

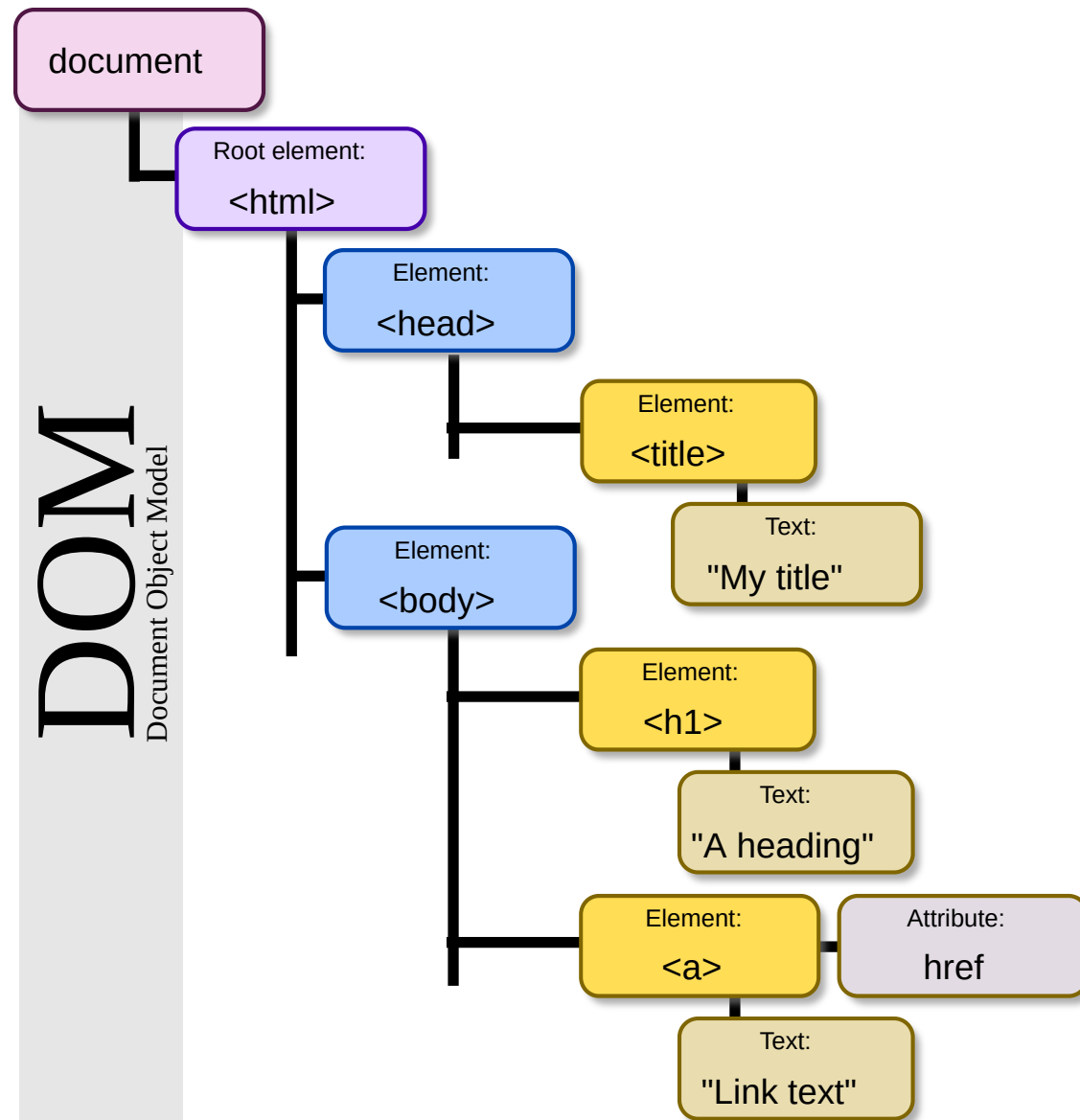
# R1.02

## DÉVELOPPEMENT D'INTERFACES WEB



# DOCUMENT OBJECT MODEL

# DOCUMENT OBJECT MODEL (DOM)

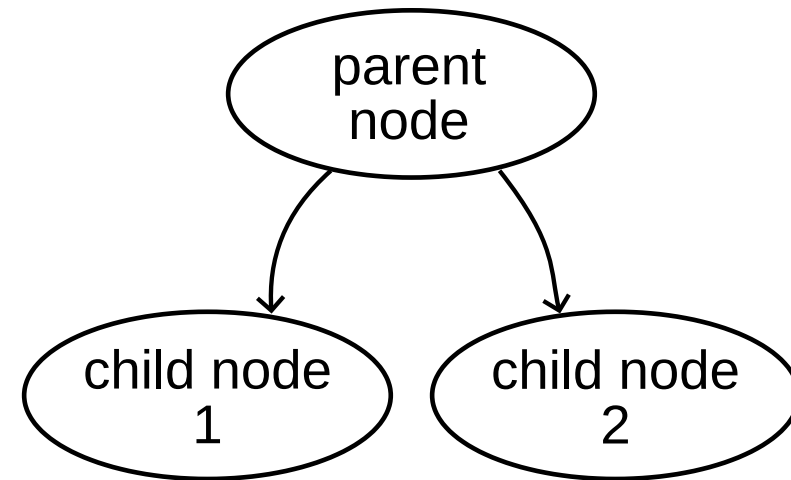


- Représentation d'une page Web avec une structure d'arbre ;
- Modèle interne utilisé par les navigateurs pour rendre une page ;
- Représentation mentale facilitant l'écriture de sélecteurs CSS ;
- Offre une API pour manipuler le DOM programmatiquement (voir cours JavaScript en BUT2).

Extrait de Wikipedia.



# NOTION D'ARBRES

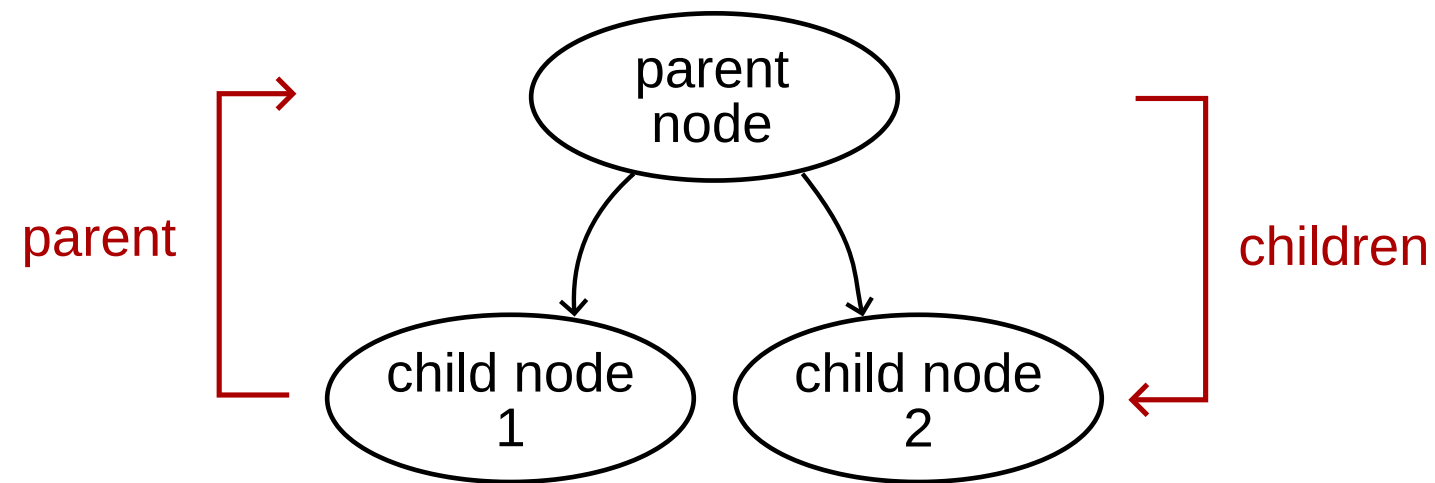


- DOM = arbre ;
- Noeuds = éléments HTML ;

Cours algorithmique en BUT2 : analyse et parcours d'arbres et de graphes.



# NOTION D'ARBRES



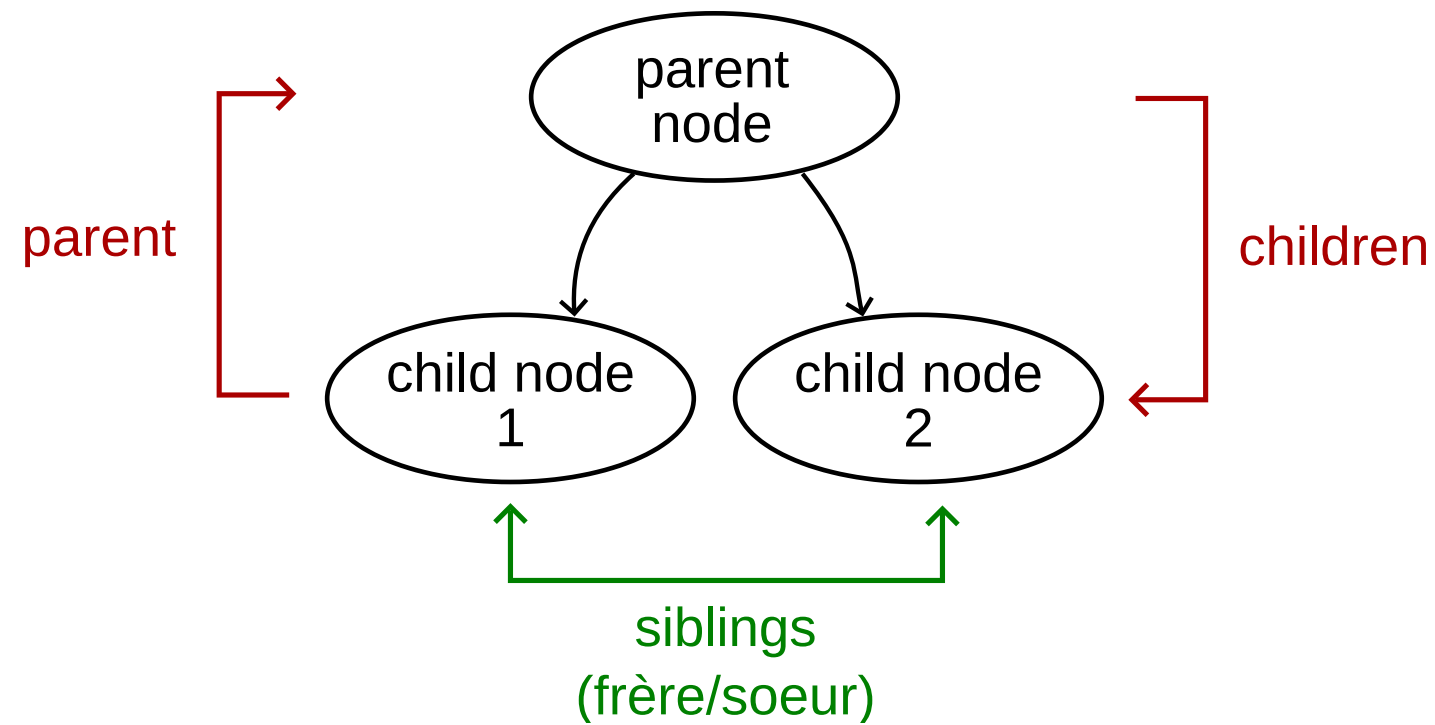
- DOM = arbre ;
- Noeuds = éléments HTML ;
- Arc = relation entre un élément parent (*parent node*) et un élément enfant (*child node*).

**Note :** un enfant n'a qu'un seul parent.

Cours algorithmique en BUT2 : analyse et parcours d'arbres et de graphes.



# NOTION D'ARBRES



- DOM = arbre ;
- Noeuds = éléments HTML ;
- Arc = relation entre un élément parent (*parent node*) et un élément enfant (*child node*).  
**Note** : un enfant n'a qu'un seul parent.
- Relation entre frères et sœurs : chaque enfant peut avoir un frère/soeur (siblings) à sa gauche et un à sa droite.  
**Note** : les enfants sont ordonnés (sens de lecture HTML).

Cours algorithmique en BUT2 : analyse et parcours d'arbres et de graphes.



# EXAMPLE

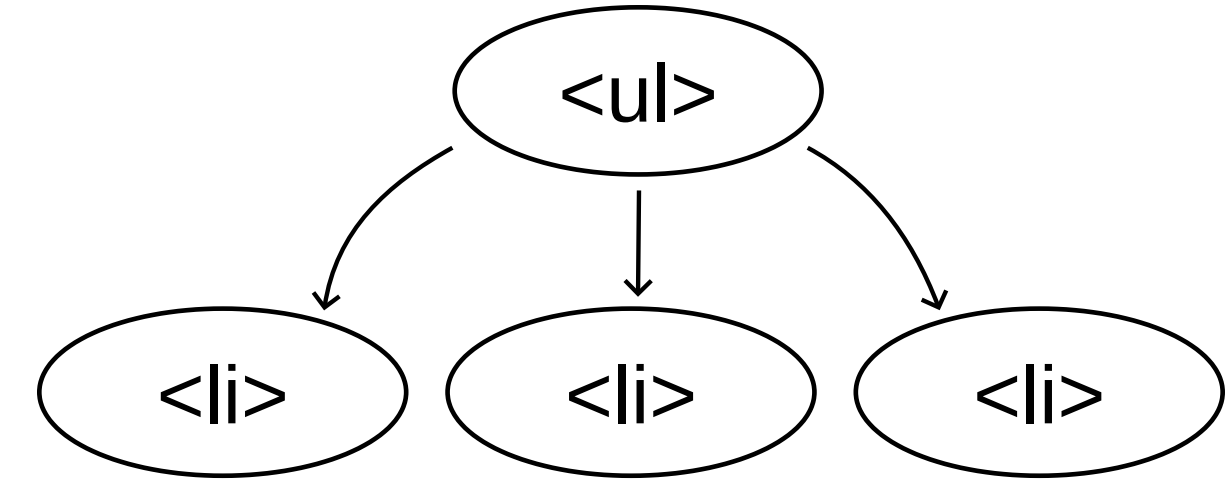
<ul>

```
<ul>
  <li>
    <a href="index.html">Accueil</a>
  </li>
  <li>
    <a href="faq.html">FAQ</a>
  </li>
  <li>
    <a href="contact.html">Me contacter</a>
  </li>
</ul>
```



# EXAMPLE

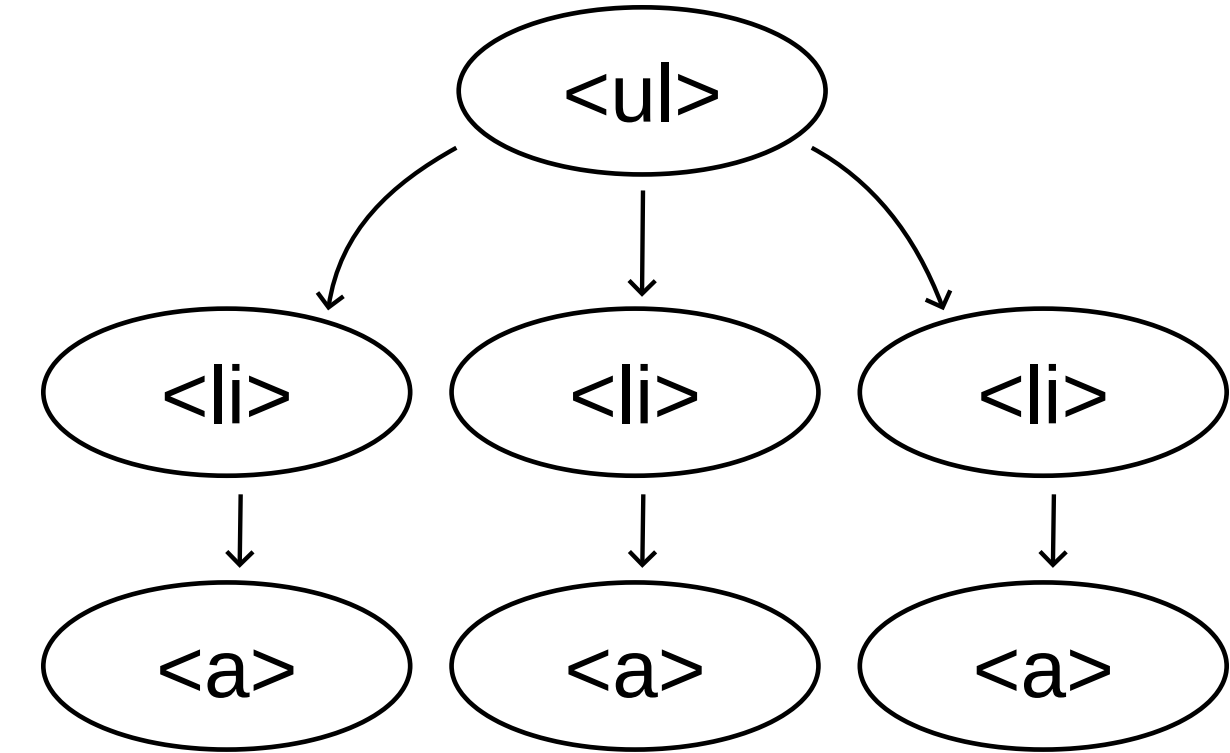
```
<ul>
  <li>
    <a href="index.html">Accueil</a>
  </li>
  <li>
    <a href="faq.html">FAQ</a>
  </li>
  <li>
    <a href="contact.html">Me contacter</a>
  </li>
</ul>
```





