IUT- Département d'Informatique
G. Rey
2016

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

1 Objectifs

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 :

- La prise en main des environnements de développement
- Les principes de base du langage JavaScript
- La découverte du DOM

Un compte rendu donnant la réponse à chacune des questions posées devra être rendu à votre responsable de TD par email à la fin de chacune des séances. L'objet (sujet) de l'email devra être le suivant :

[S4T][JS][Gx][TDx]Nom-Prénom / Nom-Prénom

Avec Gx le numéro de votre groupe (exemple G1 pour le groupe 1) et TDx le numéro du TD inscrit sur le sujet du TD (exemple TD1 pour le premier TD).

Vous ne pouvez pas faire 2 séances avec le même binôme. De même, vous ne pouvez pas faire plus de la moitié des séances seul.

Veuillez, sauf indication contraire de votre responsable de TD, faire l'ensemble des exercices marqués « Obligatoire » dans l'ordre de la feuille de TD. Les exercices « Conseillés » sont à faire si vous avez terminé les précédents ou lors de vos révisions. Les exercices « optionnels » sont là pour les plus passionnés d'entre vous.

2 Documentation (obligatoire)

Pour ce TD mais également pour les TD suivant, sauf mention contraire dans l'énoncé ou si un lien vous est fourni, vous ne devez pas aller chercher des réponses sur internet !! Les seuls sites suivants sont autorisés :

- Le site contenant les supports de cours et de TD.
- Le site de Mozilla : Mozilla Developer Network.
- Les sites du W3C : pour la validation HTML5 ou du CSS. Mais aussi pour avoir la documentation sur le DOM.

3 Prise en main de l'environnement (obligatoire)

Pour réaliser ce TD, vous aurez également besoin d'un éditeur de texte pour écrire le code de vos pages. Vous pouvez par exemple utiliser un des logiciels Notepad++, ConText, Quanta+, WebExpert,... Microsoft vous propose de télécharger gratuitement Expression Web.

Pensez à sauvegarder régulièrement votre travail et à commenter votre code. N'hésitez pas à sauvegarder les différentes versions, évolutions d'un même exercice de manière à pouvoir facilement réviser ou le réutiliser par la suite.

4 Les outils pour parcourir le Document Object Model (obligatoire)

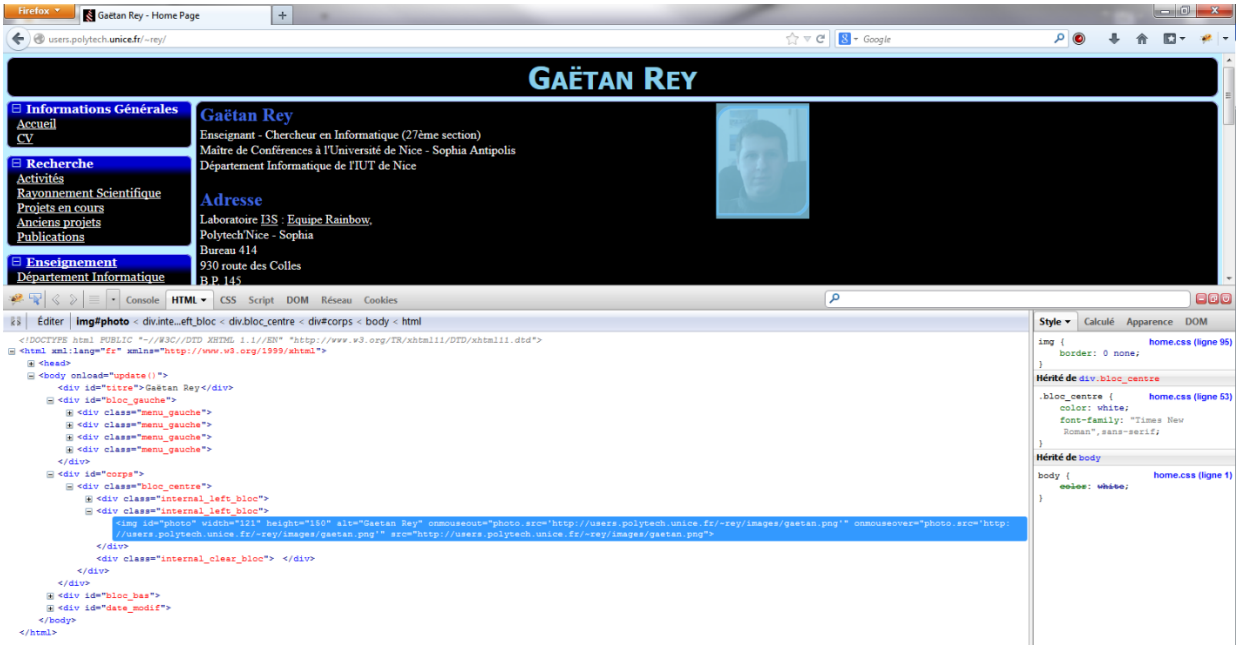
Il existe divers outils de débogage évolués pour les différents navigateurs. Ces outils vous seront très utiles pour explorer l'arbre DOM (Nous verrons plus loin dans ce TD, ce qu'est exactement le DOM) de votre page, vérifier les propriétés CSS ou encore connaitre les erreurs de vos fonctions JavaScript. Voici une rapide présentation de 3 de ces outils pour les 3 navigateurs que nous utiliserons dans ce cours.

