JAVASCRIPT: LES CONDITIONS

En JavaScript, il existe ce que l'on appelle des « structures de contrôle », elles permettent de contrôler les instructions en fonction de l'évolution du programme. Il en existe 2 sortes, les conditions et les boucles. Pour l'instant, nous allons nous pencher sur les conditions.

Une condition est un test que le programme va effectuer, et nous allons indiquer au programme ce qu'il doit faire si cette condition est vraie et ce qu'il doit faire si elle est fausse.

Un exemple :

Si l'utilisateur a plus de 18 ans, on va lui dire qu'il est majeur, si ça n'est pas le cas, on lui dira qu'il est mineur :

```
//on demande son age à l'utilisateur
let age = window.prompt("Quel âge avez-vous ?");
console.log(age);

//on parse l'age pour pouvoir le considérer comme un nombre
age = parseInt(age);

//on effectue les tests
if(age === 18){
    console.log("Vous êtes majeur");
}
else{
    console.log("Vous êtes mineur");
}
```

Le soucis ici est que à 19 ans, le programme nous dira que nous sommes mineurs. Il va donc falloir utiliser un autre opérateur.

LES OPERATEURS

Jouons à « Vrai ou faux ».

« Il pleut aujourd'hui » –> Faux !

« 5 est égal à 7 » ->Faux !

« 10 est plus grand que 2 » -> Vrai !

Nous allons pouvoir utiliser des opérateurs de comparaison, ils nous retourneront si la compraison est vrai ou fausse.

Voici les différents opérateurs possibles pour effectuer une comparaison entre 2 variables:

Opérateur	Signification
==	Egal à
!=	Différent de
===	Strictement égal
!==	Strictement différent
>	Supérieur
>=	Supérieur ou égal
<	Inférieur
<=	Inférieur ou égal

Testons nos exemples:

```
console.log(5 === 7);
```

La console nous retourne :

```
false
```

```
console.log(10>2);
```

```
true
```

IF...ELSE

Il s'agit d'une structure conditionnelle qui va permettre de tester une condition et de réaliser des instructions différentes selon qu'elle est vraie ou fausse.

Reprenons notre exemple avec l'âge :

```
if(age >= 18){
    console.log("Vous êtes majeur");
}
else{
    console.log("Vous êtes mineur");
}
```

Voilà qui est plus cohérent.

Le programme effectuera les instructions présentes entre les accolades du « if » si la condition présente dans la parenthèse retourne « true », sinon elle effectuera les instructions entre les accolades du « else ».

Le « else » n'est pas obligatoire, il est tout à fait possible de ne pas le mettre, tout est selon les besoins du programme :

```
if(age >= 18){
    console.log("Vous êtes majeur");
}
```

Dans ce cas, le programme n'agira que si la condition est à « true » mais ne fera rien si elle est « false ».

Il est également possible d'enchaîner les « if » :

```
if(age >= 100){
    console.log("Vous êtes centenaire");
}
else if(age >=18){
    console.log("Vous êtes majeur");
}
else{
    console.log("Vous êtes mineur");
}
```

Il est possible de les enchaîner autant de fois que l'on en a besoin sachant dans ce cas que si le programme rentre dans un des « if » , il ne rentrera dans aucun autre, puisque les autres se trouvent dans les « else ».

CONDITIONS MULTIPLES

Les conditions présentes dans les structures de contrôle peuvent être multiples, c'est-à-dire qu'il peut y en avoir plusieurs.

C'est là qu'interviennent les expressions ET et OU.

ET OU &&

Un exemple, nous souhaitons dire aux personnes qui ont plus de 100 ans et qui s'appellent « Raymond » : « Bonjour Raymond » :

Nous rajoutons la variables « prenom » :

```
let prenom = "Raymond";
if(age >= 100 && prenom === "Raymond"){
    console.log("Bonjour Raymond");
}
```

Le && signifie ET. Il faut alors que les 2 conditions soient « true » pour que les instructions présentes dans les accolades s'exécutent.

OU OU ||

```
let prenom = "Raymond";
if(age >= 100 || prenom === "Raymond"){
    console.log("Bonjour Raymond");
```

```
}
```

|| signifie OU, il faut alors qu'une des conditions et seulement une soit respectée pour que les instructions soient exécutées.

SWITCH

Switch permet de comparer la valeur d'une variable avec une multitude de valeurs possibles et d'agir en fonction. Elle est très pratique lorsqu'une variable doit être comparer à plusieurs valeurs. Par contre elle est à proscrire lorsqe l'on veut utiliser des « inférieurs » ou « supérieurs », elle ne va tester que l'égalité.

Exemple:

```
let nombre = window.prompt("Choisissez un nombre enter 1 et8");
nombre = parseInt(nombre);

switch(nombre){
    case 1:
        console.log("Vous gagnez un stylo");
        break;
    case 2:
        console.log("Vous gagnez un voyage");
        break;
    case 3:
        console.log("Vous gagnez une montre");
        break;
    case 4:
        console.log("Vous gagnez un ballon");
        break;
    default:
        console.log("Vous n'avez pas saisi le bon nombre");
        break;
}
```

Chaque valeur possible se trouve avec un « case », ici on compare notre variable « nombre » à des chiffres donc il n'y a pas de guillemets, mais si on comparait la variable à une chaîne de caractères, il faudrait bien sûr l'entourer de guillemets :

```
let prenom = window.prompt("Vous vous appelez ?");

switch(prenom){
    case "Pierre":
        console.log("Bonjour Pierre");
        break;
    case "Paul":
        console.log("Bonjour Paul");
        break;
    default:
        console.log("Bonjour inconnu");
        break;
}
```

Le « default » sert à définir une action si aucune des valeurs des « case » ne correspond au contenu de la variable. Il est optionnel.

Il est également possible de définir une même action pour plusieurs valeurs différentes :

```
switch(nombre){
    case 1: case 5:
        console.log("Vous gagnez un stylo");
        break;
    case 2: case 6:
        console.log("Vous gagnez un voyage");
        break;
    case 3: case 7:
        console.log("Vous gagnez une montre");
        break;
    case 4: case 8:
        console.log("Vous gagnez un ballon");
        break;
    default:
        console.log("Vous n'avez pas saisi le bon nombre");
        break;
}
```

Il suffit de les mettre à la suite.