Introduction rapide aux pages interactives avec JavaScript

Code: interactive-intro

Originaux

url: **http://tecfa.unige.ch/guides/tie/html/interactive-intro/interactive-intro.html**

url: **http://tecfa.unige.ch/guides/tie/pdf/files/interactive-intro.pdf**

Auteurs et version

- Olivier Clavel, Daniel K. Schneider

- Version: 0.1 (modifié le 23/5/01)

Prérequis

Module technique précédent: html-intro

Autres modules

Module technique suivant: php-intro

Objectifs

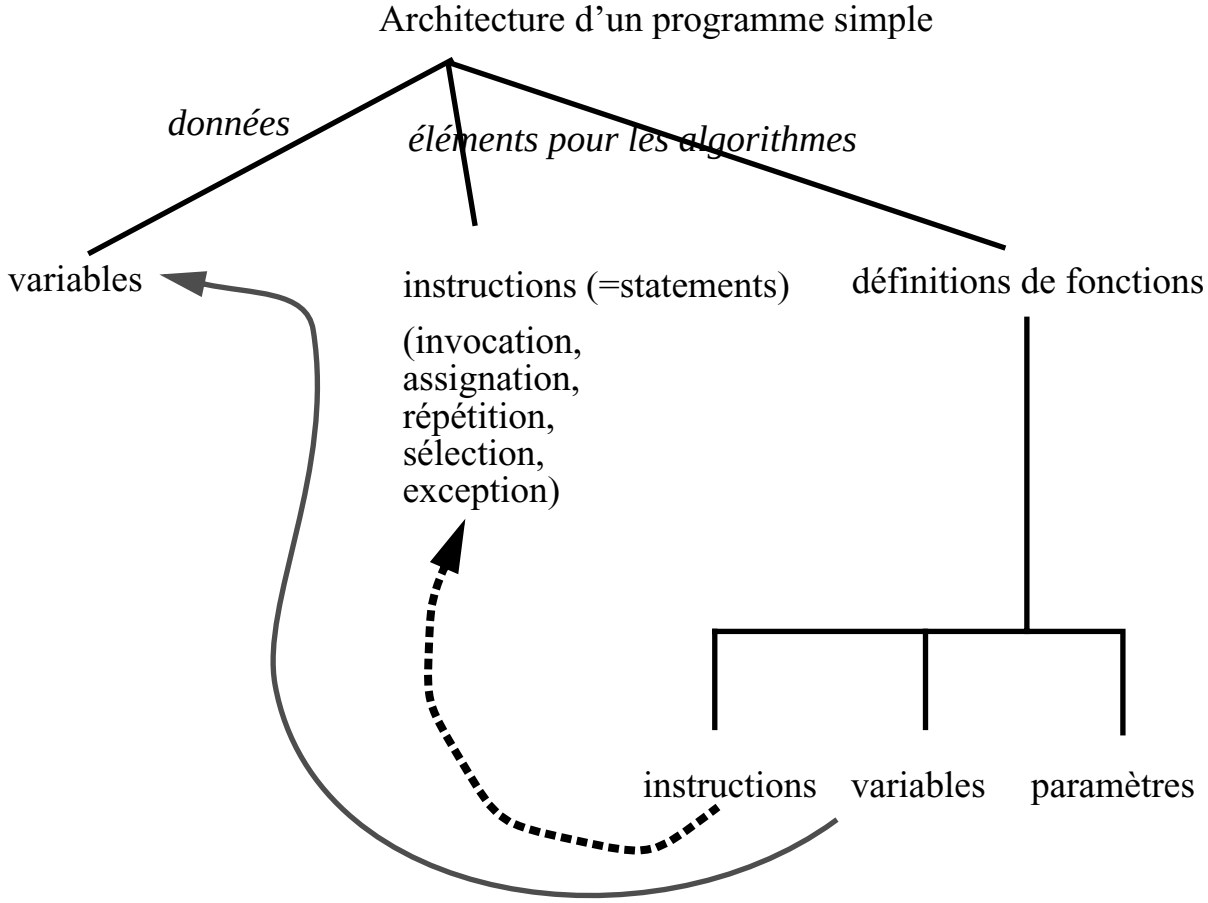
- Comprendre des concepts de programmation de base : variables, fonctions, structures de contrôles...
- Se familiariser avec les formulaires HTML et leur traitement interactif (client-side)
- Se familiariser avec le langage JavaScript

1. Table des matières détaillée

2. Introduction à quelques concepts de programmation

La syntaxe utilisée dans les exemples correspond à celle de JavaScript.

