Flexbox

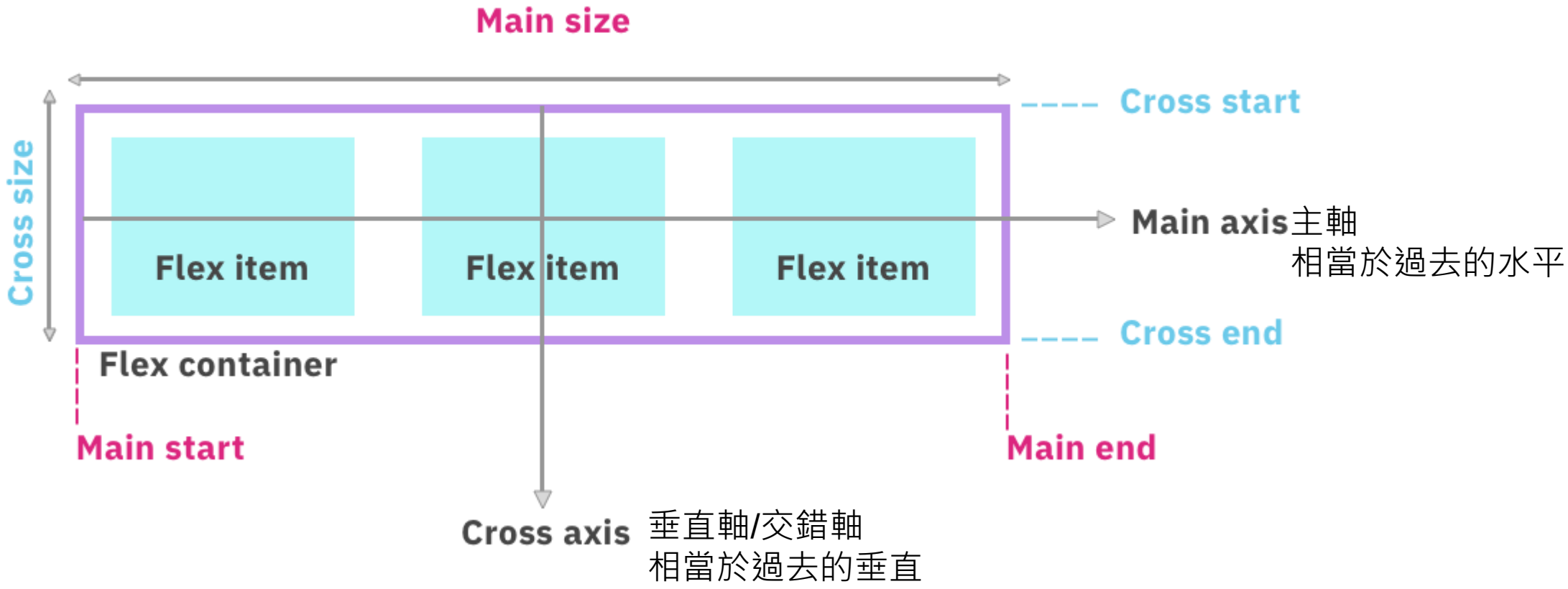
基本概念

- 過去
  - Block layout塊狀佈局模式主要是以垂直方向為主
  - Inline layout行內佈局模式主要是水平方向為主
- Flex
  - 屬於1維佈局模式(layout model)
  - Flex排版主要是調整對象的寬或是高，以更加滿足任何顯示的使用設備。

主要特色

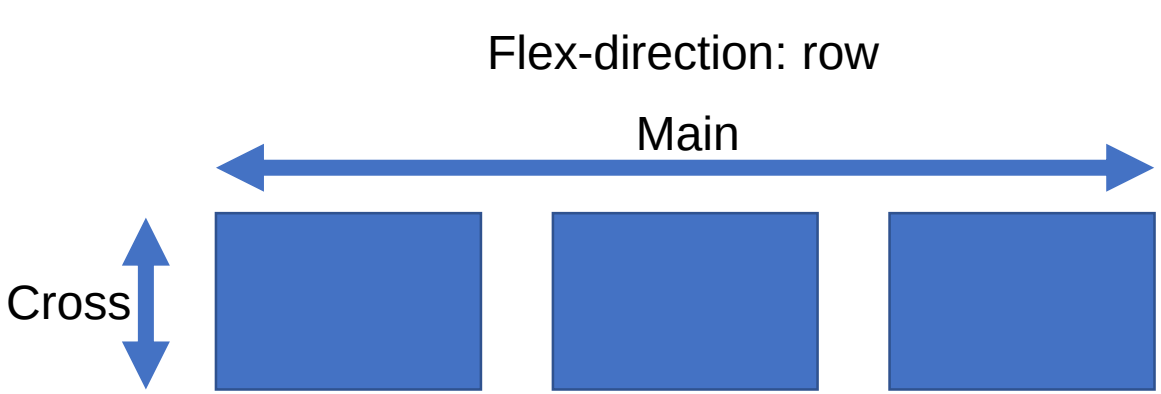
Flex container **Flex item** **Axes**

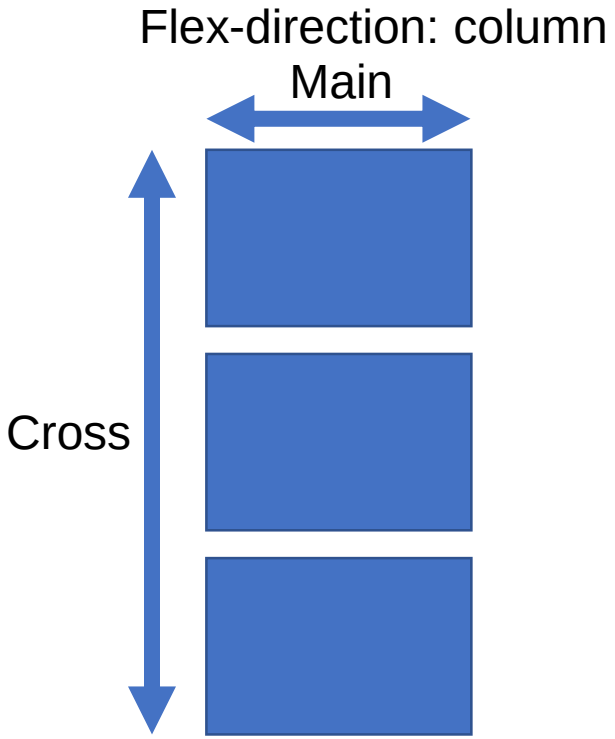
flex-direction



Flex attributes

- Main axis & Cross axis
  - Flex-direction: row、row-reverse、column、column-reverse





