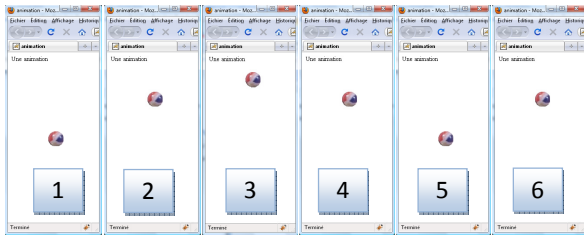


## Un exercice d'initiation à la programmation

### Description de l'exercice

Dans cet exercice, nous allons vous montrer comment faire une animation, sur une page web. Nous allons écrire une page sur laquelle une image se déplace.

Voici des copies d'écran de l'animation : la balle (1) se déplace d'abord vers le haut de la page (2), jusqu'à une position maximale (3), puis descend (4) jusqu'à une position minimale (5), remonte (6), et ainsi de suite.



Dans les pages qui suivent, nous vous indiquons les différentes étapes pour réaliser l'animation ci-dessus.

[1]

<http://iti.epfl.ch>



[2]

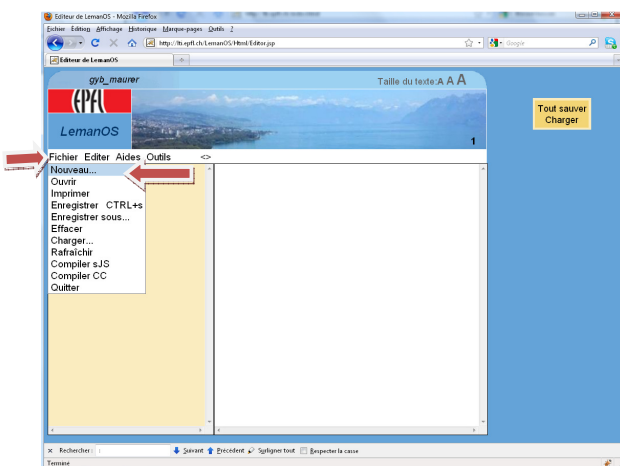
### Etape 2 : Créer un nouveau fichier sur LemanoS

Vous êtes maintenant sur l'environnement de développement LemanoS.

Nous allons commencer par créer un nouveau fichier : menu Fichier -> Nouveau

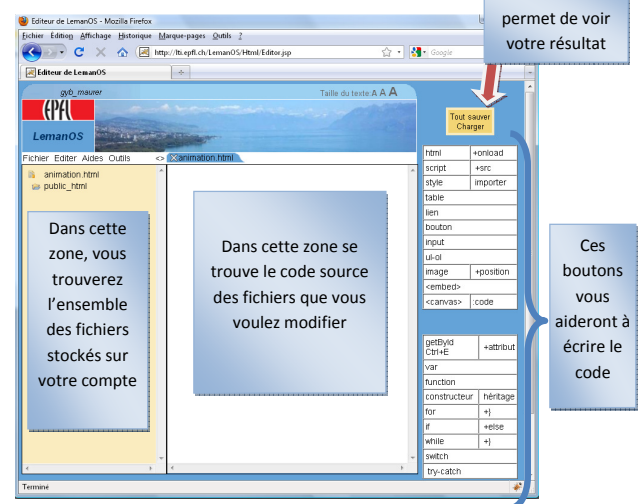
Donner, par exemple, le nom *animation.html* à votre fichier

Attention : s'assurer que le nom de votre fichier se termine bien par *.html*



[3]

