Feuille de triche HTML / CSS / Javascript

1 Javascript

1.1 Syntaxe

Les espaces ne sont pas pris en compte. Les tabulations permettent d'améliorer la lisibilité, mais ne sont pas importantes. Ces deux lignes sont équivalentes. :

```
var X =   1;
var    X     = 1;
```

Commentaires avec "//" : `// ce que je veux ici`

1.2 Variables et types de base

Déclaration de la variable "X" : `var X = 1;`
Les variables ont un type :

```
typeof(2) // number (entier ou nombre à virgule)
typeof("2") // string (chaîne de caractères)
```

Les types sont importants et changent les calculs :

```
var foo = 1;
var bar = '2';
console.log(foo + bar);  // résultat : 12 (oups)
// voilà comment faire correctement
var foo = 1;
var bar = Number('2'); // on change '2' en entier
console.log(foo + bar);  // résultat : 3 (/o/)
```

1.3 Comparaisons et conditions

On peut tester la valeur d'une variable avec "==", "!=", "<", ">" :

```
1 + 1 == 2; // true
2 > 3 ; // false
3.0*2 >= 6; // true
```

Une comparaison est une variable de type booléen :

```
(1 + 1 != 2) == false; // true (oulala)
```

Une condition permet d'exécuter du code selon la valeur d'une comparaison :

```
if(1 + 1 == 3){
        // ne s'exécute jamais
}
```

```
else if (1 + 1 != 2){
        // ne s'exécute pas non plus
}
else {
        // s'exécute si les autres comparaisons sont fausses
}
```

1.4 Boucles

Les boucles servent à effectuer une action plusieurs fois. Elles sont gouvernées par un booléen. Utiliser les boucles for quand on sait l'avance le nombre d'itérations. :

```
var res = 0 // on veut calculer la somme de 1 à 11
for(var i=1; i < 11; i = i + 1){
        res = res + i;
} // res vaut 1 + 2 + ... + 10 = 55
```

Des fois, on ne sait pas combien d'itérations sont nécessaires pour arriver au résultat. On utilise les boucles while :

```
var valeur = 500; res = 0;// on cherche res * 7 = 500;
while(res * 7 < valeur){
        res += 1;
} // res = 72// évidemment, on peut utiliser la division...
```

Attention, les boucles peuvent ne jamais rencontrer le critères d'arrêt, on parle alors de boucles infinies...

1.5 Fonctions

Les fonctions servent à isoler des morceaux de code, il y a quelque chose en entrée, on fait quelque chose avec. Pour appeler une fonction, il suffit d'écrire son nom et de mettre des paramètres entre par
Utilisation de fonction :

```
console.log("quelque chose"); // écrit
Math.random(); // tire un nombre au hasard
Math.floor(2.3); // renvoie la valeur tronquée (2 ici)
alert("quelque chose"); // ouvre une popup
prompt("Nom :"); // ouvre une popup qui demande un nom
```

Déclaration de fonction :

```
function le_nom_que_lon_veut(param1, param2) {
    // ce que l'on veut
    return param1 + param2 // par exemple
}
```

2 HTML

2.1 Balises

Les balises permettent de structurer le document HTML, chacune à une particularité décrite dans le standard HTML5.
Détail d'une balise :

- chevron ouvrant "<", un nom, des attributs optionnels, et un chevron fermant ">"
- balise "normales" : `<article>...</article>`
- balise "auto fermantes" : `<input ... />`
- ajout d'une classe : `<input class="ma_classe" />`
- ajout d'un identifiant : `<input id="mon_id" />`

Dans notre cas, nous n'utiliserons quasiment que des balises normales. Les classes et identifiants permettent de les manipuler plus facilement.
Exemple : `<article>` ou encore : `<img src="monImage.png" alt="">`
Liste complète des balises HTML :

- `https://www.w3schools.com/tags/ref_byfunc.asp`
- `http://www.simplehtmlguide.com/cheatsheet.php`

2.2 Structure de base

page HTML minimale :

```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <title>Document</title>
        <script>console.log("Du JS ici");</script>
        <style>Du css ici</style>
</head>
<body>
        <!-- contenu ici -->
</body>
</html>
```

2.3 Séparations de sections

Titres : `<h1></h1>` , `<h2></h2>` Bloc de contenu : `<div></div>` Paragraphe : `<p></p>`

2.4 Interaction

bouton :

```
<button onclick="fonction_a_apeller()" type="button">
        Text
</button>
```

champs de texte : `<input type="text"/>`

3 CSS