資料來源 https://www.w3.org/TR/css-flexbox-1/#property-index

Flex container

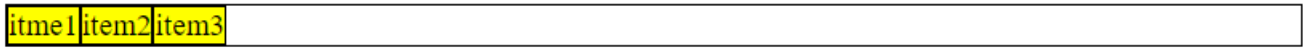
- 網頁排版使用flex需要一個容器，此時就需要使用到display屬性，將其設定為flex或是inline-flex。
- 在此容器中的元素item，就會繼承並具有flex元素屬性，預設
  - flex-direction: row
  - Item : Start line in main axis
  - Item : Shrink in main axis
  - Item : stretch in cross axis
  - Flex-basis: auto
  - Flex-wrap: nowrap

試試看增加更多的item會發生什麼事

```html
<div class="flexContainer">
    <div class="item item1">itme1</div>
    <div class="item item2">item2</div>
    <div class="item item3">item3</div>
</div>
```

```css
.flexContainer {
    display: flex;
    border: 1px solid black;
}
.item {
    border: 1px solid black;
    background: yellow;
    text-align: center;
    font-size: larger;
}
```

flex-direction: row
Item : Start line in main axis

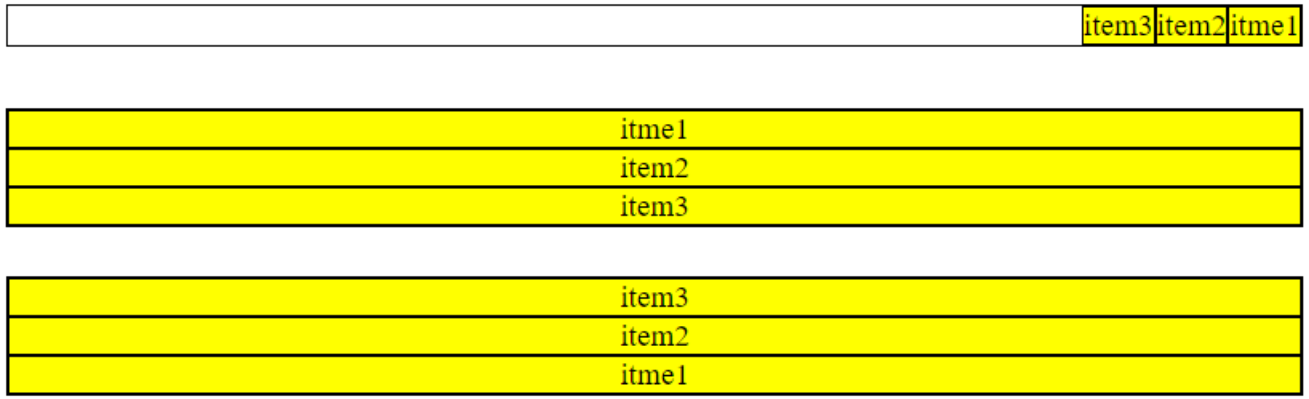


Item : stretch in cross axis
Item的高與container的高一樣

Item : Shrink in main axis

Flex屬性

- Flex-direction: row、row-reverse、column、column-reverse



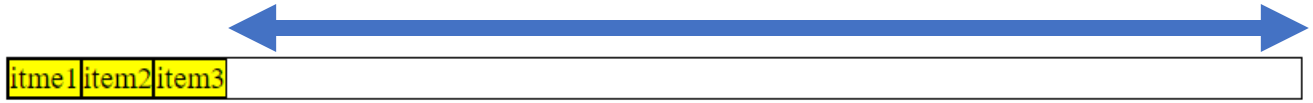
- flex-wrap: wrap;

Flex屬性

- flex-flow為flow-direction和flow-wrap的縮寫
- 例如

```
flex-direction: row-reverse;
flex-wrap: wrap;
=
flex-flow: row-reverse wrap;
```

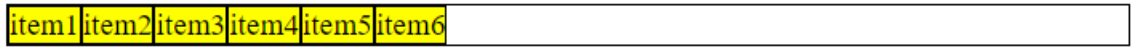
Flex元素item的屬性



  - 主要用來調整控制剩餘可使用的空間
- Flex-basis:預設為auto(其值為0)，或是設為content根據設定調整大小。
- Flex-grow:以flex-basis為延伸的基礎，在主軸方向上增加元素寬度，通常設定值為正整數。
- Flex-shrink:與flex-grow相反，主要用來縮小元素寬度
- Flex: 縮寫，包含三個屬性，分別代表為(flex-grow, flex-shrink, flex-basis)

Flex-basis

- 假設設定.flexContainer寬度為700px，
  - .item設定flex-basis不作設定 (也就是預設auto為0，其根據文字調整)

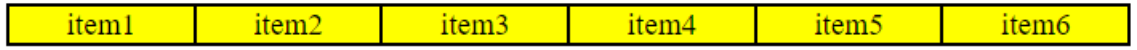


  - .item設定flex-basis為100px (每個元素占100px)

  - .item設定flex-basis為200px (自動平均分配整體寬度)

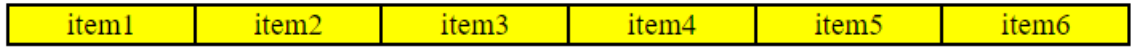
Flex-basis

- 假設設定.flexContainer寬度為700px，而.item設定width為200px
  - .item設定flex-basis不作設定



  - .item設定flex-basis為0

  - .item設定flex-basis為auto



  - .item設定flex-basis為content

  - .item設定flex-basis為100px

Flex-grow

- 以三個item為例作說明
- 分別設定如下

```
.item1 {
    flex-basis: 200px;
    background: yellow;
}
.item2 {
    flex-basis: 200px;
    background: lightgreen;
}
.item3 {
    flex-basis: 200px;
    background: tomato;
}
```

等同

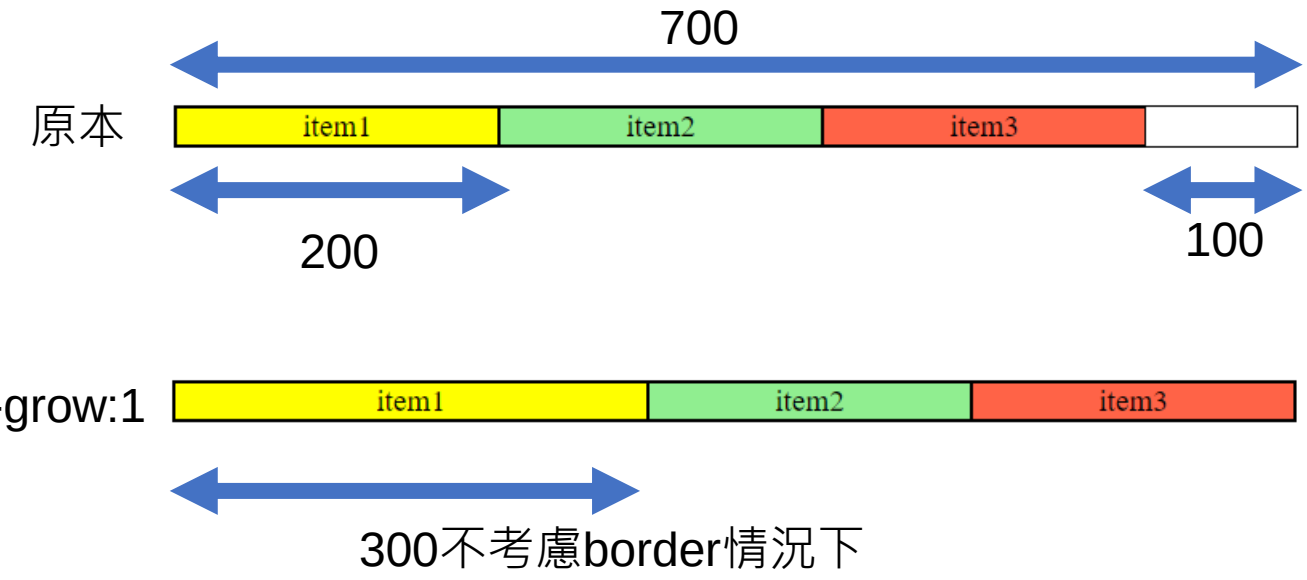

```
.item1 {
    flex-basis: 200px;
    flex-grow: 0;
    background: yellow;
}
.item2 {
    flex-basis: 200px;
    flex-grow: 0;
    background: lightgreen;
}
.item3 {
    flex-basis: 200px;
    flex-grow: 0;
    background: tomato;
}
```

