

# Intégration Web

MMI - Travaux Pratiques #4 S1

# Danielo JEAN-LOUIS

## Développeur front-end

# But du cours

- Voir les bases de la programmation d'un site web
  - Langage HTML
  - Langage CSS
- Sensibilisation au web : design et programmation
  - Accessibilité
  - Bonnes pratiques
- Avoir les connaissances pour développer un site web

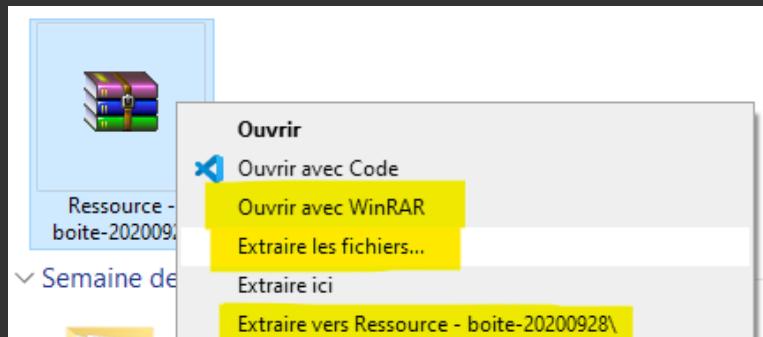
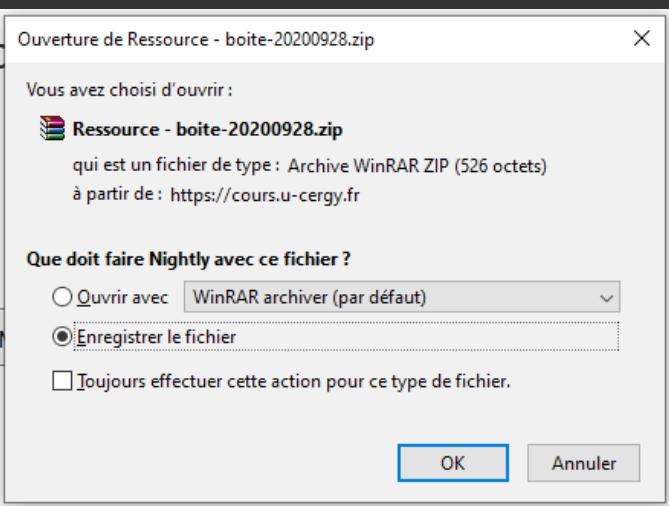
# Récapitulatif du tp précédent

- Les pseudo-formats et pseudo-éléments
- Les transitions css
- La propriété css "transform" et ses fonctions

## Source :

- <https://github.com/DanYellow/presentations/blob/master/mmi/html/travaux-pratiques/numero-3/presentation.pdf>

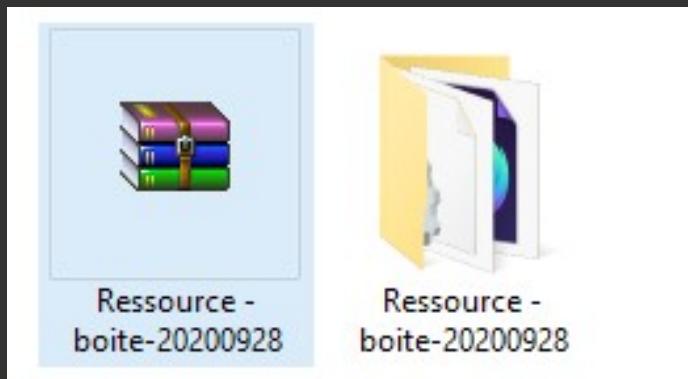
# Petit point sur les ressources de cours



- Les ressources se présentent sous la forme d'une archive une fois téléchargées
- L'archive doit être impérativement extraite sinon vous ne serez pas capables de réaliser correctement les tp

Menu contextuel sur l'archive (clic droit).  
Choisir un des choix surligné.

# Petit point sur les ressources de cours



- Une fois extrait, vous devez obtenir un dossier avec le contenu de l'archive
- Libre à vous de travailler dans le dossier OU en copier le contenu pour le coller ailleurs
- **En aucun vous devez travailler dans l'archive**

# Ressource du cours

**Travaux Pratiques #4 > "Ressource - Responsive" sur  
ENT**

2007

# iPhone



- Annoncé en janvier 2007
- Sort en juin de la même année
- Premier vrai smartphone
- Premier vrai navigateur mobile  
(Safari mobile)
  - Vrais débuts du web mobile

iPhone 11 Pro Max

# Smartphone / tablette

- Points à prendre en compte sur ces appareils :
  - Écran plus petits
  - Interface tactile → nouvelles interactions :
    - zoom / double toucher / *swipe*
  - Pas de clic droit (enfin à l'époque)
  - La mobilité (l'utilisateur n'est pas tout le temps en wi-fi)
    - Mode hors-ligne
  - Orientation de l'écran
  - Représente la majorité de la fréquentation web
  - Définition élevée de nos jours
  - Notifications et Push
  - [...]

Source :

- <https://developer.apple.com/design/human-interface-guidelines/ios/user-interaction/gestures/> - anglais

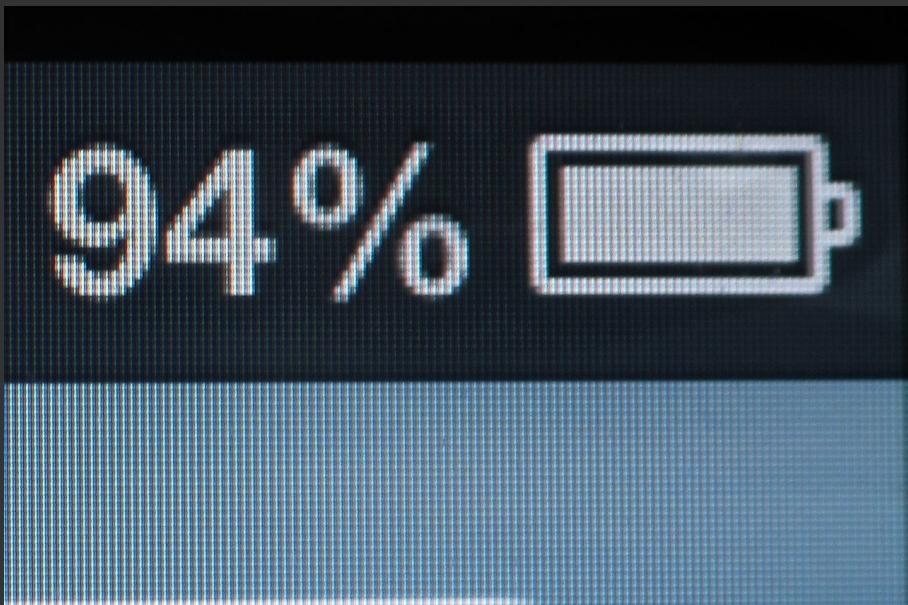
# Point technique – Définition / résolution

- La définition représente le nombre de pixels (ou points) qu'un écran peut afficher
  - Exemple : une image 3000 x 2000 a une définition 6 millions de pixels
- La résolution représente la densité de pixels. Autrement dit le nombre pixels par pouce (1 pouce = 2,54cm)
  - Plus la densité est élevée, plus les traits sont détaillés. Retina est le terme utilisé par Apple pour désigner les écrans haute résolution (apparition avec l'iPhone 4)