2.1 La notion de programme



Un *programme* est composé d'une série *d'instructions*, parfois regroupées en *fonctions*, qui compose un *algorithme* permettant de traiter des *données*.

Les *données* sont manipulées par le biais de *variables*.

Les *fonctions* peuvent recevoir des *paramètres* (données), effectuent des calculs ou des actions en utilisant des *variables* et peuvent retourner une valeur à la fin de leur exécution. Elles sont liées à une instruction dans le programme ou à un évènement qui survient (l'utilisateur clique sur un bouton par exemple)

L'algorithme contient une ou plusieurs *structures de contrôle* qui permettent au programmeur de décider les calculs et les actions à entreprendre. Ces structures permettent de sélectionner une action en fonction de la valeur d'une variable, de répéter une action de multiples fois...

2.2 Variable

Une variable est une zone de mémoire informatique contenant une valeur pouvant varier au cours de l'éxécution d'un programme.

- Il s'agit d'un "conteneur" auquel on donne un nom (e.g. **`score`**).
- La valeur contenue peut être de différents types

```
entier (integer) : 134
réel (float) : 345.38743
chaîne de caractère (string) : "réponse 3 : Lausane"
```

- Pour donner une valeur à la variable, on fait une *assignation*.

```
score = 34;
On lit de gauche à droite "mettre la valeur 34 dans la variable score"
  (non pas "score égale 34", ce n'est pas un opérateur d'égalité)

feedback = "Vous avez gagné !";
```

- On peut réutiliser la valeur de la variable à tout moment

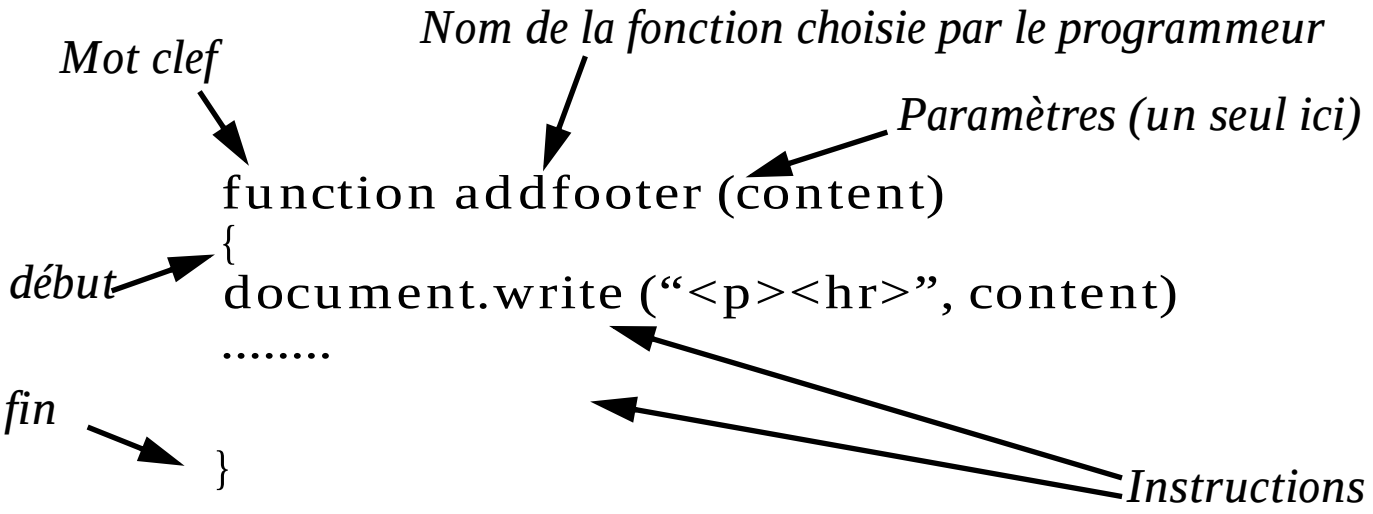
```
write(feedback); (écrit le contenu de la variable feedback)
```

- Avec Javascript, il n'est pas obligatoire (mais fortement conseillé) de déclarer et d'initialiser les variables. Cela permet de s'assurer que chaque variable est bien définie et contient une valeur par défaut avant d'effectuer des opérations avec. Cela fait gagner du temps pour trouver les erreurs dans la phase de mise au point du programme.

```
var score = 0;
```

2.3 Fonction

Ensemble d'instructions identifié par un nom unique permettant d'effectuer une action, un calcul...