Flex-grow:0代表不作延伸，以自己的flex-basic作為基礎

Flex-grow

- .item1設定flex-grow:1

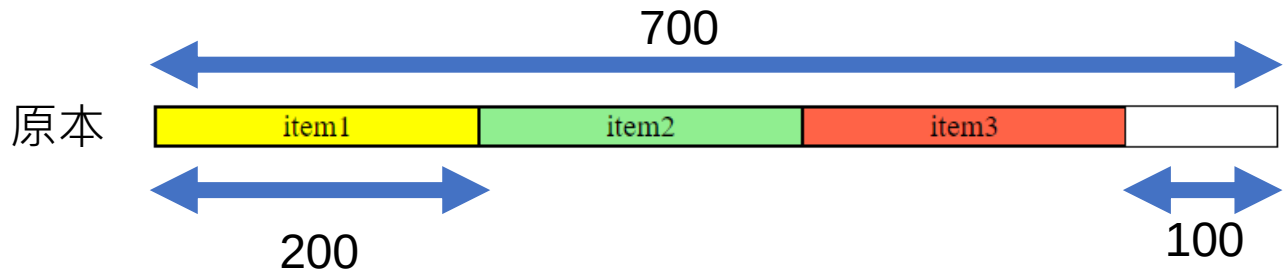
```
.item1 {
    flex-basis: 200px;
    flex-grow: 1;
    background: yellow;
}
```



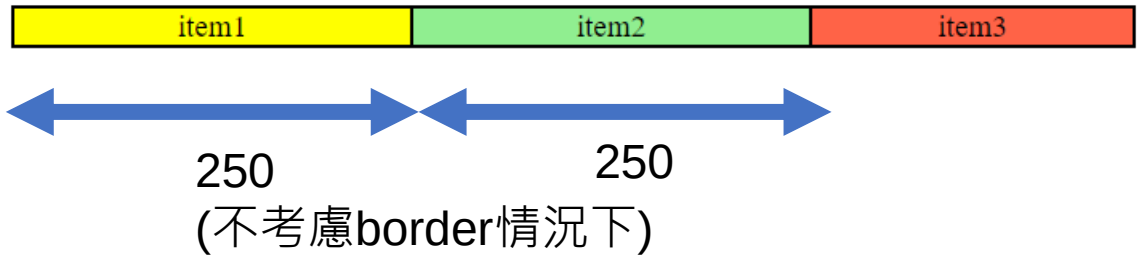
Flex-grow

- .item1和.item2設定flex-grow:1

```css
.item1 {
    flex-basis: 200px;
    flex-grow: 1;
    background: yellow;
}
.item2 {
    flex-basis: 200px;
    flex-grow: 1;
    background: lightgreen;
}
```



Item1: flex-grow:1
Item2: flex-grow:1

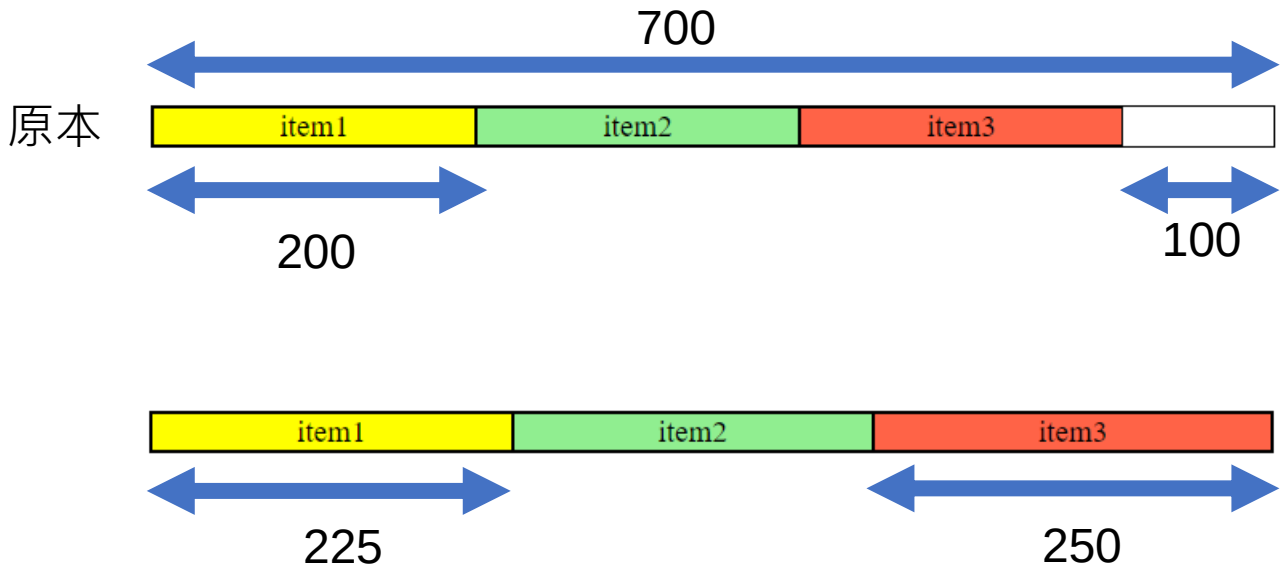


(不考慮border情況下)

Flex-grow

- .item1和.item2設定flex-grow:1，.item3設定flex-grow:2

```
.item1 {
    flex-basis: 200px;
    flex-grow: 1;
    background: yellow;
}
.item2 {
    flex-basis: 200px;
    flex-grow: 1;
    background: lightgreen;
}
.item3 {
    flex-basis: 200px;
    flex-grow: 2;
    background: tomato;
}
```



(不考慮border情況下)

