LOG4420

Sites web dynamiques et transactionnels

Automne 2012

Contrôle périodique

Enseignant : Michel Gagnon

Cet examen comporte 5 questions

Durée : 2 heures

Pondération : 25% de la note finale

Directives :
Toute documentation permise.
Calculatrice non programmable permise.

1 Programmation en Scala (4 points)

Vous savez qu'en Scala, la fonction *map* permet d'appliquer une fonction à tous les items d'une liste :

```
scala> List(1,2,3,4).map(math.pow(_,2))
res0: List[Double] = List(1.0, 4.0, 9.0, 16.0)
```

Lorsqu'appliqué à une liste, le résultat est une liste de listes. Voici un exemple de code qui ne conserve que les valeurs supérieures à 9 :

```
scala> List(List(10,2,3),List(40,50,6)).map(_.filter(_ > 9))
res1: List[List[Int]] = List(List(10), List(40, 50))
```

On aimerait ici utiliser une fonction *flatMap*, qui prend le résultat et retourne une liste unique :

```
scala> List(List(10,2,3),List(40,50,6)).flatMap(_.filter(_ > 9))
res2: List[List[Int]] = List(10, 40, 50))
```

a) (2 pts) Décrivez le plus brièvement possible ce que fait la fonction suivante (si vous le désirez, vous pouvez fournir un exemple d'application) :

```
def mystere(ls: List[Any]): List[Any] = ls flatMap {
   case ms: List[_] => mystere(ms)
   case e => List(e)
}
```

b) (2 pts) Définissez une fonction *flatMap(Liste, fonction)*, qui prend une liste d'entiers et une fonction, et qui retourne une liste contenant le résultat de l'application de la fonction à tous les entiers de la liste. Par exemple :

```
scala> flatMap(List(List(1,2,3),List(4,5,6)),_.map(_+10))
res0: List[Int] = List(11, 12, 13, 14, 15, 16)
```

Dans votre définition, ne faites pas appel à la méthode prédéfinie *flatMap* de la classe List. Complétez l'ébauche de solution suivante :

```
def flatMap(list:List[List[Int]],fonction:List[Int]=>List[Int]):List[Int] = ...
```

2 CSS (4 points)

Soient les fichiers HTML et CSS illustrés aux figures 1a et 1b. On remarque que le résultat obtenu (figure 1c) est différent de celui désiré (figure 1d) : le premier hyperlien n'est pas affiché correctement.

a) (1 pt) Expliquez pourquoi le résultat obtenu est différent du résultat désiré.

b) (1 pt) Une mauvaise solution consisterait à changer l'ordre des règles dans le fichier CSS. Proposez une meilleure solution.

c) (1 pt) Dans le fichier HTML de la figure 1, dites ce qui sera sélectionné par le sélecteur CSS suivant : `div > p + p`.

d) (1 pt) Nous aimerions que le logo de Poly se trouve dans le coin supérieur droit du bloc de l'article. On utilise donc la règle suivante :

```
#logoPoly {
   position: absolute;
   right: 1%;
   top: 1%;
}
```

Malheureusement, cela ne fonctionne pas. On obtient plutôt le résultat illustré à la figure 2a, alors qu'on devrait avoir celui illustré à la figure 2b. Expliquez ce qui ne fonctionne pas et montrez comment corriger la situation.

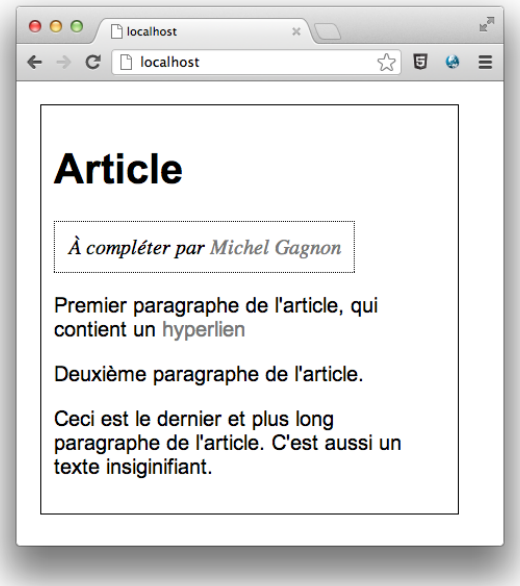
```html
<!doctype html>
<html>
  <head>
    <meta charset="utf8">
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="article">
      <h1>Article</h1>
      <div class="warning"> À compléter par
        <a class="responsable" href="#">Michel Gagnon</a>
      </div>
      <div class="content">
        <p> Premier paragraphe de l'article, qui contient
            un <a href="#"> hyperlien </a> </p>
        <p> Deuxième paragraphe de l'article.  </p>
        <p> Ceci est le dernier et plus long paragraphe
        de l'article.
            C'est aussi un texte insiginifiant.  </p>
      </div>
      <img id="logoPoly" src="logoPoly.png">
    </div>
  </body>
</html>
```

