

Cours de programmation

Les instructions de base :

Les boucles :

JavaScript permet de créer des blocs et de les nommer. Ce nom peut être ensuite utilisé par les instructions *break* et *continue* pour leur indiquer où continuer l'exécution du programme.

• Exemple :

```
NomBloc : {  
  instruction 1 ;  
  instruction 2 ;  
}
```

L'instruction *while* permet de répéter l'exécution d'un bloc tant que la condition passée en paramètre est vérifiée (C'est-à-dire correspond à la valeur *true*). Si la condition est testée après l'exécution du bloc d'instructions, alors le bloc d'instructions sera au moins exécuté une fois. Il est possible de tester la condition avant l'exécution du bloc. Dans ce cas, il se peut que le bloc d'instructions ne soit jamais exécuté.

• Exemple :

```
do {  
  instruction 1 ;  
  instruction 2 ;  
} while (a < b )
```

• Exemple :

```
while (a < b ) {  
  instruction 1 ;  
  instruction 2 ;  
}
```

Vous pouvez créer une boucle qui se répète à l'infini en créant une condition qui est toujours vérifiée.

• Exemple :

```
while (true) {  
  instruction 1 ;  
  instruction 2 ;  
}
```

Comme l'instruction *while*, l'instruction *for* est utilisée pour répéter l'exécution d'une série de commandes. Cependant, cette instruction est plus intéressante car elle permet en outre de préciser une commande à

exécuter avant la première itération, une commande à exécuter après chaque première itération et enfin la condition qui doit être vérifiée avant d'entamer une nouvelle itération. Dans la majorité des cas, l'instruction *for* est utilisée lorsqu'on désire avoir un entier qui balaye toutes les valeurs entre deux bornes. Il faut noter que chacun des paramètres de l'instruction *for* est optionnel.

• Exemple :

```
for (i=5; i<=10; i++) {  
instruction 1 ;  
instruction 2 ;  
}
```

L'instruction *for* peut être associée au mot *in*. Cette instruction permet d'avoir une variable qui balaye l'ensemble des index d'un tableau ou l'ensemble des propriétés d'un objet selon qu'on lui passe en paramètre un tableau ou un objet. En fait, la notion de propriété et d'index d'un tableau est très proche.

Les interruptions de boucles :

Il est possible d'interrompre une boucle créée à l'aide de l'instruction *while* ou *for*. En effet, si vous utilisez l'instruction *break*, la boucle est immédiatement stoppée et on passe aux instructions suivant le bloc. Par contre si vous utilisez l'instruction *continue*, toutes les instructions du bloc qui suivent sont ignorées et on recommence la série d'instruction du bloc (si bien sûr la condition est toujours vérifiée).

Il est possible de nommer les boucles afin de préciser quelle boucle est considérée lors de l'utilisation de l'instruction *continue*.

• Exemple :

```
Ligne:  
for (i=5; i<=10; i++) {  
instruction 1 ;  
instruction 2 ;  
}  
continue Ligne;
```

Les fonctions :

Nous avons déjà vu comment déclarer de nouvelles fonctions. Mais nous allons approfondir la question en utilisant le passage de paramètres. Pour cela, il suffit d'utiliser la structure décrite dans l'exemple suivant. On peut faire en sorte qu'une fonction retourne une valeur en utilisant l'instruction **return**.

• Exemple :

```
function Prgm1()  
{  
var num1=45;  
Prgm2(num1,'Texte');  
}
```

```
function Prgm2(num2,chaine1)
{
// A ce moment, on a num1=num2 et chaine1='Texte'

return 3;
}
```

Les instructions conditionnelles :

La première instruction conditionnelle que nous allons voir est la plus simple, bien que les autres ne soient pas compliquées.

• Exemple :

```
var c = a;
if (a < b)
var c = b;
c = c * 2;
```

A l'aide de cet exemple, on constate qu'au début "c" est égal à "a". Ensuite, si la propriété "a<b" est vérifiée, alors "c" est égal à "b". Par conséquent, "c" contient la valeur la plus grande entre "a" et "b". Enfin, la valeur "c" est multipliée par 2 quoiqu'il arrive. Si on veut exécuter un bloc d'instructions après la condition, il faut utiliser les accolades.

Nous pouvons compliquer la fonction en utilisant l'instruction *else*. Celle-ci permet d'introduire des instructions à exécuter dans le cas où la condition n'est pas vérifiée. Les deux exemples suivants produisent exactement le même résultat que le précédent.

• Exemple :

```
if (a < b) {
    var c = b;
}
else {
    var c = a;
}

c = c * 2;
```

• Exemple :

```
if (a < b) {
    var c = b;
    c = c * 2;
}
else {
```

```
var c = a;  
c = c * 2;  
}
```

Enfin, nous avons l'instruction *switch* qui permet de sélectionner un bloc d'instructions à exécuter en fonction de la valeur d'une expression passée en paramètre. On associe à chaque valeur un bloc d'instructions à l'aide du mot *case*. Toutes les valeurs non traitées provoquent l'exécution du bloc nommé portant le nom *default*. La valeur peut être une valeur numérique ou une chaîne de caractères.

• Exemple :

```
switch (valeur) {  
  case 1 :  
    instruction1  
    instruction2  
    break  
  
  case 2 :  
    instruction1  
    instruction2  
    break  
  
  case 'abc' :  
    instruction1  
    instruction2  
    break  
  
  default :  
    instruction1  
    instruction2  
}
```



Copyright © 2000–2001 ProgWeb Tous droits réservés
Dernière mise à jour : 22/10/2001