練習

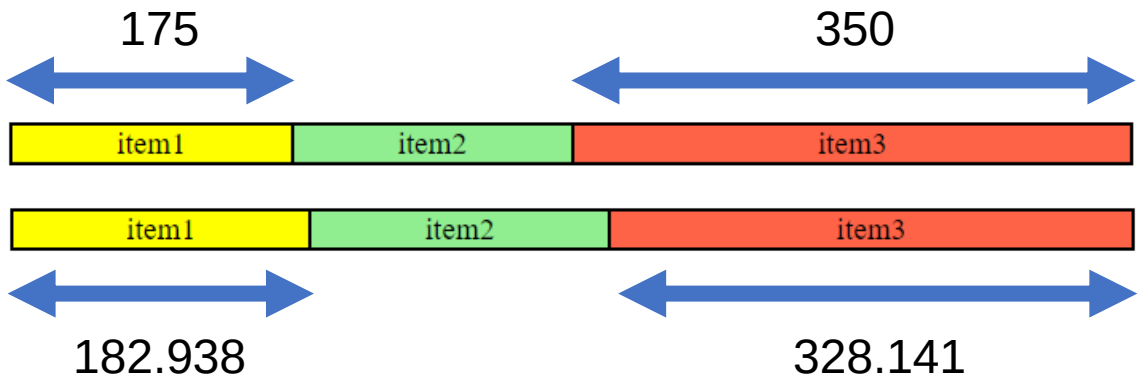
- 想想看如何使用flex-grow設定三個item3的寬度為item1和item2的兩倍大小

Flex-basis: auto vs. 0

-

```
flex-basis: 0;
```

```
flex-basis: auto;
```

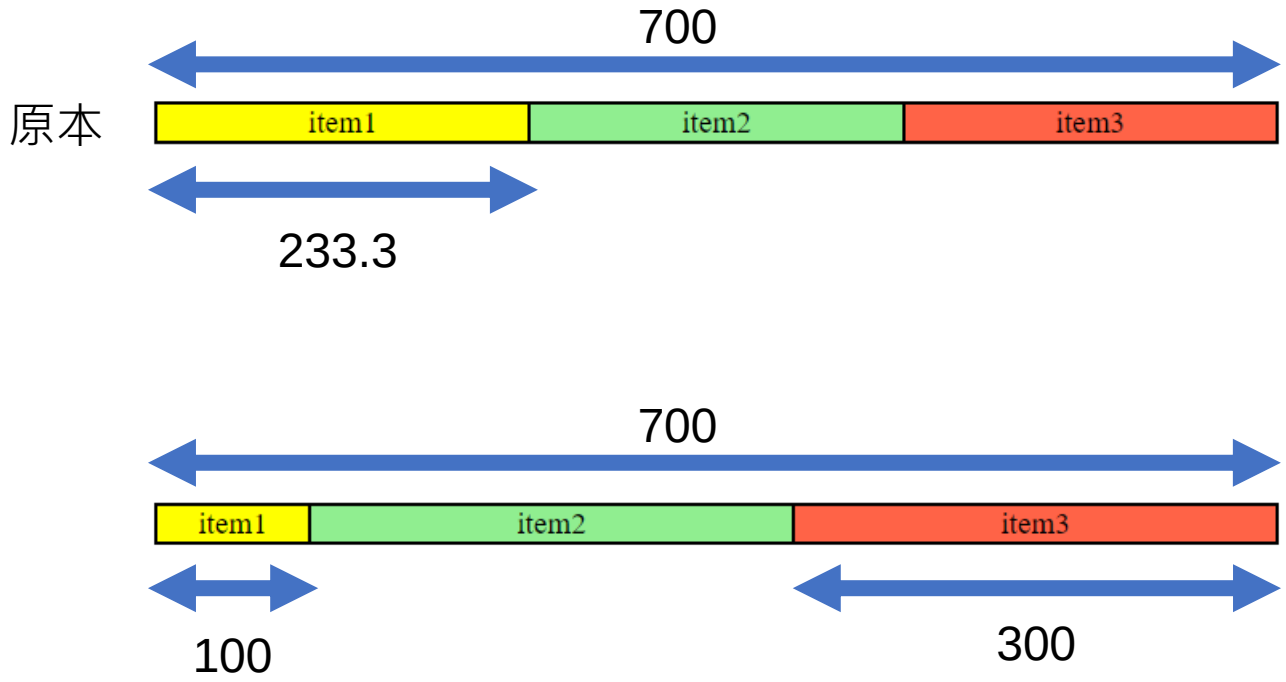


- Auto情況：根據父元素等分3份後，剩餘空間在切割成四份
- 0情況：依據本身自我的寬度為0，700px等分四份

Flex-shrink

```css
.item1 {
    flex-basis: 300px;
    background: yellow;
}
.item2 {
    flex-basis: 300px;
    background: lightgreen;
}
.item3 {
    flex-basis: 300px;
    background: tomato;
}
```

```css
.item1 {
    flex-basis: 300px;
    flex-shrink: 1;
    background: yellow;
}
.item2 {
    flex-basis: 300px;
    flex-shrink: 0;
    background: lightgreen;
}
.item3 {
    flex-basis: 300px;
    flex-shrink: 0;
    background: tomato;
}
```



多出來的空間300*3-700=200

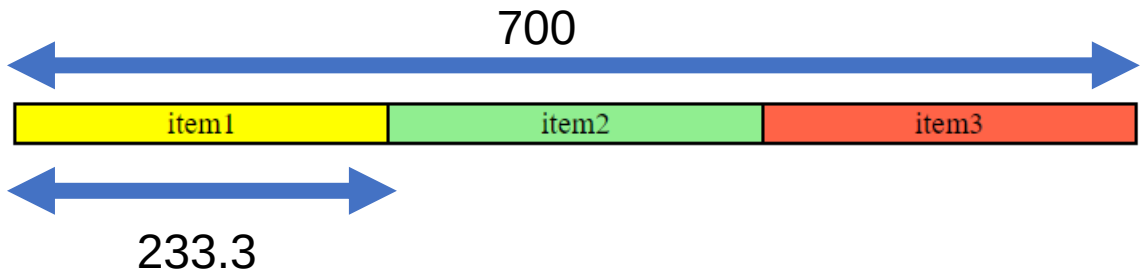
itme1: 300px-(300px*1 /300px*1)*200px=100px

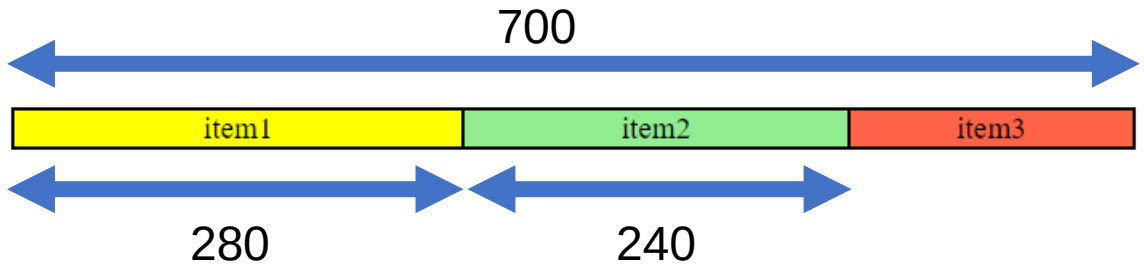
Flex-shrink

```
.item1 {
    flex-basis: 300px;
    background: yellow;
}
.item2 {
    flex-basis: 300px;
    background: lightgreen;
}
.item3 {
    flex-basis: 300px;
    background: tomato;
}
```

```
.item1 {
    flex-basis: 300px;
    flex-shrink: 1;
    background: yellow;
}
.item2 {
    flex-basis: 300px;
    flex-shrink: 3;
    background: lightgreen;
}
.item3 {
    flex-basis: 300px;
    flex-shrink: 6;
    background: tomato;
}
```