- Une fonction peut accepter des paramètres (données)
- Une fois qu'elle est définie, on peut l'appeler depuis n'importe quel endroit du programme, y compris depuis une autre fonction.

```
addfooter("Fait à TECFA");
```

- Si la fonction retourne une valeur on peut la stocker dans une variable par exemple

```
score = calcule(reponse1, reponse2);
```

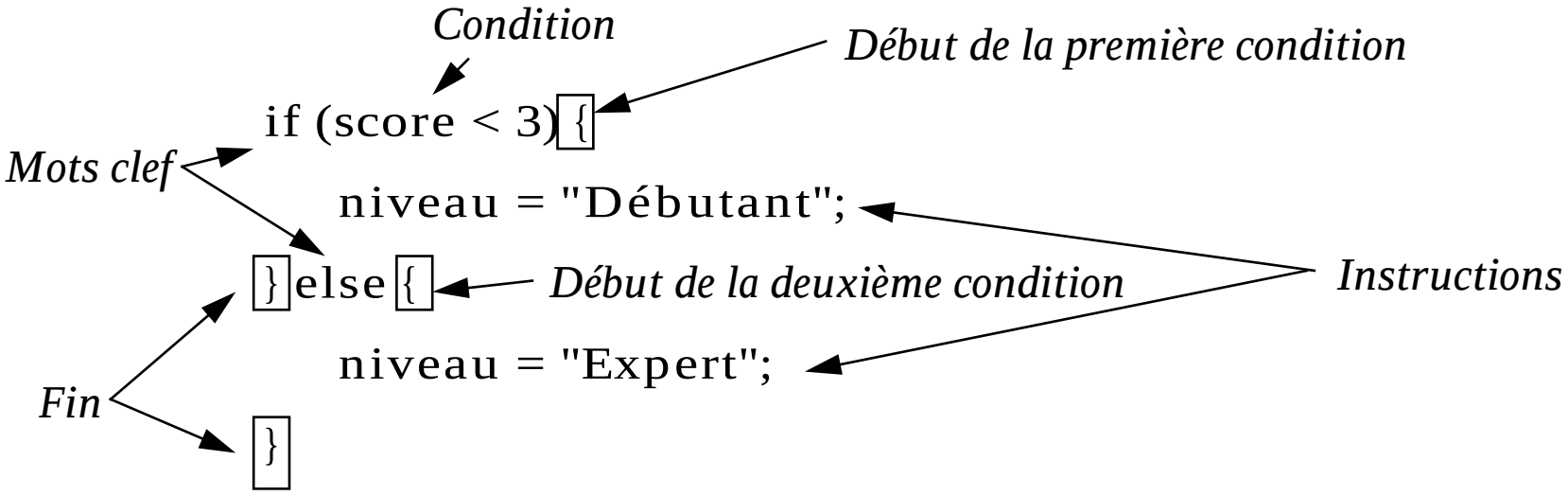
- Les variables et paramètres définis dans une fonction seront local à la fonction (on ne les "voit" pas dans le reste du programme).

2.4 Structures de contrôle

Les structures de contrôle permettent de controler l'éxécution d'un programme en faisant des test sur les valeurs d'une variable. Il existe plusieurs structure de contrôle. Seulement 2 sont présentées ici.

