

Animations ++

ADRAR / **PÔLE NUMERIQUE**

- > **INFRASTRUCTURES** SYSTÈMES & RÉSEAUX
- > **CYBERSÉCURITÉ** INFRASTRUCTURES & APPLICATIONS
- > **DEVOPS / SCRIPTING** & AUTOMATISATION
- > **DEVELOPPEMENT** WEB & MOBILE
- > **TRANSFORMATION NUMERIQUE DES ENTREPRISES**

www.adrar-numerique.com

Animations ++: Transitions

Les transitions permettent de faire des animations simples en passant d'un état A à un état B;

Pour faire cela, nous allons utiliser deux propriétés :

- transition-property => Permet de choisir quelle(s) propriété(s) CSS va être animée
- transition-duration => Permet de choisir le temps sur lequel se déroule l'animation

Animations ++: Transitions

Prenons par exemple une `<div>` avec les paramètres suivants :

```

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>TRANSFORMATION</title>
    <link rel="stylesheet" type="text/css" href="app.css">
  </head>

  <body>
    <div></div>
  </body>
</html>
  
```

```

div
{
  width: 200px;
  height: 150px;
  background-color: blue;
}

div:hover
{
  background-color: red;
}
  
```



Appliquons nos deux propriétés précédentes :

```

div
{
  width: 200px;
  height: 150px;
  background-color: blue;
  transition-property: background-color;
  transition-duration: 1s;
}

div:hover
{
  background-color: red;
}
  
```

Animations ++: Transitions

Lorsque notre curseur passera sur la <div>, il y aura une transition entre le « background-color: blue » et le « background-color: red » de une seconde :

```
div
{
  width: 200px;
  height: 150px;
  background-color: blue;
  transition-property: background-color;
  transition-duration: 1s;
}

div:hover
{
  background-color: red;
}
```

Note : La valeur de durée peut être en milliseconde (ms).

Animations ++: Transitions

Vous pouvez appliquer la transition sur plusieurs propriétés en même temps, en les séparant par des virgules :

```
div
{
  transition-property: background-color, width, height;
}
```

Animations ++: Transitions

Nous pouvons aussi ajouter une propriété supplémentaire à nos transitions, la propriété « transition-delay » qui comme son nom l'indique, permet de jouer une animation, mais avec un délai au démarrage :

```
div
{
  width: 200px;
  height: 150px;
  background-color: blue;
  transition-property: background-color;
  transition-duration: 1s;
  transition-delay: 0.4s;
}

div:hover
{
  background-color: red;
}
```

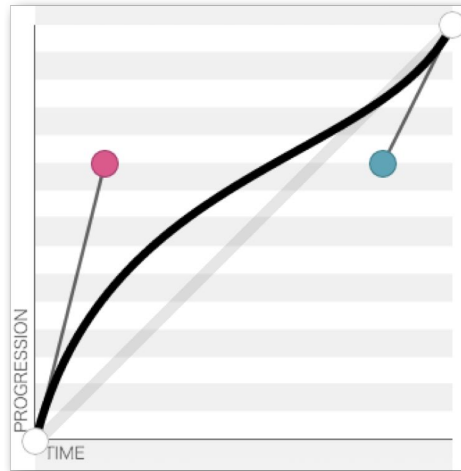
Animations ++: Transitions

Ajoutons encore plus à notre transition, avec la propriété
« transition-timing-function » qui permet de choisir l'effet qu'aura la transition :

- Ease : rapide au début et lent à la fin.
- Linear : vitesse régulière.
- ease-in : lent au début et de plus en plus rapide.
- ease-out : rapide au début, et de plus en plus lent.
- ease-in-out : lent au départ et à la fin, rapide entre.

Animations ++: Transitions

Toujours avec la propriété « transition-timing-function », nous pouvons aussi créer notre propre fonction à l'aide d'une fonction cubic-bezier().



Lien vers le site : <https://cubic-bezier.com/>

Animations ++: Transitions

Nous pouvons aussi utiliser la super-propriété « transition » qui permet, du coup, de réunir toutes les propriétés précédentes en une seule :

```
div
{
  width: 200px;
  height: 150px;
  background-color: blue;
  transition: background-color 6s ease;
}
```

Et pour optimiser encore plus, nous pouvons combiner les super-propriétés :

```
div
{
  width: 200px;
  height: 150px;
  background-color: blue;
  transition: background-color 2s ease, width 1s linear;
}

div:hover
{
  background-color: red;
  width: 500px;
}
```

Animations ++: Mise en pratique (20 min)

- Dans un nouveau répertoire, créez un fichier HTML avec le HTML minimum et un fichier CSS. Liez-les ensemble.
- Ajoutez un titre et un paragraphe avec du texte.
- Ajoutez un effet sur le titre pour qu'il passe de la couleur rouge au vert au survol de la souris, l'animation durera 1s et sera linéaire.
- Ajoutez un effet de rotation au clic sur le paragraphe (2 tours en 2s), de manière linéaire.
- Testez les autres accélérations (ease, ease-in,...).

