	False
nom	Nvarchar(100)	False
dept	Nvarchar(100)	False
dateinscription	DateTime	False

Table: Students

La première colonne, la colonne ID, a deux propriétés particulières. D'abord, vous devez marquer la colonne ID comme la colonne de **clé primaire**. Après avoir sélectionné la colonne ID, cliquez-droit sur le bouton de la souris, puis définir la clé primaire (c'est l'icône qui ressemble à une clé). Deuxièmement, vous devez marquer la colonne id comme une colonne d'identité. Dans la fenêtre Propriétés des Colonnes, faites défiler jusqu'à la section Spécifications du Compte et étendre. Modifier la propriété « est d'identité » à Oui.

Après avoir créé la table, enregistrez la table. Ajoutez-y quelques enregistrements : Cliquez-droit sur la table **Students** dans la fenêtre de l'Explorateur de serveurs et sélectionnez le menu Afficher les données de la table. Entrez une liste d'étudiants.

CREATION DU MODELE

Nous avons ensuite besoin de créer un ensemble de classes (modèle objet) pour représenter notre base de données. Nous allons, donc, exploiter le « Entity Framework » de dotNET pour générer les classes de notre modèle de base. (unetable -->une entité).

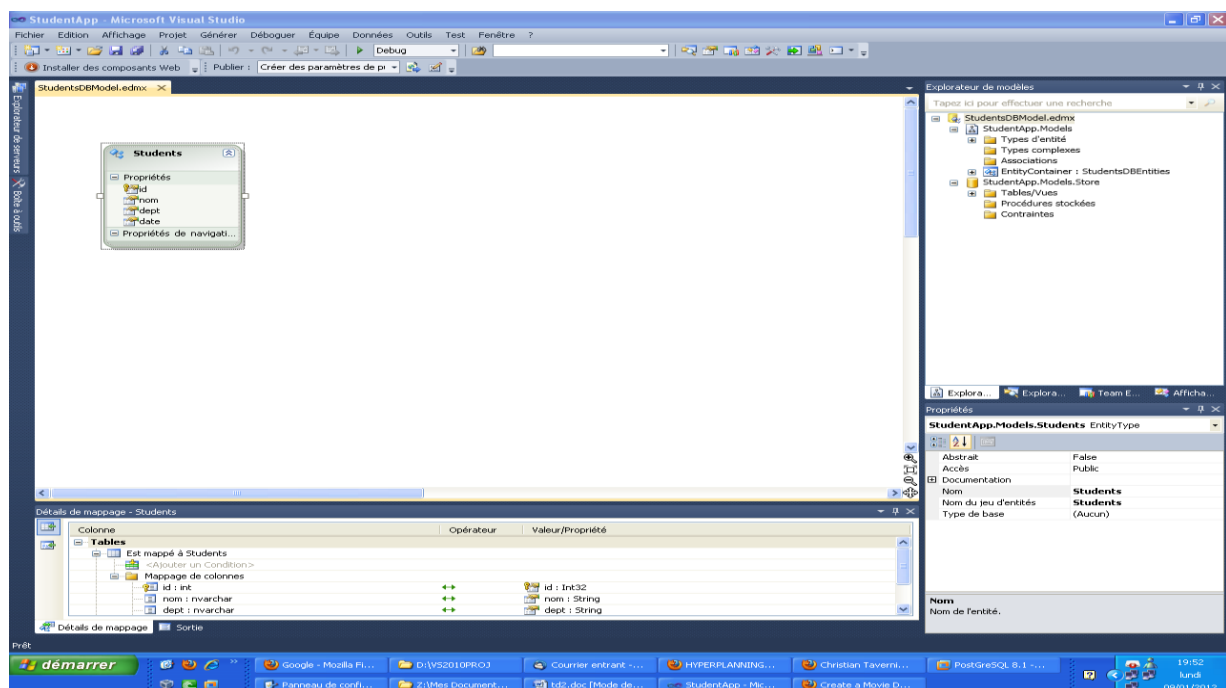
Le framework ASP.NET MVC n'est pas liée à l'**Entity Framework** de Microsoft. Vous pouvez créer vos modèles de classes de votre base de données en tirant profit d'une variété de solutions Objet Relationnel Mapping (OR/M), y compris des outils de LINQ to SQL, Subsonic, et NHibernate.

Suivez ces étapes pour lancer l'Assistant Entity Data Model:

- Cliquez-droit sur le dossier Modèles dans la fenêtre de l'Explorateur de solutions et sélectionner l'option Ajouter, Nouvel élément.
- Sélectionnez la catégorie de données et sélectionnez l'ADO.NET Entity Data Model.
- Donnez le nom **StudentsDBModel.edmx** au modèle de données et cliquez sur le bouton Ajouter.