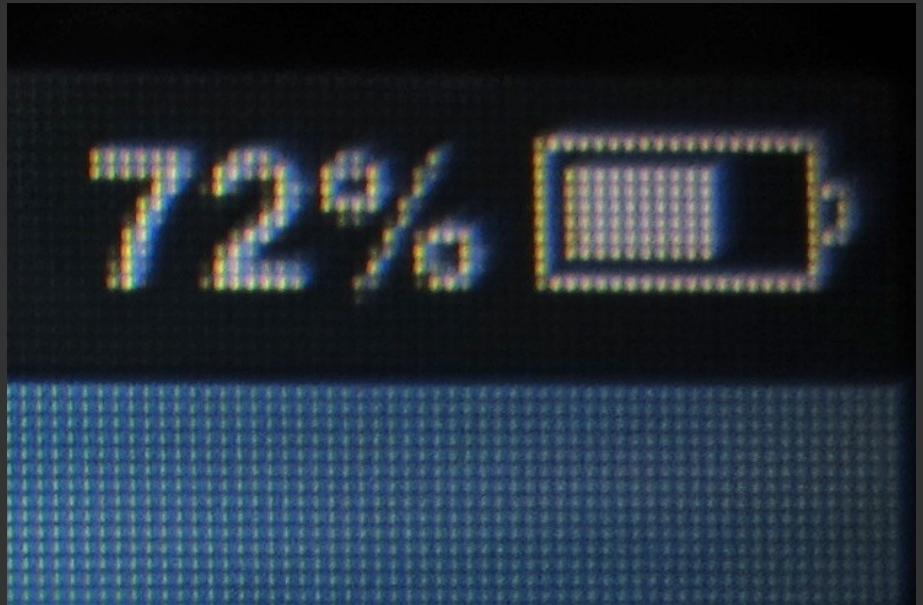
## Sources :

- [https://fr.wikipedia.org/wiki/D%C3%A9finition\\_d%C2%7C%C3%A9cran](https://fr.wikipedia.org/wiki/D%C3%A9finition_d%C2%7C%C3%A9cran)
- [https://www.frandroid.com/comment-faire/comment-fonctionne-la-technologie/342762\\_difference-entre-definition-resolution](https://www.frandroid.com/comment-faire/comment-fonctionne-la-technologie/342762_difference-entre-definition-resolution)
- <https://www.paintcodeapp.com/news/iphone-6-screens-demystified>

# Petit point technique – Définition / résolution



Affichage Retina  
(source wikipedia)



Affichage non Retina  
(source wikipedia)

## Sources :

- [https://fr.wikipedia.org/wiki/D%C3%A9finition\\_d%C2%A7%C3%A9cran](https://fr.wikipedia.org/wiki/D%C3%A9finition_d%C2%A7%C3%A9cran)
- [https://www.frandroid.com/comment-faire/comment-fonctionne-la-technologie/342762\\_difference-entre-definition-resolution](https://www.frandroid.com/comment-faire/comment-fonctionne-la-technologie/342762_difference-entre-definition-resolution)
- [https://fr.wikipedia.org/wiki/%C3%89cran\\_Retina](https://fr.wikipedia.org/wiki/%C3%89cran_Retina)
- <https://www.paintcodeapp.com/news/iphone-6-screens-demystified>

# Responsive design

- Terme inventé par Ethan Marcotte en 2010
- Désigne Concept, une approche du web :
  - Site fluide (le site se comporte comme un liquide), voir sources
  - Images fluides
  - Utilisation des media queries
  - **Ce n'est pas une technologie**

## Sources :

- <https://alistapart.com/article/responsive-web-design/> - en
- <https://mdn.github.io/css-examples/learn/rwd/liquid-width.html> – Site fluide
- [https://developer.mozilla.org/fr/docs/Apprendre/CSS/CSS\\_layout/Responsive\\_Design](https://developer.mozilla.org/fr/docs/Apprendre/CSS/CSS_layout/Responsive_Design)
- <http://lehollandaisvolant.net/tuto/responsive-css>
- <https://www.alsacreations.com/article/lire/1615-cest-quoi-le-responsive-web-design.html>

# Un peu de vocabulaire

- Responsive design : vu précédemment
- Fixed design : le web "ancien", le site (et ses éléments) ont une taille définie et si ça ne rentre pas... tant pis
- Adaptive design : forme évoluée du fixed design, les designers proposent des designs propres à "chaque écran" et les développeurs s'arrangent
- Fluid design : Le site et ses éléments ont des tailles en pourcentage

## Sources :

- <https://medium.com/@popart.studio/fluid-vs-adaptive-vs-responsive-design-62de51e036bd> – en
- <https://www.alsacreations.com/article/lire/1615-cest-quoi-le-responsive-web-design.html>



**Responsive Web Design  
est le mot magique**

# Avant le Responsive design – Version mobile

- Nécessite un serveur pour la gestion
- Double charge de travail :
  - version PC et mobile à gérer
- Techniques de détection de mobile pas fiable
- Utilisation du javascript
  - Lourdeur potentielle des scripts

## Sources :

- <https://alistapart.com/article/responsive-web-design/> - en
- <https://mdn.github.io/css-examples/learn/rwd/liquid-width.html> – Site fluide
- [https://developer.mozilla.org/fr/docs/Apprendre/CSS/CSS\\_layout/Responsive\\_Design](https://developer.mozilla.org/fr/docs/Apprendre/CSS/CSS_layout/Responsive_Design)
- <http://lehollandaisvolant.net/tuto/responsive-css>
- <https://www.alsacreations.com/article/lire/1615-cest-quoi-le-responsive-web-design.html>

# Images fluides oui, optimisation avant tout

```
img {  
  max-width: 100 %;  
  height: auto;  
}
```

Ces deux lignes rendent vos images responsives, elles gardent également leurs proportions

- Ces deux déclarations rendent nos images "fluides" tout en gardant leur proportion
- Elles ne dépasseront jamais la largeur de l'écran
- Pour autant, ces images ne sont pas forcément optimisées

## Sources :

- <http://lehollandaisvolant.net/tuto/responsive-css/#astuce-image>

# Point accessibilité – Images

- Les mobiles utilisent souvent la data pour accéder à Internet Mettre des images de 15 Mo (même très belle) consomme beaucoup de données
  - Limiter la taille des images. Préférer des images qui font 3 Mo ou moins
- Formats à utiliser :
  - jp(e)g / png / .gif
  - webp : format avec compression sans perte par Google
  - AVIF : format prometteur avec compression sans perte
    - Non géré par tous les navigateurs à l'heure actuelle (10/2020)
  - svg : pour les logos – format vectoriel

## Sources :

- <https://jakearchibald.com/2020/avif-has-landed/> - en
- <https://fr.wikipedia.org/wiki/WebP>
- <https://squoosh.app/> - pour compresser vos images

# Point accessibilité – Images

- jpeg : format d'image avec **compression avec perte**
  - Extension : .jpg ou .jp(e)g
- png : format d'image avec **compression sans perte**
  - Extension : .png
  - Gère la transparence
- gif : format d'image avec **compression avec perte**
  - Extension : .gif
  - Gère les animations
  - Limité à 256 couleurs → Inadapté pour vos photos de vacances

## Sources :

- <https://graphilink.fr/jpg-png-gif-differences/>

# La tête dans le responsive

```
<meta name="viewport" content="width=device-width,initial-scale=1">
```

- Balise meta indispensable pour le responsive design
- Notre meta en détails :
  - width=device-width : Indique au navigateur que le site doit avoir une largeur égale à celle de l'écran
  - initial-scale=1 : Définit le zoom par défaut du site
- Accepte la valeur "user-scalable=no". Ne jamais l'utiliser, elle empêche l'utilisateur de zoomer

## Sources :

- [https://developer.mozilla.org/fr/docs/Apprendre/CSS/CSS\\_layout/Responsive\\_Design](https://developer.mozilla.org/fr/docs/Apprendre/CSS/CSS_layout/Responsive_Design)