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

Il est fortement conseiller d'avoir un de ces outils (ou un équivalent) d'installés pour la réalisation de ce TD.

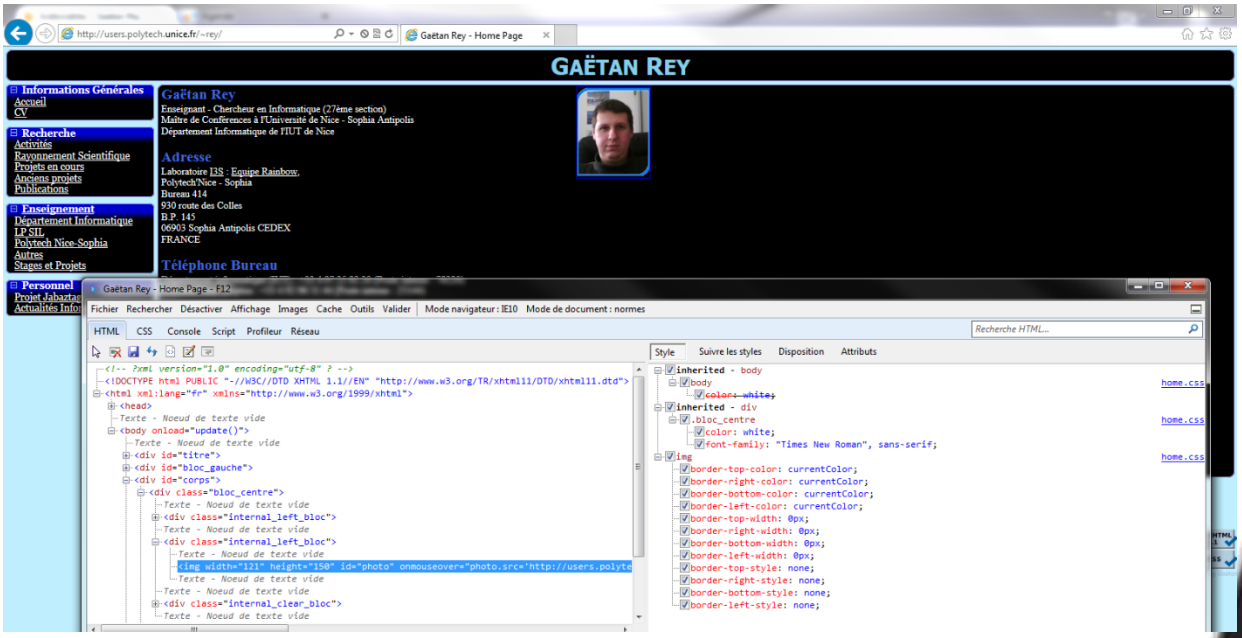
4.1 FireBug, une extension pour Firefox 23