原本





多出來的空間300*3-700=200

Itme1: 300px-(300px*1 /300px*1+300px*3+300px*6)*200px=280px

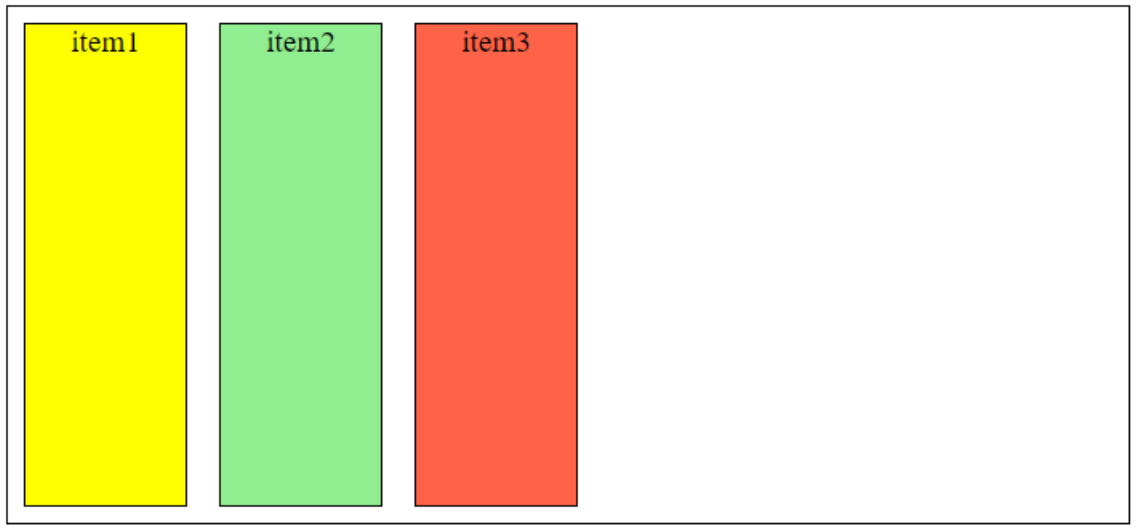
flex內的元素間佈局與排列

- justify-content
  - 目的讓元素根據主軸進行排列對齊(水平線)
  - flex-start (default)、flex-end、center、space-around、space-between、space-evenly

- align-items
  - 目的讓元素根據垂直軸進行排列對齊(垂直線)
  - stretch (default)、center、flex-start、flex-end、baseline

以三個不同高度的item舉例

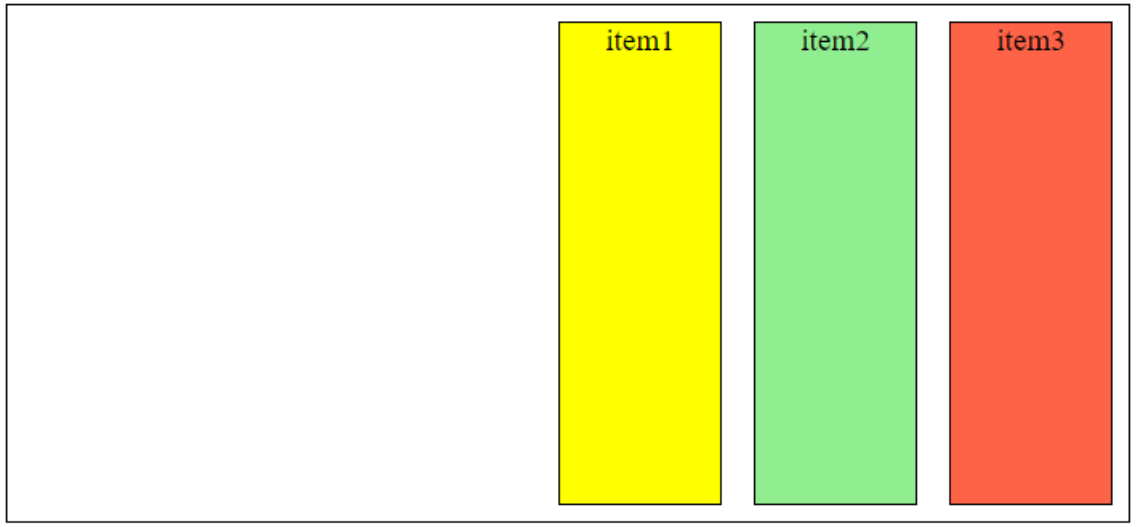
```css
.flexContainer {
    display: flex;
    border: 1px solid black;
    width: 700px;
}
.item {
    border: 1px solid black;
    background: yellow;
    text-align: center;
    font-size: larger;
    width: 100px;
    margin: 10px;
}
.item1 {
    min-height: 100px;
    background: yellow;
}
.item2 {
    min-height: 200px;
    background: lightgreen;
}
.item3 {
    min-height: 300px;
    background: tomato;
}
```