Après avoir cliqué sur le bouton Ajouter, l'Assistant Entity Data Model apparaît. Suivez ces étapes pour terminer l'assistant:

- Sélectionner « **Generate from database** ».
- Sur cet écran, accepter les données proposées par défaut. Suivant
- Dans **Your Database Objects**, choisir la table Students. Changer l'espace de nom en *StudentApp.Models* puis Terminer.



Changer le nom de la classe générée (Students) en (Student) ; chaque instance de la classe Student représente, en effet, un seul étudiant de la base de données. Pour cette raison nous parlons du mappage relationnel/objet (Table Students \leftrightarrow Classe Student) : ORM

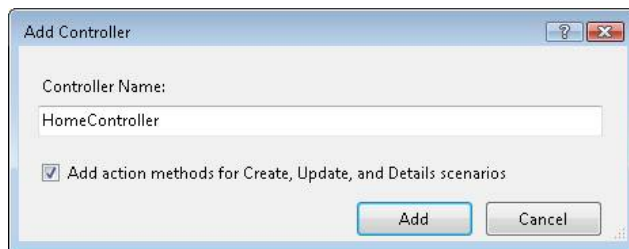
Sauvegarder.

CREATION DE ASP.NET MVC CONTROLLER

Dans la fenêtre de l'Explorateur de solutions, cliquez-droit sur le dossier Controllers et sélectionnez l'option Ajouter contrôleur.

Dans la boîte de dialogue Ajouter Controller, entrez le nom **HomeController** et cochez la case Ajouter des méthodes d'action pour les scénarios Création, mise à jour, détails.

Cliquez sur le bouton Ajouter pour ajouter le nouveau contrôleur à votre projet.



Après avoir effectué ces étapes, le contrôleur est créé. Remarquez qu'il contient des méthodes nommées INDEX, DETAILS, CREATE et EDIT. Dans les sections suivantes, nous allons ajouter le code nécessaire à ces méthodes.

LISTING DES ENREGISTREMENTS DE LA TABLE STUDENT

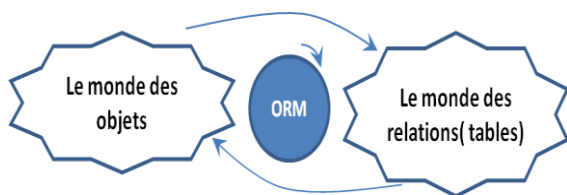
La méthode Index() du contrôleur d'accueil est la méthode par défaut pour une application ASP.NET MVC. Lorsque vous exécutez une application ASP.NET MVC, la méthode Index () est la première méthode appelée du contrôleur.

Nous utiliserons la méthode Index () pour afficher la liste des enregistrements de la table de base **Students**. Pour le faire, nous profiterons de la classe **Student** que nous avons créé plus tôt pour instancier les enregistrements de la base avec la méthode Index ().

J'ai modifié la classe HomeController dans le listing afin qu'elle contienne un nouveau champ privé nommé _db. La classe StudentsDBEntities représente notre modèle de base et nous utiliserons cette classe pour communiquer avec notre base de données.

Résumons-nous :

StudentsDBEntities : pour accéder aux données de la base de manière transparente (la tuyauterie nécessaire a été générée par VS) : c'est la couche O/R M.



Student : un objet de la classe Student représente un enregistrement.

```
public class HomeController : Controller
{
    //
    // GET: /Home/
    private StudentsDBEntities _db = new StudentsDBEntities();

    public ActionResult Index()
    {
        return View(_db.Students.ToList());
    }
}
```

NB : `using StudentApp.Models;`//ajouter, s'il ne trouve pas la classe **StudentsDBEntities**.

La méthode Index () retourne une vue nommée Index. Nous avons besoin de créer cette vue pour afficher la liste des enregistrements de la base de données. Suivez ces étapes:

- D'abord Sauvegarder puis Générer la solution (Menu Générer)
- Cliquez-droit sur la méthode Index () dans l'éditeur de code, puis sélectionnez l'option Ajouter une vue.
- Dans la boîte de dialogue vérifiez que la case Créer une vue fortement typée est cochée.
- Afficher la classe de données : StudentApp.Models.Student
- Contenu de la vue : List
- Cliquez sur le bouton Ajouter pour créer la nouvelle vue.

La vue Index affiche tous les enregistrements de la table de base des étudiants dans un tableau HTML. La vue contient une boucle For Each qui parcourt chaque étudiant représenté par la propriété ViewData.Model.