Firebug est un outil de développement web sous forme d'une extension pour Mozilla Firefox et SeaMonkey qui permet de déboguer, modifier et contrôler le HTML, le CSS, le DOM, le XHR et le JavaScript d'une page web. [Wikipédia]



4.2 Outils de développement d'Internet Explorer 10

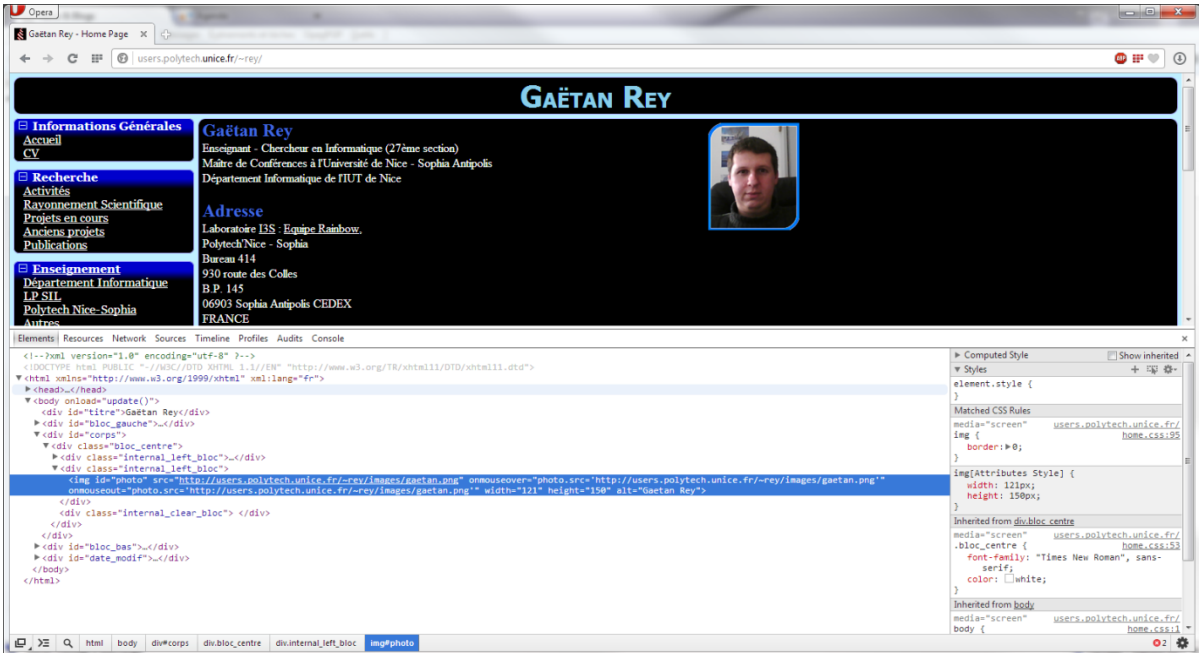
Les outils de développement (F12) ont été introduits dans Windows Internet Explorer 8 et mis à jour avec de nouvelles fonctionnalités dans Windows Internet Explorer 9. Les outils de développement (F12) d'Internet Explorer 10 ajoutent le débogage de threads de travail Web et la prise en charge de plusieurs sources de script. [Microsoft]



Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

4.3 DragonFly dans Opera 23

Dragonfly Opera est une application JavaScript utilisée pour débogger les pages Web locales et distantes. [Opera]



5 Rappel de cours (obligatoire)

Lors de ce TD, l'ensemble des pages web écrites devra (sauf mention contraire) être constitué de pages HTML5 valide ne contenant pas de mise en forme. La mise en forme sera dans un ou plusieurs fichiers CSS valides. De même, le code JavaScript sera également dans des fichiers séparés.

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives mais aussi pour les serveurs2. C'est un langage orienté objet à prototype, c'est-à-dire que les bases du langage et ses principales interfaces sont fournies par des objets qui ne sont pas des instances de classes, mais qui sont chacun équipés de constructeurs permettant de créer leurs propriétés, et notamment une propriété de prototypage qui permet d'en créer des objets héritiers personnalisés. En outre, les fonctions sont des objets de première classe.

