

Présentation de la syntaxe

La structure conditionnelle en JavaScript est très proche syntaxiquement de celle vue précédemment en langage descriptif algorithmique.

Tout à fait classiquement le bloc d'instructions à exécuter dans le cas où la condition testée est vraie est délimité par un jeu d'accolades ({}). Il est aussi possible de prévoir une séquence d'instructions alternative avec le mot clé `else`. Cette séquence sera aussi encadrée par des accolades.

Le langage JavaScript est assez permissif quant au positionnement de ces accolades. Ainsi vous rencontrerez dans les scripts les constructions suivantes :

```
if (condition)
{
    Actions_1;
}
else
{
    Actions_2;
}
```

ou


```
if (condition) {
    Actions_1;
}
else {
    Actions_2;
}
```

ou

```
if (condition) { Actions_1; } else { Actions_2; }
```

avec :

- condition représentant un test de comparaison générant un résultat booléen `true` ou `false`,
- `Actions_1` et `Actions_2` représentant des séquences d'instructions (en général réparties sur plusieurs lignes).

 Attention pour effectuer un test de comparaison en égalité l'opérateur est le double égal (`==`) à ne pas confondre avec le simple égal (`=`) qui sert à effectuer les comparaisons.

Dans les tests de comparaison avec des constantes on préférera par exemple :

```
if (5 == compteur)
{
    Actions_1;
}
```

```
}  
else  
{  
    Actions_2;  
}
```

à

```
if (compteur == 5)  
{  
    Actions_1;  
}  
else  
{  
    Actions_2;  
}
```

car l'étourderie consistant à confondre le `==` avec le `=` est plus facile à détecter dans le premier cas, JavaScript renvoyant une erreur. Dans le second cas (si vous mettez un `=` au lieu d'un `==`) la séquence `Actions_1` sera systématiquement exécutée car l'évaluation de `compteur = 5` donnerait toujours `true` (JavaScript estimant que l'affectation est réussie).