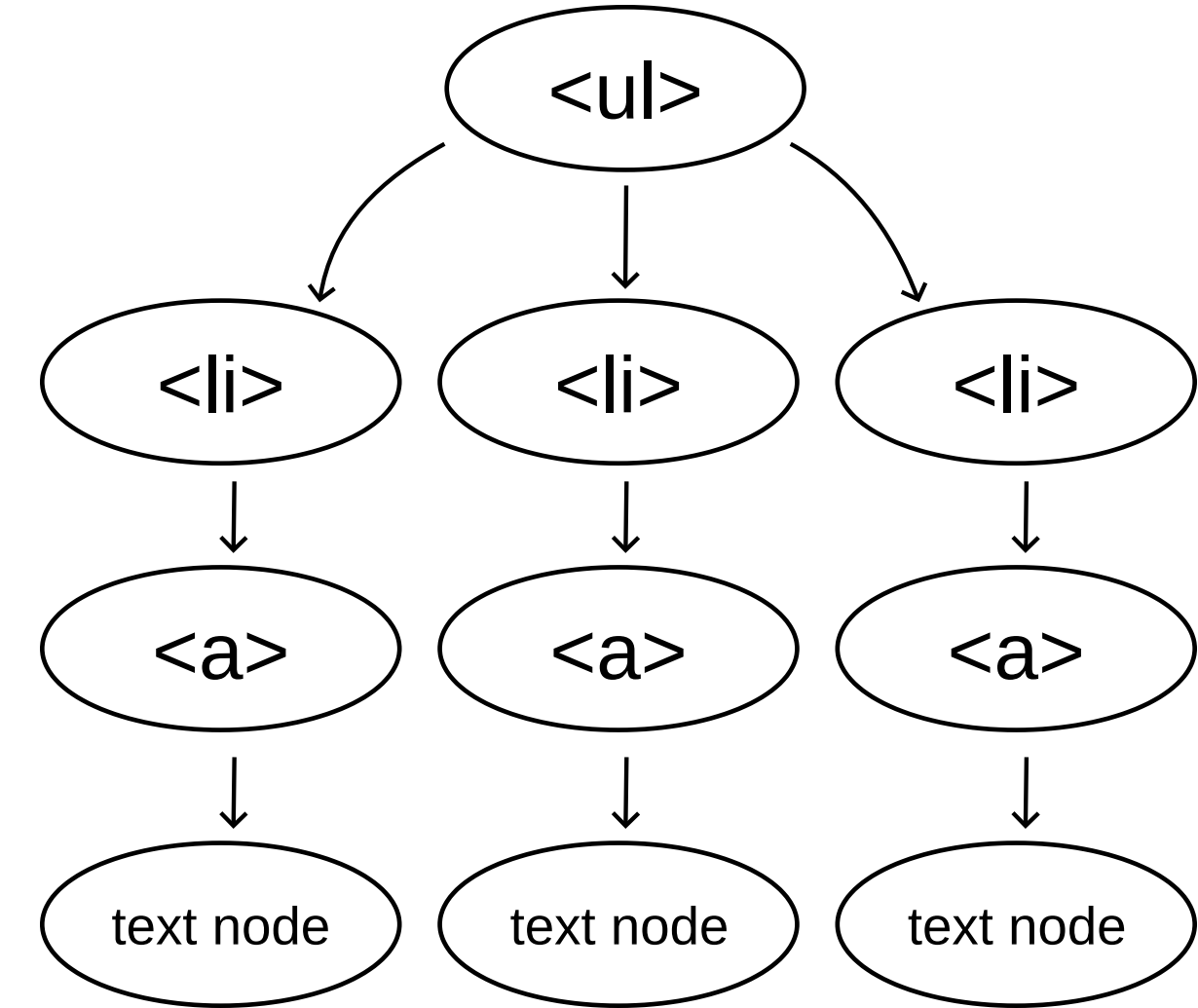
# EXAMPLE

```
<ul>
  <li>
    <a href="index.html">Accueil</a>
  </li>
  <li>
    <a href="faq.html">FAQ</a>
  </li>
  <li>
    <a href="contact.html">Me contacter</a>
  </li>
</ul>
```



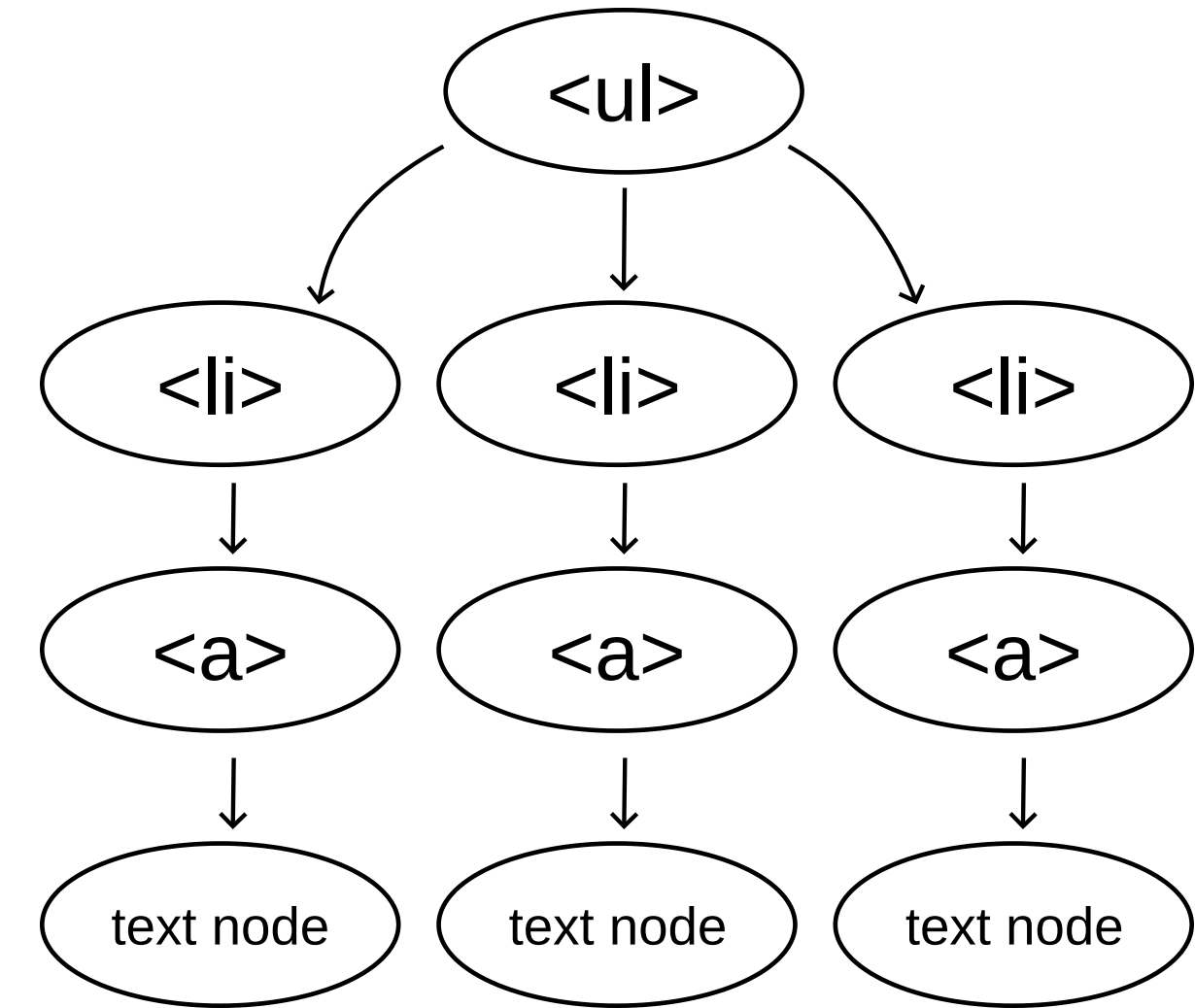
# EXAMPLE

```
<ul>
  <li>
    <a href="index.html">Accueil</a>
  </li>
  <li>
    <a href="faq.html">FAQ</a>
  </li>
  <li>
    <a href="contact.html">Me contacter</a>
  </li>
</ul>
```



# EXAMPLE

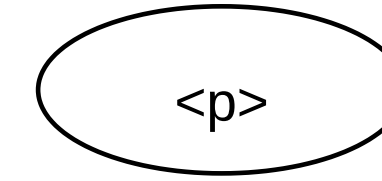
```
<ul>
  <li>
    <a href="index.html">Accueil</a>
  </li>
  <li>
    <a href="faq.html">FAQ</a>
  </li>
  <li>
    <a href="contact.html">Me contacter</a>
  </li>
</ul>
```



**Note :** en CSS, les sélecteurs s'appliquent aux noeuds éléments et pas aux noeuds textuels.



# EXEMPLE 2



```
<p>  
  Ceci est un  
  <strong>paragraphe</strong> avec  
  de la <u><i>mise en forme.</i></u>  
</p>
```

Rendu :

Ceci est un paragraphe avec de la *mise en forme.*



# EXEMPLE 2

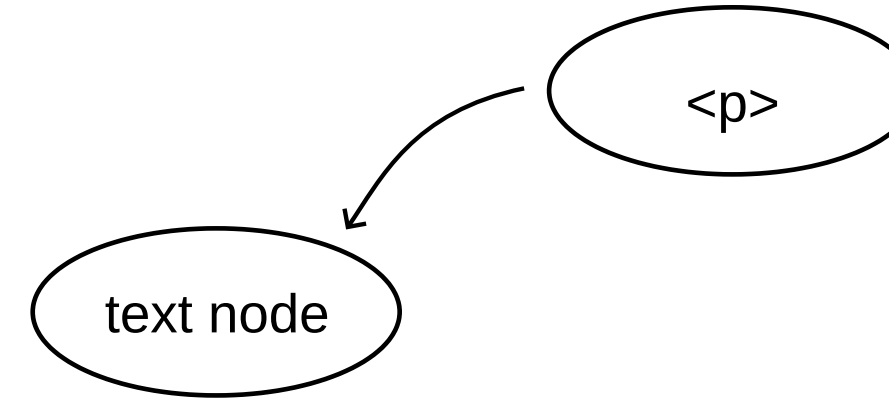
<p>

Ceci est un

<strong>paragraphe</strong> avec

de la <u><i>mise en forme.</i></u>

</p>



Rendu :

Ceci est un paragraphe avec de la *mise en forme.*

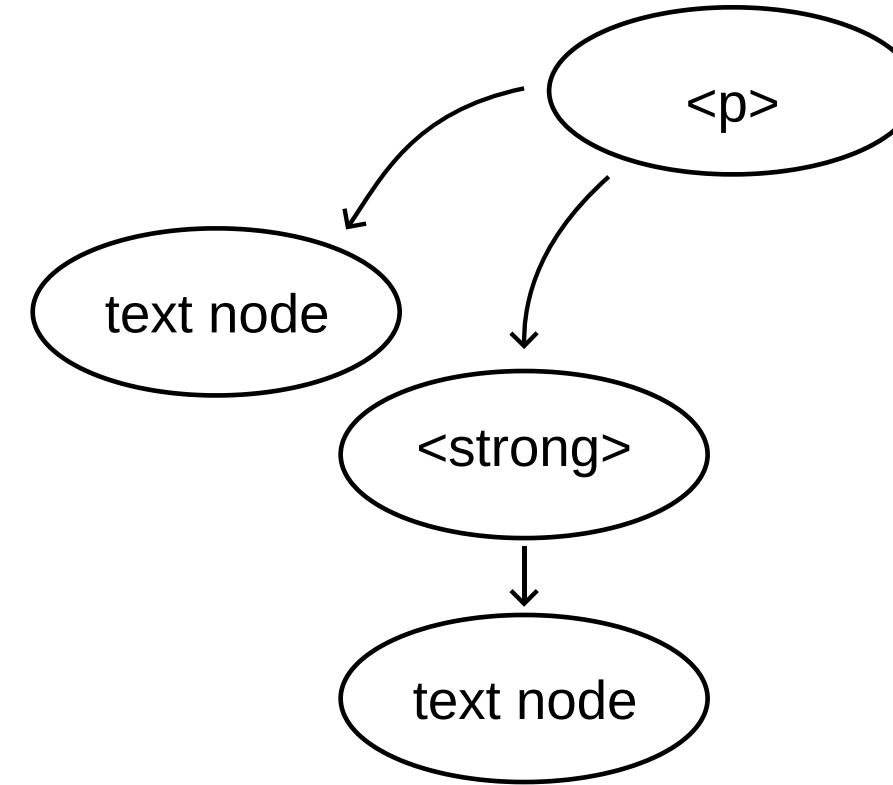


# EXEMPLE 2

```
<p>  
  Ceci est un  
  <strong>paragraphe</strong> avec  
  de la <u><i>mise en forme.</i></u>  
</p>
```

Rendu :

Ceci est un paragraphe avec de la *mise en forme.*

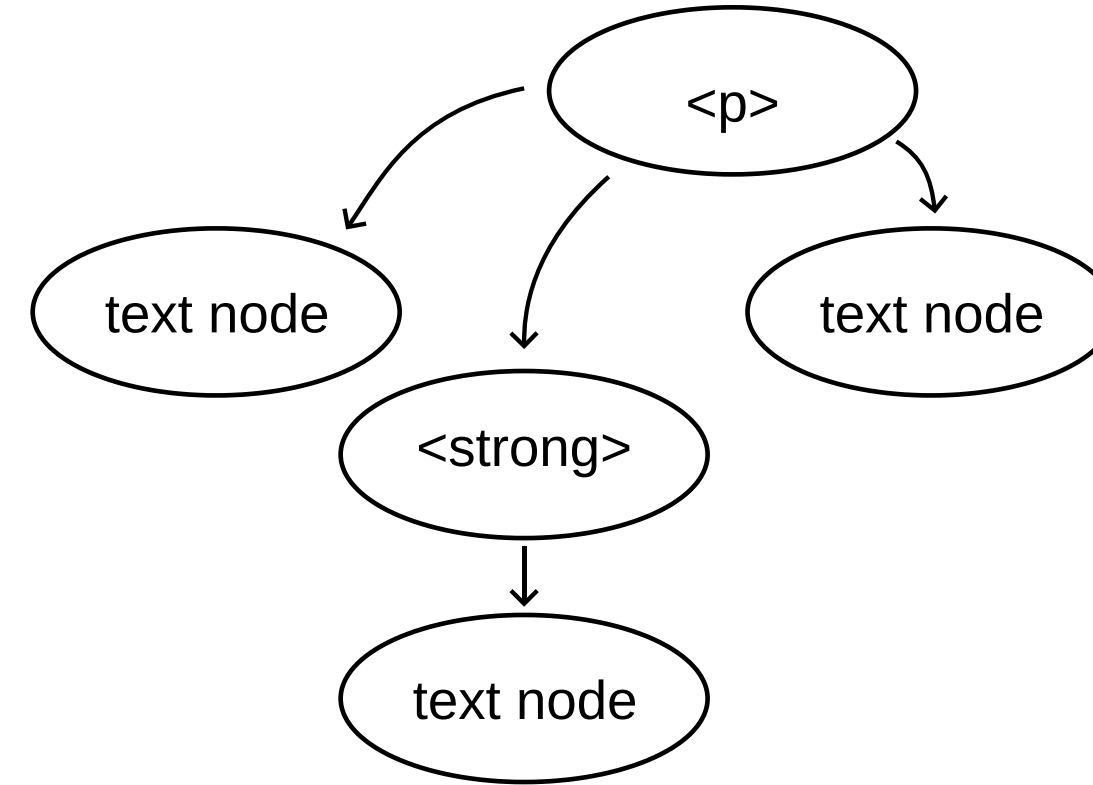


# EXEMPLE 2

```
<p>  
  Ceci est un  
  <strong>paragraphe</strong> avec  
  de la <u><i>mise en forme.</i></u>  
</p>
```

Rendu :

Ceci est un paragraphe avec de la *mise en forme.*

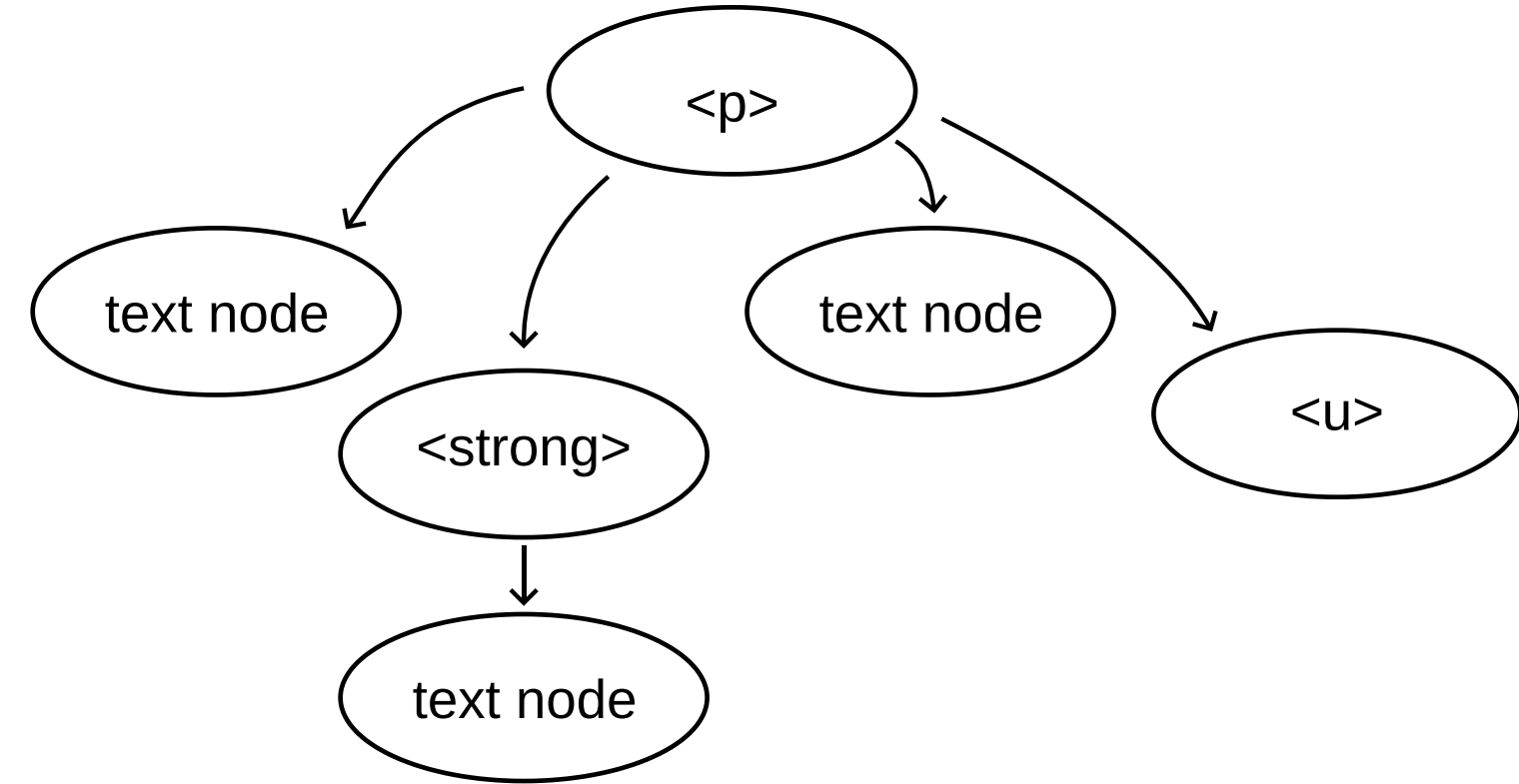


# EXEMPLE 2

```
<p>  
  Ceci est un  
  <strong>paragraphe</strong> avec  
  de la <u><i>mise en forme.</i></u>  
</p>
```

Rendu :