Animations ++: Keyframes

Rentrons maintenant dans les animations beaucoup plus complexes. Pour cela nous allons utiliser les « keyframes ». Contrairement aux transitions, qui n'effectuent une action que d'un état A à un état B, les « keyframes » permettent de définir autant d'étapes que voulu.

Voici un exemple de ce qu'on peut faire uniquement en HTML/CSS, avec des animations :

<https://codepen.io/tadywankenobi/pen/QbWNGR/>

Animations ++: Keyframes

En premier, il va nous falloir créer notre animation, étape par étape. Pour cela, nous allons les définir dans la règle « @keyframes » :

```
@keyframes masuperanimation {  
  0% {  
    transform: translateX(0px);  
  }  
  50% {  
    transform: translateX(150px);  
  }  
  100% {  
    transform: translateX(150px) rotate(30turn);  
  }  
}
```

Cette animation, qui porte le nom de « masuperanimation », nom que l'on peut bien entendu changer à sa guise, se présente en plusieurs étapes :

Animations ++: Keyframes

En premier, à 0% de l'avancé, nous donnons l'instruction à l'élément de ne pas bouger (une translation de 0px)

En second, à 50% de l'avancé, nous donnons l'instruction à l'élément de bouger de 150px vers la droite

En dernier, à 100% de l'avancé, nous donnons l'instruction à l'élément de bouger de 150px ET de faire une rotation de 30 tours

```
@keyframes masuperanimation {  
  0% {  
    transform: translateX(0px);  
  }  
  50% {  
    transform: translateX(150px);  
  }  
  100% {  
    transform: translateX(150px) rotate(30turn);  
  }  
}
```

Animations ++: Keyframes

Enfin, pour appliquer cette animation à notre élément, il suffit simplement d'utiliser la super-propriété « animation » et de mettre en valeur le nom de notre animation en plus de sa durée :

```
@keyframes masuperanimation {  
  0% {  
    transform: translateX(0px);  
  }  
  50% {  
    transform: translateX(150px);  
  }  
  100% {  
    transform: translateX(150px) rotate(30turn);  
  }  
}  
  
div  
{  
  margin: 100px;  
  width: 200px;  
  height: 150px;  
  background-color: #008;  
}  
  
div:hover  
{  
  animation: masuperanimation 2s;  
}
```

Animations ++: Keyframes

Qui dit super-propriété, dit qu'elle est composée de plusieurs propriétés, dont voici une liste non-exhaustive :

- Animation-name : nom de l'animation.
- Animation-duration : durée d'exécution de l'animation.
- Animation-delay : délai avant l'exécution de l'animation.
- Animation-timing-function : pattern d'accélération.
- Animation-iteration-count : nombre de répétitions de l'animation (la valeur « infinite » existe pour permettre à l'animation de tourner en boucle).
- Animation-fill-mode : permet d'indiquer l'état final de l'animation (« forwards » pour qu'il reste à sa dernière étape ou « backwards » pour qu'il revienne dans son état d'origine).
- Animation-direction : permet de choisir le sens de lecture de l'animation.

Animations ++: Mise en pratique (40 min)

- Reprenez l'exercice précédent.
- Ajoutez trois boutons dans une `<div>` à la suite du paragraphe.
- Ajoutez une animation de translation pour le titre en partant d'une position négative jusqu'à 0 en 2s (pour donner un effet d'arrivée sur la page).
- Ajoutez une animation sur le paragraphe pour qu'il se « déplie » en modifiant sa taille (width & height) en allant de 0 à 100% en 1 seconde.
- Au survol du curseur, ajoutez :
 - Sur le bouton 1, une rotation de 0 à 10 degrés puis à -10 degrés en 0,2s en « infinite » et « alternate » pour que l'animation tourne en boucle de manière fluide (donne un effet de tremblement).
 - Sur le bouton 2, une translation de 0 à -10px sur l'axe Y, en 0,2s en « infinite » et « alternate » pour que l'animation tourne en boucle de manière fluide (donne un effet de sursaut).
 - Sur le bouton 3, un agrandissement de 1 à 1,3 fois, en 0,5s de manière linéaire en « infinite » et « alternate » pour que l'animation tourne en boucle de manière fluide (donne un effet de battement).