

# Intégration Web

MMI 1 – TP#3 S2

Danielo **JEAN-LOUIS**  
Michele **LINARDI**

# CSS Transform

- Permet d'appliquer des transformations sur une balise
  - rotation, translation, mise à l'échelle... via des fonctions définies
    - voir second lien des sources

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform>
- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function>

# CSS Transform

- **Ce n'est pas un moteur physique**
  - Ex : Appliquer une translation ne va déplacer les éléments aux alentours
- Ne fonctionne pas sur toutes les balises (voir source encart "Attention")

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform>

# CSS Transform – Exemple de code

```
transform: scale(2, 0.5);
```

L'élément va voir son échelle être modifiée :

- 200 % en largeur
- 50 % en longueur

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform>

# CSS Transform – Exemple de code

```
transform: scale(2, 0.5) rotate(270deg);
```

transformation 1

transformation 2

Attention l'ordre des transformations ont leur importance  
(Les transformations sont séparées par des espaces)

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform>

# Point accessibilité – CSS Transform

- Proposez une version sans animation (voir sources)
  - Des utilisateurs peuvent avoir des maux de tête à cause des transformations

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/@media/prefers-reduced-motion>
- <https://www.w3.org/WAI/WCAG21/Understanding/animation-from-interactions> - anglais

# Point accessibilité – CSS Transform

- Le code suivant désactive les animations pour les utilisateurs qui n'en veulent pas

Pas besoin de recopier, ce code est dans la ressource du TP. Et la source est le troisième lien parmi les sources ci-dessous

```
@media (prefers-reduced-motion: reduce) {  
  *, ::before, ::after {  
    animation-delay: -1ms !important;  
    animation-duration: 1ms !important;  
    animation-iteration-count: 1 !important;  
    background-attachment: initial !important;  
    scroll-behavior: auto !important;  
    transition-duration: 0s !important;  
    transition-delay: 0s !important;  
  }  
}
```

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/@media/prefers-reduced-motion>
- <https://www.w3.org/WAI/WCAG21/Understanding/animation-from-interactions> - anglais
- <https://web.dev/prefers-reduced-motion/#conclusions> - anglais



# CSS Transform

- Les transformations, par défaut, sont calculées depuis le centre de la balise
  - Possibilité de déplacer ce point d'origine
    - Propriété "transform-origin"

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform>

# Pratiquons ! - CSS Transform (Partie 1)

Pré-requis :

- Avoir la ressource ressources/transform

A télécharger ici : A MODIFIER

Connectez-vous avec votre compte Teams/Office 365 pour télécharger la ressource

# Ça fonctionne mais...




Notre texte n'est pas très visible (contraste)

**Que pouvons-nous faire pour corriger ce problème ?**

**Et si nous ajoutions un fond, un arrière-plan... sans  
rajouter de nouvelles balises**

# Pseudo-éléments

- Permet d'ajouter du contenu sans pour autant ajouter de nouvelles balises
- Syntaxe




```
mon-sélecteur::pseudo-element {  
    /* propriétés CSS */  
}
```

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

# Pseudo-éléments



```
mon-sélecteur::pseudo-element {  
    /* propriétés CSS */  
}
```

Remarquez la présence de deux deux-points pour signaler un pseudo-élément.

Si vous n'en mettez qu'un ce n'est pas grave.

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

# Pseudo-éléments

- Ne fonctionnent pas sur toutes les balises
  - Exemple : `<img>` ou `<br>`
- Acceptent toutes les propriétés CSS

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>



# Pseudo-éléments

- Le contenu des pseudo-éléments **n'apparaît pas** dans l'onglet "Inspecteur" du navigateur
  - On dit que l'élément est "émulé"
- Les pseudo-éléments `::before` et `::after` sont ceux que vous allez utiliser le plus

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

# Pseudo-éléments – ::before / ::after

- Permettent d'ajouter des éléments avant ou après le contenu ciblé
  - Les deux pseudo-éléments peuvent être appliqués en même temps
- N'existent pas pour les lecteurs d'écran
  - Ne pas placer de données critiques dedans

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

# Pseudo-éléments – ::before / ::after

- **Nécessite la propriété "content"** pour s'afficher. Contenu pouvant être :
  - Une chaîne de caractères (qui peut être vide)
  - Une image (non-redimensionnable)
  - Un compteur
  - ...
- S'affiche en ligne ("inline") par défaut

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>


# Pseudo-éléments

- Il est possible de combiner pseudo-éléments et pseudo-classe au sein du même sélecteur

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

# Pseudo-éléments



```
mon-sélecteur:hover::before {  
    /* propriétés CSS */  
}
```

Ici on cible au survol (:hover), le pseudo-élément ::before.  
Autrement dit : on fait apparaître un pseudo-élément au survol

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

# Pseudo-éléments

- Par convention, on écrit les pseudo-éléments (et pseudo-classes) **après** les sélecteurs simples

```
mon-sélecteur {  
    /* propriétés CSS */  
}  
  
mon-sélecteur::pseudo-classe {  
    /* propriétés CSS */  
}  
  
mon-sélecteur::pseudo-élément {  
    /* propriétés CSS */  
}
```

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

# Pratiquons ! - CSS Transform (Partie 2/3/4)

Pré-requis :

- Avoir la ressource ressources/transform

A télécharger ici : A MODIFIER

Connectez-vous avec votre compte Teams/Office 365 pour télécharger la ressource

**Questions ?**