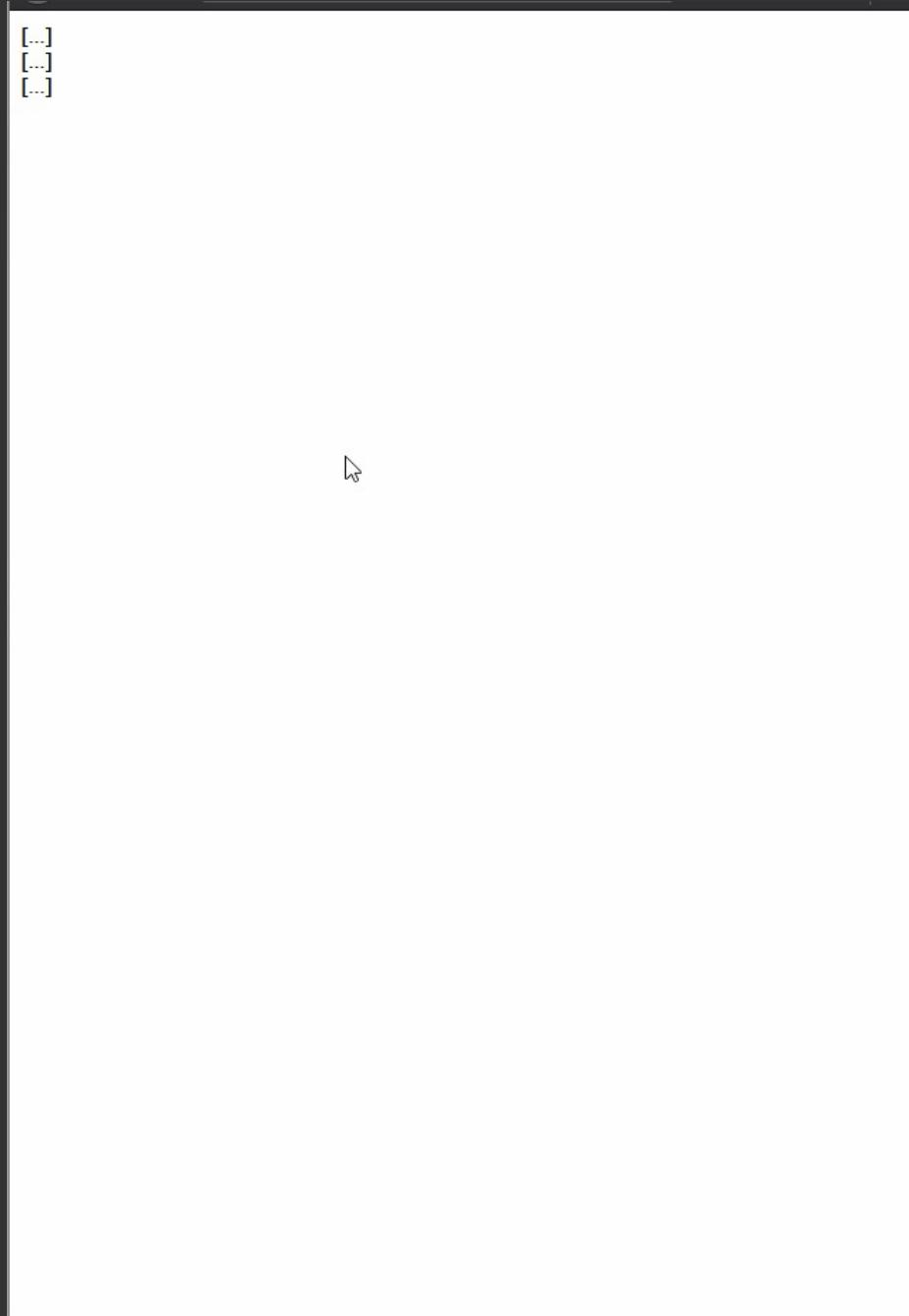
# Tester son site mobile sur PC

- Le mode responsive (ou vue adaptative) des navigateurs permet de tester son site à des résolutions différentes. **Ce mode garde le rendu du navigateur**
- Sur MacOS, il est possible de brancher son iPhone / iPad et tester l'inspecteur de Safari
- Raccourci clavier : ctrl + shift (maj) + m (Chrome / Brave / Firefox)

## Sources :

- <https://www.idownloadblog.com/2019/06/21/how-to-use-safari-web-inspector-ios-mac/> - en

# Mode mobile sur PC - Firefox



# Mode mobile sur PC - Chrome

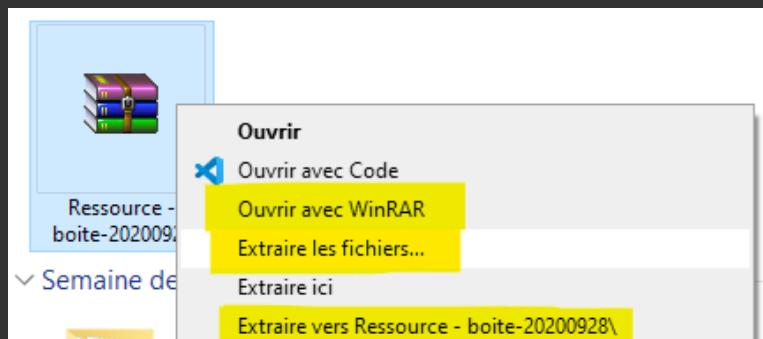
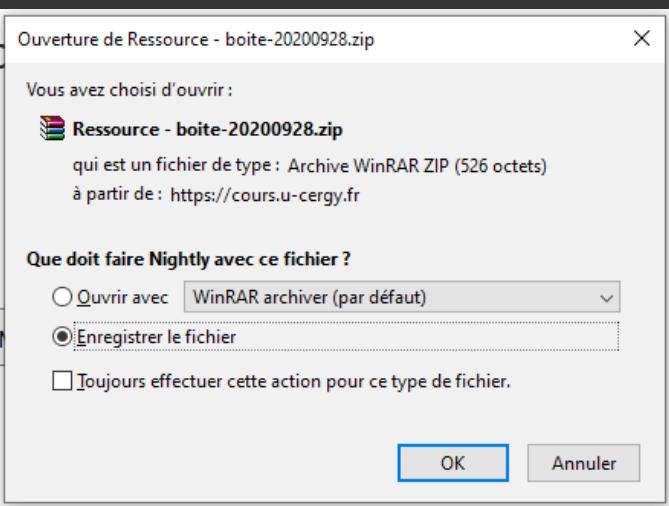


[...]  
[...]  
[...]

# Ressource du cours sur ENT

## Travaux Pratiques #4 > "Ressource – Images responsives"

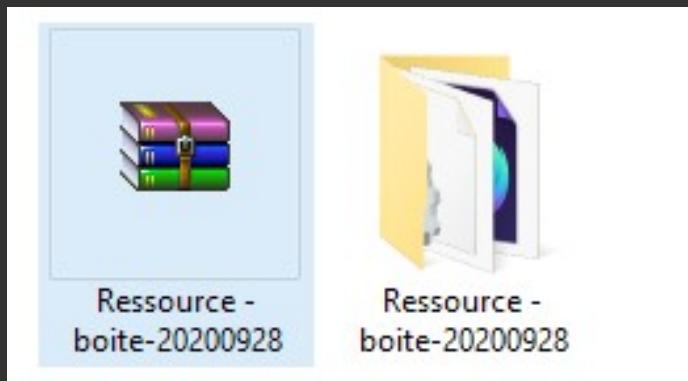
# Petit point sur les ressources de cours



- Les ressources se présentent sous la forme d'une archive une fois téléchargées
- L'archive doit être impérativement extraite sinon vous ne serez pas capables de réaliser correctement les tp

Menu contextuel sur l'archive (clic droit).  
Choisir un des choix surligné.