Ceci est un paragraphe avec de la *mise en forme.*



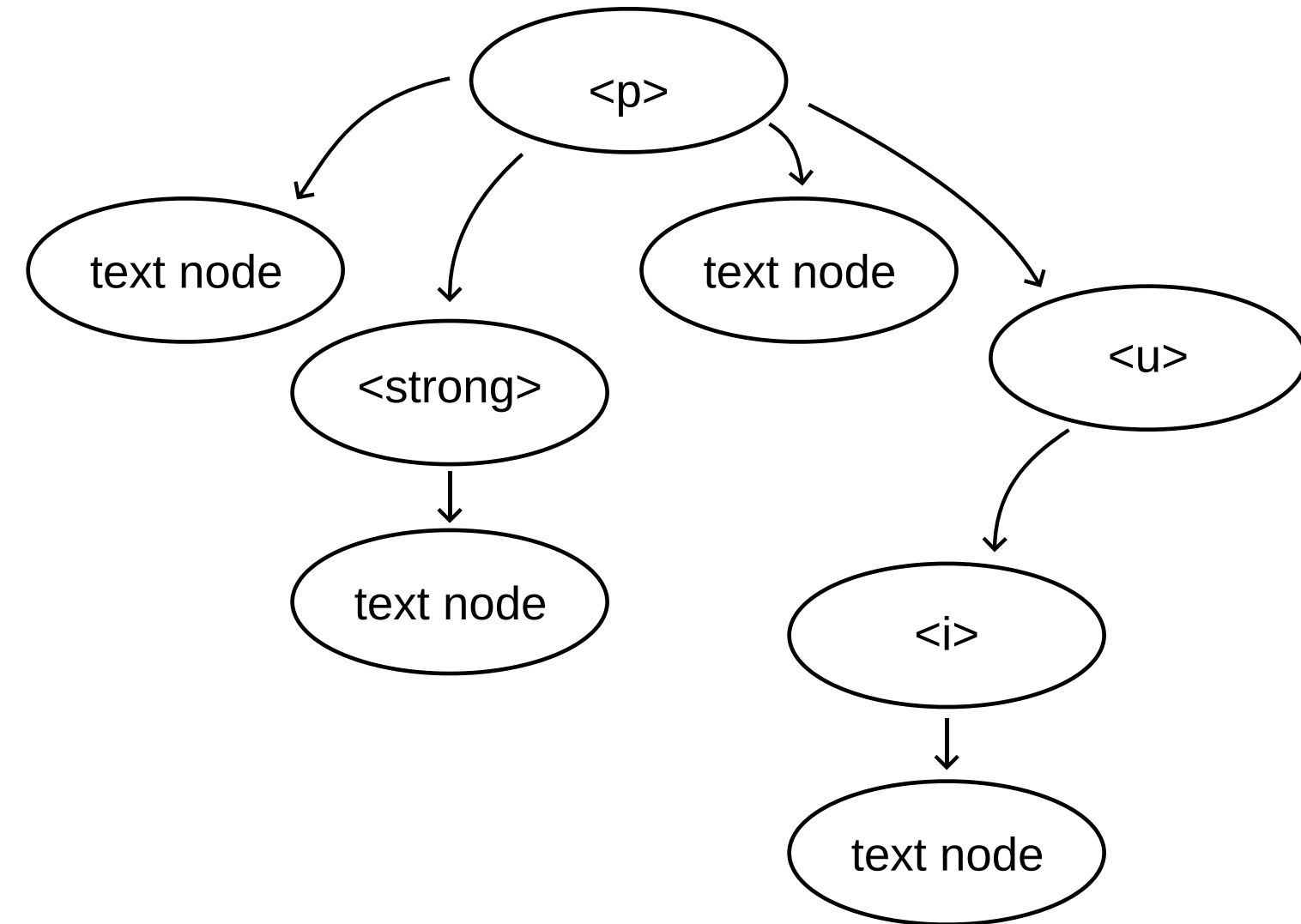


# EXEMPLE 2

```
<p>  
  Ceci est un  
  <strong>paragraphe</strong> avec  
  de la <u><i>mise en forme.</i></u>  
</p>
```

Rendu :

Ceci est un paragraphe avec de la *mise en forme.*

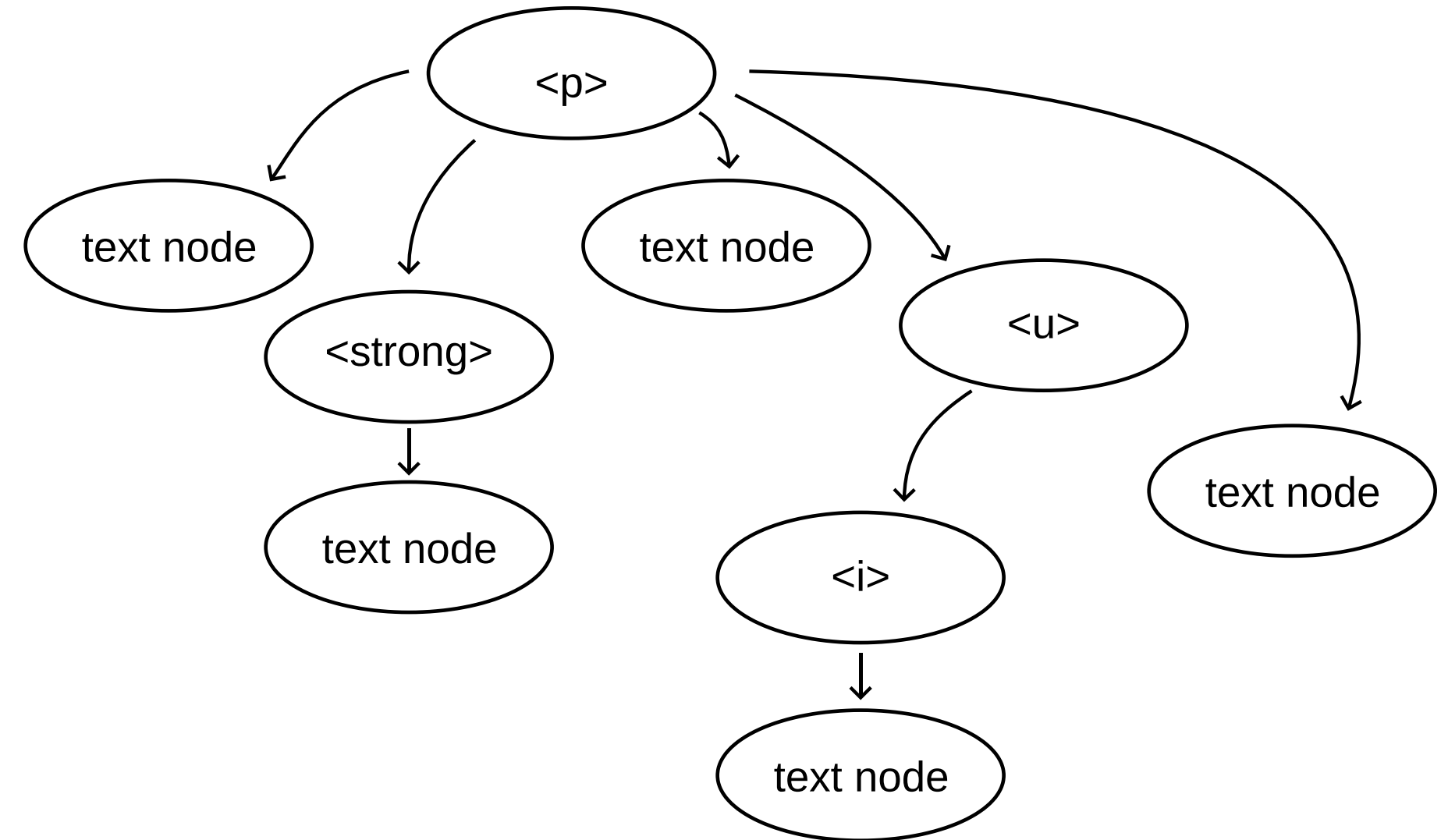


# EXEMPLE 2

```
<p>  
  Ceci est un  
  <strong>paragraphe</strong> avec  
  de la <u><i>mise en forme.</i></u>  
</p>
```

Rendu :

Ceci est un paragraphe avec de la *mise en forme.*

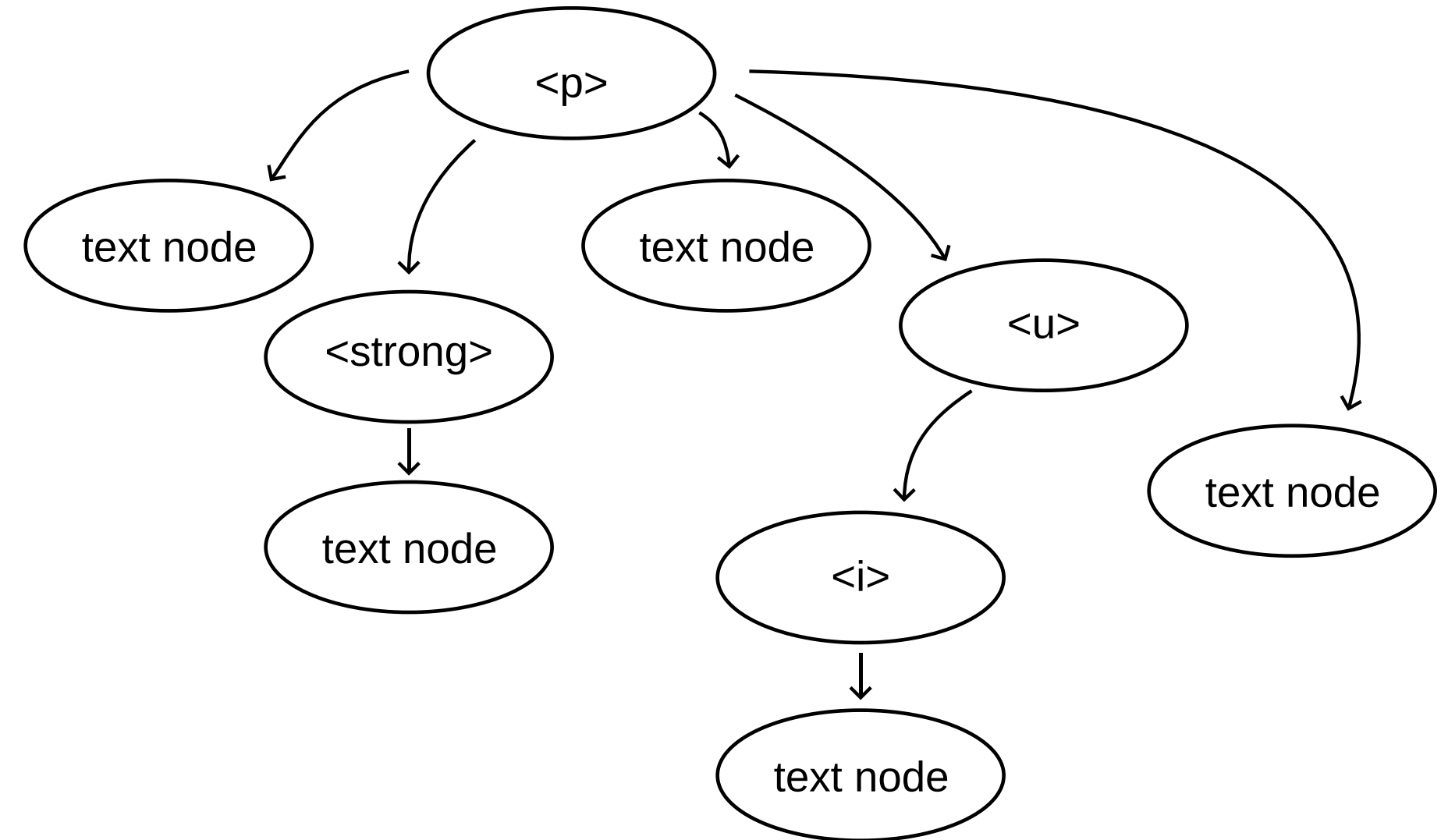


# EXEMPLE 2

```
<p>  
  Ceci est un  
  <strong>paragraphe</strong> avec  
  de la <u><i>mise en forme.</i></u>  
</p>
```

Rendu :

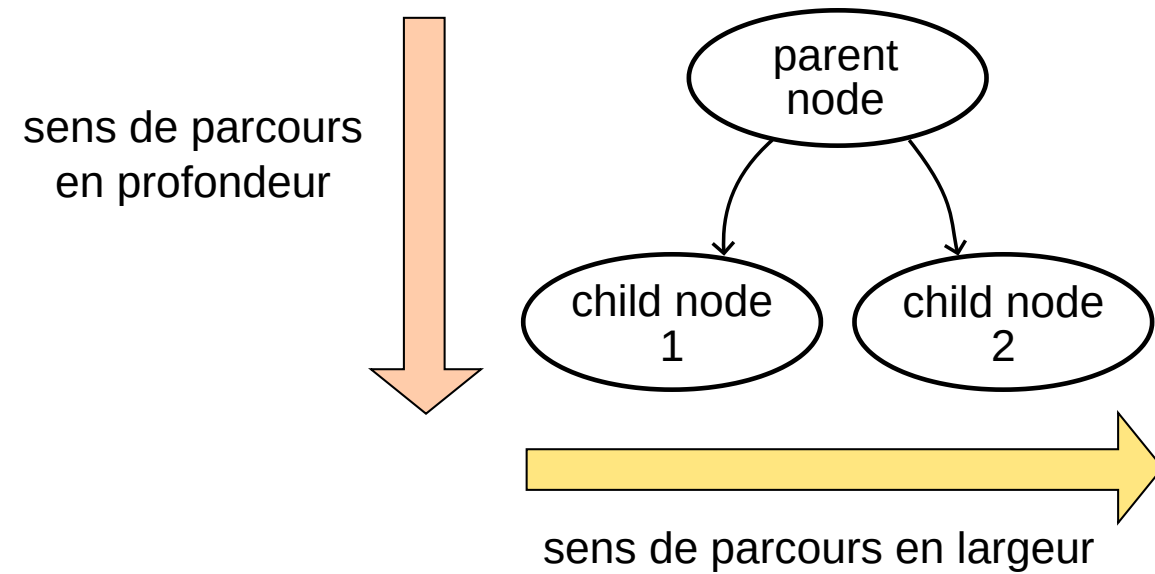
Ceci est un **paragraphe** avec de la *mise en forme.*



**Note :** un espace dans le code HTML est un caractère. Cela devient donc un noeud textuel. En JavaScript (BUT2), il faudra se rappeler de cette notion de noeuds textuels car elle a plus d'importance qu'en CSS.



# LIEN ENTRE LE DOM ET CSS



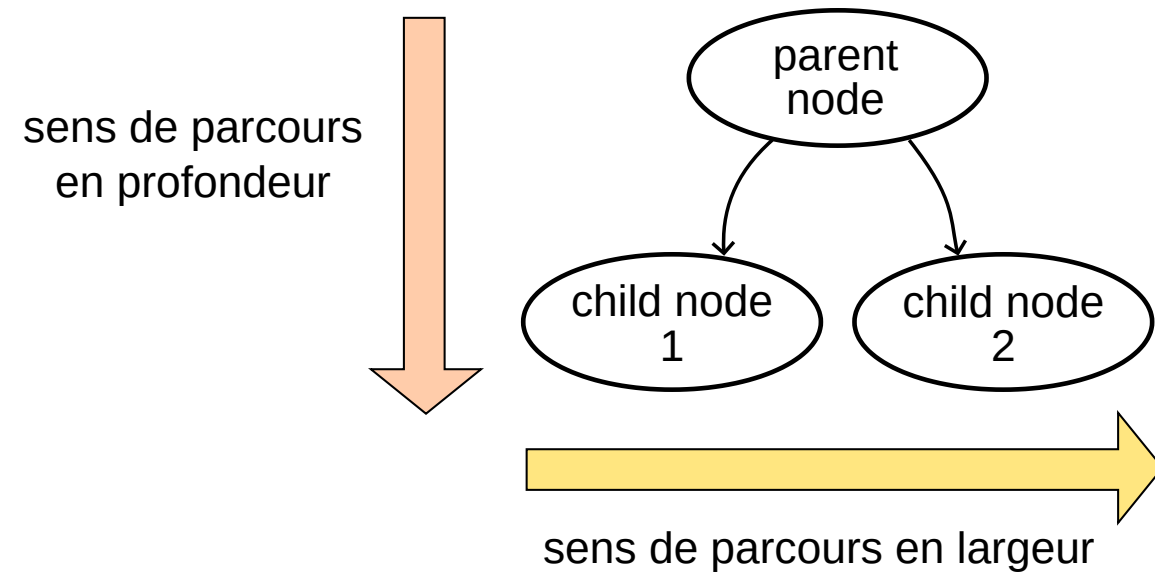
- Parcours en profondeur : sélecteurs « » (espace) et « > »
- Parcours en largeur : sélecteurs « + » et « ~ » (tilde)

## Exemples :

<code>ul li { /* ... */ }</code>	<code>/* tous les enfants */</code>
<code>ul &gt; li { /* ... */ }</code>	<code>/* enfants directs seulement */</code>
<code>li ~ li { /* ... */ }</code>	<code>/* éléments adjacents et non adjacents */</code>
<code>li + li { /* ... */ }</code>	<code>/* élément adjacent seulement */</code>



# LIEN ENTRE LE DOM ET CSS



- Parcours en profondeur : sélecteurs « » (espace) et « > »
- Parcours en largeur : sélecteurs « + » et « ~ » (tilde)

**Note :** on ne remonte pas et on ne revient pas en arrière.

## Exemples :

<code>ul li { /* ... */ }</code>	<code>/* tous les enfants */</code>
<code>ul &gt; li { /* ... */ }</code>	<code>/* enfants directs seulement */</code>
<code>li ~ li { /* ... */ }</code>	<code>/* éléments adjacents et non adjacents */</code>
<code>li + li { /* ... */ }</code>	<code>/* élément adjacent seulement */</code>



# RETOUR SUR CSS

## LE POSITIONNEMENT FLEX

# POSITIONNEMENT FLEX

- L'un des positionnements les plus importants à maîtriser aujourd'hui
- Ancêtre : inline-block, float
- Avantages : plus de contrôle, adaptation à l'écran
- Un autre positionnement important : grid (ne sera pas présenté dans ce cours)

## Usage :

```
nav ul {  
  display: flex;  
}
```

- **display**: flex; initialise un conteneur flex ;
- Tous les fils directs d'un contenu flex sont des items flex.