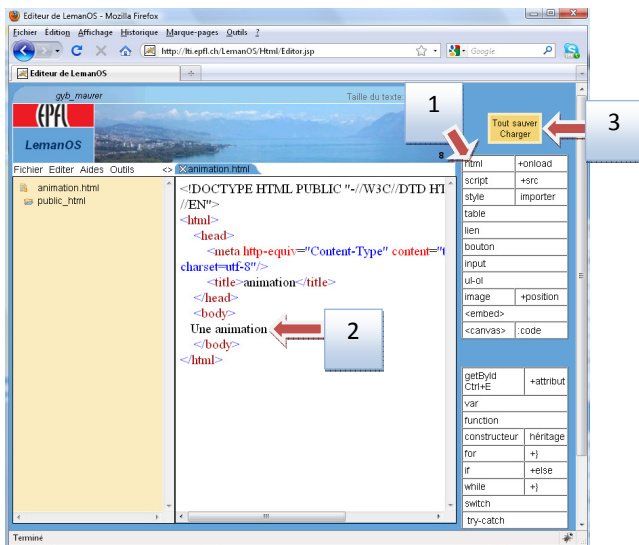
### Etape 3 : Les différentes parties de LemanoS



[4]

#### Etape 4 : Créer la structure d'une page HTML

1. Un cliquant sur le bouton html, la structure d'une page web est créée
2. Pour que du texte apparaisse sur la page, il suffit de l'écrire entre la balise <body> et la balise </body>
3. Pour voir votre résultat, cliquez sur le bouton **Tout sauver Charger**



[5]

#### Etape 5 : Insérer une image

Nous allons maintenant ajouter une image à notre page.

Complétons donc notre fichier, en y ajoutant une balise <img> :

```
<body>
Une animation 
</body>
```

#### Explications

- La balise <img> permet d'afficher une image.
- La balise <img> possède un attribut src qui permet de dire quelle image afficher.
- A l'adresse <http://lti.epfl.ch/images/bille.gif> se trouve l'image de la bille qui sera affichée sur notre page

#### Etape 6 : Positionner une image

Pour définir la position d'une image, on complète notre code ainsi :

```
<body>
Une animation

</body>
```

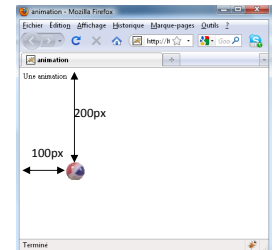
Attention à bien respecter la syntaxe. En particulier, prendre garde à ne pas inverser les deux-points et les points-virgules. Il y a 3 instructions, contenant des deux-points

- position :absolute
- top :200px
- left :100px

Les instructions sont séparées par des points-virgules.

En cliquant sur le bouton « tout sauver charger », vous devriez obtenir le résultat ci-contre.

Maintenant, l'image n'est plus positionnée juste à côté du texte, mais à 200 pixels du haut de la fenêtre et à 100 pixels de la gauche de la fenêtre.



[6]

#### Etape 7 : Ecrire un programme

Pour écrire un programme, on écrit une balise <script> et </script> à l'intérieur de la page, comme dans l'exemple ci-dessous.

Entre la balise <script> et la balise </script>, on écrit les instructions du programme. L'instruction que nous avons écrite est `alert('une animation')`.

Cliquez sur le bouton « tout sauver charger » pour voir quelle est l'effet de cette instruction

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>animation</title>
<script>
alert('une animation')
</script>
</head>
<body>
Une animation

</body>
</html>
```

#### Etape 8 : Une fonction

Modifions le programme ainsi :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>animation</title>
<script>
function maFonction(){
alert('une animation')
}
</script>
</head>
<body>
Une animation

</body>
</html>
```

Vous constaterez que la fenêtre surgissante n'apparaît plus. L'instruction `alert('une animation')` n'est donc plus exécutée.

[7]

#### Etape 9 : Appel d'une fonction

Nous voulons que l'instruction `alert('une animation')` soit exécutée lorsque l'utilisateur clique sur l'image.

Il faut donc **appeler** la fonction `maFonction()` lorsque l'utilisateur clique sur l'image.

