CSS : petits compléments

au programme...

1. ::before et ::after
2. compteurs
3. media queries
4. transformations et transitions

::before et ::after

::before et ::after

Les pseudo-éléments `::before` et `::after` permettent l'insertion de contenu avant ou après un élément.
Le contenu se définit grâce à la propriété `content`.
Des propriétés de style peuvent également être précisées.

content

content

`content` peut prendre comme valeur :

- **chaîne** une chaîne de caractères, entre `"`
- **url()** le plus souvent pour désigner une image
- **counter()** la valeur d'un compteur
- **attr(A)** une chaîne de caractère correspond à la valeur de l'attribut `A`
- **open-quote**|**close-quote** guillemets ouvrant/fermant définis par `quotes`

ex-content.html

compteurs

On peut manipuler des compteurs via :

- `counter-reset` : propriété qui crée et réinitialise le compteur précisé en valeur
  Chaque élément qui déclare la propriété `counter-reset` crée sa propre instance locale « courante » de compteur.
- `counter-increment` : propriété qui incrémente le compteur « courant » précisé en valeur
- *`counter()`* : affiche la valeur du compteur « courant » passé en paramètre

ex-content2.html

au programme...

au programme...

media queries

Objectif :

- adapter la feuille de style en fonction du périphérique de consultation du document :
  - type de média (écran, impression, etc.)
  - propriétés (dimensions, couleurs, ...)

media queries

Objectif :

- adapter la feuille de style en fonction du périphérique de consultation du document :
  - type de média (écran, impression, etc.)
  - propriétés (dimensions, couleurs, ...)

media query

Une ***media query***, ou **requête de média**, est une expression booléenne permettant de préciser la portée de règles CSS.

media queries

Objectif :

- adapter la feuille de style en fonction du périphérique de consultation du document :
  - type de média (écran, impression, etc.)
  - propriétés (dimensions, couleurs, ...)

media query

Une ***media query***, ou **requête de média**, est une expression booléenne permettant de préciser la portée de règles CSS.

rappel : dans le mécanisme de cascade le filtre par **media** est le premier appliqué

définition

- dans la balise <link> de <head>

```
<link href="style.css" rel="stylesheet" media="(max-width:800px)"/>
<link href="style-print.css" rel="stylesheet" media="print"/>
```

définition

- dans la balise <link> de <head>

```
<link href="style.css" rel="stylesheet" media="(max-width:800px)"/>
<link href="style-print.css" rel="stylesheet" media="print"/>
```

- dans la feuille de style

```
@media screen and (max-width:  1000px) {
   img.exemple {
      width :  60%;
   }
}
```

critères

- media
  `screen`, `print`, `tv`, `braille`, `all`, ...

media queries sur MDN

critères

- media
  `screen`, `print`, `tv`, `braille`, `all`, ...
- propriétés
  la plupart peuvent être précédées de `min-` ou `max-`

media queries sur MDN

critères

- media
  `screen`, `print`, `tv`, `braille`, `all`, ...
- propriétés
  la plupart peuvent être précédées de `min-` ou `max-`
  - `width` (`min-width`, `max-width`), `height` (*`min-height:640px`*)

media queries sur MDN

critères

- media
  `screen`, `print`, `tv`, `braille`, `all`, ...
- propriétés
  la plupart peuvent être précédées de `min-` ou `max-`
  - `width` (`min-width`, `max-width`), `height` (*`min-height:640px`*)
  - `device-width`, `device-height` (*`max-device-width:800px`*)

media queries sur MDN

critères

- media

  `screen`, `print`, `tv`, `braille`, `all`, ...

- propriétés

  la plupart peuvent être précédées de `min-` ou `max-`

  - `width` (`min-width`, `max-width`), `height` (*`min-height:640px`*)
  - `device-width`, `device-height` (*`max-device-width:800px`*)
  - `orientation` (*`portrait`*, *`landscape`*)

media queries sur MDN

critères

- media

  `screen`, `print`, `tv`, `braille`, `all`, ...

- propriétés

  la plupart peuvent être précédées de `min-` ou `max-`

  - `width` (`min-width`, `max-width`), `height` (*`min-height:640px`*)
  - `device-width`, `device-height` (*`max-device-width:800px`*)
  - `orientation` (*`portrait`*, *`landscape`*)
  - `aspect-ratio` (*16/9*, *4/3*, etc.)

media queries sur MDN

critères

- media

  `screen`, `print`, `tv`, `braille`, `all`, ...