# LES DIFFÉRENTES PROPRIÉTÉS CSS

Nom	Description	Application
flex-direction	Change l'axe principal et secondaire	Conteneur Flex
justify-content	Alignement sur l'axe principal	
gap	Espacement sur l'axe principal	
flex-wrap	Passage (ou non) à la ligne en cas d'espace restreint	
align-items	Alignement sur l'axe secondaire	
flex-basis	Taille initiale de l'item	Items Flex
flex-grow	Autorise l'élargissement au delà de la taille initiale	
flex-shrink	Autorise le rétrécissement en deçà de la taille initiale	

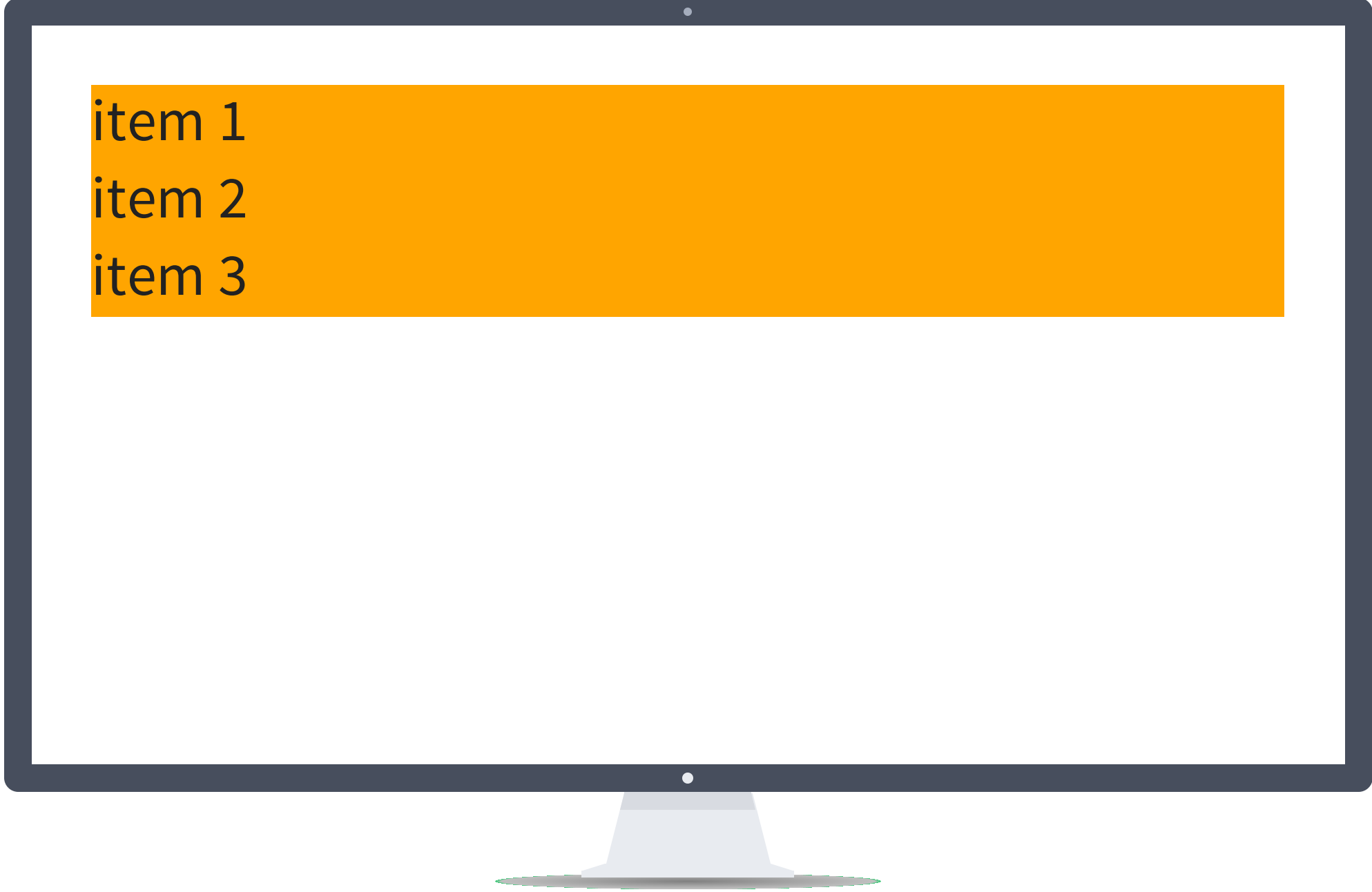




# EXAMPLE

index.html

```
1 <div class="container">
2   <div>item 1</div>
3   <div>item 2</div>
4   <div>item 3</div>
5 </div>
```



item 1  
item 2  
item 3

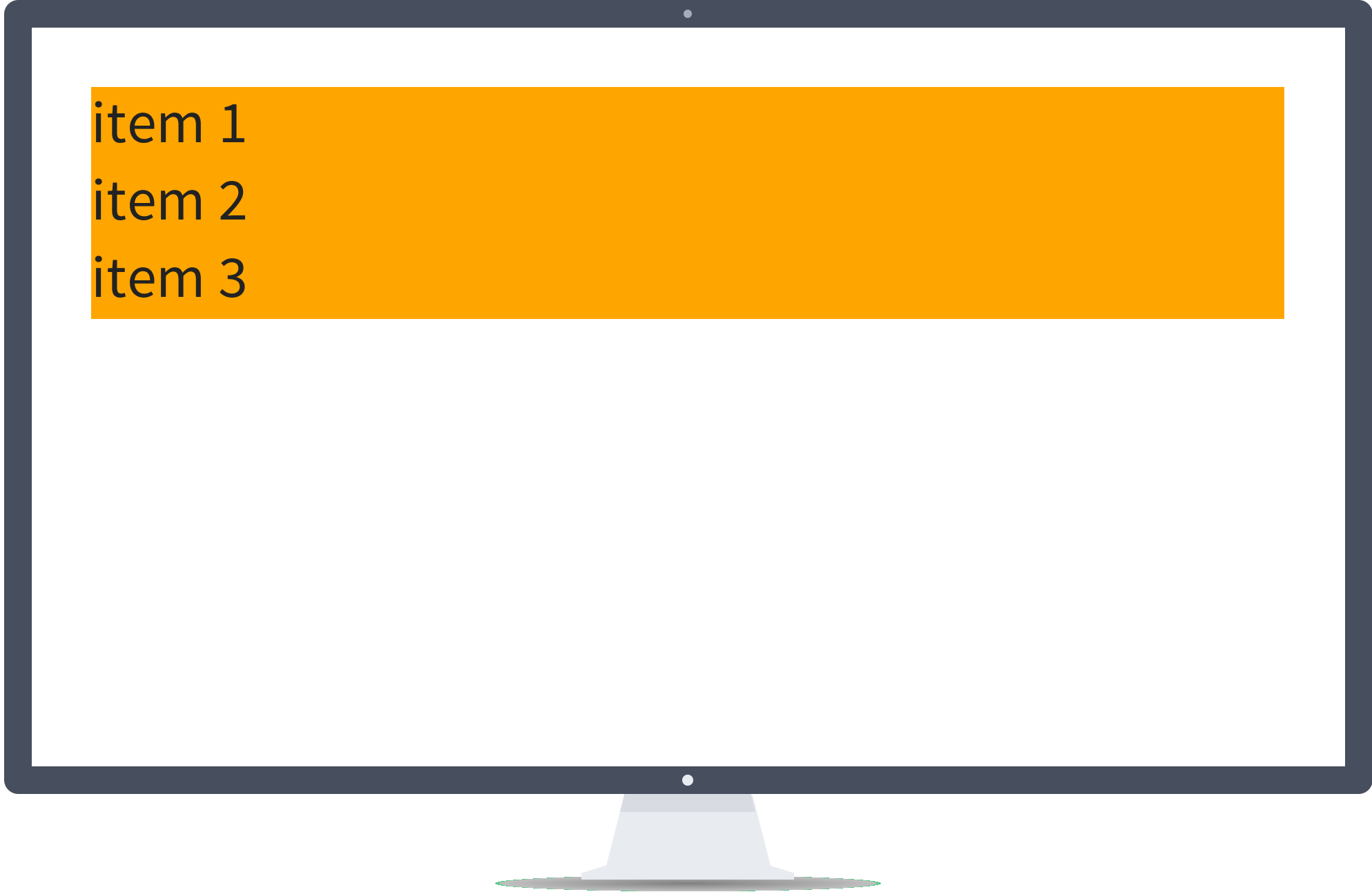


# EXAMPLE

index.html

style.css

```
1 .container {  
2   display: flex;  
3   background-color: orange;  
4 }  
5  
6 .container div {  
7   background-color: yellow;  
8   border: thick solid black;  
9   text-align: center;  
10  line-height: 50px;  
11 }
```



item 1  
item 2  
item 3

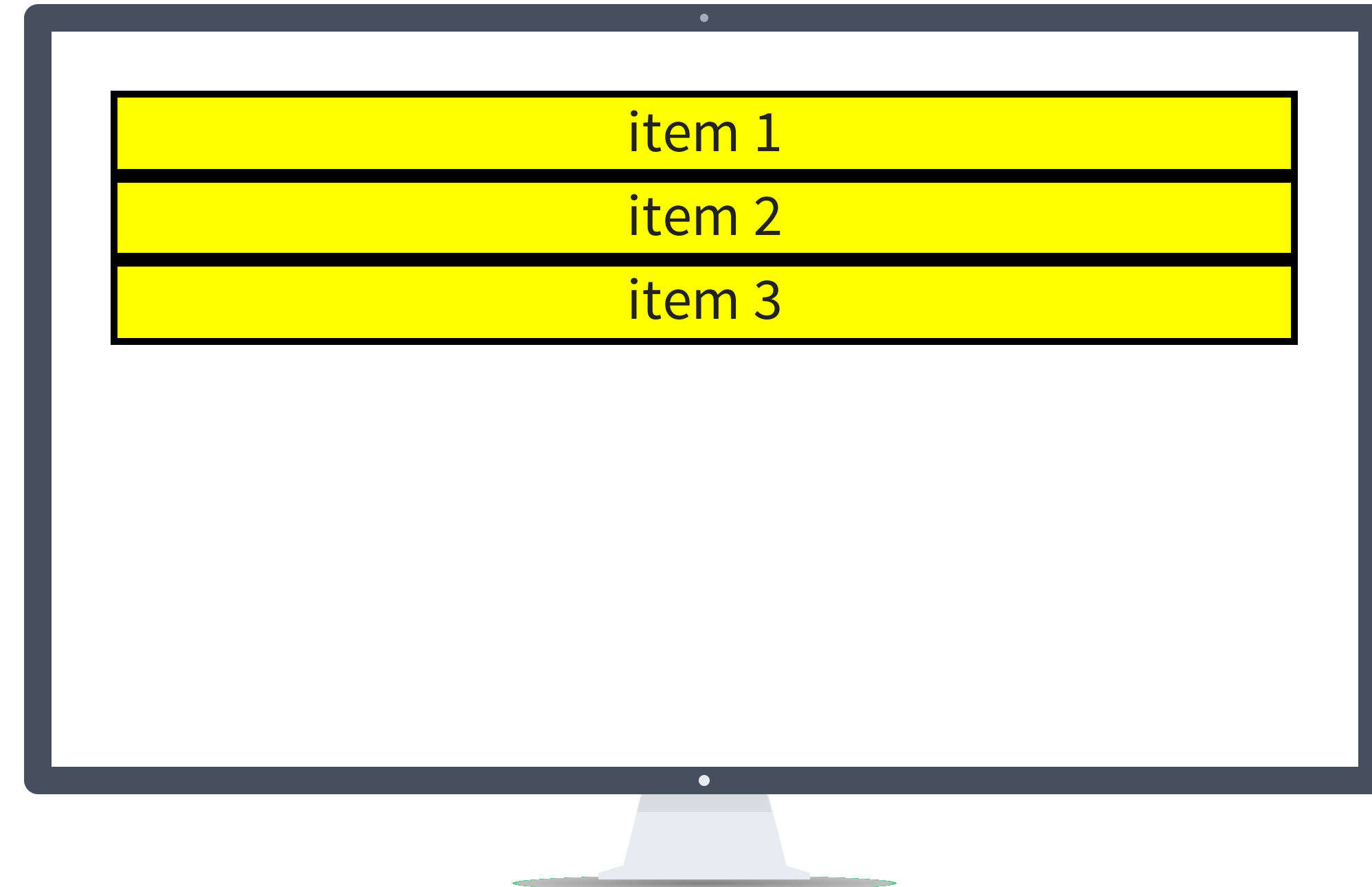


# EXAMPLE

index.html

style.css

```
1 .container {  
2   display: flex;  
3   background-color: orange;  
4 }  
5  
6 .container div {  
7   background-color: yellow;  
8   border: thick solid black;  
9   text-align: center;  
10  line-height: 50px;  
11 }
```

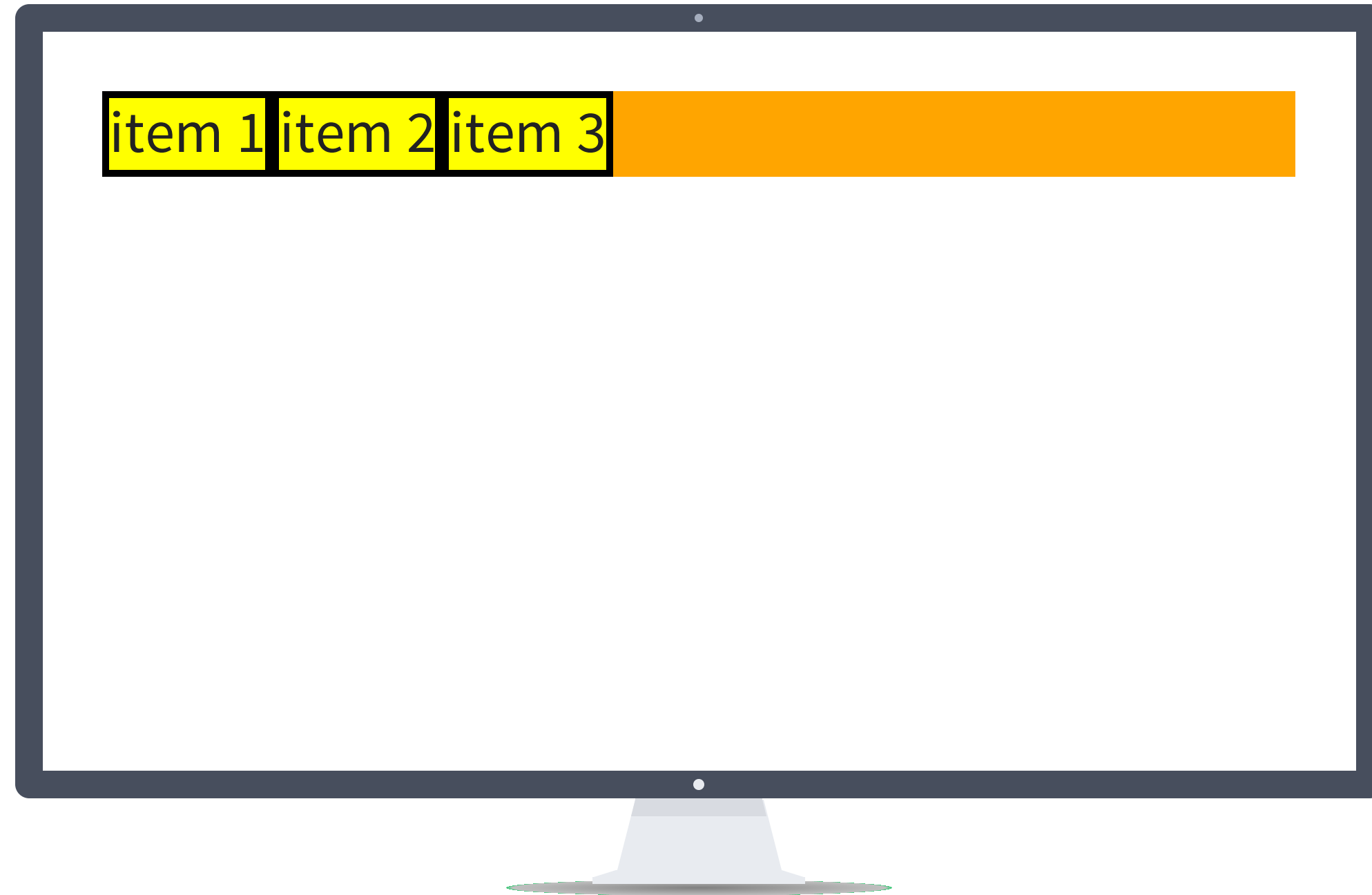


# EXAMPLE

index.html

style.css

```
1 .container {  
2   display: flex;  
3   background-color: orange;  
4 }  
5  
6 .container div {  
7   background-color: yellow;  
8   border: thick solid black;  
9   text-align: center;  
10  line-height: 50px;  
11 }
```



# POSITIONNEMENT FLEX : NOTION D'AXES

2 axes sont utilisés :

- Axe primaire (par défaut horizontal)
- Axe secondaire (par défaut, vertical)
- Remplacement de l'axe primaire/secondaire avec **flex-direction**

```
1 .container {  
2   flex-direction: row|column|row-reversed|column-reversed;  
3 }
```



# ALIGNEMENT

Alignement sur l'axe principal :

```
justify-content: flex-start | flex-end | start | end | center | space-between | space-around | space-evenly ;
```

**Note** : en dehors des trois valeurs space-\*, pas d'espace entre les éléments. Pour forcer un espace :

```
gap: <valeur><unit> ;
```

Alignement sur l'axe secondaire :

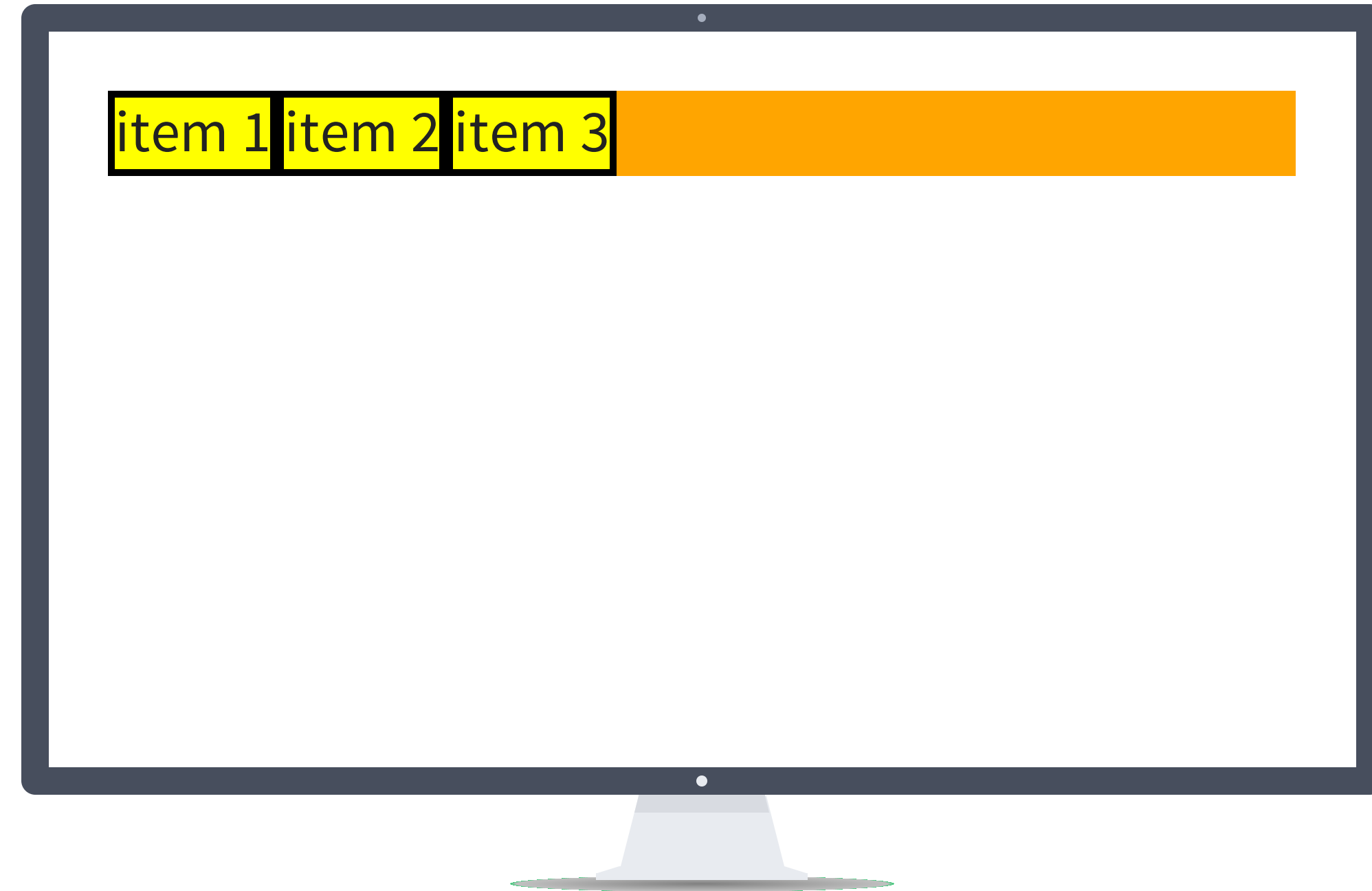
```
align-items: start | end | center | stretch ;
```