A. La sélection : if... else

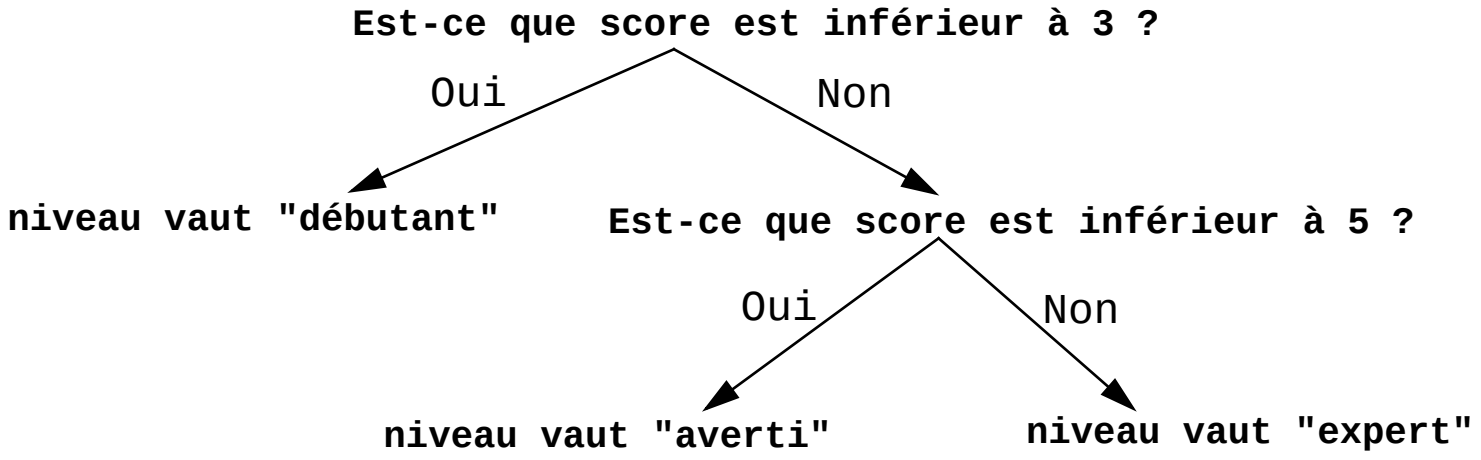
On peut résumer en quelques mots cette structure à "si une condition est vraie alors faire ceci sinon faire cela".



- On peut imbriquer plusieurs sélections : "si la condition est vraie alors faire ceci sinon, si cette autre condition est vraie alors faire ceci, sinon faire cela";

```
if (score < 3) {
    niveau = "débutant";
} else {
    if (score < 5) {
        niveau = "averti";
    } else {
        niveau = "expert";
    }
}
```

Voici l'arbre de décision associé à l'exemple ci-dessus.



B. La boucle incrémentale: for...

```
for (<expression de début>; <condition>; <expression de fin>) {
   expression 1;
   expression 2;
   ...
}
```

- L'*expression de début* est exécutée quand on rentre dans la boucle pour la première fois.
- la *condition* est évaluée à chaque passage dans la boucle. Si elle est vrai, on exécute les expressions de la boucle. Si elle est fausse, on sort de la boucle.
- L'*expression de fin* est exécutée quand toutes les expressions de la boucle on été exécutées (avant le passage suivant dans la boucle).

Exemple : calcul d'un produit factoriel (n! = 1 * 2 *... * n-2 * n-1 * n)

```
var n = 12;
var prodfact = 1;
for (i=1; i<=n; i++) {
    prodfact = prodfact * i;
}
document.write("le produit factoriel de" + n + " vaut " + prodfact);
```

C'est sur, ca va pas nous servir tous les jours :))

3. Introduction à JavaScript

3.1 Origine du langage JavaScript

- JavaScript a été développé par Netscape (d'abord sous le nom LiveScript)
- Il existe une version Microsoft (appelée parfois JScript)
- Le nom "JavaScript" reflète un certain voisinage syntaxique avec JAVA (mais ca n'a rien à voir, c'est une pure raison de marketing)

3.2 Utilisation principale de JavaScript

- Applications interactives (par ex. tests et quiz)
- Pages HTML plus riches (par ex. "highlighting"), DHTML
- Génération de pages HTML selon le profil de l'utilisateur
- Gestion de plugins, versions de Java etc.
- Simples animations
- Vérification de formulaires avant traitement sur un serveur.

3.3 Principe et fonctionnement de JavaScript par l'exemple

Le code JavaScript est exécuté par le navigateur qui recoit la page (client-side) : c'est interactif (pas besoin de faire appel au serveur de nouveau, tout se passe en local).