Allez dans le fichier Site.Master (Views/shared) et remplacez « Mon Application MVC » par « Student Information System ».

Exécutez votre application en appuyant sur la touche F5.

***Nota** : chaque fois que c'est possible, regarder le code source de la page web dans votre navigateur. C'est un moyen d'apprendre comment, finalement, le code HTML est « fabriqué » à la volée par APS.NET.*

AJOUTER DES ENREGISTREMENTS

La vue **Index** que nous avons créé dans la section précédente inclut un lien « Create new » pour créer des enregistrements dans la base de données (mais pour le moment ce lien ne fait rien). Allons de l'avant et mettant en œuvre la logique nécessaire pour créer de nouveaux enregistrements.

Le contrôleur **HomeController** contient deux méthodes nommée **Create ()**. La première méthode n'a pas de paramètres. Elle utilisée pour afficher un formulaire HTML servant à créer un nouvel étudiant dans la base de données.

La seconde a un paramètre **FormCollection**. Cette deuxième méthode surchargée est appelée lorsque le formulaire HTML pour créer un étudiant est envoyé au serveur. Notez que

cette seconde méthode, a un attribut **AcceptVerbs** qui empêche la méthode d'être appelée à moins qu'une opération **HTTP POST** soit exécutée.

```
[HttpPost]
public ActionResult Create( [Bind(Exclude="Id")] Student studentToCreate)
{
    // TODO: Add insert logic here
    if (!ModelState.IsValid)
        return View();

    _db.AddToStudents(studentToCreate);

    _db.SaveChanges();

    return RedirectToAction("Index");
}
```

Nous allons créer la vue Create.

1. Right-click the Create() method in the code editor and select the menu option **Add View**.
2. Verify that the checkbox labeled **Create a strongly-typed view** is checked.
3. From the **View content** dropdown list, select the value *Create*.
4. From the **View data class** dropdown list, select the value *StudentApp.Student*
5. Click the **Add** button to create the new view.

Le formulaire HTML généré par la boîte de dialogue **Add View** génère un champ Id dans le formulaire. Comme la colonne Id est une colonne d'identité, vous n'avez pas besoin de ce champ de formulaire et vous pouvez le supprimer. Dans la vue Create (page Create.aspx) supprimer :

```
<div class="editor-label">
    <%: Html.LabelFor(model => model.id) %>
</div>

<div class="editor-field">
    <%: Html.TextBoxFor(model => model.id) %>
    <%: Html.ValidationMessageFor(model => model.id) %>
</div>
```

Exécutez et Testez

EDITER UN ENREGISTREMENT

Dans les sections précédentes, nous avons discuté de la façon dont vous pouvez lister et créer des enregistrements. Dans cette dernière section, nous discutons de la façon dont vous pouvez modifier les enregistrements.

Premièrement, nous avons besoin de générer le formulaire de modification. Cette étape est facile, car Visual Studio va générer le formulaire de modification pour nous automatiquement.

Ouvrez la classe HomeController.cs dans l'éditeur de code Visual Studio et suivez ces étapes:

Faites un clic droit de la méthode Edit () dans l'éditeur de code, puis sélectionnez l'option Add View. Vérifiez la case Créer une vue fortement typée. Choisir Edit dans le menu déroulant. Appuyer sur le bouton Ajouter.

Une nouvelle vue nommée Edit.aspx dans Vues \Home. Cette vue contient un formulaire HTML pour éditer un enregistrement.

La vue Edit contient un champ de formulaire HTML qui correspond à la propriété ID. Parce que vous ne voulez pas éditer la valeur de la propriété ID, vous devez supprimer ce champ du formulaire (Vue Edit.aspx).

Enfin, nous avons besoin de modifier le contrôleur Home afin qu'il supporte l'édition un enregistrement. La classe HomeController mise à jour est présentée dans le Listing :

```
// GET: /Home/Edit/5
public ActionResult Edit(int id)
{
    var studentToEdit = (from m in _db.Students
                        where m.id == id //Attention : respectez la casse
                        select m).First();
    return View(studentToEdit);
}

// POST: /Home/Edit/5
[HttpPost]
public ActionResult Edit(Student studentToEdit)
{
    var originalStudent = (from m in _db.Students
                        where m.id == studentToEdit.id
                        select m).First();
    if (!ModelState.IsValid)
        return View(originalStudent);
    _db.ApplyCurrentValues(originalStudent.EntityKey.EntitySetName, studentToEdit);
    _db.SaveChanges();
    return RedirectToAction("Index");
}
```

HttpGet

ASP.NET sait quel Edit appeler en utilisant [HttpPost] comme discriminant