(a) Source HTML

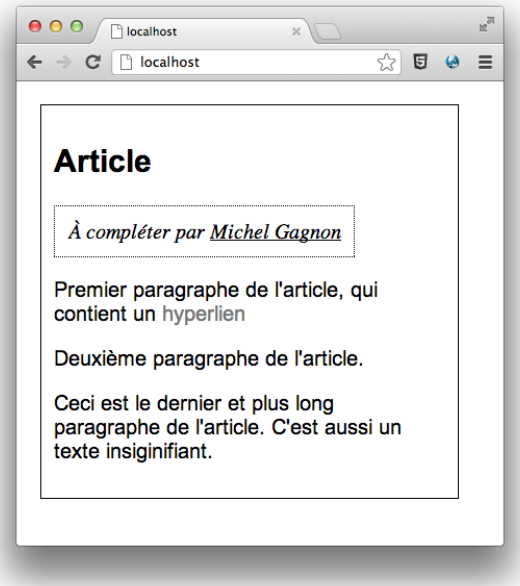
```css
.warning {
    text-align: center;
    border: dotted 1px;
    display: inline-block;
    padding: 10px;
    font-style: italic;
    font-family: times;
}
.warning a {
    text-decoration: underline;
    color: inherit;
}
.article {
    border: solid 1px;
    padding: 10px;
    width: 300px;
    font-family: arial;

}
.article a {
    text-decoration: none;
    color: #999;
}
#logoMini {
display: none;
}
```

(b) Source *style.css*

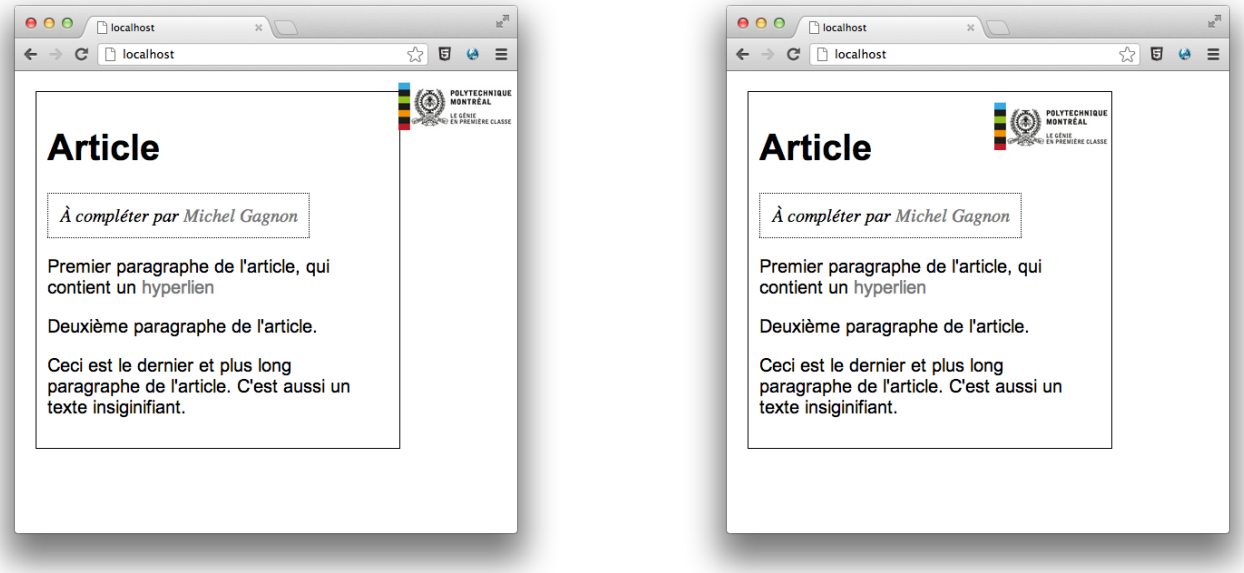


(c) Résultat obtenu



(d) Résultat désiré

FIGURE 1 – Code source et résultats pour la question 2a.



(a) Résultat obtenu

(b) Résultat désiré

FIGURE 2 – Résultats pour la question 2d.

3 HTML 5 (4 points)

a) (2 pts) HTML5 ajoute plusieurs nouveaux éléments dans les formulaires (email, url, number, etc,). Indiquez deux autres nouveautés importantes apportées par HTML5, pour le traitement des formulaires, et expliquez l'avantage de ces nouveautés.

b) (2 pts) Soit le segment de code Javascript suivant :

```
function ajouter(valeur) {
  if (localStorage.total) {
    localStorage.total = parseInt(localStorage.total) + valeur;
  }
  else {
    localStorage.total = valeur;
  }
}
```

Expliquez l'utilité de ce code.

4 HTTP (3 points)

a) (1,5 pt) Soit la réponse suivante retournée par un serveur web :

```
HTTP/1.1 304 Not Modified
Last-Modified: Mon, 24 Jan 2011 21:13:24 GMT
```

Expliquez pourquoi cette réponse a été retournée.

b) (1,5 pt) Dans une réponse à une requête, le serveur moodle.polymtl.ca nous retourne la réponse suivante :

```
HTTP/1.1 200 OK
Date: Mon, 15 Oct 2012 14:24:31 GMT
Set-Cookie: MOODLEID_=%25F3%259; expires=Fri, 14-Dec-2012 14:24:31 GMT; path=/course
```

Expliquez l'impact de cette réponse dans nos prochaines requêtes au serveur moodle.polymtl.ca.