C'est un langage basé objet qui permet d'interagir avec les éléments qui se trouvent dans le navigateur.

Pour illustrer l'utilisation de ce langage, nous allons travailler avec un petit fomulaire HTML et procéder à son traitement de 2 manière différentes.

A. Formulaire HTML très simple avec des boutons radio

url: http://tecfa.unige.ch/guides/tie/html/interactive-intro/examples/simple-form.html

```
<form action="" method="post" name="questions">
  <H2>Avez-vous déjà écrit des pages HTML avec un éditeur ?</H2>
  <P>
    <INPUT TYPE="radio" NAME="reponse1" VALUE="0"> Jamais
    <INPUT TYPE="radio" NAME="reponse1" VALUE="1"> Parfois
    <INPUT TYPE="radio" NAME="reponse1" VALUE="2"> Souvent
  </P>
  <H2>Avez vous déjà ecrit du code HTML à la main ?</H2>
  <P>
    <INPUT TYPE="radio" NAME="reponse2" VALUE="0"> Jamais
    <INPUT TYPE="radio" NAME="reponse2" VALUE="1"> Parfois
    <INPUT TYPE="radio" NAME="reponse2" VALUE="2"> Souvent
  </P>
  <INPUT TYPE="submit" VALUE="Soumettre">
</FORM>
```

Déclaration du formulaire

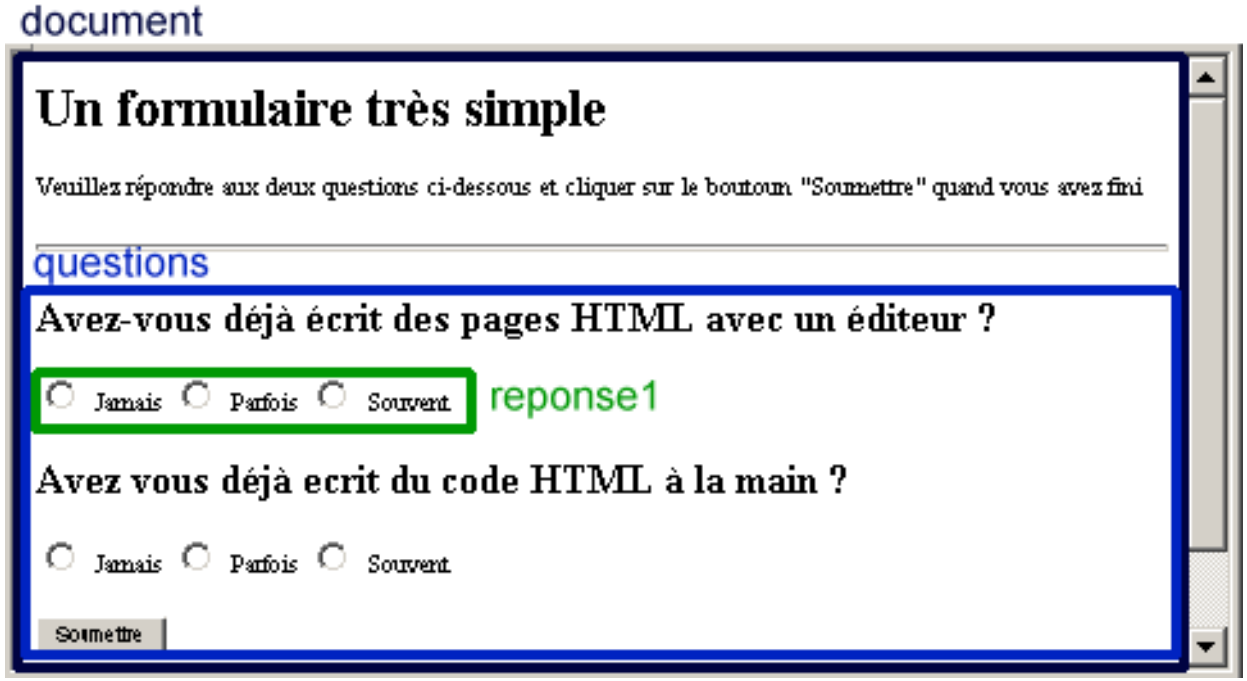
Groupe de boutons radio

Bouton de soummission

Fin du formulaire

- Les champs du formulaire sont inclus entre les balises *<FORM>* et *</FORM>*
- L'attribut *NAME* permet de nommer un élément
- L'attribut *ACTION* donne l'URL de la ressource qui traitera l'information
- L'attribut *VALUE* donne la valeur de l'élément qui sera transmise ou le texte à afficher pour les boutons.