Le langage a été créé en 1995 par Brendan Eich (Brendan Eich étant membre du conseil d'administration de la fondation Mozilla à cette époque) pour le compte de Netscape Communications Corporation. Le langage, actuellement à la version 1.8.2, est une implémentation de la 3e version de la norme ECMA-262 qui intègre également des éléments inspirés du langage Python. La version 1.8.5 du langage est prévue pour intégrer la 5e version du standard ECMA3. [Wikipédia]

5.1 Introduction

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.

JavaScript est un langage non typé. Cela signifie qu'il n'est pas forcément utile de déclarer les variables qui vont être utilisées et encore moins d'indiquer leur type. En fait, une variable est implicitement déclarée dès son apparition et typée par son contexte d'utilisation (ses affectations). Le type d'une variable peut donc changer en fonction de son utilisation.

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

Un programme JavaScript s'intègre directement dans votre page HTML entre deux balises <script> et </script>. Si vous souhaité avoir du code XHTML valide, il est préférable d'ajouter les balises CDATA comme présenté ci-dessous :

```
<script type="text/javascript">
  /*<![CDATA[*/
   ... Insérez ici votre code JavaScript ...
  /*]]>*/
</script>
```

Vous pouvez, et cela est recommandé si vous souhaitez partager des fonctions JavaScript entre plusieurs pages HTML, utiliser un fichier externe pour stocker l'ensemble de vos fonctions JavaScript :

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

JavaScript peut intervenir dans un document HTML de deux façons :

1. Il s'exécute au chargement de la page. Le programme JavaScript a pour objet d'écrire dans le document du code HTML pouvant s'adapter dynamiquement à plusieurs facteurs tels le type de configuration matérielle et/ou logicielle du client (navigateur, plugins…), le contenu éventuel de certains cookies, la date, l'heure du moment présent, etc.
2. Il s'agit de scripts qui ne s'exécuteront non pas au moment du chargement de la page, mais une fois celle-ci chargée, lorsque l'utilisateur va interagir avec elle au moyen des différents objets qu'elle contient (liens, boutons, champs de texte…), mais aussi par des actions sur l'environnement (déplacement de la fenêtre, activation d'une autre fenêtre, déplacement de la souris, frappe au clavier…). Ces réactions seront donc provoquées par des événements qui se seront produits après la fin de chargement de la page.

5.2 Introduction au Document Object Model

Document Object Model a pour acronyme DOM. C'est une API pour du (x)html et XML valide. En suivant le DOM, les programmeurs peuvent construire des documents, naviguer dedans, ajouter, modifier ou effacer des éléments à ces documents. En tant que recommandation du W3C, l'objectif du DOM est de fournir une interface de programmation standard pour être utilisée par tous (applications, OS). Suivant le DOM, un document a une structure logique d'arbre (ou de forêt). Le nom DOM provient de l'approche retenue par le W3C : une approche proche des modèles objets (propriétés, méthodes, description).

Le DOM a pour but d'uniformiser la manipulation des documents "web", notamment par les scripts. Cependant, afin de rester indépendant, le DOM est écrit en OMG (Object Managment Group) IDL (interface definition language) tout en proposant les liens pour une transcription dans les différentes implémentations (i.e. langage de programmation) comme ECMAscript (une sorte de standardisation de javascript). Chaque niveau de DOM correspond à des itérations successives, enrichissant au fur et à mesure le DOM. La traduction d'un DOM dans les faits n'est pas immédiate. Aussi, je vous invite à consulter le DOM Level 2.

La différence principale entre DOM level 2 (core) et DOM HTML Level 2 (application du DOM à l'html) est l'existence de méthodes conçue pour les scripts. En plus, il y a la spécialisation des classes :

- HTMLDocument hérite de l'interface document
- HTMLElement hérite de l'interface élément (qui hérite de node)

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

D'un point de vue DOM, la différence entre HTML et xhtml est l'aspect sensible à la casse. En particulier les noms des balises sont en minuscule (pour getElementsByTagName). Le niveau 2 appliqué à l'HTML est défini sur les pages DOM HTML du W3C.