# EXAMPLE

```
1 .container {  
2   background-color: orange;  
3   display: flex;  
4   justify-content: center;  
5   gap: 50px;  
6   height: 200px;  
7   align-items: end;  
8 }
```

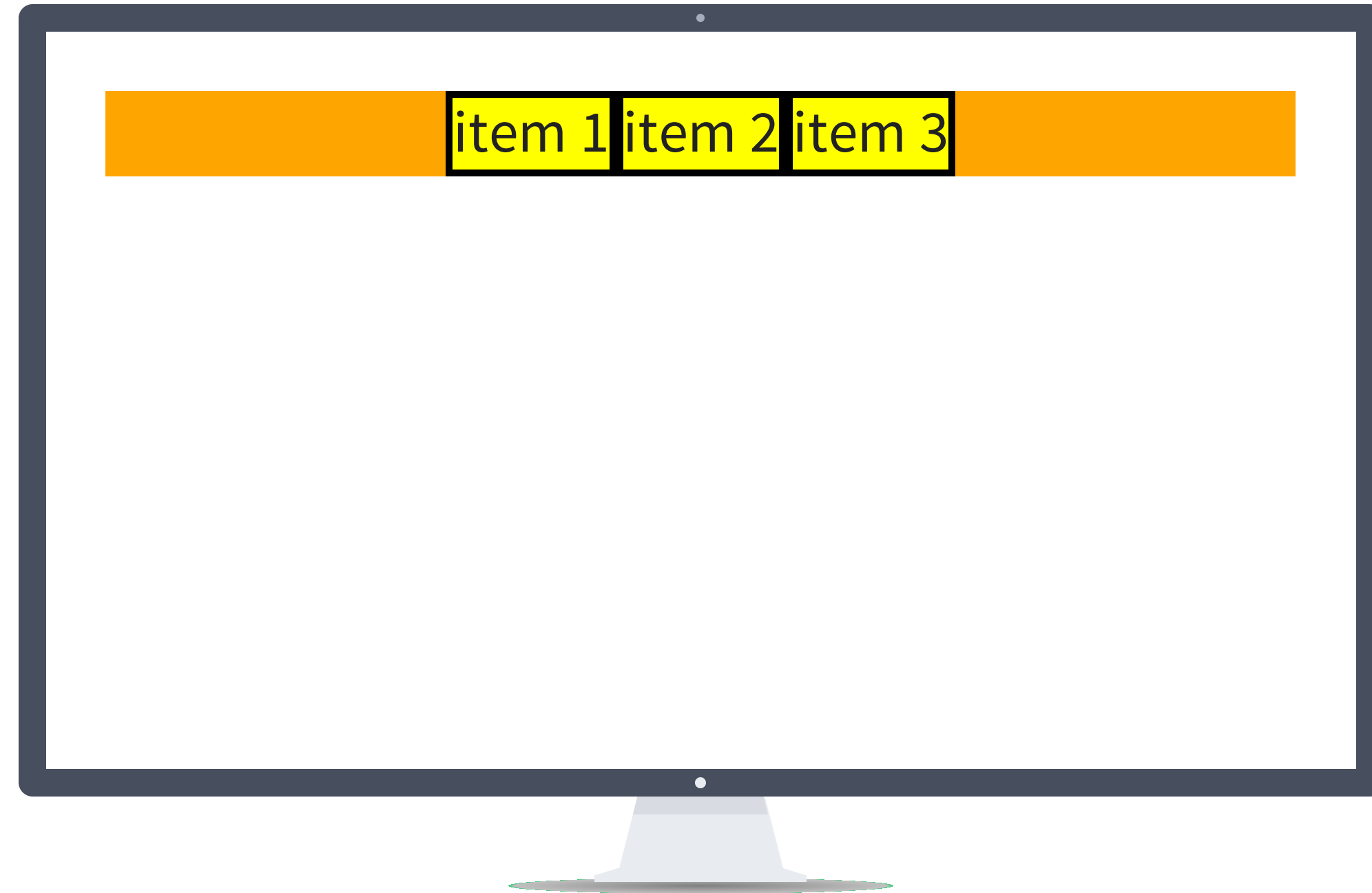
Note : par défaut : `align-items` : stretch.



# EXAMPLE

```
1 .container {  
2   background-color: orange;  
3   display: flex;  
4   justify-content: center;  
5   gap: 50px;  
6   height: 200px;  
7   align-items: end;  
8 }
```

Note : par défaut : `align-items` : stretch.

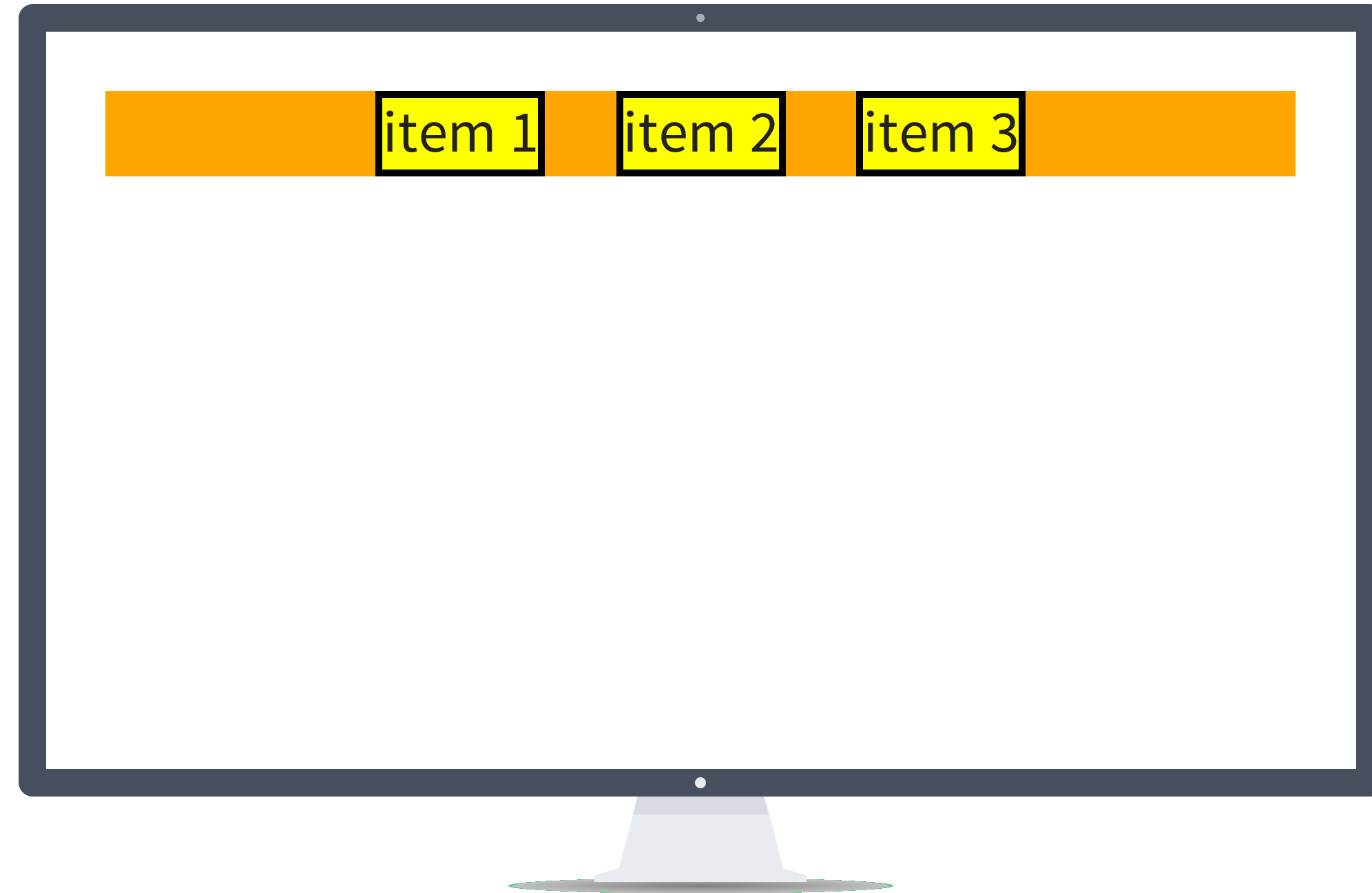




# EXAMPLE

```
1 .container {  
2   background-color: orange;  
3   display: flex;  
4   justify-content: center;  
5   gap: 50px;  
6   height: 200px;  
7   align-items: end;  
8 }
```

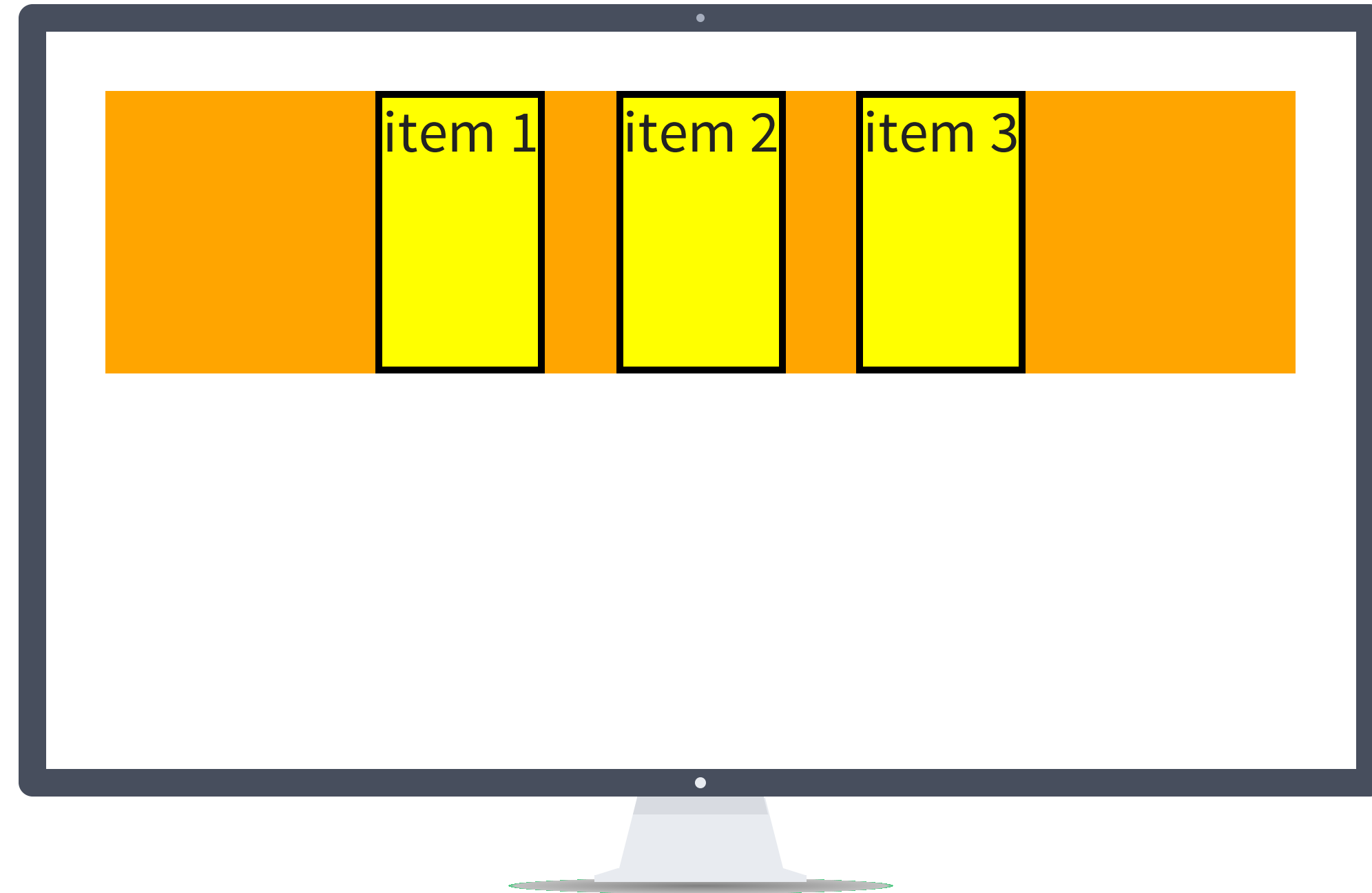
Note : par défaut : **align-items**: stretch.



# EXAMPLE

```
1 .container {  
2   background-color: orange;  
3   display: flex;  
4   justify-content: center;  
5   gap: 50px;  
6   height: 200px;  
7   align-items: end;  
8 }
```

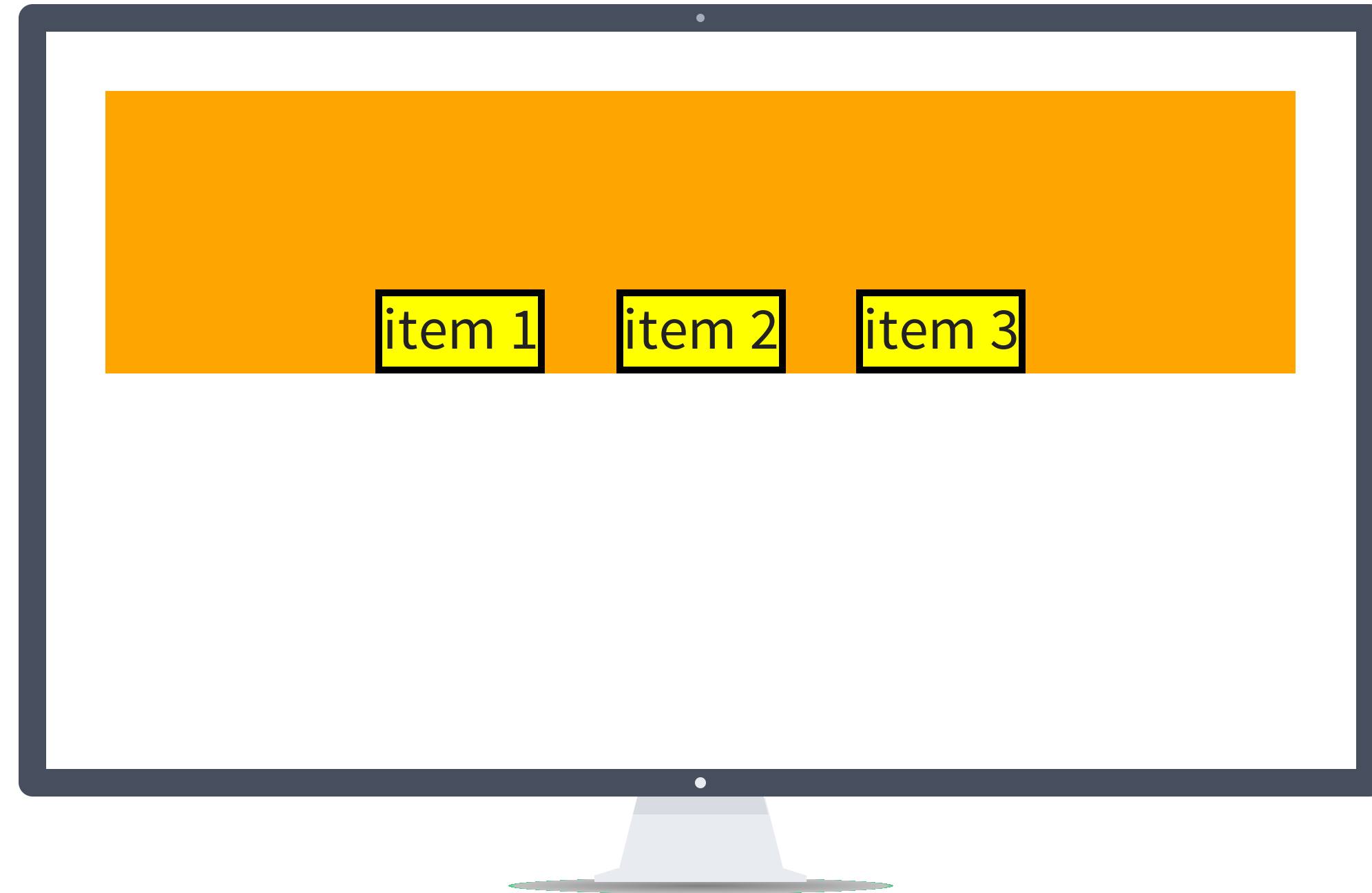
Note : par défaut : `align-items` : stretch.



# EXAMPLE

```
1 .container {  
2   background-color: orange;  
3   display: flex;  
4   justify-content: center;  
5   gap: 50px;  
6   height: 200px;  
7   align-items: end;  
8 }
```

Note : par défaut : `align-items` : stretch.