# Petit point sur les ressources de cours



- Une fois extrait, vous devez obtenir un dossier avec le contenu de l'archive
- Libre à vous de travailler dans le dossier OU en copier le contenu pour le coller ailleurs
- **En aucun cas vous devez travailler dans l'archive**
- Une fois fait, glissez le dossier dans VS Code

# Pratiquons ! - Notre image responsive

Pré-requis :

- Avoir la ressource "images-responsives"
- Ajouter une balise `<img />` avec une image relativement large (+2000px)

# Pratiquons ! - Notre image responsive

Pré-requis :

- Avoir la ressource "images-responsives"
- Avoir une balise <img /> avec une image relativement large
- Ajouter une classe "img-responsive" dans le fichier css style.css avec les déclarations suivantes :

**max-width: 100 %;**  
**height: auto;**

- Ajouter la classe à l'image
- Redimensionnez votre fenêtre, votre image s'adapte maintenant à votre fenêtre

**Notre image est responsive mais...**

**Elle est trop grande pour nos  
terminaux mobiles**

# Attributs srcset / sizes et balise <picture>

- Permettent de gérer des images en fonction de l'appareil
  - Taille d'écran
  - Résolution
  - Orientation (uniquement la balise picture)
- Chargent uniquement l'image pertinente
  - Économie de bande passante
- Gérés par les navigateurs modernes

## Sources :

- <https://www.alsacreations.com/article/lire/1621-responsive-images-srcset.html>
- <https://www.smashingmagazine.com/2014/05/responsive-images-done-right-guide-picture-srcset/> - anglais
- [https://developer.mozilla.org/fr/docs/Apprendre/HTML/Comment/Ajouter\\_des\\_images\\_adaptatives\\_%C3%A0\\_une\\_page\\_web](https://developer.mozilla.org/fr/docs/Apprendre/HTML/Comment/Ajouter_des_images_adaptatives_%C3%A0_une_page_web)

# Attributs srcset / sizes et balise <picture>

- Ils ne sont pas interchangeables
- Si l'image doit changer son contenu en fonction de l'écran → balise <picture>
  - Par exemple, je veux afficher une draisienne sur mobile, un tricycle sur tablette et un vélo sur smartphone
- Si l'image doit changer de taille en fonction du périphérique (largeur d'écran, définition / densité de pixels) → attribut srcset sur la balise <img />

## Sources :

- <https://www.alsacreations.com/article/lire/1621-responsive-images-srcset.html>
- <https://www.smashingmagazine.com/2014/05/responsive-images-done-right-guide-picture-srcset/> - anglais
- [https://developer.mozilla.org/fr/docs/Apprendre/HTML/Comment/Ajouter\\_des\\_images\\_adaptatives\\_%C3%A0\\_une\\_page\\_web](https://developer.mozilla.org/fr/docs/Apprendre/HTML/Comment/Ajouter_des_images_adaptatives_%C3%A0_une_page_web)

# Attribut srcset

- Attribut "srcset" liste les sources des images, le navigateur choisira la plus pertinente tout seul.
- Valeurs séparées par une virgule
- Image peut être associée à une résolution ou une taille

The diagram illustrates the structure of the `srcset` attribute. It shows a green `srcset` attribute with a pink box around the value `"mon-imageHD.jpg 2x"`. This value is divided into two parts: `mon-imageHD.jpg` and `2x`. A blue box surrounds the `2x` part. To the right of the `2x` box is a vertical line with a bracket pointing to the text "descripteur (ici de densité)". Below the `srcset` attribute, a pink box contains the text "source de mon image".

```
srcset="mon-imageHD.jpg 2x"  
      |  
      +-- descripteur  
           (ici de densité)  
      |  
      +-- source de  
           mon image
```

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/img#attr-srcset>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/img#attr-sizes>

# Attribut srcset – descripteur

- Descripteur "x"
  - Désigne la densité de pixel de l'écran
  - 1x représente un écran normal / 2x un écran HD...
  - Accepte un nombre décimal. Exemple "3x"
- Descripteur "w"
  - Désigne la largeur de l'image (pas de l'écran)
  - Accepte un entier positif. Exemple "320w"

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/img#attr-srcset>

# Attribut srcset – En application

```

```

Le code ci-dessus dit au navigateur "j'ai une image de 320 px de largeur et une autre de 640 px de largeur, prend la plus adaptée et si tu ne connais pas l'attribut srcset affiche mon-image.jpg"

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/img#attr-srcset>

# Attribut sizes

- Lié à la balise <img />
- Fonctionne de pair avec l'attribut "srcset"
  - Uniquement s'il y a des descripteurs de largeur "w"
- Éléments séparés par une virgule

**sizes="**(max-width : 320px) 280px, 900px"

condition sur le média  
(media query)

taille de l'image pour la  
condition donnée

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/img#attr-srcset>

# Pratiquons ! - Attributs srcset / sizes

Pré-requis :

- Avoir la ressource "images-responsives"
- Ajouter une balise `<img />` avec les attributs `srcset` et `src`
- Mettre comme valeur pour l'attribut `src`, une image du dossier "assets/images/attribut-srcset/"
- Définir la valeur de l'attribut `srcset` avec autant de valeurs qu'il y a d'images dans le dossier "assets/images/attribut-srcset/" avec le descripteur de densité (x) jusqu'à 3x
- Tester avec le mode responsive du navigateur (ctrl + maj + m sur firefox)

**Notre image change maintenant en fonction de l'écran**

# Pratiquons ! - Attributs srcset / sizes

Pré-requis :

- Avoir la ressource "images-responsives"
- Avoir une balise <img /> avec les attributs srcset et src
- Remplacer les descripteurs de densité par des descripteurs de largeur avec comme valeur le nombre contenu dans le nom de l'image
  - Ex : image-750.png → 750w
- Ajouter l'attribut sizes avec des media queries et une taille
  - Ex : (max-width : 480px) 100vw
  - Ne pas oublier la valeur par défaut
- Tester avec le mode responsive du navigateur (ctrl + maj + m sur firefox)

**Notre image change maintenant en fonction de l'écran et possède une taille en fonction de l'écran**

# Balise picture

- Utilise la balise <img /> comme valeur par défaut
- Fonction de pair avec la balise <source/>
  - Fonctionne avec les attributs srcset et media
- A utiliser si l'image doit changer (rognage par exemple) en fonction du périphérique
- Permet de gérer différent format d'image quand certains ne sont pas gérés par le navigateur
  - Ex : le format png n'est pas géré alors le navigateur va gérer un autre format

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/picture>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source>

# Balise picture – balise source

## Dans le cas de la balise <picture>

- Fonctionne avec les attributs "srcset" et "media/type"
  - srcset : définit un lien vers une image
  - media : définit la condition sur le media
  - type : définit le format de l'image (au format MIME)
  - Il faut au moins l'attribut "media" ou "type" avec l'attribut "srcset"
- Une balise <source> par image

Une fois tout ça définit, le navigateur décide tout seul

### Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/picture>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source#attr-media>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source#attr-srcset>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source#attr-type>

# Balise <picture> – En application

```
<picture>
  <source srcset="image-mini.jpg" media="(min-width:
320px)" />
  <source srcset="image-large.jpg" media="(min-width:
600px)"/>
  
</picture>
```

Le code ci-dessus dit au navigateur "j'ai une image de 320 px de largeur et une autre de 640 px de largeur, prend la plus adaptée et si aucune condition est remplie ou que tu ne connais pas la balise source charge la "mon-image.jpg"