6 Exercice 1 : Commençons doucement avec l'objet document (obligatoire)

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 ».

6.1 La propriété document.title

Ajoutez un id= « titre » à votre h1.

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 connaitre 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 ?

6.2 Les propriétés innerHTML, innerText, outerHTML, outerText et 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.

- Quelles différences existe-t-il entre les 5 propriétés de cette section ?
- Y a-t-il une différence avec Internet Explore, FireFox et d'autres navigateurs à votre disposition concernant ces propriétés ?

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

6.3 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 2015 par Rey Gaëtan » 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. Voici une rapide présentation (mais partielle) de celui-ci extraite du site de Mozilla :

```
Constructeur
new Date()
new Date(milliseconds)
new Date(dateString)
new Date(year, month, day [, hour, minute, second, millisecond ])
Méthodes
Date.prototype.getDate()
      Renvoie le jour du mois (1-31) pour la date spécifiée selon l'heure locale.
Date.prototype.getDay()
      Renvoie le jour de la semaine (0-6) pour la date spécifiée selon l'heure locale.
Date.prototype.getFullYear()
      Renvoie l'année (avec 4 chiffres pour une année à 4 chiffres) pour la date spécifiée selon
l'heure locale.
Date.prototype.getHours()
      Renvoie l'heure (0-23) pour la date spécifiée selon l'heure locale.
Date.prototype.getMilliseconds()
      Renvoie les millièmes de secondes (0-999) pour la date spécifiée selon l'heure locale.
Date.prototype.getMinutes()
      Renvoie les minutes (0-59) pour la date spécifiée selon l'heure locale.
Date.prototype.getMonth()
      Renvoie le mois (0-11) pour la date spécifiée selon l'heure locale.
Date.prototype.getSeconds()
      Renvoie les secondes (0-59) pour la date spécifiée selon l'heure locale.
Date.prototype.getTime()
      Renvoie la valeur numérique de la date spécifiée sous la forme du nombre de millisecondes
depuis le 1er janvier 1970, 00:00:00 UTC (négativement pour les dates antérieures).
```

Quand plusieurs personnes éditent un même fichier, il est probable d'avoir plusieurs auteurs. Donc plusieurs balise meta avec le champ name="author" (chaque balise ne doit identifier qu'un seul auteur de la page).

- Comment modifier votre code pour qu'il permette de sélectionner le 1er auteur de la liste ?
- Même question avec le dernier auteur de la liste.

7 Exercice 2 : l'objet Date (obligatoire)

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

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

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 clique sur 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 ?

7.1 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. Voici une rapide présentation (mais partielle) des deux fonctions. Cette documentation est extraite du site de Mozilla :

window.setInterval

Appelle une fonction de manière répétée, avec un certain délai fixé entre chaque appel.

```
intervalID = window.setInterval(fonction,delai[,param1,param2, ...]);
intervalID = window.setInterval(code,delai);
```

où

intervalID est un ID unique d'intervalle qui peut être passé à window.clearInterval()
fonction est la fonction qui doit être appelée de manière répétée.
code, dans la syntaxe alternative, est une chaîne représentant le code à exécuter de manière répétée.
delai est le nombre de millisecondes (millièmes de seconde) que setInterval() doit attendre avant chaque appel de fonction.

window.setTimeout

Exécute un morceau de code ou une fonction après un délai déterminé.

```
timeoutID = window.setTimeout(fnct, delai[, param1, param2, ...]);
timeoutID = window.setTimeout(code, delai);
```

où

timeoutID est l'identificateur numerique du timeout, qui peut être utilisé avec window.clearTimeout.
fnct est la fonction que vous désirez exécuter après delai millisecondes.
code est, dans la syntaxe alternative, une chaîne contenant le code à exécuter après delai millisecondes. (L'utilisation de cette syntaxe n'est pas recommandée pour les mêmes raisons que l'utilisation d'eval()).
delai est le nombre de millisecondes (millièmes de seconde) après lequel la fonction doit être appelée. Le délai réel peut s'avérer plus long (cf. doc en anglais).