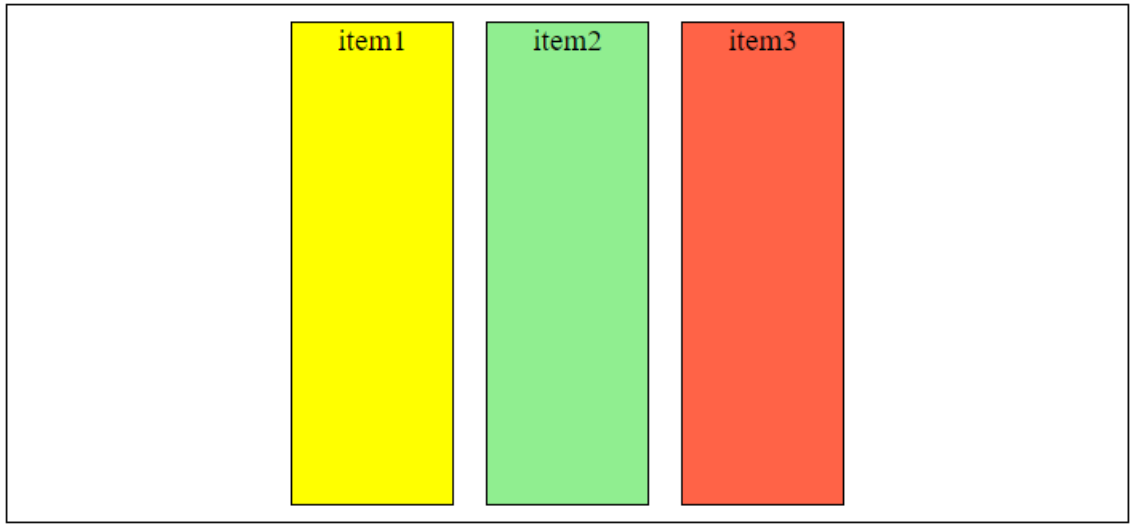
justify-content

- 同學可以試試看其他的

justify-content: flex-end



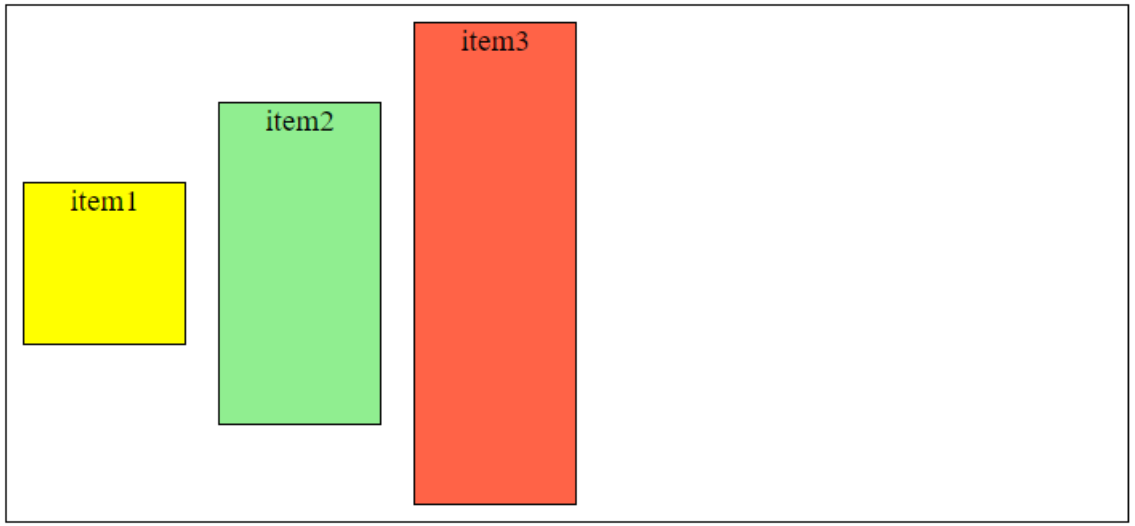
Justify-content: center



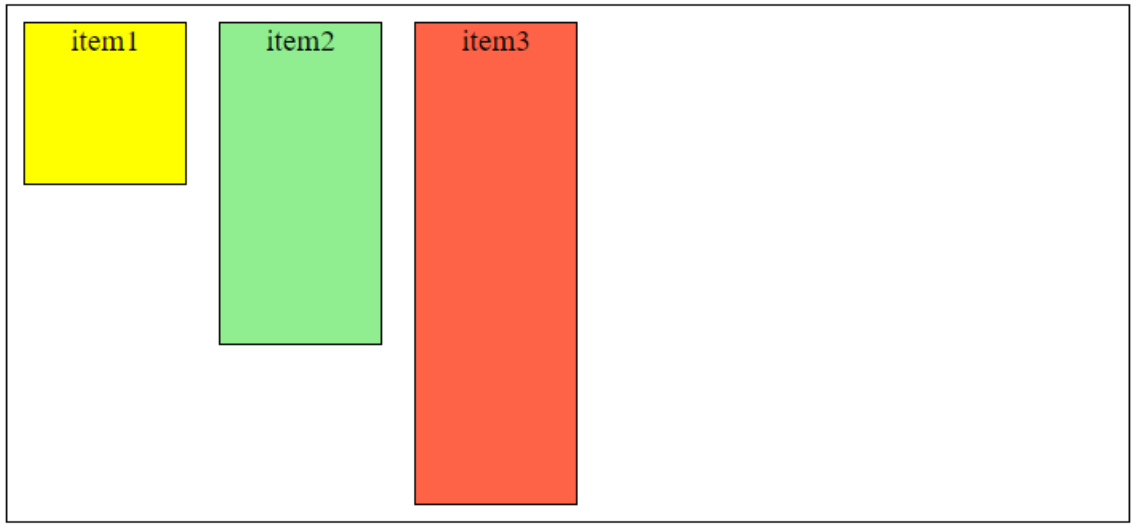
align-items

同學可以試試看其他的

center



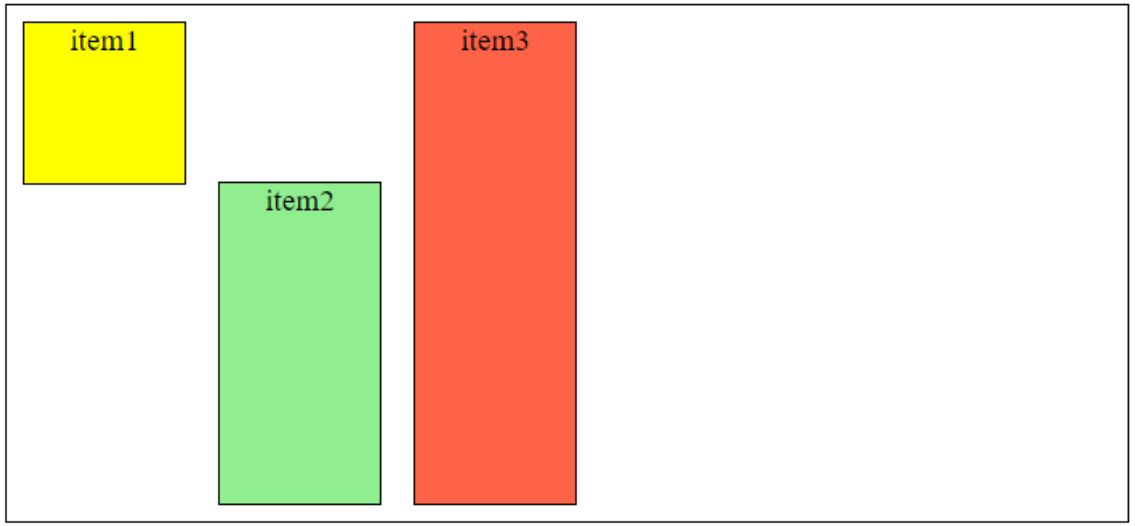
flex-start



align-self

- 使用時機：不理父元素的設定，以自己設定值為主
- 屬性值：auto(default)、stretch、center、flex-start、flex-end、baseline