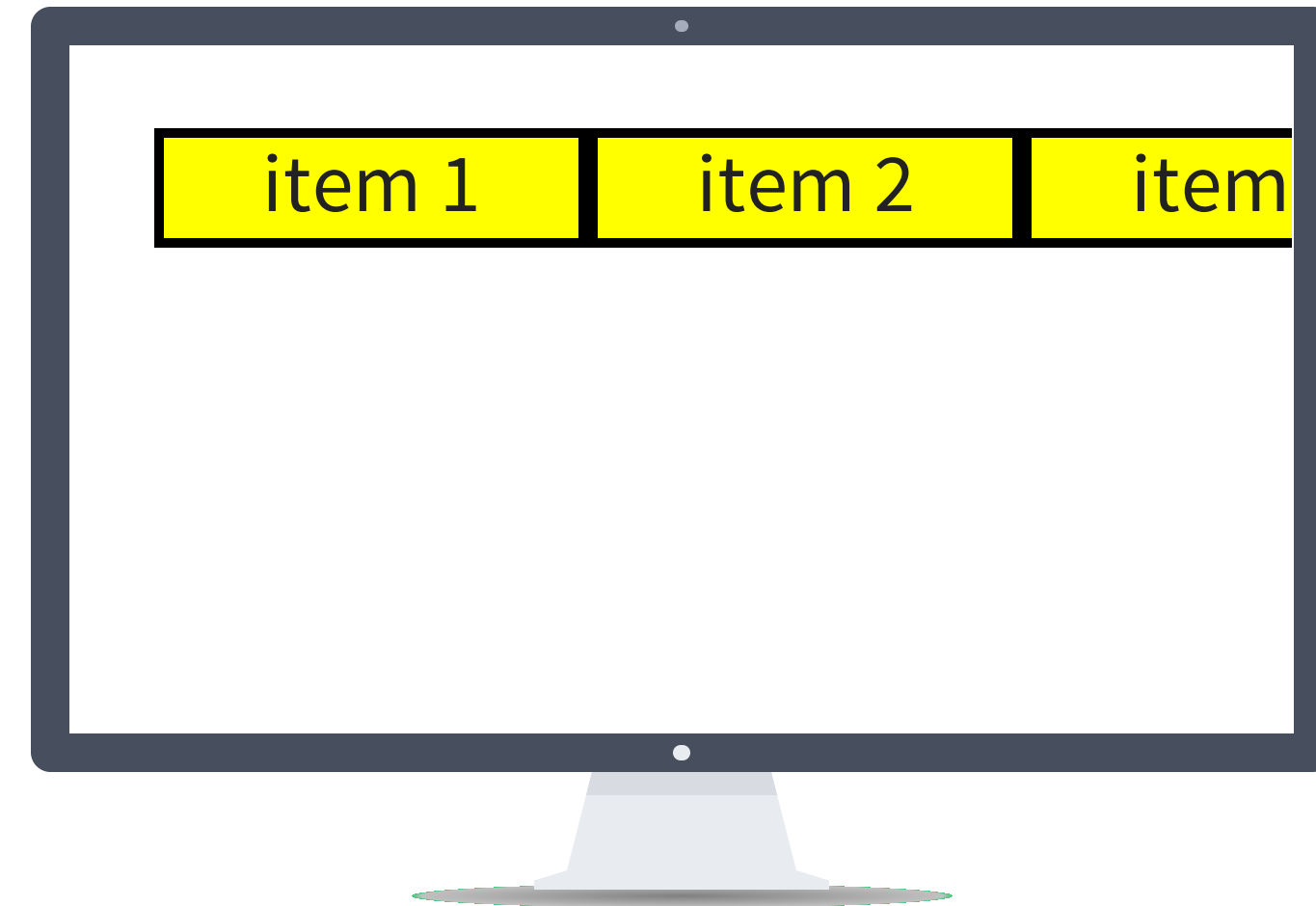
# FLEX-WRAP

Passage à la ligne avec `flex-wrap` :

```
flex-wrap: nowrap|wrap|wrap-reverse ;
```

Exemple :

```
1 .container {  
2   flex-wrap: nowrap;  
3   flex-wrap: wrap;  
4   flex-wrap: wrap-reverse;  
5 }
```



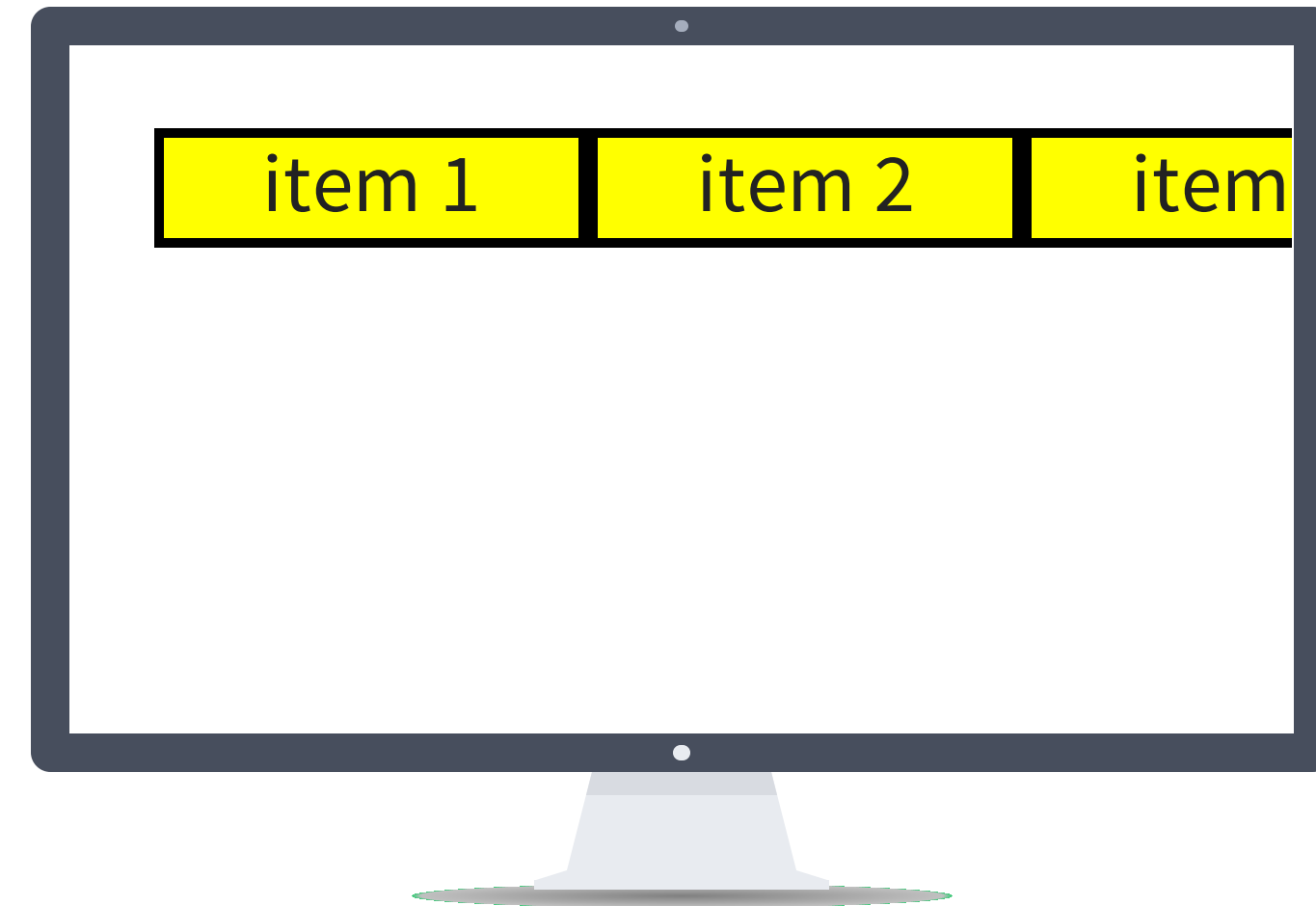
# FLEX-WRAP

Passage à la ligne avec **flex-wrap** :

```
flex-wrap: nowrap|wrap|wrap-reverse ;
```

Exemple :

```
1 .container {  
2   flex-wrap: nowrap;  
3   flex-wrap: wrap;  
4   flex-wrap: wrap-reverse;  
5 }
```



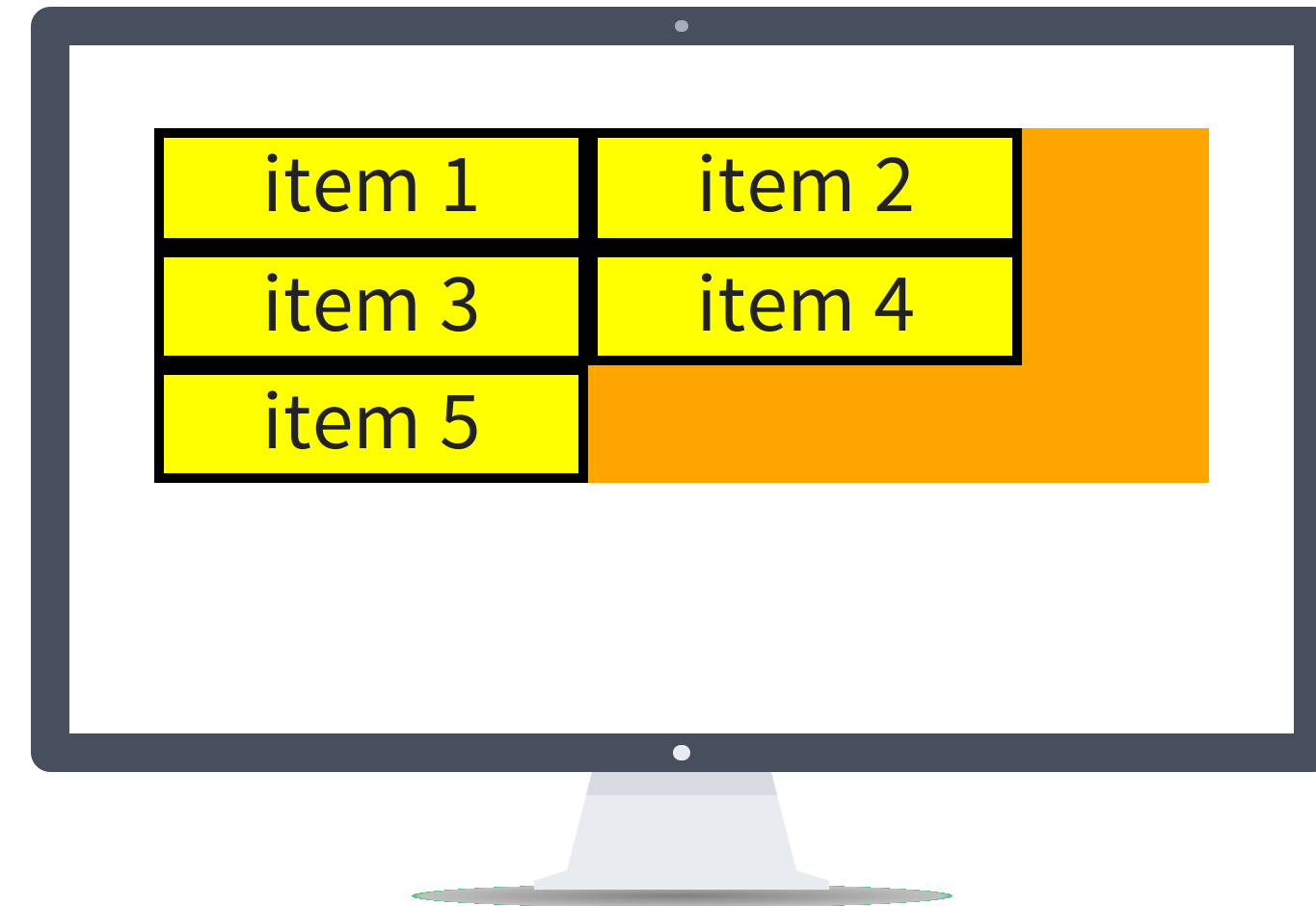
# FLEX-WRAP

Passage à la ligne avec `flex-wrap` :

```
flex-wrap: nowrap|wrap|wrap-reverse ;
```

Exemple :

```
1 .container {  
2   flex-wrap: nowrap;  
3   flex-wrap: wrap;  
4   flex-wrap: wrap-reverse;  
5 }
```



# FLEX-WRAP

Passage à la ligne avec `flex-wrap` :

```
flex-wrap: nowrap|wrap|wrap-reverse ;
```

Exemple :

```
1 .container {  
2   flex-wrap: nowrap;  
3   flex-wrap: wrap;  
4   flex-wrap: wrap-reverse;  
5 }
```



# PROPRIÉTÉS APPLIQUÉES AUX ITEMS

## flex-grow :

Facteur d'élargissement d'un item Flex, par rapport aux autres items appartenant au même conteneur. ;

## flex-shrink :

Facteur de rétrécissement d'un item Flex par rapport aux autres items appartenant au même conteneur ;

## flex-basis :

Taille initiale d'un item.

```
1 .container {  
2   flex-grow: 0;  
3   flex-shrink: 1;  
4   flex-basis: auto;  
5 }
```

Par défaut :

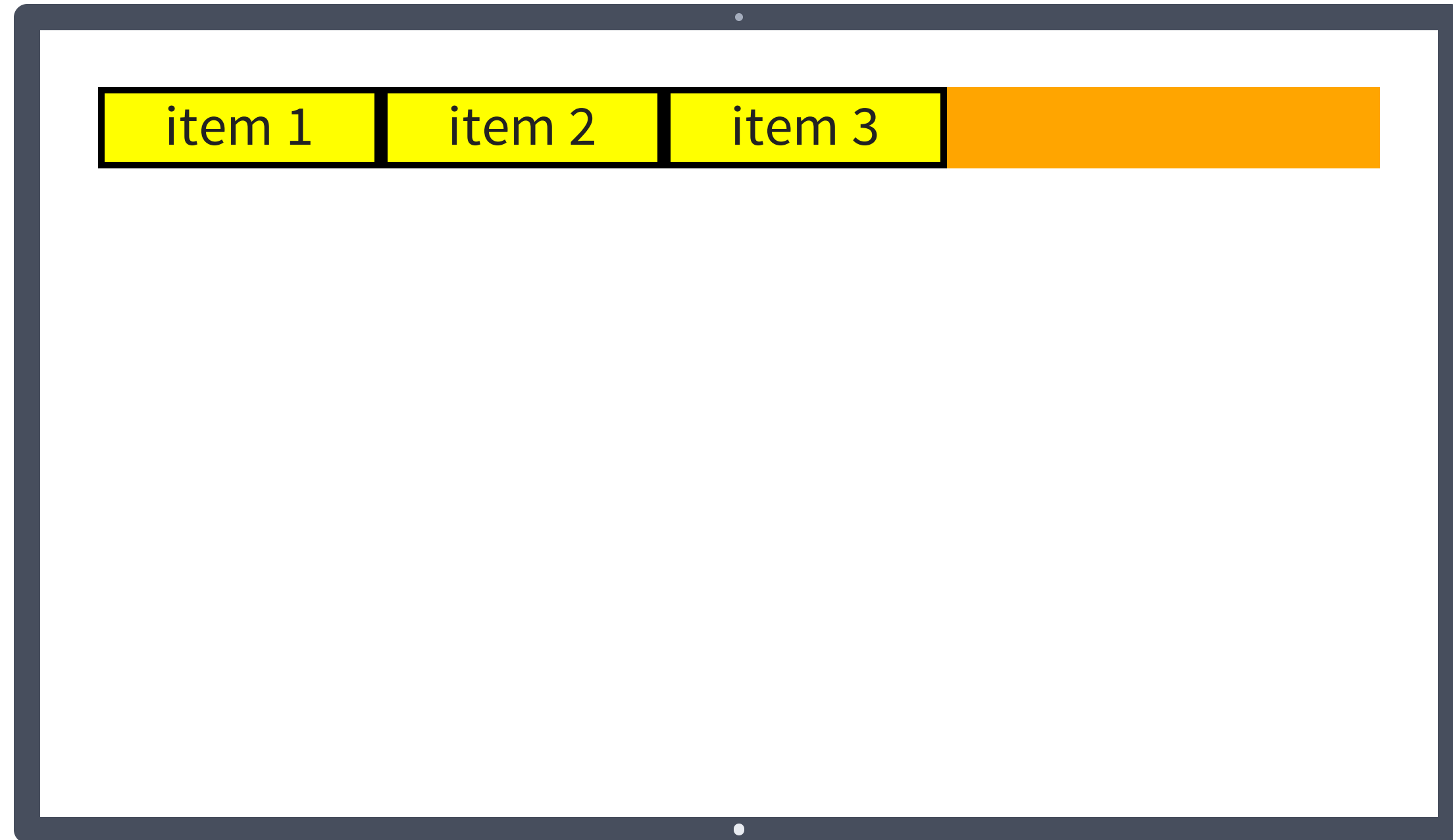
- Un item ne grossit pas, mais peut rétrécir ;
- Taille initiale = largeur (propriété *width*) si axe principal horizontal ou hauteur (propriété *height*) si axe principal vertical.



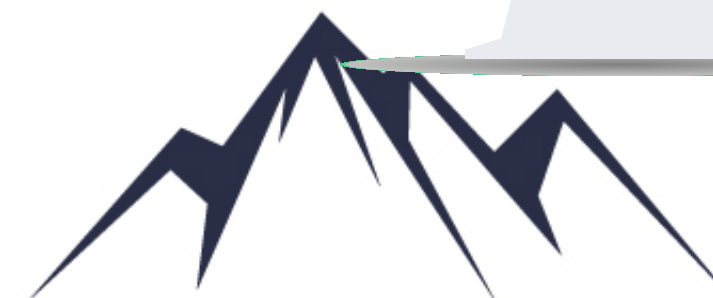


# EXEMPLE (ÉLARGISSEMENT)

```
1 .container div {  
2   flex-basis: 220px;  
3 }  
4  
5 .container div:first-child {  
6   flex-grow: 1;  
7 }  
8  
9 .container div:last-child {  
10  flex-grow: 3;  
11 }
```

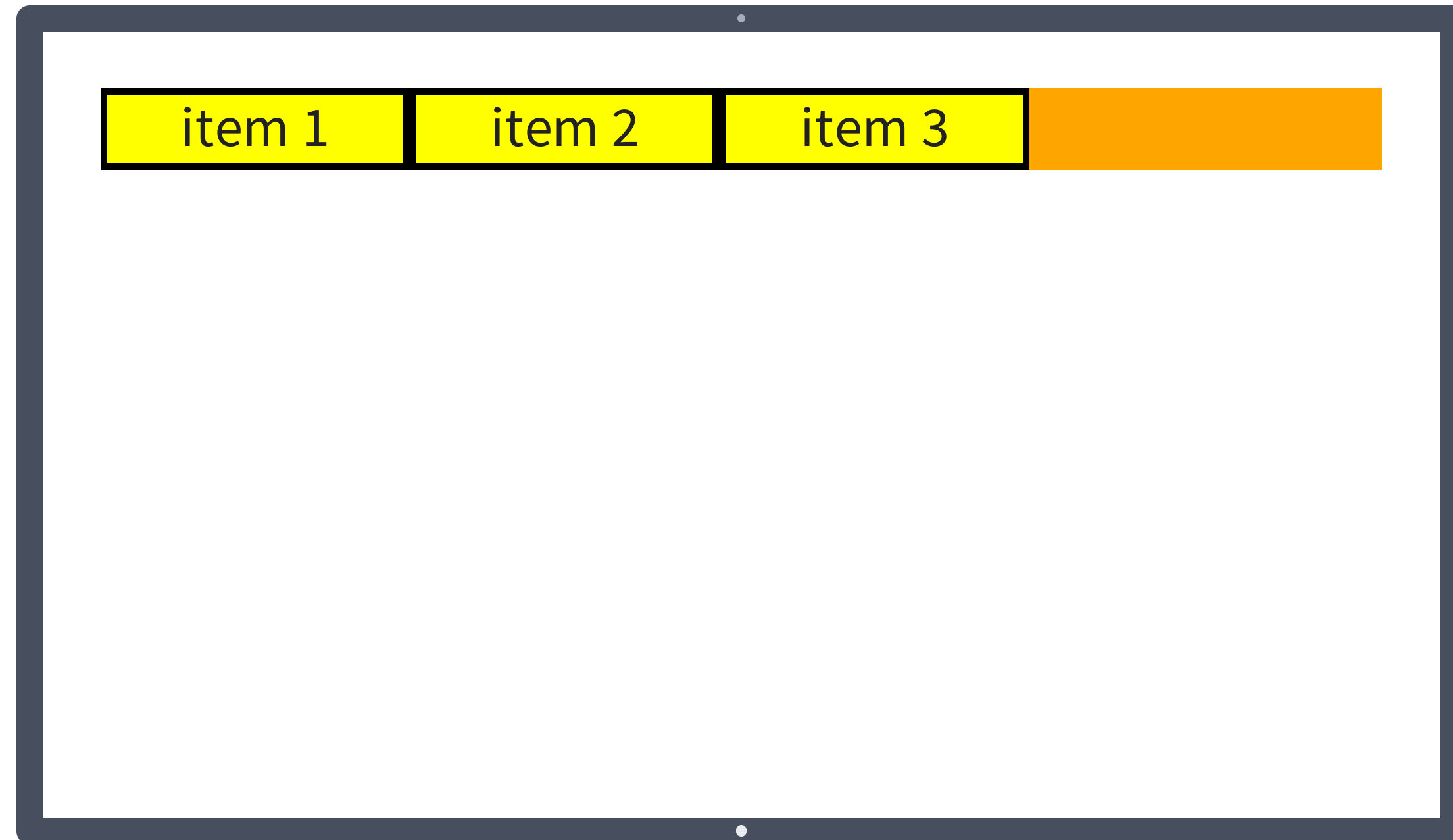


**Note :** si un seul item s'élargit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur d'élargissement.



# EXEMPLE (ÉLARGISSEMENT)

```
1 .container div {  
2   flex-basis: 220px;  
3 }  
4  
5 .container div:first-child {  
6   flex-grow: 1;  
7 }  
8  
9 .container div:last-child {  
10  flex-grow: 3;  
11 }
```