Une fois que votre horloge fonctionne avec les deux méthodes, ajoutez une balise div avec l'id « grafHorloge ». Cette balise contiendra le texte et les balise img que vous souhaitez.
Vous devez maintenant écrire une fonction majGrafH() qui affiche de manière graphique une horloge dans cette balise. Pour cela vous utiliserez les images de ce lien.

- Laquelle des deux méthodes de window avez-vous utilisé ? Pourquoi ?

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

8 Exercice 3 : HTML, CSS et JavaScript (obligatoire)

8.1 Champ Texte et Couleur d'arrière-plan

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);
}
```

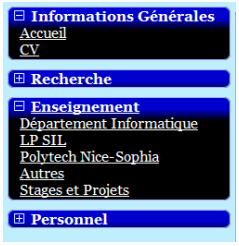
Faites-en sorte que la zone de texte devienne rouge si le texte entré n'est pas un nombre et si l'utilisateur tape un nombre, alors le fond doit devenir vert. Attention, si la zone de texte est vide, elle doit être de couleur blanche.

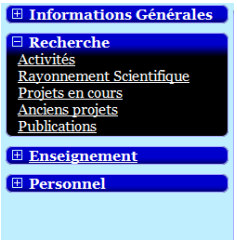
- Quel évènement avez-vous utilisé ?
- Comment avez-vous fait changer la couleur du champ texte ?

8.2 Menu déroulant

Rappel *: la propriété CSS* display, *peut prendre (entre autre) une des valeurs suivantes :*
none: le bloc ne sera pas affiché.
inline: le bloc sera considéré comme étant une seule ligne
block: spécifie un bloc.

A l'aide de cette propriété et d'une petite fonction en JavaScript, réalisez un petit menu déroulant dans le même genre que celui des images ci-dessous. Ce menu sera contenu dans une balise aside. Pensez bien à ajouter les icones plus et moins à votre menu. Attention, les icones devront être ajoutées depuis le Javascript et non dans le HTML.





9 Exercice 4 : parcours de l'arbre DOM (obligatoire)

Pour finir de petit TD, nous allons écrire une fonction « recherche() » qui se déclenchera au clic sur un bouton. Elle ira lire le contenu d'un champ texte et surlignera dans la page tous les mots correspondant au texte de ce champ texte.

Commencez par ajouter un paragraphe qui contiendra le champ texte et le bouton.

Votre fonction recherche devra, si c'est sa première utilisation, faire une sauvegarde de la page (on va dire seulement du corps de celle-ci) dans une variable. Si ce n'est pas la première utilisation, elle restaurera la page des modifications apportées préalablement.

Programmation Web – client riche
M4103 - TD n° 1 (séance 1 et 2)

Ensuite vous allez devoir parcourir le DOM pour recherche le texte souhaité dans toute le page et le remplacer par une balise span. Attention, il ne faut remplacer que le texte et pas les balises HTML. Notre nouvelle balise span sera de la classe CSS « select » et contiendra le texte recherché. La classe CSS « select » devra mettre l'arrière-plan de la balise en jaune.

Pour faire cet exercice, vous aurez besoin d'utiliser les méthodes de gestion des nœuds (cours diapositive n°60), ainsi que les propriétés « childNodes » et nodeType (cours diapositive n°58).

Une fois que cela fonctionne, ajoutez un deuxième champ texte à votre page. A chaque lettre saisie, il appellera une fonction « rechercheInteractive() » dont le but est similaire à la fonction « recherche() » précédente.

10 Conclusion

J'espère que ce premier TD vous a permis de découvrir ou redécouvrir les bases de JavaScript et vous a donné une idée des possibilités qu'offre ce langage. Dans le prochain TD nous explorerons d'autres propriétés, nous développerons des exemples un peu plus complexes et nous irons un peu plus loin dans les subtilités du langage.