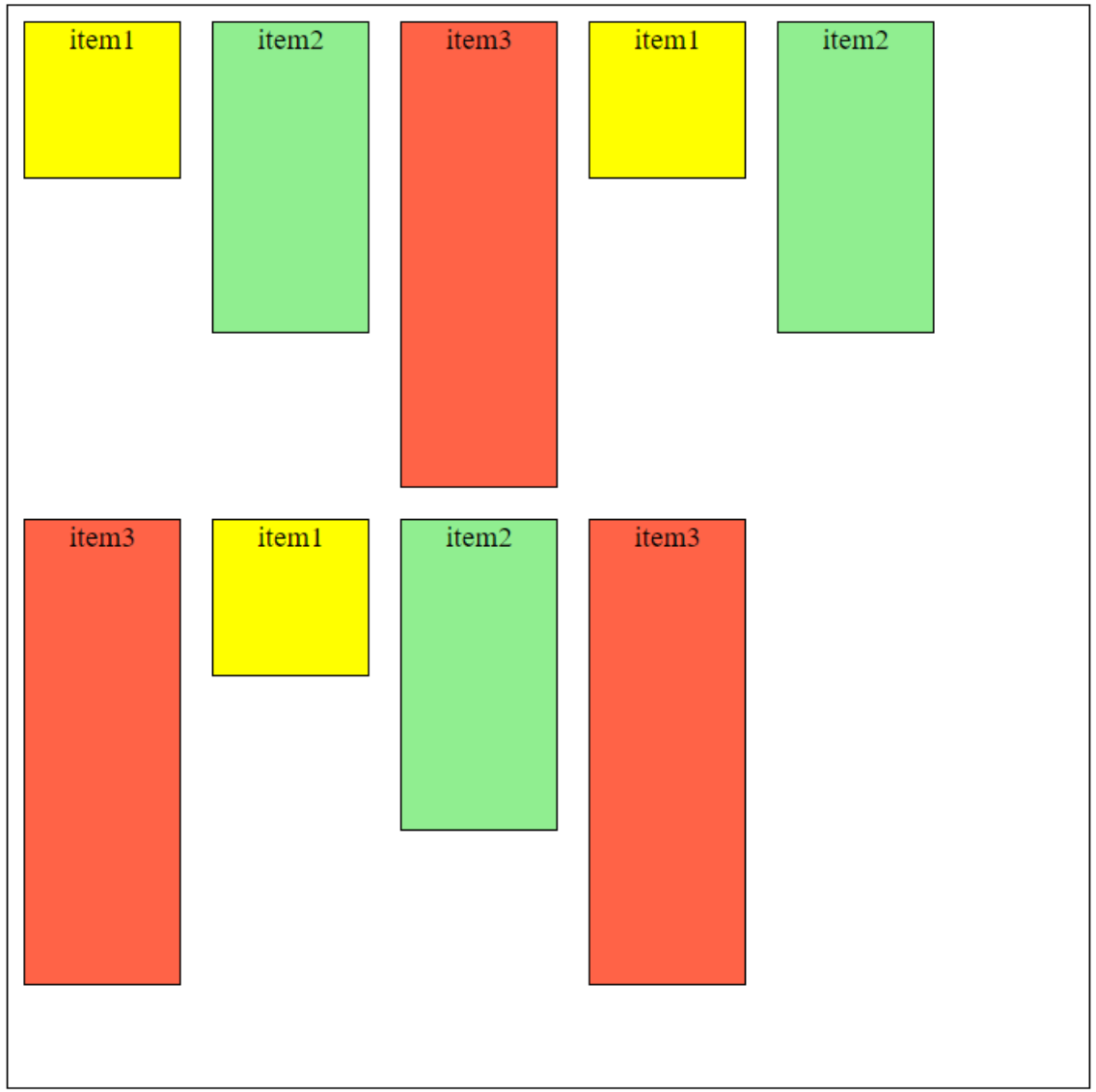
flex-end



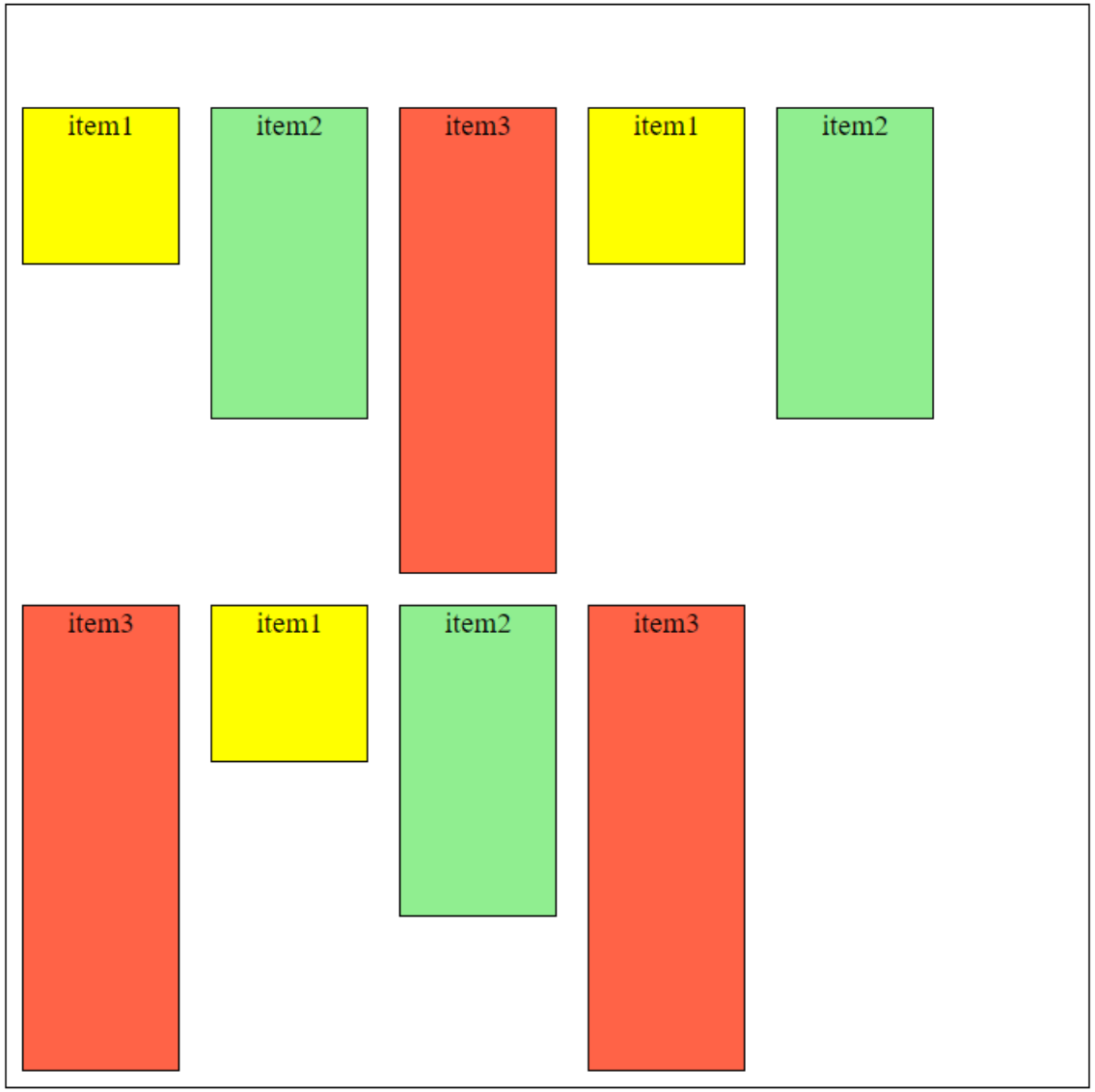
align-content

- 使用時機：有多個元素item超過一行時使用
- 要搭配flex-wrap: wrap一起使用
- 屬性值：stretch (default)、flex-start、flex-end、center、space-around、space-between、space-evenly
- 當有設定高度時，stretch不會有作用。