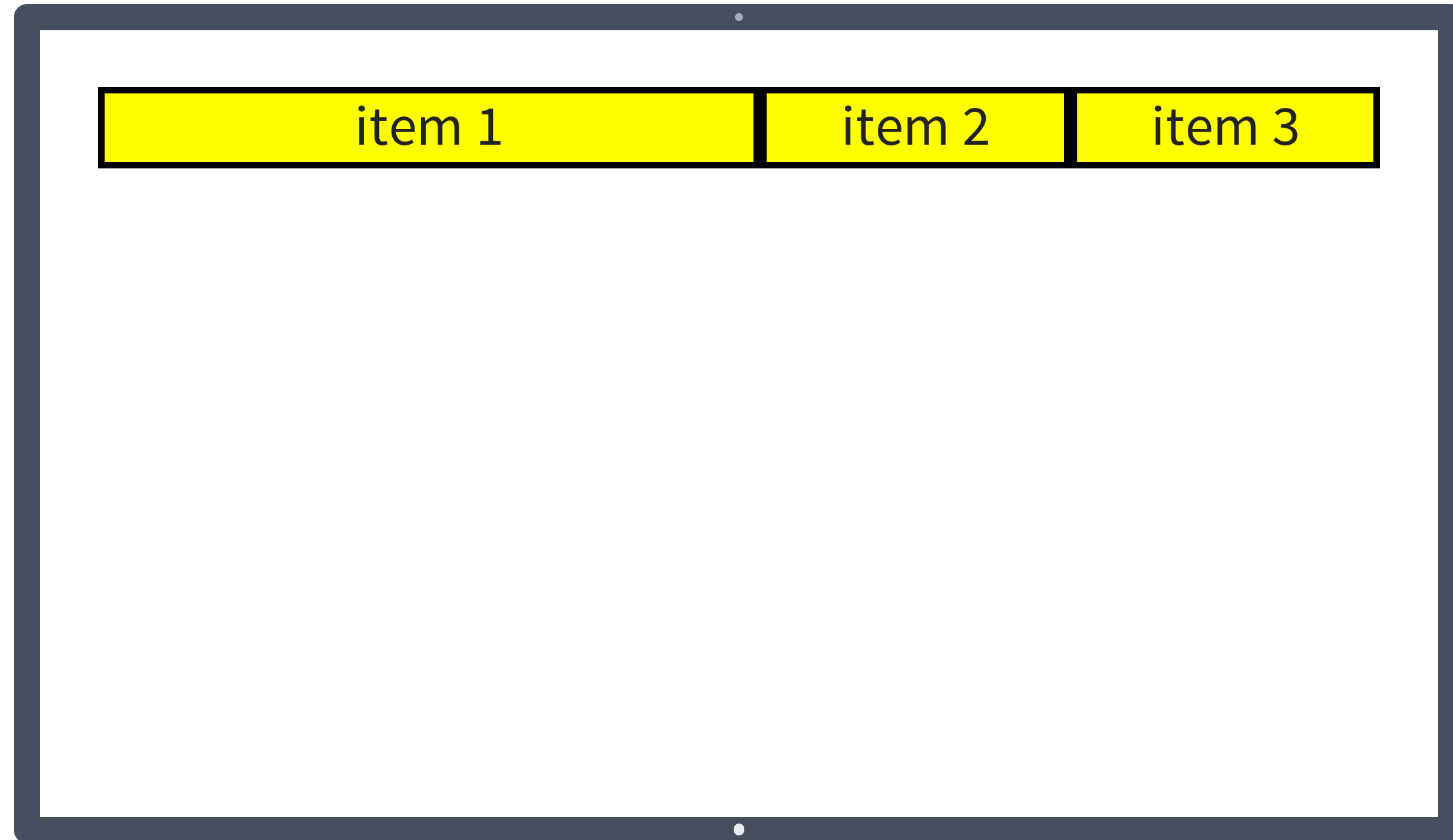


**Note :** si un seul item s'élargit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur d'élargissement.



# EXEMPLE (ÉLARGISSEMENT)

```
1 .container div {  
2   flex-basis: 220px;  
3 }  
4  
5 .container div:first-child {  
6   flex-grow: 1;  
7 }  
8  
9 .container div:last-child {  
10  flex-grow: 3;  
11 }
```

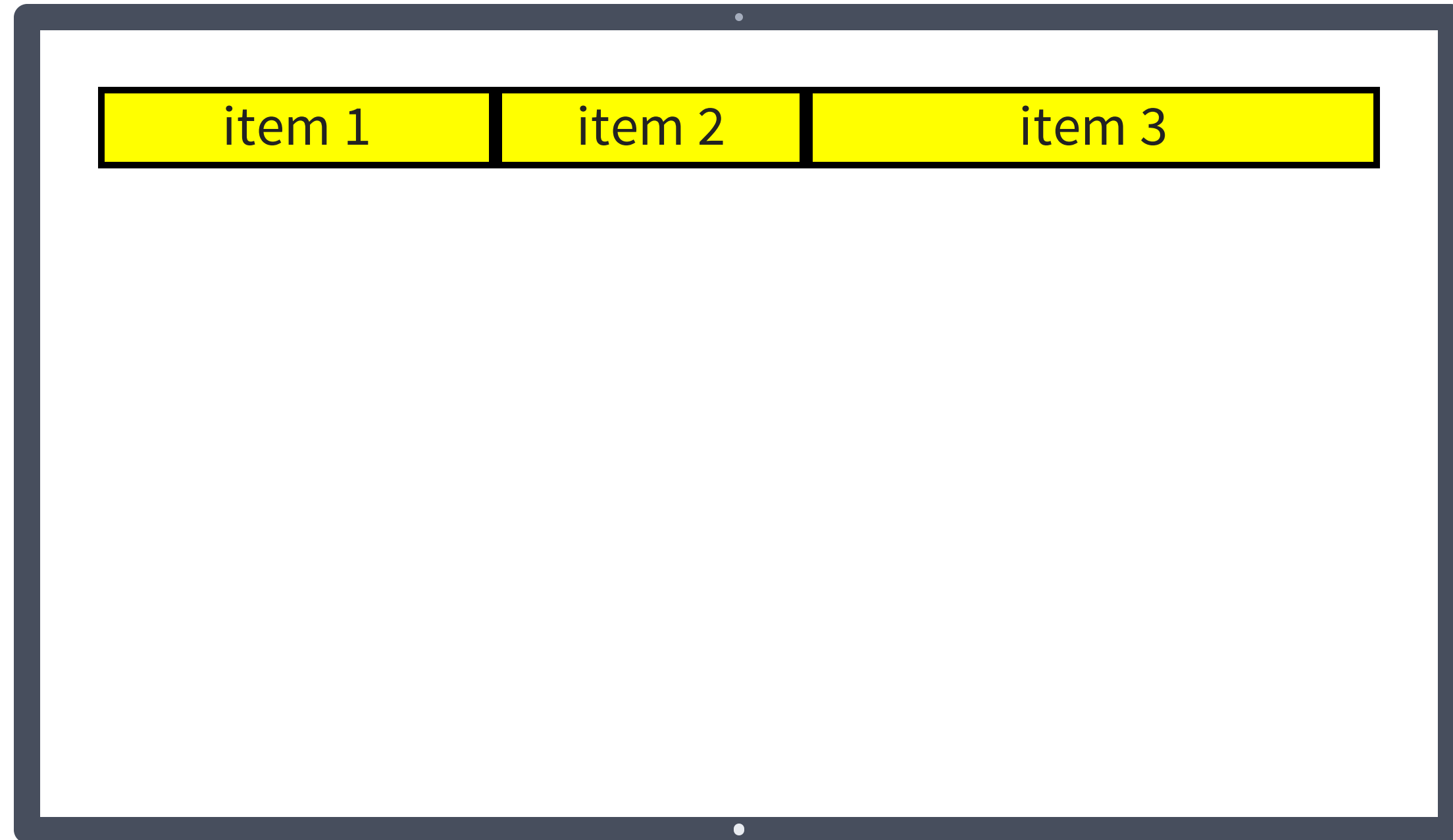


**Note :** si un seul item s'élargit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur d'élargissement.

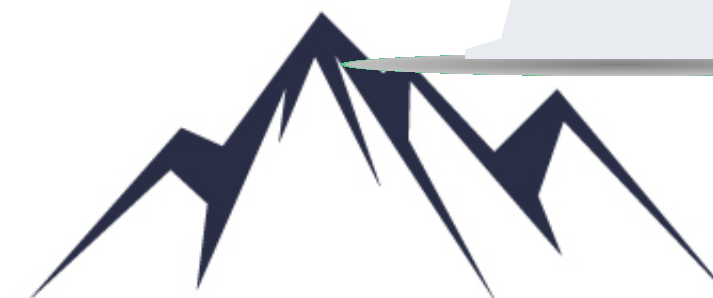


# EXEMPLE (ÉLARGISSEMENT)

```
1 .container div {  
2   flex-basis: 220px;  
3 }  
4  
5 .container div:first-child {  
6   flex-grow: 1;  
7 }  
8  
9 .container div:last-child {  
10  flex-grow: 3;  
11 }
```

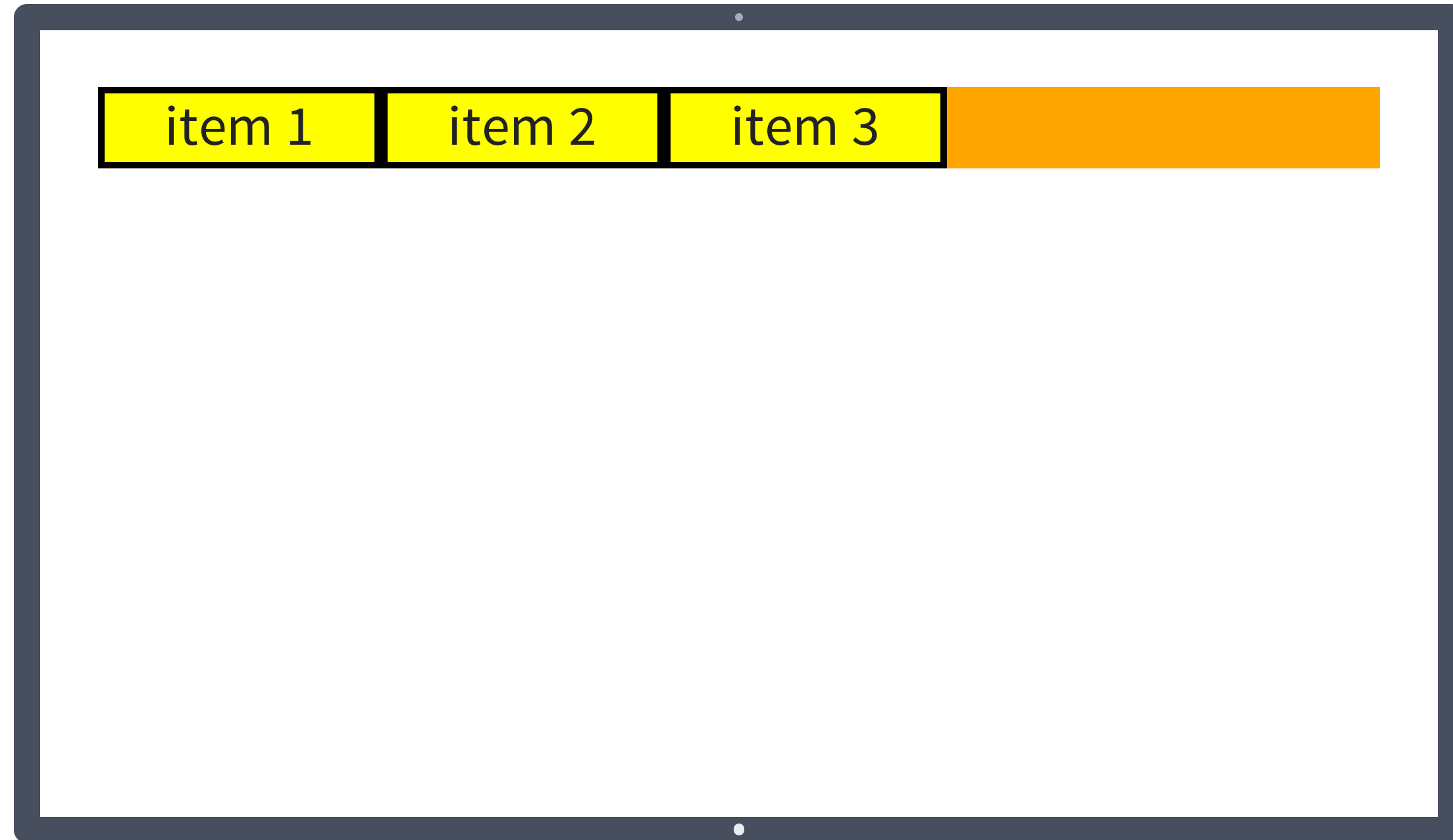


**Note :** si un seul item s'élargit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur d'élargissement.

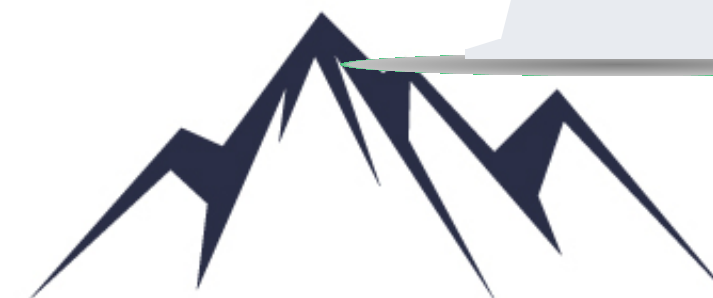


# EXEMPLE (RÉTRÉCISSEMENT)

```
1 .container div {  
2   flex-basis: 450px;  
3 }  
4  
5 .container div:first-child {  
6   flex-shrink: 2;  
7 }  
8  
9 .container div:last-child {  
10  flex-shrink: 0;  
11 }
```

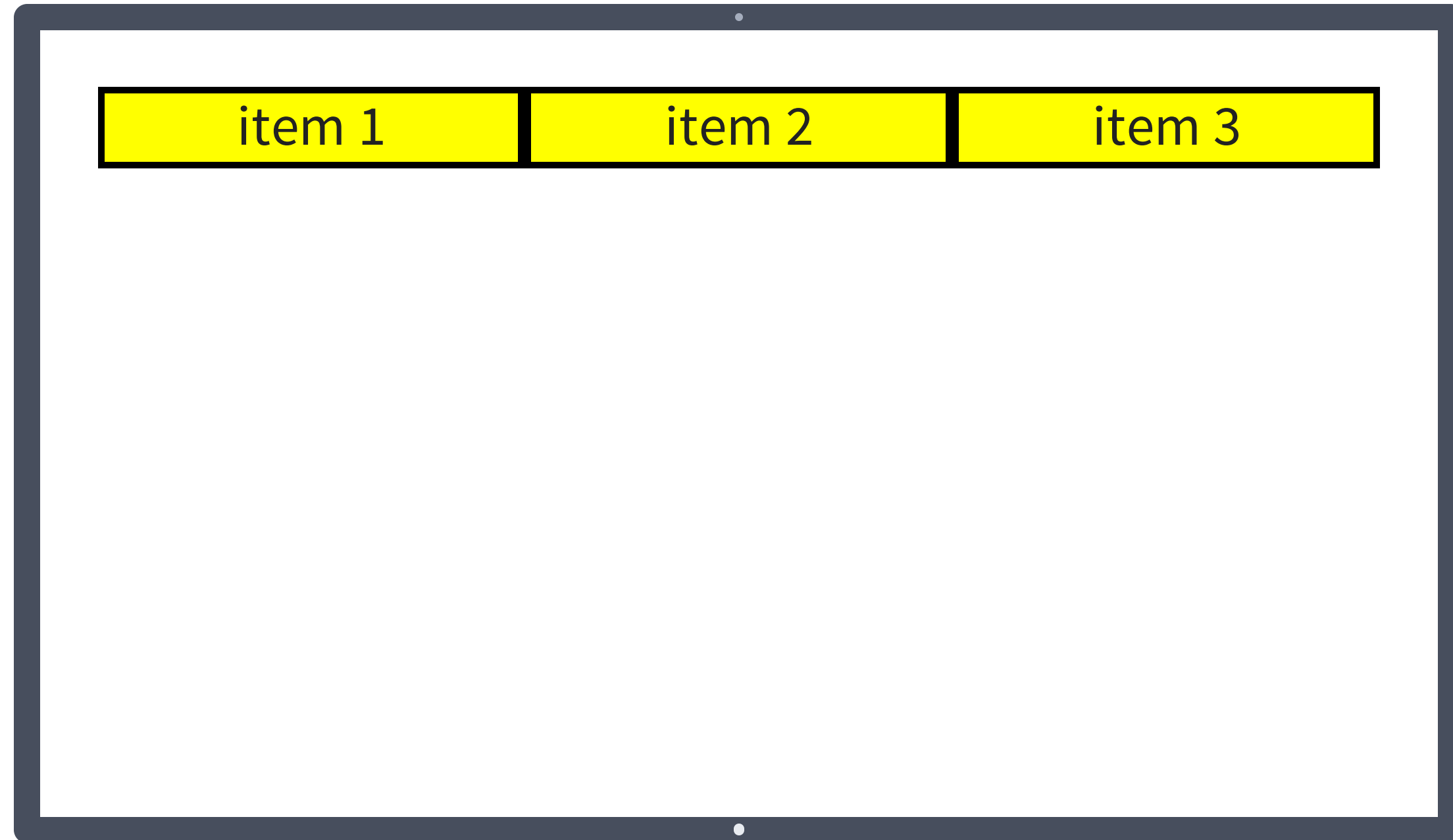


**Note :** si un seul item se rétrécit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur de rétrécissement.

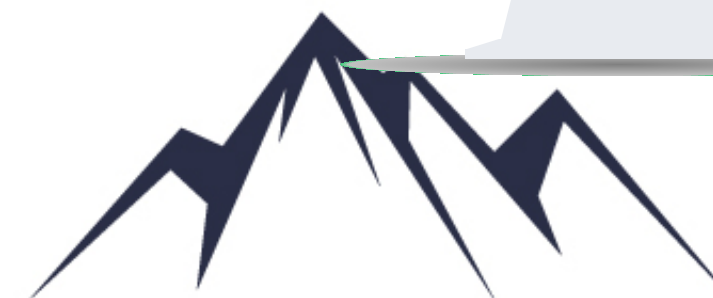


# EXEMPLE (RÉTRÉCISSEMENT)

```
1 .container div {  
2   flex-basis: 450px;  
3 }  
4  
5 .container div:first-child {  
6   flex-shrink: 2;  
7 }  
8  
9 .container div:last-child {  
10  flex-shrink: 0;  
11 }
```