5 DOM et Javascript (5 points)

a) (4 pts) Soit le source HTML suivant :

```
<!doctype html>
<html>
  <head>
    <meta charset="utf8">
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="boite"><img src="logoMini.png"></div>
    <div class="boite"></div>
    <div class="boite"></div>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></script>
    <script src="monscript.js"></script>
  </body>
</html>
```

Le résultat dans le navigateur est tel qu'illustré à la figure 3. Lorsqu'on clique sur un des trois carrés, on veut que l'image se déplace à cet endroit. Écrivez le code javascript qui permet d'obtenir ce comportement. Un aide-mémoire du DOM est fourni en annexe.

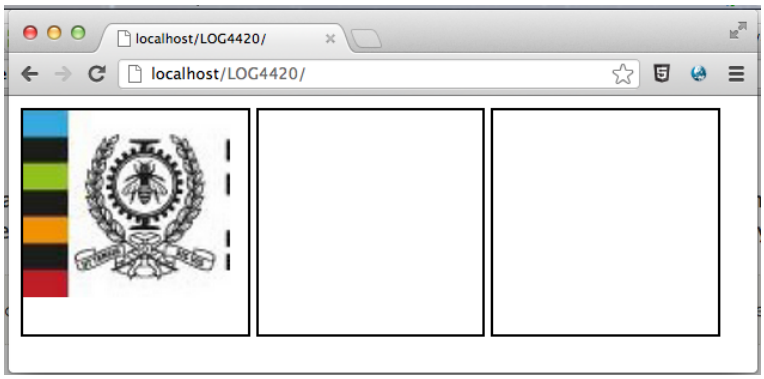


FIGURE 3 – Affichage

b) (1 pt) Soit le code Javascript suivant :

```
function creerObjet(proto) {
   function F() {}
   F.prototype = proto
   return new F()
}

var chien1 = {race: "beagle", age: 23 };
var chien2 = creerObjet(chien1);
chien1.anniversaire = function() { this.age++ }
chien2.anniversaire();
alert(chien2.age);
```

Dites ce qui sera affiché, lors de l'exécution de ce script : 23, 24 ou *undefined*. Expliquez pourquoi.

Annexe - DOM

Propriétés et méthodes de l'object *document* :

getElementById()	Retourne l'élément possédant le ID passé en paramètre.
getElementsByTagName()	Retourne une collection contenant tous les éléments dont la balise correspond à celle passée en paramètre.
getElementsByClassName()	Retourne une collection contenant tous les éléments dont la classe correspond à celle passée en paramètre.
querySelector()	Retourne le premier élément correspondant au sélecteur CSS passé en paramètre.
querySelectorAll()	Retourne une collection contenant tous les éléments correspondant au sélecteur CSS passé en paramètre.
createElement()	Crée un élément correspondant à la balise passée en paramètre.
createTextNode()	Crée un nouveau noeud texte.

Propriétés et méthodes associées à un noeud N du DOM :

innerHTML	Contient tout le contenu HTML inclus dans un noeud.
childNodes	Collection contenant les noeuds fils, incluant les noeuds texte.
children	Collection contenant les noeuds fils, excluant les noeuds texte.
firstChild	Retourne le premier noeud fils
firstElementChild	Retourne le premier fils, en ne considérant pas les noeuds texte.
lastChild	Retourne le dernier noeud fils
lastElementChild	Retourne le dernier fils, en ne considérant pas les noeuds texte.
nextSibling	Retourne le noeud qui a le même parent que N et qui est situé immédiatement après.
nextElementSibling	Retourne le noeud qui a le même parent que N et qui est situé immédiatement après, en ne considérant pas les noeuds texte.
previousSibling	Retourne le noeud qui a le même parent que N et qui est situé immédiatement avant.
previousElementSibling	Retourne le noeud qui a le même parent que N et qui est situé immédiatement avant, en ne considérant pas les noeuds texte.
parentNode	Retourne le noeud parent.
textContent	Retourne tout le contenu textuel d'un noeud (incluant ses descendants).
nodeType	Retourne le type d'un noeud. En particulier : 1 = noeud élément, 2 = noeud attribut, 3 = noeud texte.
style	Permet d'obtenir et de changer le style d'un élément. Exemple : `monElement.style.color = blue`.
className	Permet d'obtenir et de changer la classe d'un élément.
addEventListener(ev,gest,cap)	Associe un gestionnaire d'événement à un événement. Si le troisième argument est $true$ le traitement se fait aussi au cours de la capture (pas seulement lors de la propagation vers le haut (bubbling).
appendChild()	Ajoute le noeud passé en paramètre après tous les noeuds fils de N.
insertBefore(nouv,ref)	Ajoute le noeud $nouv$ juste avant le noeud ref, qui est un noeud fils de N.