align-content: flex-start

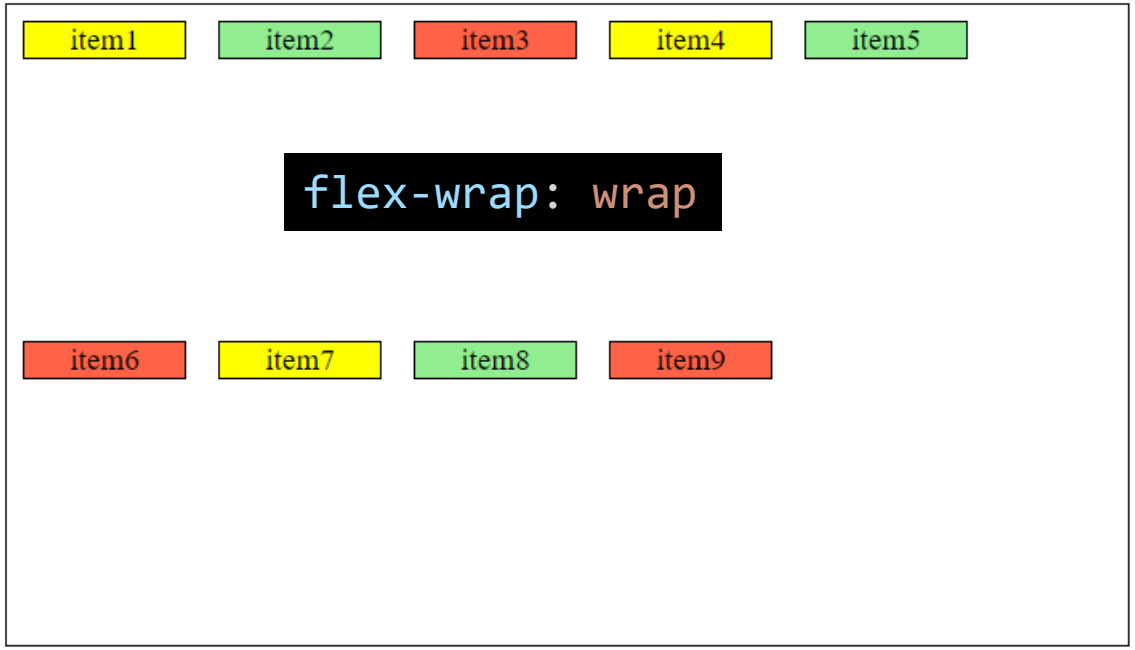


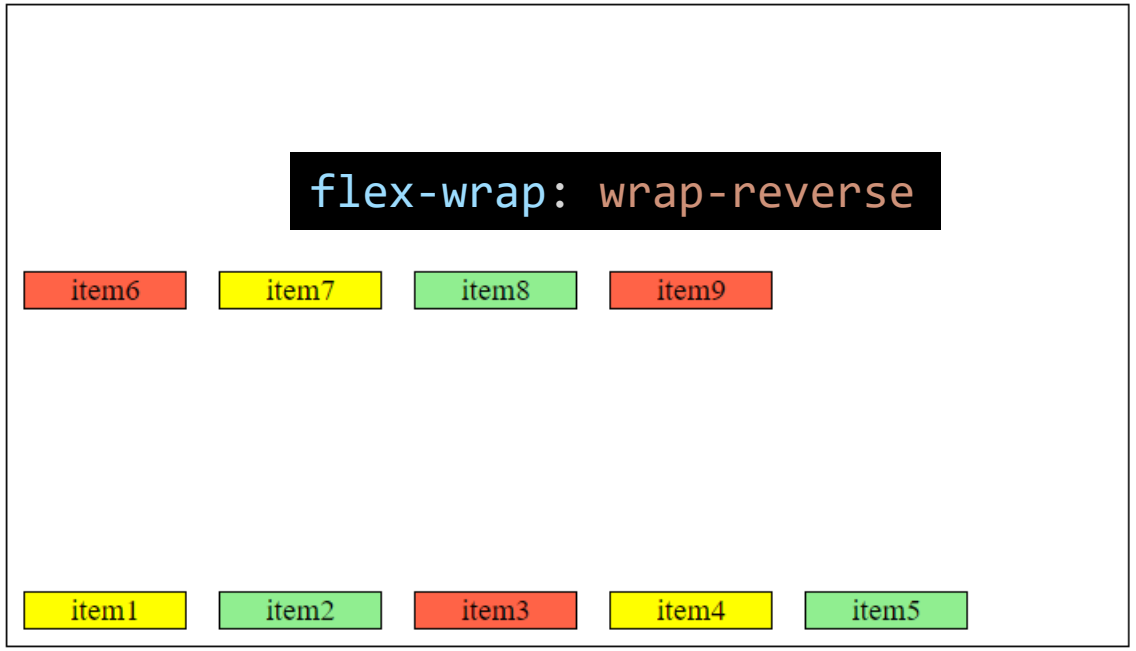
align-content: flex-end



Flex-wrap

- 使用時機：多個元素撐滿容器時是否要換行
- 屬性值：nowrap (default)、wrap、wrap-reverse
- 設定如下：取消所有item高度，將容器高度設為400px

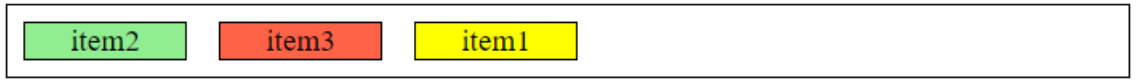




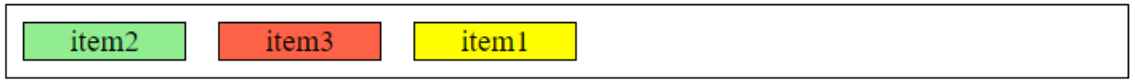
order

- 使用時機：用來指定元素的排列出現順序
- 屬性值：整數值

原本




```css
.item1 {
    order: 3;
    background: yellow;
}
.item2 {
    order: 1;
    background: lightgreen;
}
.item3 {
    order: 2;
    background: tomato;
}
```



學習任務

- 使用flex設計如右網頁排版
- main內放入6個圖片。
- 版面大小不拘，但需要排列整齊。