- propriétés

  la plupart peuvent être précédées de `min-` ou `max-`

  - `width` (`min-width`, `max-width`), `height` (*`min-height:640px`*)
  - `device-width`, `device-height` (*`max-device-width:800px`*)
  - `orientation` (*`portrait`*, *`landscape`*)
  - `aspect-ratio` (*16/9*, *4/3*, etc.)
  - `color` (*`min-color:16`*)

media queries sur MDN

critères

- media

  `screen`, `print`, `tv`, `braille`, `all`, ...

- propriétés

  la plupart peuvent être précédées de `min-` ou `max-`

  - `width` (`min-width`, `max-width`), `height` (*`min-height:640px`*)
  - `device-width`, `device-height` (*`max-device-width:800px`*)
  - `orientation` (*`portrait`*, *`landscape`*)
  - `aspect-ratio` (*16/9*, *4/3*, etc.)
  - `color` (*`min-color:16`*)
  - etc.

media queries sur MDN

opérateurs

- `and` réalise le **et** logique de requêtes

```
@media screen and (max-width:  1000px)
```

opérateurs

- `and` réalise le **et** logique de requêtes
  ```
  @media screen and (max-width:  1000px)
  ```
- séparation par des virgules : équivaut au **ou** logique
  ```
  @media screen and (max-width:  800px) , orientation:landscape
  ```

opérateurs

- `and` réalise le **et** logique de requêtes
  `@media screen and (max-width: 1000px)`
- séparation par des virgules : équivaut au **ou** logique
  `@media screen and (max-width: 800px) , orientation:landscape`
- `not` inverse le résultat de la requête (porte sur l'ensemble)
  `not print and (monochrome)` ≡ `not (print and monochrome)`

opérateurs

- `and` réalise le **et** logique de requêtes
  `@media screen and (max-width: 1000px)`
- séparation par des virgules : équivaut au **ou** logique
  `@media screen and (max-width: 800px) , orientation:landscape`
- `not` inverse le résultat de la requête (porte sur l'ensemble)
  `not print and (monochrome)` ≡ `not (print and monochrome)`
- `only` pour compatibilité ancien navigateur

opérateurs

- `and` réalise le **et** logique de requêtes
  `@media screen and (max-width: 1000px)`
- séparation par des virgules : équivaut au **ou** logique
  `@media screen and (max-width: 800px) , orientation:landscape`
- `not` inverse le résultat de la requête (porte sur l'ensemble)
  `not print and (monochrome)` ≡ `not (print and monochrome)`
- `only` pour compatibilité ancien navigateur

exemple-mediaQueries.html

au programme...

au programme...

transformations

- la propriété `transform` permet d'appliquer des tranformations à la boîte d'un élément :
  - translations : `translate`, `translateX`, `translateY`
  - rotations : `rotate`
  - homothéties : `scale`, `scaleX`, `scaleY`
  - inclinaisons : `skewX`, `skewY`

transformations

- la propriété `transform` permet d'appliquer des tranformations à la boîte d'un élément :
  - translations : `translate`, `translateX`, `translateY`
  - rotations : `rotate`
  - homothéties : `scale`, `scaleX`, `scaleY`
  - inclinaisons : `skewX`, `skewY`

```
transform :  translateX(50px) skewY(5deg);
```

exemple-transition-transform.html

transitions

- les transitions permettent d'animer les changements de valeurs de propriétés en les rendant progressifs

transition sur MDN

transitions

- les transitions permettent d'animer les changements de valeurs de propriétés en les rendant progressifs
- les transitions s'appliquent entre une valeur initiale et une valeur finale et concernent nombres, longueurs, couleurs

transition sur MDN

transitions

- les transitions permettent d'animer les changements de valeurs de propriétés en les rendant progressifs

- les transitions s'appliquent entre une valeur initiale et une valeur finale et concernent nombres, longueurs, couleurs
- les valeurs intermédiaires sont calculées par le navigateur

transition sur MDN

transitions

- les transitions permettent d'animer les changements de valeurs de propriétés en les rendant progressifs
- les transitions s'appliquent entre une valeur initiale et une valeur finale et concernent nombres, longueurs, couleurs
- les valeurs intermédiaires sont calculées par le navigateur
- les transitions sont contrôlées par la propriété `transition` : qui agrègent les propriétés
  - `transition-property` la propriété concernée
  - `transition-duration` la durée de la transition
  - `transition-delay` la pause avant la transition
  - `transition-timing-function` la fonction de calcul des valeurs intermédiaires

transition sur MDN

```
transition :  color 1s;
transition :  padding 1s 0.5s ease-out;
```

```
transition :  left 1s 0.5s ease-out, background-color 1s 0s linear;
```

équivaut à

```
transition-property :  left, background-color;
transition-duration :  1s;
transition-delay :  0.5s, 0s;
transition-timing-function :  ease-out,linear;
```

à suivre...

Javascript