## Sources :

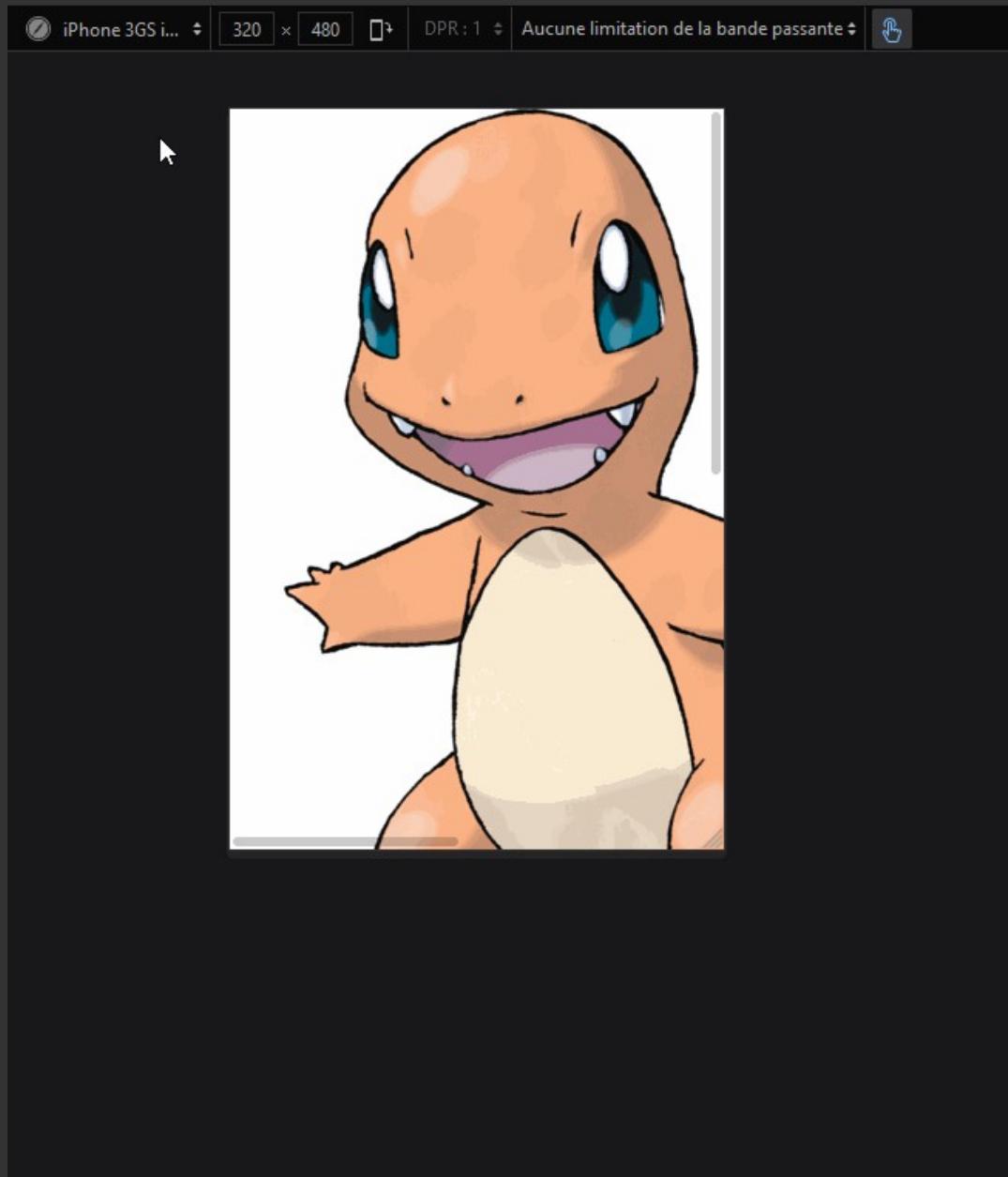
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/picture>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source#attr-media>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source#attr-srcset>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/source#attr-type>

# Pratiquons ! - Balise <picture>

Pré-requis :

- Avoir la ressource "images-responsives"
- Ajouter la structure de code pour la structure <picture> (il y a un début de structure dans les notes de la slide précédente)
  - Gérer 3 résolutions
    - 320px (iPhone 5 / SE)
    - 640px (Samsung GS 10 / iPhone X)
    - 768px (iPad en mode portrait)
- Utiliser les images contenues dans le dossier "assets/images/balise-picture/"

# Pratiquons ! - Balise <picture>



# Media queries

- Introduit avec CSS3
- Permet de définir un comportement pour des conditions données
  - Taille limite d'écran (hauteur ou largeur)
  - Gestion de certaines propriétés css
  - Orientation de l'écran
  - ...
- Une des trois composantes du "Responsive Web Design"
- Gestion d'opérateurs logiques
  - and / only / not
  - Plusieurs conditions peuvent être appliquées
- Au service des terminaux mobiles
  - Ajustement du design pour ces terminaux
- Peut être géré directement dans le css ou le html via la balise <link />

## Sources :

- [https://developer.mozilla.org/fr/docs/Web/CSS/Requ%C3%Aates\\_m%C3%A9dia/Utiliser\\_les\\_Media\\_queries](https://developer.mozilla.org/fr/docs/Web/CSS/Requ%C3%Aates_m%C3%A9dia/Utiliser_les_Media_queries)
- <https://www.alsacreations.com/article/lire/930-css3-media-queries.html>

# Media queries – Dans le css

- Nécessite la règle @ (at) @media et ses conditions

```
@media screen and (max-width: 640px) {  
    .titre-principal {  
        font-size: 25px;  
    }  
}
```

Sélecteurs css

Condition sur le média

Ici on définit que ces règles ne s'applique que sur un **écran ayant une largeur de 640px maximum inclus**

## Sources :

- [https://developer.mozilla.org/fr/docs/Web/CSS/Requ%C3%Aates\\_m%C3%A9dia/Utiliser\\_les\\_Media\\_queries](https://developer.mozilla.org/fr/docs/Web/CSS/Requ%C3%Aates_m%C3%A9dia/Utiliser_les_Media_queries)
- [https://developer.mozilla.org/fr/docs/Web/CSS/R%C3%A9gles\\_@](https://developer.mozilla.org/fr/docs/Web/CSS/R%C3%A9gles_@)

# Pratiquons ! - Media queries

Pré-requis :

- Avoir la ressource "media-queries"
- Éditer la règle @media de façon à ce que le texte s'affiche plus gros sur un appareil dont la largeur d'écran est inférieure ou égale à 320px
- Ajouter une règle @media pour gérer un écran dont la taille est supérieure ou égale à 400px et appliquer des déclarations
  - Ex : ajouter des marges latérales

# Pratiquons ! - Media queries

- Ajouter une règle @media pour gérer un écran ayant l'orientation paysage (landscape en anglais)
- Éditer / ajouter des règles @media pour changer la mise en page de la page en fonction de vos envies

# Media queries – Dans le html

- Utilisation de la balise <link> avec l'attribut "media"
- Plus besoin d'utiliser la règle @, l'attribut media remplissant le rôle

```
<link rel="stylesheet"  
      media="screen and (max-width: 640px)"  
      href="smartphone.css" type="text/css" />
```

Chemin vers le fichier css

Condition sur le média pour le fichier entier

Ici on définit que le contenu du fichier "smartphone.css" ne s'applique que sur un **écran ayant une largeur de 640px maximum inclus**

## Sources :

- [https://developer.mozilla.org/fr/docs/Web/CSS/Requ%C3%Aates\\_m%C3%A9dia/Utiliser\\_les\\_Media\\_queries](https://developer.mozilla.org/fr/docs/Web/CSS/Requ%C3%Aates_m%C3%A9dia/Utiliser_les_Media_queries)
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/link>

# Formatage du texte – rem / em

- Unités relatives
- rem se base sur l'élément racine (r pour root) de la page, souvent le font-size défini sur <html> avec une valeur de 16px
- em sera plus utilisé pour les marges notamment de la balise <p>

## Sources :

- <https://24ways.org/2006/compose-to-a-vertical-rhythm/> - en

# Formatage du texte – rem

- 1 rem représente 100 % de la valeur du parent
- Exemple, si on définit "font-size: 16px" au niveau de la balise <html>, une balise <h1> qui a la déclaration "font-size : 1.5rem" a la valeur de 24px une fois converti en pixels
- Calcul pixel → rem :

$$taille \text{ en rem} = \frac{\text{taille de l' élément}}{\text{taille racine}}$$