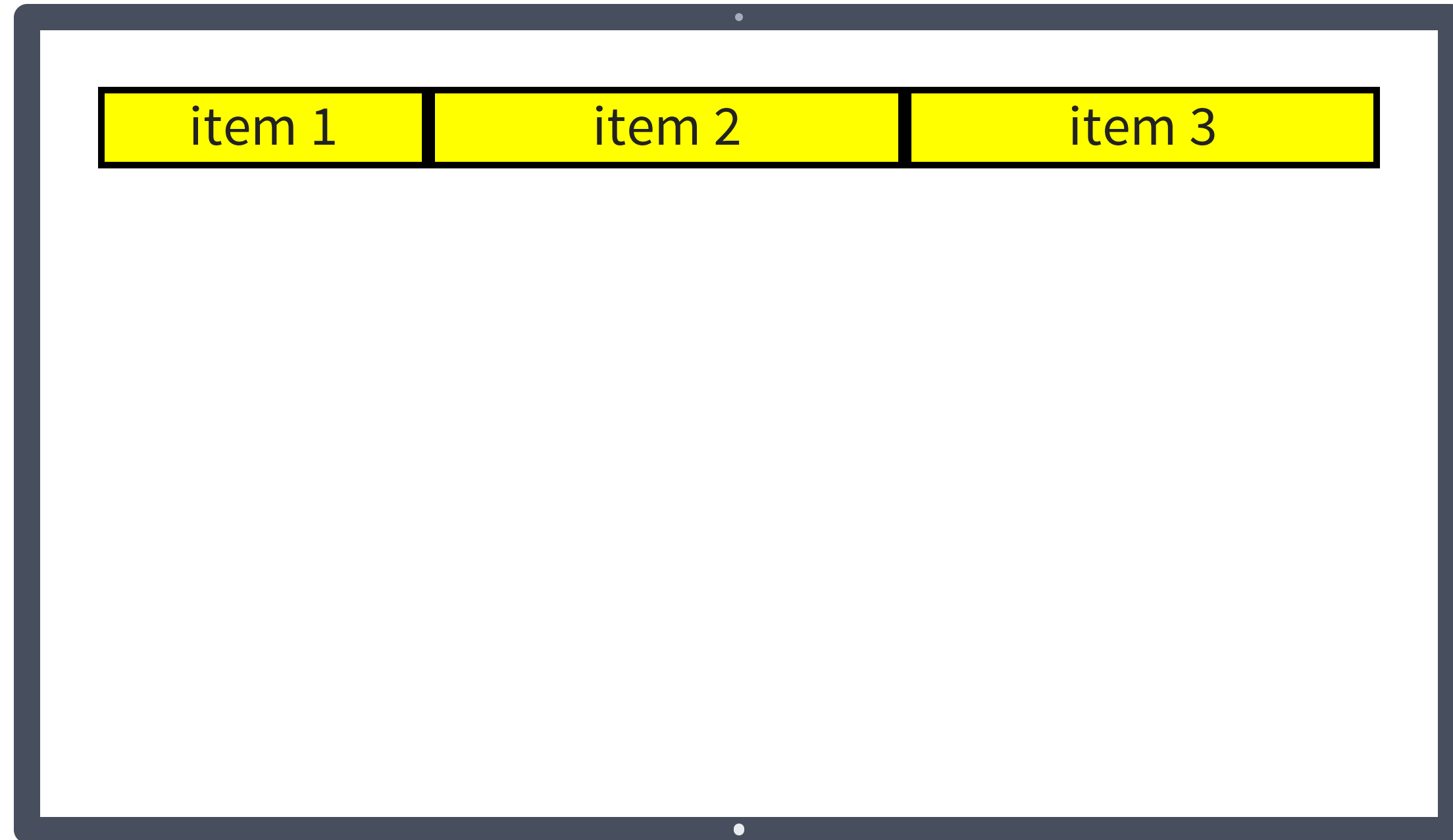


**Note :** si un seul item se rétrécit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur de rétrécissement.



# EXEMPLE (RÉTRÉCISSEMENT)

```
1 .container div {  
2   flex-basis: 450px;  
3 }  
4  
5 .container div:first-child {  
6   flex-shrink: 2;  
7 }  
8  
9 .container div:last-child {  
10  flex-shrink: 0;  
11 }
```

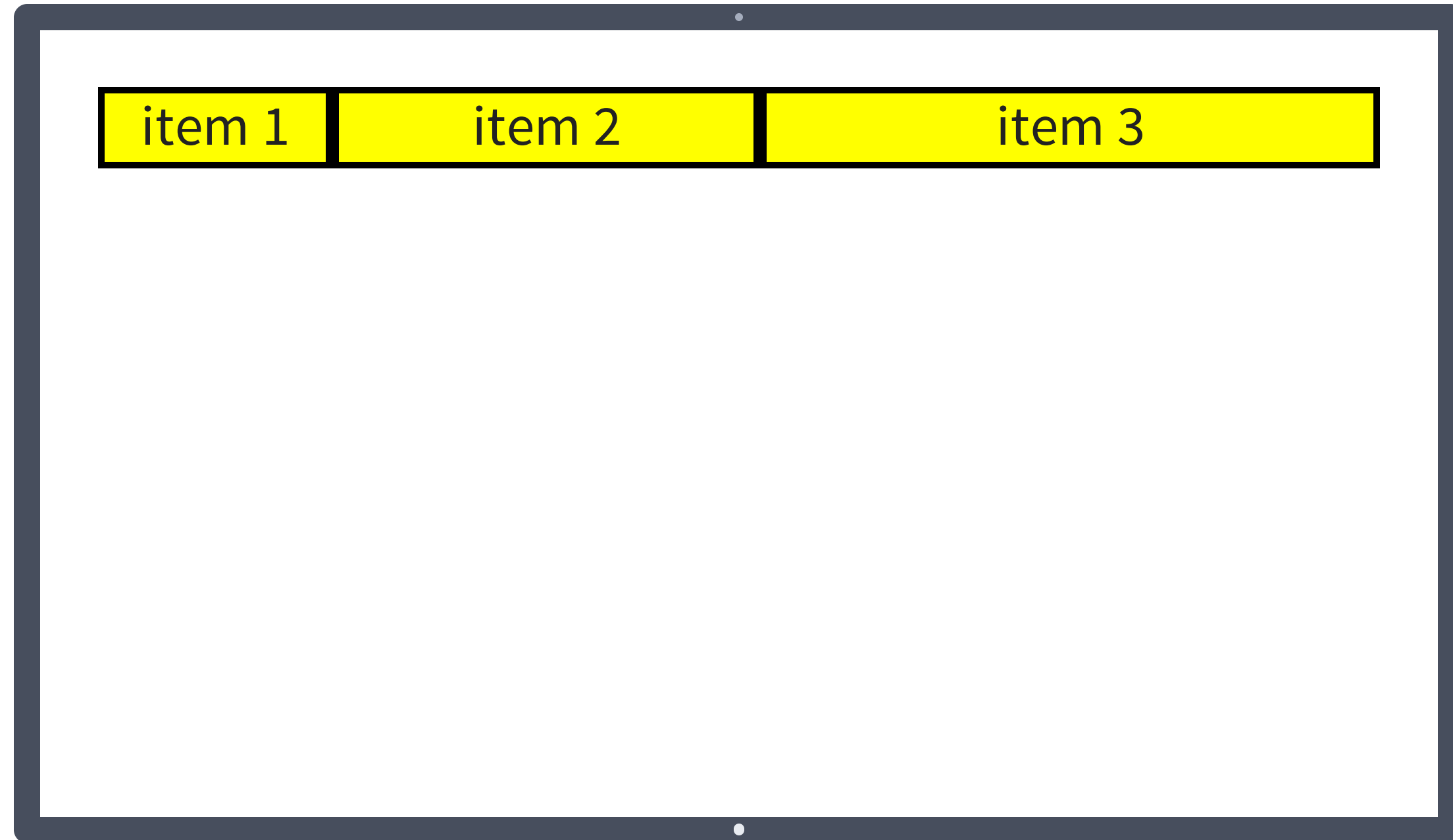


**Note :** si un seul item se rétrécit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur de rétrécissement.



# EXEMPLE (RÉTRÉCISSEMENT)

```
1 .container div {  
2   flex-basis: 450px;  
3 }  
4  
5 .container div:first-child {  
6   flex-shrink: 2;  
7 }  
8  
9 .container div:last-child {  
10  flex-shrink: 0;  
11 }
```

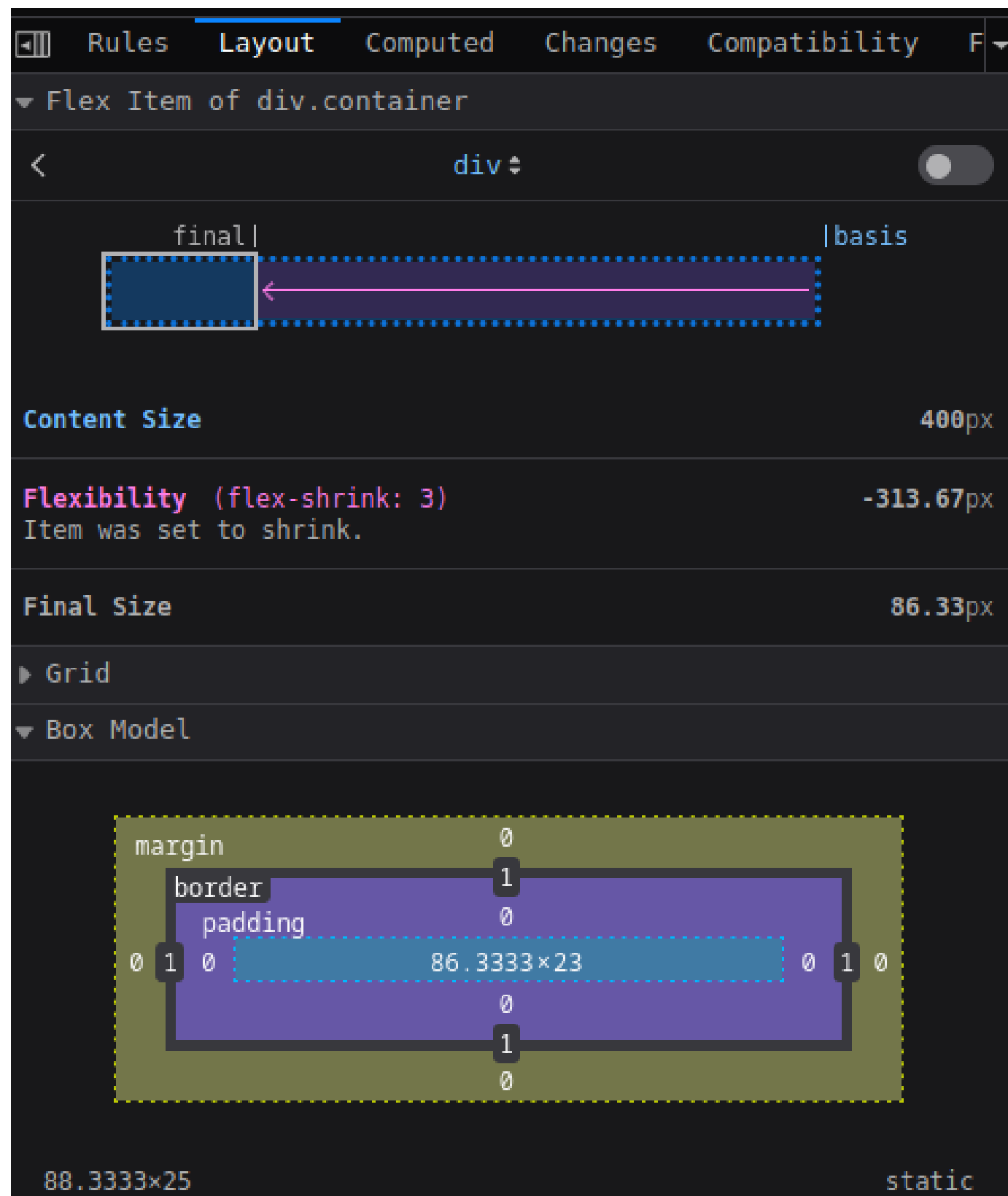


**Note :** si un seul item se rétrécit, n'importe quelle valeur ( $\geq 1$ ) peut être utilisée pour le facteur de rétrécissement.





# SAVOIR EXPLOITER L'INSPECTEUR



L'inspecteur vous fournit un tas d'information :

- La taille initiale
- Si l'élément est destiné à être élargi ou rétréci
- En fonction, la quantité de pixels tronqués / ajoutés
- La taille finale :

$$taille_{finale} = taille_{initiale} + ajout$$

ou

$$taille_{finale} = taille_{initiale} - troncation$$



index.html

```
1 <div class="container">
2   <div class="item">
3     <div class="subitem">1</div>
4     <div class="subitem">2</div>
5   </div>
6   <div class="item">
7     <div class="subitem">3</div>
8     <div class="subitem">4</div>
9   </div>
10 </div>
```

## ÉLÉMENT CONTENEUR/ITEM

- Un élément HTML peut à la fois être item et conteneur ;
- Dans ce cas-là, il est à la fois item d'un conteneur parent, et conteneur d'items enfants ;
- Toutes les propriétés CSS relatives au Flex s'appliquent.



```
1 .container { /* conteneur flex */
2   display: flex;
3   flex-direction: column;
4 }
5
6 .item { /* item/conteneur flex */
7   display: flex;
8   flex-grow: 1; /* car item */
9   gap: 10px; /* car conteneur */
10 }
11
12 .subitem { /* item */
13   flex-grow: 1;
14 }
```

## ÉLÉMENT CONTENEUR/ITEM

- Un élément HTML peut à la fois être item et conteneur ;
- Dans ce cas-là, il est à la fois item d'un conteneur parent, et conteneur d'items enfants ;
- Toutes les propriétés CSS relatives au Flex s'appliquent.



# APPLICATION

index.html

```
1 <body>
2   <header>header</header>
3   <main>
4     <section>section</section>
5     <aside>aside</aside>
6   </main>
7   <footer>footer</footer>
8 </body>
```

header

section

aside

footer



# APPLICATION

index.html

style.css

```
1  body {
2      display: flex;
3      flex-direction: column;
4      height: 100%;
5  }
6
7  header, footer {
8      flex-basis: 200px;
9  }
10
11 main {
12     display: flex;
13     flex-direction: row;
14     flex-grow: 1;
15 }
```

header

section

aside

footer



# APPLICATION

index.html

style.css

```
1  body {
2      display: flex;
3      flex-direction: column;
4      height: 100%;
5  }
6
7  header, footer {
8      flex-basis: 200px;
9  }
10
11 main {
12     display: flex;
13     flex-direction: row;
14     flex-grow: 1;
15 }
```

header

section

aside

footer

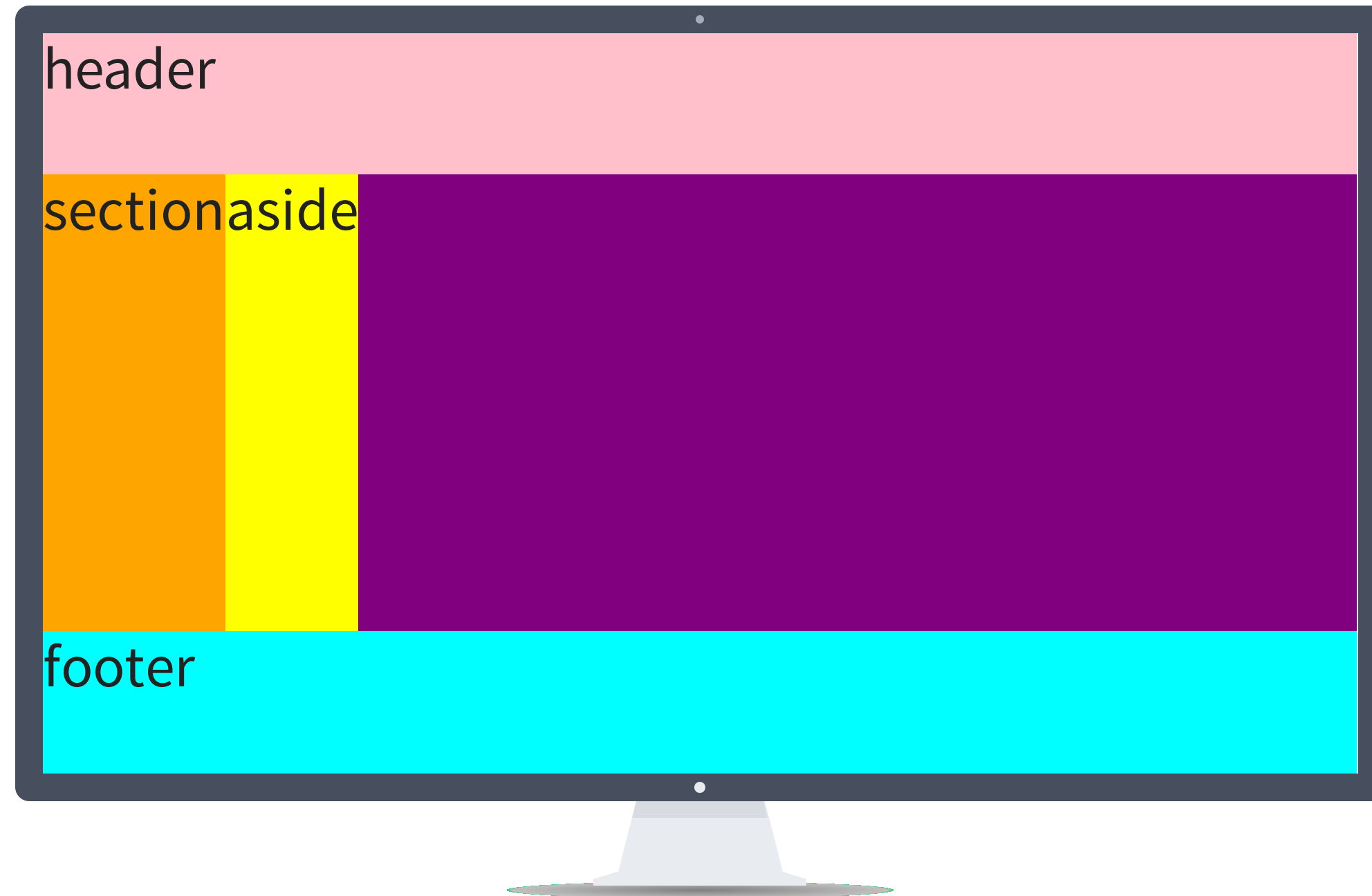


# APPLICATION

index.html

style.css

```
1  body {
2    display: flex;
3    flex-direction: column;
4    height: 100%;
5  }
6
7  header, footer {
8    flex-basis: 200px;
9  }
10
11 main {
12   display: flex;
13   flex-direction: row;
14   flex-grow: 1;
15 }
```



# APPLICATION

index.html

style.css

style.css (suite)

```
1 main {
2     display: flex;
3     flex-direction: row;
4     flex-grow: 1;
5 }
6
7 aside {
8     flex-basis: 200px;
9 }
10
11 section {
12     flex-grow: 1;
13 }
```