B. Objets, propriétés et méthodes



Chaque élément de la page correspond à un objet qui s'insère dans un arbre hiérarchique. "*les boutons radio réponse1 sont dans le formulaire questions qui se trouve dans le document par défaut*". On nomme l'objet en écrivant sa hiérarchie :

```
document.questions.reponse1
```

Voici quelques exemple pour accéder à des propriétés ou a des méthodes :

```
soumettre le formulaire questions (méthode):
document.question.submit();

mettre le nombre d'éléments du formulaires questions dans une variable
(propriété):
elements = document.questions.length;

envoyer un popup d'alerte avec un bouton "ok" (méthode) :
alert("attention, vous devez remplir le champ nom")
```

C. Les évènements

JavaScript permet de faire réagir le programmes à des évènement qui se passent dans le navigateur. On donne alors une instruction (exécuter une fonction, modifier une variable, changer une propriété d'un objet...) lorsque cet évènement survient.

Exemples d'évènements : click sur un bouton, chargement de la page, soumision d'un formulaire, entrée du curseur de la souris sur un objet...

D. Traitement du formulaire I : gestion des évènements

url: http://tecfa.unige.ch/guides/tie/html/interactive-intro/examples/action-form.html
url: (code) http://tecfa.unige.ch/guides/tie/html/interactive-intro/examples/action-form.txt

On va ici traiter chaque click que fait l'utilisateur sur les boutons radio avant de soumettre le formulaire. On donnera le résultat dans une boite d'alerte.

On commence par déclarer une variable pour chaque réponse :

```
var rep1 = null;
var rep2 = null;
```

Initialiser les variables à la valeur *null* permet de tester si l'utilisateur a bien répondu aux questions avant de soumettre.

Pour chaque bouton radio, on rajoute une option *onClick* qui permet d'éxécuter une instruction à chaque fois que le bouton en question est cliqué.

```
<INPUT TYPE="radio" NAME="reponse1" VALUE="0" onClick="rep1=0">
```

Ensuite, on écrit une fonction qui sera appelée lorsque le formulaire est soumis :

```
function displayScore (rep1, rep2) {
     var score = 0;
     if (rep1 == null || rep2 == null) {
          alert("Il faut répondre aux questions avant de soumettre");
     } else {
          score = rep1 + rep2;
          if (score == 4) {
               alert("Vous êtes un expert");
          } else {
               if (score >= 2) {
            alert("Vous avez une certaine maîtrise de HTML");
           } else {
            alert("Vous êtes un novice, continuez à vous entrainer");
           }
        }
     }
}
```

Enfin, on indique au formulaire d'utiliser cette fonction JavaScript lors de la soumission

```
<form method="post" name="questions" action="javascript:displayScore(rep1,rep2)">
```

E. Traitement du formulaire II : intérrogation des valeurs

url: http://tecfa.unige.ch/guides/tie/html/interactive-intro/examples/value-form.html
url: (code) http://tecfa.unige.ch/guides/tie/html/interactive-intro/examples/value-form.txt

Cet exemple et plus efficace au niveau de la programmation mais plus difficile à comprendre pour des débutants. Voici la fonction *displayScore* modifiée et commentée:

```
function displayScore () {

  // petit artifice pour eviter un message d'erreur quand on recommence sans recharger la page.
  document.questions.submit.checked = false;

  // on déclare la variable score et on initialise à 0
  var score = 0;

  // on regarde combien d'élément il y a dans le formulaire
  elements = document.questions.length;

  // on fait ensuite un boucle sur le nombre d'éléments pour les passer en revue un par un.
  var i;
  for (i = 0; i < elements; i++) {
    // on regarde si l'élément est "checked". Si oui, on calcule le score
    if (document.questions.elements[i].checked) {
   score = score + eval(document.questions.elements[i].value);
    }
  }
  // feedback
  ...
```

Le traitement pour le feedback est le même que dans l'exemple précédent.