## Sources :

- <https://24ways.org/2006/compose-to-a-vertical-rhythm/> - en

# Formatage du texte – text-align

- Permet de modifier l'alignement du texte
- Aligné à gauche par défaut (`text-align: left;`)
- Accepte les valeurs suivantes :
  - `left`, `right`, `center`, `justify`
- La valeur "justify" donne accès à une nouvelle propriété "text-justify"

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/text-align>
- <https://developer.mozilla.org/fr/docs/Web/CSS/text-justify>

# Formatage du texte – text-decoration

- Décore le texte
- Propriété raccourcie
  - text-decoration-line
  - text-decoration-color
  - text-decoration-style
- Permet par exemple de souligner ou barrer du texte avec une certaine couleur

## Source :

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

# Pratiquons ! - Décorons du texte

Pré-requis :

- Avoir le contenu de la ressource "Ressource texte à décorer" sur ENT

- Changer l'alignement du paragraphe

- Changer la "font-family" du texte.

Exemples de polices sûres :

- Arial, Impact, "Times New Roman", Verdana, Tahoma (voir lien en-dessous pour plus de polices)

Source :

- <https://www.cssfontstack.com/> - liste des polices "sûres"

# Pratiquons ! - Décorons du texte

- Au sein du même paragraphe :
  - Souligner une partie du texte
  - Augmenter une partie du texte
  - Diminuer une partie du texte
  - Ajouter une couleur d'arrière-plan sur une partie du texte
- La partie doit être différente à chaque fois

# Quiz

## "Test - Texte" sur ENT

Conditions du test :

- ~10 minutes
- Tentatives illimitées
- Meilleure des notes prise en compte

# Ressource du cours

**Travaux Pratiques #3 > "Ressource – Site web" sur ENT**

# Site web

- Les pages peuvent être statiques ou dynamiques
- Ensemble infinie de pages web liées par des liens hypertextes
- Les pages sur le même domaine sont considérées comme faisant partie du même site web.
- Si mon site a un lien vers le site de duckduckgo ça ne veut pas dire que mon site fait partie de duckduckgo et vice-versa

## Sources :

- <https://24ways.org/2006/compose-to-a-vertical-rhythm/> - en

# Balise <a>

- a pour "anchor" ou "ancre" en français
- Permet de définir un lien dans la page ou une autre page sur le web
- Utilise l'attribut "href" pour définir l'hypertexte
- Historiquement les liens s'affichaient soulignés
- Par défaut un lien s'ouvre dans le même onglet

```
<a href="https://cours.u-cergy.fr">Mon ENT</a>
```

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/a>

# Balise <a> - ancre sur la même page

- Permet d'atteindre un élément dans la page grâce à son paramètre "id"
- Un élément sur la même page doit avoir un id avec une valeur
- Une balise <a> doit avoir comme valeur pour l'attribut "href" le même id précédé du signe croisillon (#)

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/a>

# Pratiquons ! - Lien interne

Pré-requis :

- Avoir le contenu de la ressource "Ressource – Site web" sur ENT
- Ajouter un élément avec un id
- Ajouter une balise <a> avec comme attribut "href" et la valeur "#le-nom-de-votre-id"

# Pratiquons ! - Lien interne

[Hello test](#) Cras molestie diam eu metus malesuada facilisis. Curabitur nec consequat purus. Mauris gravida volutpat tortor, at vestibulum metus egestas sed. Ut non felis in tortor iaculis hendrerit. Suspendisse ornare tincidunt odio, et faucibus eros volutpat a.

Avant clic

[Hello test](#) Cras molestie diam eu metus malesuada facilisis. Curabitur nec consequat purus. Mauris gravida volutpat tortor, at vestibulum metus egestas sed. Ut non felis in tortor iaculis hendrerit. Suspendisse ornare tincidunt odio, et faucibus eros volutpat a.

Après clic

# Pseudo-classe

- Permet d'appliquer des déclarations css sur un élément dans un état spécifique
- Vous ne pouvez pas associer plusieurs fois la même pseudo-classe
- Syntaxe :
  - un sélecteur html + ":" + le nom de la pseudo classe
  - exemple : **.ma-classe:pseudo-classe { border: 2px solid blue;**  
**}**

## Sources :

- <https://www.creativejuiz.fr/blog/css-css3/difference-entre-pseudo-element-et-pseudo-classe>
- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes>

# Pseudo-classe – <a> dans tous ses états

- La balise <a> est souvent associée aux pseudo-classes suivantes :
  - :link : désigne les liens non visités
  - :visited : désigne les liens visités
  - :hover : désigne les liens au survol
  - :active : désigne le moment entre le clic et le relâchement d'un lien
- Pour éviter tout problème de css, il est préférable de les écrire dans l'ordre suivant :
  - :link — :visited — :hover — :active.

## Sources :

- <https://www.creativejuiz.fr/blog/css-css3/difference-entre-pseudo-element-et-pseudo-classe>
- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes>

# Pratiquons ! - Lien interne

Pré-requis :

- Avoir le contenu de la ressource "Ressource – Site web" sur ENT avec une balise <a> dans le code
- Rajouter les pseudo-classes (:link, :visited, :hover, :active) pour la balise <a>
- Mettre un style différent (color, text-decoration...) au choix pour chacune de ces pseudo-classes

# Pseudo-classe – <a> dans tous ses états

- Prend également en compte la pseudo-classe "focus"
  - N'est pas influencé par l'ordre vu précédemment
  - Visible uniquement au "focus" clavier du lien

## Sources :

- <https://www.creativejuiz.fr/blog/css-css3/difference-entre-pseudo-element-et-pseudo-classe>
- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes>

# Balise <a> - lien externe

- Utilisation de l'attribut "href"
- Permet de faire des liens absous ou relatifs
  - Vers page web, fichiers (jpg, pdf, doc)...
- Il est préférable d'ouvrir un lien dans le même onglet
  - Attribut "target" permet de changer ce comportement grâce à l'attribut "\_blank"
- Indiquer si on ouvre une nouvelle page ou lien
- Éviter les cta (call to action) trop obscurs comme "cliquer ici", il est vaut mieux mettre l'action que ça va effectuer. Exemple "Accéder à ENT"

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/a>

# Pratiquons ! - Lien externe

Pré-requis :

- Avoir le contenu de la ressource "Ressource – Site web" sur ENT avec une balise <a> dans le code
- Faire pointer votre lien (ou un autre) vers l'autre page présente dans la ressource "description.html"
- Dans le fichier "description.html" faire pointer un lien vers la page "index.html"
- Dans le fichier "description.html" faire pointer une image vers une image

# Balise <a> - attribut href

- Permet de faire des liens vers des ancrés ou des pages distantes
- Schéma url : permet d'effectuer des actions spécifiques
  - mailto: envoyer une adresse mail
  - tel: composer un numéro de téléphone

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/a>

# Quiz

## "Test - Liens" sur ENT

Conditions du test :

- ~10 minutes
- Tentatives illimitées
- Meilleure des notes prise en compte

# Petit point accessibilité – Les liens

- Les liens interne peuvent être utilisés pour accéder rapidement au contenu principal
- Les liens ne doivent pas être utilisés pour être des pseudo-boutons, la balise `<button>` est là pour ça
- Des liens proches peuvent mener à des mauvais clics, si possible éviter cette juxtaposition ou mettre un espace suffisamment grand entre eux
- Sur mobile, on cible avec les doigts, périphérique plus gros, il faut donc s'arranger pour que les éléments cliquables soient plus gros sur appareils mobiles

