

# 1. CSS responsive (media-query , flex, ...)

## media-query

```

/*d'abord les règles pour tous médias et tailles */
h1 { font-size: 72px}
h1 , h2 { color : #0ca027 }
nav { height: 50px; background-color : lightgray; text-align : center}
p { width: 200px; height: 50px; background-color: #85c4db; /*bleu ciel */ }

/* ensuite nouvelles regles pour taille > 600ps de large) */
@media screen and (min-width : 600px){
  body { background-color : #f4b6a6}
}

/* ensuite nouvelles regles pour taille < 600ps de large) */
@media screen and (max-width : 600px){
  body { background-color : #f0f296}
  h1 { font-size: 32px}
  h2 { color : black;}
  nav { display : none; }
}

```

### Critères de sélection pour "media-query" :

<b>height</b>	hauteur de la fenêtre
<b>width</b>	largeur de la fenêtre
<b>device-height</b>	hauteur du périphérique (ex : tablette)
<b>device-width</b>	largeur du périphérique
<b>orientation</b>	portrait ou paysage
<b>media</b> <b>screen</b> <b>handheld</b> <b>print</b> ... <b>all</b>	<b>type de média</b> écran classique mobile (smartphone) imprimante

combinaisons de critères possibles avec **not** , **only** , **and** , **or** ...

**Attention** (pour une page html devant avoir un comportement "responsive" sur un smartphone , il ne faut pas oublier le paramètre "viewport" :

```
<html>
  <head> ...
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <!-- for responsive css on mobile device -->
</head>
```

### Flex box

Une boîte flexible (**flexbox**) permet de disposer les éléments (en horizontal ou en vertical) d'une manière responsive / flexible .

Il faut définir l'axe principal (**flex-direction**) ayant une des valeurs suivantes :

- **row**
- *row-reverse* (inverse au sens d'écriture lui même associé à la langue)
- **column**
- *column-reverse*

L'axe secondaire (cross-axis) est toujours perpendiculaire à l'axe principal .

Exemple :

```
.box {
  display: flex;
  flex-direction: row;
}
```

NB :

Si les éléments d'une boîte (en ligne par exemple) sont trop grands pour tenir sur une même ligne , ceux-ci seront :

- soit affichés sur plusieurs lignes (**flex-wrap: wrap**)
- soit éventuellement rétrécis pour tenir sur une même ligne ou vont provoquer un dépassement (**flex-wrap: nowrap** / valeur par défaut) .

Syntaxe synthétique :

**flex-flow: row wrap** (ou row nowrap) (ou column nowrap) ...

### Exemple complet (avec images et textes qui s'ajustent) :

#### *flex-page.html*

```
<html>
<head>
  <title>flex-page</title>
  <link rel="stylesheet" type="text/css" href="/css/styles-flex.css" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
</head>
<body>
  <h1>page responsive</h1> <h2>via flex box</h2> <hr/>
  <div class="mywrapbox">
    <div class="myitem proportionalText">-OneOneOneOne-</div>
    <div class="myitem proportionalText">*Two*Two*</br>
      *Two*Two*</br>*Two*Two*</div>
    <div class="myitem proportionalText">-ThreeThreeThree-</div>
    <div class="myitem proportionalText">*FourFourFour*</div>
  </div>
  <div class="mybox">
    <!-- les images 1 , 2 et 3 sont de différentes tailles -->
    <div class="mycell"></div>
    <div class="mycell"></div>
    <div class="mycell"></div>
  </div>
</body>
</html>
```

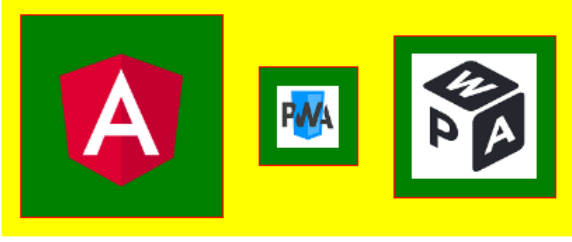
-OneOneOneOne-

\*Two\*Two\*  
\*Two\*Two\*  
\*Two\*Two\*

-ThreeThreeThree-

\*FourFourFour\*



<p>-OneOneOneOne-      *Two*Two*                                   *Two*Two*                                   *Two*Two*                                   *FourFourFour*</p> 	<p>4ème texte sur la ligne d'après car "wrap" sur 1ère box.</p> <p>Centrage vertical vis à vis du texte de plus grande taille '*TwoTwo&lt;br/&gt;...&lt;br/&gt;...Two*'</p> <p>Les images sont automatiquement réduites en taille car "nowrap" sur 2 ème box          et <code>img { max-width:100%;}</code> .</p>
---	--

## styles-flex.css

```

/* align-items : center ou stretch ou flex-start ou flex-end
   pour contrôler l'alignement sur l'axe secondaire si les éléments n'ont pas la même hauteur */
.mybox {
    background-color: yellow;
    display: flex;
    flex-flow: row nowrap;
    align-items : center;
}

.mywrapbox {
    display: flex;
    flex-flow: row wrap;
    align-items : center;
}

/* flex: flex-grow, flex-shrink, flex-basis
   flex-basis = taille occupée ou bien auto (automatiquement calculée)
   flex-grow et flex-shrink sont des coefficients (valeurs relatives par rapport autres éléments)*/
.myitem {
    flex: 1 1 auto;
    text-align: center;    padding: 2px;
}

.mycell {
    flex: 1 1 auto;
    padding: 10px;    margin: 10px;    background-color: green;
    border: 1px solid red;    text-align: center;
}

img { max-width:100%;} /* pour shrink sur images si trop grandes */

```

*/\* attention , l'ordre des min-width est important du plus petit au plus grand !!!*

*xs, sm , md , lg , xl \*/*

**@media screen and (max-width : 36em){**

.proportionalText { **font-size: 80%**; color:red; }

}

**@media screen and (min-width:36em ) {**

.proportionalText { **font-size: 100%**; color:black; }

}

**@media screen and (min-width:48em ) {**

.proportionalText { **font-size: 110%**; color:green; }

}

**@media screen and (min-width:64em){**

.proportionalText { **font-size: 120%**; color:blue;}

}

**@media screen and (min-width : 80em){**

.proportionalText { **font-size: 130%**; color:red;}

}