Voici les instructions à écrire pour appeler une fonction lorsque l'utilisateur clique sur l'image :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>animation</title>
<script>
function maFonction(){
alert('une animation')
}
</script>
</head>
<body>
Une animation

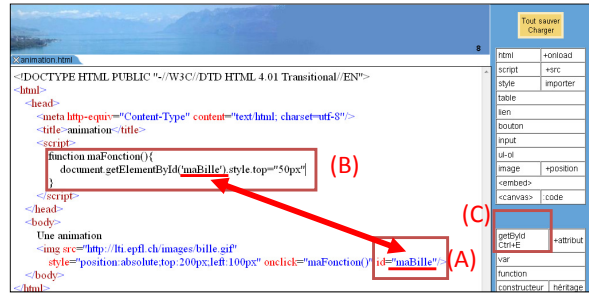
</body>
</html>
```

Charger la page. En cliquant sur la bille, la fenêtre surgissante apparaîtra.

[8]

## Etape 10 : Modifier les attributs de l'image

Plutôt que d'afficher une fenêtre surgissante lorsque l'utilisateur clique sur l'image, nous aimerions déplacer cette dernière vers le haut.



1. Ajouter un identifiant (id) à l'image (A)
2. Modifier la fonction *maFonction()*, pour comme dans l'exemple ci-dessus (B). Le bouton *getElementById* vous facilitera l'écriture de l'instruction *document.getElementById(...)*

Charger la page et cliquez sur l'image pour voir le résultat

## Etape 11 : Une variable pour changer la position de l'image

```
<script>
var pos=200
function maFonction(){
  pos=pos-10
  document.getElementById('bille').style.top=pos+"px"
}
</script>
```

[9]

## Etape 12 : Appeler la fonction régulièrement

On écrivant *onclick="setInterval(maFonction,100)"*, plutôt que *onclick="maFonction()"*, nous avons une animation. Charger la page et cliquez sur l'image pour voir le résultat

```
Une animation

```

## Etape 13 : Tester la position de la bille

Probablement avez-vous remarqué que la bille finit par sortir de la fenêtre. Nous aimerions éviter ceci, en arrêtant la bille à une certaine position.

Pour ce faire, on va faire un test (*if*). En ajoutant le test *if(pos>50)*, l'instruction *pos=pos-10*, nous sera exécutée que si la valeur de *pos* est plus grande que 50.

```
<script>
var pos=200
function maFonction(){
  if(pos>50){
    pos=pos-10
  }
  document.getElementById('bille').style.top=pos+"px"
}
</script>
```

## Etape 14 : Changer la direction de la bille

Plutôt que d'arrêter la bille à la position 50, on va lui faire changer de direction. Modifiez votre programme ainsi :

```
<script>
var pos=200
var delta=10
function maFonction(){
  if(pos<50){
    delta=-10
  }
  pos=pos-delta
  document.getElementById('bille').style.top=pos+"px"
}
</script>
```

**Explications :** nous avons introduit une nouvelle variable, nommée *delta*. Cette variable vaut 10 au départ. La bille va donc monter. Lorsque la position de la bille est inférieure à 50 pixels du haut de la fenêtre, *delta* prend la valeur -10, et la bille va redescendre.

[10]

## Etape 15 : Un deuxième test

Pour que la bille alterne entre une position de 50 pixels et une position de 300 pixels, on ajoute un deuxième test : lorsque la bille atteint une position de 300 pixels, on attribue à nouveau 10 à *delta*.

```
<script>
var pos=200
var delta=10
function maFonction(){
  if(pos<50){
    delta=-10
  }
  if(pos>300){
    delta=10
  }
  pos=pos-delta
  document.getElementById('bille').style.top=pos+"px"
}
</script>
```

## Etape 16 (facultative) :

### Une écriture plus compacte pour le même programme

Il est possible d'écrire le même programme, mais de façon plus compacte.

Dans le programme précédent, nous avons deux tests séparés :

- `if(pos<50)`
- `if(pos>300)`

Il est possible de ne faire qu'un test, sur deux conditions :

- `if(pos>50 || pos>300)`

Les doubles barres (||) indiquent un **OU** logique : le test est vrai, si l'une, ou l'autre, ou les deux conditions sont vérifiées.

```
<script>
var pos=200
var delta=10
function maFonction(){
  if(pos<50 || pos>300)
    delta=-delta
  pos=pos-delta
  document.getElementById('bille').style.top=pos+"px"
}
</script>
```

[11]