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/a#Accessibilit%C3%A9>

# Pseudo-classe - :hover

- Permet d'appliquer des déclarations css sur un élément au survol
- Applicable sur n'importe quelle balise
- Idéal pour effectuer des transitions d'élément(s)
- Peut de cibler plusieurs balises en même temps grâce au système d'imbrications

## Sources :

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

# Pseudo-classe - :hover

- Jouer avec le principe des imbrications de balises html et règles css
- Indiquer le comportement des enfants lors du survol (:hover) du parent

```
.ma-balise:hover p {      <div class="ma-balise">
    color: red;           <p>Mon texte</p>
}                           </div>
```

Ici, on cible les <p> quand <div class="ma-balise"> est survolée

## Sources :

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

# Ressource du cours

**Travaux Pratiques #3 > "Ressource – hover" sur ENT**

# Pratiquons ! - :hover sur sur balise

Pré-requis :

- Avoir le contenu de la ressource "Ressource – hover" sur ENT
- Ajouter un nouveau sélecteur "conteneur-survol:hover" dans le fichier css
- Changer la couleur du span contenu dans la balise au survol de la classe "conteneur-survol"

**Ça fonctionne... mais ça manque de dynamisme**

# Transitions css

- Permet de faire une transition entre deux états (valeur de propriétés)
- Propriété "transition"
- Souvent lancées au survol ou au focus ou modification de classe (nécessite javascript)
- Mettre les transitions sur le sélecteur et non le sélecteur + pseudo-classe
- Animations "simples" comparé aux animations
- Rajoute du "dynamisme"

## Sources :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_Transitions/Utiliser\\_transitions\\_CSS](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Transitions/Utiliser_transitions_CSS)
- <https://developer.mozilla.org/en-US/docs/Web/CSS/transition>
- [https://developer.mozilla.org/fr/docs/Web/CSS>Liste\\_propri%C3%A9t%C3%A9s\\_CSS\\_anim%C3%A9es](https://developer.mozilla.org/fr/docs/Web/CSS>Liste_propri%C3%A9t%C3%A9s_CSS_anim%C3%A9es)

# Transitions css

- Propriété raccourcie :
  - transition-property : propriétés qui vont changer
  - transition-duration : durées des transitions
  - transition-timing-function : description de la courbe d'évolution des valeurs intermédiaires
  - transition-delay : délai avant le lancement de la transition
- Si les valeurs ne sont pas définies, celles par défaut seront utilisées

## Sources :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_Transitions/Utiliser\\_transitions\\_CSS](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Transitions/Utiliser_transitions_CSS)
- <https://developer.mozilla.org/en-US/docs/Web/CSS/transition>
- [https://developer.mozilla.org/fr/docs/Web/CSS>Liste\\_propri%C3%A9t%C3%A9s\\_CSS\\_anim%C3%A9es](https://developer.mozilla.org/fr/docs/Web/CSS>Liste_propri%C3%A9t%C3%A9s_CSS_anim%C3%A9es)

# Transitions css

**transition:** margin-right **2s** linear **.5s;**

-property

-duration

-delay

-timing-function

## Sources :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_Transitions/Utiliser\\_transitions\\_CSS](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Transitions/Utiliser_transitions_CSS)
- <https://developer.mozilla.org/en-US/docs/Web/CSS/transition>
- [https://developer.mozilla.org/fr/docs/Web/CSS>Liste\\_propri%C3%A9t%C3%A9s\\_CSS\\_anim%C3%A9es](https://developer.mozilla.org/fr/docs/Web/CSS>Liste_propri%C3%A9t%C3%A9s_CSS_anim%C3%A9es)

# Pratiquons ! - :hover sur sur balise

Pré-requis :

- Avoir le contenu de la ressource "Ressource – hover" sur ENT avec une pseudo-classe "hover" sur le sélecteur "conteneur-survol"
- Ajoutons une transition sur le sélecteur ".conteneur-survol:hover"

```
.conteneur-survol:hover .titre {  
    [...]  
    transition-property: color;  
    transition-duration: durée choisie;  
    transition-timing-function: linear;  
    transition-delay: durée choisie;  
}
```

# Pratiquons ! - :hover sur sur balise

- Corrigeons notre code :
  - Retirer la transition dans le sélecteur ".conteneur-survol:hover"
  - Mettre la transition sur la classe ".titre"

```
.titre {  
  [...]  
  transition-property: color;  
  transition-duration: durée choisie;  
  transition-timing-function: linear;  
  transition-delay: durée choisie;  
}
```

Maintenant la transition est présente quand on entre et sort de l'élément ".conteneur-survol"

# Transitions css

- Presque toutes les propriétés css sont animables
- Utilisation de la propriété "transform" pour effectuer des transitions plus élaborées
- Un élément modifié par la propriété transform garde sa place dans le flux. Elle ne fait donc pas bouger les éléments juxtaposés

## Sources :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_Transitions/Utiliser\\_transitions\\_CSS](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Transitions/Utiliser_transitions_CSS)
- <https://developer.mozilla.org/en-US/docs/Web/CSS/transition>
- [https://developer.mozilla.org/fr/docs/Web/CSS>Liste\\_propri%C3%A9t%C3%A9s\\_CSS\\_anim%C3%A9es](https://developer.mozilla.org/fr/docs/Web/CSS>Liste_propri%C3%A9t%C3%A9s_CSS_anim%C3%A9es)
- <https://www.alsacreations.com/tuto/lire/873-transitions-css3-animations.html>

# Pratiquons ! - Essayons les transitions

Pré-requis :

- Avoir le contenu de la ressource "Ressource – transform" sur ENT le contenu du dossier "exercice" servira de base de travail
- Premières transitions



Objectif



Base

# Pratiquons ! - Essayons les transitions

Première transition en détails :

- Opacité de l'image de fond
- Apparition du sous-titre
- Déplacement du sous-titre, du titre et l'image



État initial



État final

# Pratiquons ! - Essayons les transitions

- Appliquons les sélecteurs css les plus simples : l'opacité et sa transition sur l'image

```
.element-conteneur.transition-1 img {  
    opacity: 0.7;  
    transition: opacity 0.35s;  
}
```

```
.element-conteneur.transition-1:hover img {  
    opacity: 1;  
}
```

# Pratiquons ! - Essayons les transitions

- Vous devriez obtenir ceci



# Transitions css – Propriété transform

- Propriété CSS3
- Permet d'appliquer des transformations sur un élément comme la translation ou la rotation
- Accepte comme valeurs des fonctions définies
- Possibilité d'appliquer plusieurs fonctions en même temps espacées
- N'utilise pas l'accélération matérielle (GPU)

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform>
- <https://www.alsacreations.com/article/lire/1418-css3-transformations-2d.html>

# Propriété transform – Fonction translate

- Permet de déplacer un élément sur l'axe x et y
- Accepte comme valeur une longueur
- Possibilité de se déplacer sur l'axe z avec la fonction translate3d – Accepte trois paramètres
- Propriété raccourcie
  - translateX(), translateY(), translateZ()
- translate() n'utilise pas l'accélération matérielle contrairement à translate3d()

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function/translate3d>
- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function/translate>
- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function#Translations>
- <https://www.alsacreations.com/astuce/lire/1565-acceleration-materielle-au-service-de-vos-animations-css.html>

# Pratiquons ! - Essayons les transitions

- Toujours sur l'image appliquons la translation

```
.element-conteneur.transition-1 img {  
    opacity: 0.7;  
    transition: opacity 0.35s, transform 0.35s;  
    transform: translate3d(-40px, 0, 0);  
}
```

```
.element-conteneur.transition-1:hover img {  
    opacity: 1;  
    transform: translate3d(0, 0, 0);  
}
```

# Pratiquons ! - Essayons les transitions

- Vous devriez obtenir ceci



# Pratiquons ! - Essayons les transitions

- Déplaçons le titre

```
.element-conteneur.transition-1 h2 {  
    [...]  
    transform: translate3d(0, 40px, 0);  
}
```

```
.element-conteneur.transition-1:hover h2 {  
    transform: translate3d(0, 0, 0);  
}
```

# Pratiquons ! - Essayons les transitions

- Puis le sous-texte

```
.element-conteneur.transition-1 p {  
    [...]  
    transition: opacity 0.2s, transform 0.35s;  
    transform: translate3d(0, 60px, 0);  
}
```

```
.element-conteneur.transition-1:hover p {  
    transform: translate3d(0, 0, 0);  
    opacity: 1;  
}
```

# Pratiquons ! - Essayons les transitions

- Et normalement, on devrait obtenir nos transitions cibles



# Transitions css – Bonnes pratiques

- Préférer translate / translate3d au lieu des propriétés top, margin, width... pour des questions de performances
- Inutile de rendre accessible une animation, en revanche, les informations importantes doivent être accessibles
- Faire attention à la vitesse des animations, trop lentes ou rapides, elles peuvent nuire à l'expérience utilisateur

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function#Translations>
- <https://www.alsacreations.com/astuce/lire/1565-acceleration-materielle-au-service-de-vos-animations-css.html>

# Transitions css – Bonnes pratiques

- Éviter les zones de survol trop petites surtout si elles affichent des données critiques
- Ne pas oublier qu'une animation joue notamment sur la persistance rétinienne

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function#Translations>
- <https://www.alsacreations.com/astuce/lire/1565-acceleration-materielle-au-service-de-vos-animations-css.html>

# Pratiquons ! - Essayons les transitions

Deuxième transition en détails :

- Apparition d'une barre blanche
- Déplacement du titre
- Mise à l'échelle et rotation de l'image de fond



État initial



État final

# Pratiquons ! - Essayons les transitions

- Déplaçons la barre blanche

```
.element-conteneur.transition-2 figcaption::before {  
    compléter  
}
```

```
.element-conteneur.transition-2:hover figcaption::before {  
    compléter  
}
```

# Pseudo-éléments

- Permet de cibler une partie du contenu sans pour autant ajouter de nouvelles balises
- N'existe pas pour les lecteurs d'écran → Ne pas mettre de données critiques
- Syntaxe :
  - un sélecteur html + ":" + le nom de la pseudo classe
  - exemple : **.ma-classe::pseudo-element { color: blue; }**

## Sources :

- <https://www.creativejuiz.fr/blog/css-css3/difference-entre-pseudo-element-et-pseudo-classe>
- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-%C3%A9%C3%A9ments>
- <https://accessibleweb.com/question-answer/how-is-css-pseudo-content-treated-by-screen-readers/> - en
- <https://developer.mozilla.org/fr/docs/Web/CSS/:first-line>

# Pseudo-éléments

- Vous ne pouvez pas associer plusieurs fois le même pseudo-élément à une balise
- Ne s'affiche pas dans le code source
- ::before / ::after sont certainement ce que vous utiliserez le plus

## Sources :

- <https://www.creativejuiz.fr/blog/css-css3/difference-entre-pseudo-element-et-pseudo-classe>
- <https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-%C3%A9%C3%A9ments>
- <https://accessibleweb.com/question-answer/how-is-css-pseudo-content-treated-by-screen-readers/> - en
- <https://developer.mozilla.org/fr/docs/Web/CSS/::first-line>

# Pseudo-éléments - ::before /::after

- Permettent d'ajouter des éléments avant ou après un contenu
- **Nécessite la propriété "content"** pour s'afficher. Contenu pouvant être :
  - une chaîne de caractères, une image (qui ne peut pas être redimensionnée), rien (une chaîne vite), un compteur

## Sources :

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

# Pseudo-éléments - ::before /::after

- Acceptent toutes les déclarations css
- Ne fonctionne pas sur les éléments remplacés comme <img> ou <br>
- Le contenu dans l'attribut "content" n'est pas sélectionnable

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/::after>
- <https://developer.mozilla.org/fr/docs/Web/CSS/::before>
- <https://a.singlediv.com/> - Exemple de ce qu'on peut faire avec ces pseudo-éléments

# Pratiquons ! - Essayons les transitions

- Déplaçons la barre blanche

```
.element-conteneur.transition-2 figcaption::before {  
    compléter  
}
```

```
.element-conteneur.transition-2:hover figcaption::before {  
    compléter  
}
```

# Pratiquons ! - Essayons les transitions

- Déplaçons le titre
  - Le <h2> part d'où il est, et descend de 40px sur l'axe y

```
.element-conteneur.transition-2 h2 {  
    compléter  
}
```

```
.element-conteneur.transition-2:hover h2 {  
    compléter  
}
```

# Pratiquons ! - Essayons les transitions

- Sur l'image appliquons ses transitions
- Pour rappel, il nous faut changer :
  - échelle (scale) et rotation (rotateZ)

```
.element-conteneur.transition-2 img {  
    compléter  
}
```

```
.element-conteneur.transition-2:hover img {  
    compléter  
}
```

# Transitions css – Fonction scale()

- Permet de changer la mise à l'échelle d'un élément
- Accepte jusqu'à deux valeurs (axe x, axe y)
- Ces valeurs doivent être des nombres décimaux négatifs ou positifs
- `scale(1)` = élément garde son échelle de base (sur les deux axes)
- Fonction raccourcie des fonctions "scaleX" et "scaleY"

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function#Translations>
- <https://www.alsacreations.com/astuce/lire/1565-acceleration-materielle-au-service-de-vos-animations-css.html>

# Pratiquons ! - Essayons les transitions

- Appliquons la mise à l'échelle (1.5) sur l'image

```
.element-conteneur.transition-2 img {  
    compléter  
}
```

```
.element-conteneur.transition-2:hover img {  
    compléter  
}
```

# Transitions css – Fonction rotate()

- Permet de faire pivoter d'un élément selon un axe (centre / centre par défaut)
- Ces valeurs doivent être des valeurs valides pour un angle. Ex : -15deg ou 3.142rad
- Ne déforme pas l'élément
- Possède des fonctions sœurs : rotateX, rotateY, rotateZ
- Possède une version 3d : rotate3d()

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function/rotate>
- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function/rotate3d>
- <https://developer.mozilla.org/fr/docs/Web/CSS/transform-function#Rotation>

# Pratiquons ! - Essayons les transitions

- Appliquons la la rotation (15deg) sur l'image

```
.element-conteneur.transition-2 img {  
    compléter  
}
```

```
.element-conteneur.transition-2:hover img {  
    compléter  
}
```

# Pratiquons ! - Essayons les transitions

- Et normalement, on devrait obtenir nos transitions cibles



# Quiz

"Test - Pseudo-élément / pseudo-classe et transitions"  
sur ENT

Conditions du test :

- ~10 minutes
- Tentatives illimitées
- Meilleure des notes prise en compte

# Propriétés max-width et max-height

- Définit la dimension maximale d'un élément
  - Permet donc d'éviter qu'un élément dépasse un taille
- Prioritaires comparé à leur équivalent fixe
- Géré par tous les navigateurs modernes
- Très utiles pour le responsive
- Possèdent des propriétés "inverses" : min-width / max-width :
  - Les propriétés "min-" surchargent les propriétés "max-" et "width / height"

## Sources :

- <https://developer.mozilla.org/fr/docs/Web/CSS/max-width>
- <https://developer.mozilla.org/fr/docs/Web/CSS/max-height>
- <https://developer.mozilla.org/fr/docs/Web/CSS/min-width>
- <https://developer.mozilla.org/fr/docs/Web/CSS/min-height>
-

# Pratiquons ! - Essayons les transitions

- Appliquons la propriété "max-width" sur l'image

```
.element-conteneur.transition-2 img {  
    max-width: 100%;  
}
```



**Questions ?**