UI BASICS

LAYOUT

CONTENT

- CSS Flexbox – Intro

- **Flex Container Properties**

  - CSS Flexbox – display:flex;
  - CSS Flexbox – flex-direction
  - CSS Flexbox – flex-wrap
  - CSS Flexbox – flex-flow
  - CSS Flexbox – justify-content
  - CSS Flexbox – align-items
  - CSS Flexbox – align-content

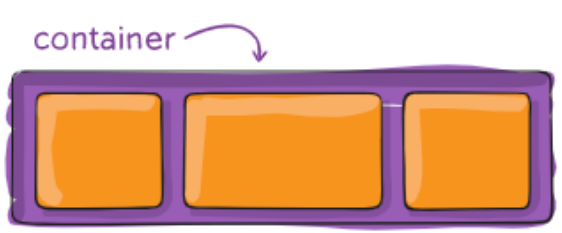
- **Flex Item Properties**

  - CSS Flexbox – order
  - CSS Flexbox – flex-grow
  - CSS Flexbox – flex-shrink
  - CSS Flexbox – flex-basis
  - CSS Flexbox – flex
  - CSS Flexbox – align-self

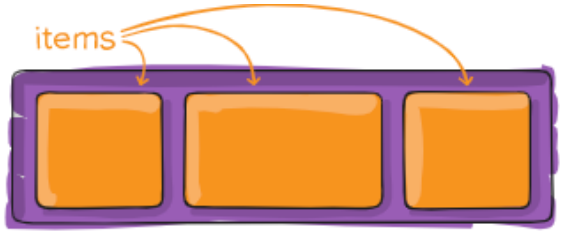
CSS Flexbox

- **Flex** is a *display* property like *block, inline or inline block.*
- Defining the **display property of a div** as *flex*, automatically creates a *flex container* and all it children become its *flex items.*

display: flex;



- **flex-direction**
- **flex-wrap**
- **flex-flow**
- **justify-content**
- **align-items**
- **align-content**

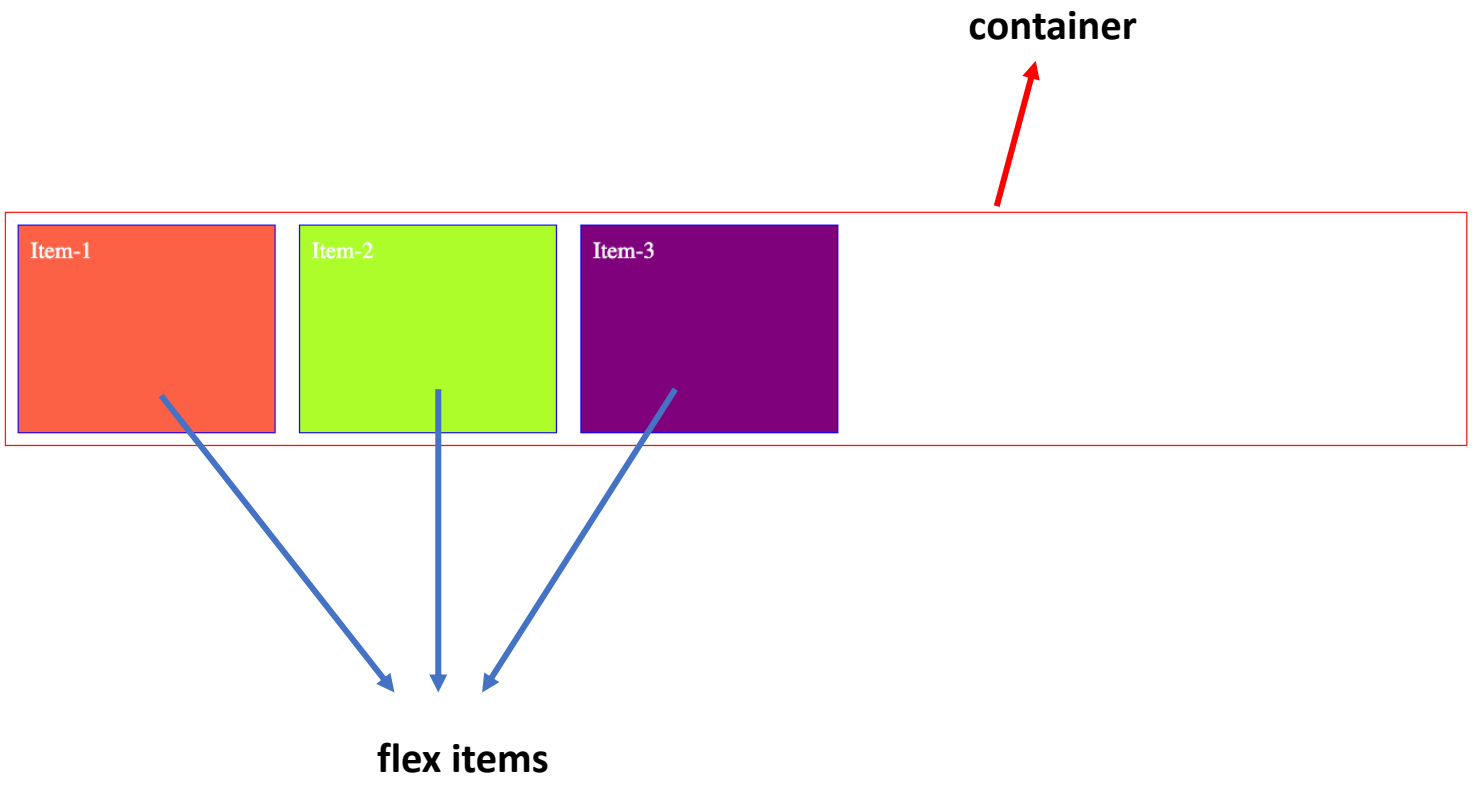


- **order**
- **flex-grow**
- **flex-shrink**
- **flex-basis**
- **flex**
- **align-self**

CSS Flexbox

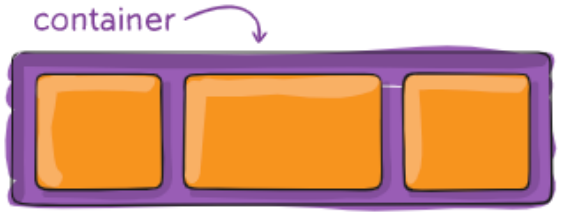
```css
.container {
    display: flex;
    width: 100%;
    height: 200px;
    border: 1px solid red;
}
.item {
    margin: 10px;
    padding: 10px;
    width: 200px;
    font-size: 20px;
    color: white;
}
.tomato {
    background: tomato;
}
.purple {
    background: purple;
}
.green {
    background: greenyellow;
}
```

```html
<div class="container">
    <div class="item tomato">Item-1</div>
    <div class="item green">Item-2</div>
    <div class="item purple">Item-3</div>
</div>
```

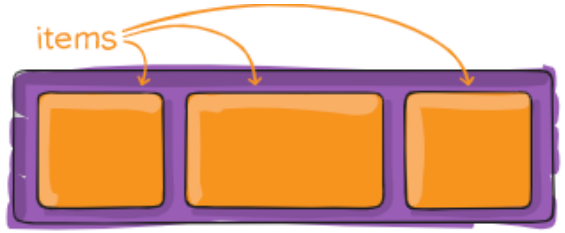


LAYOUT

CSS Flexbox



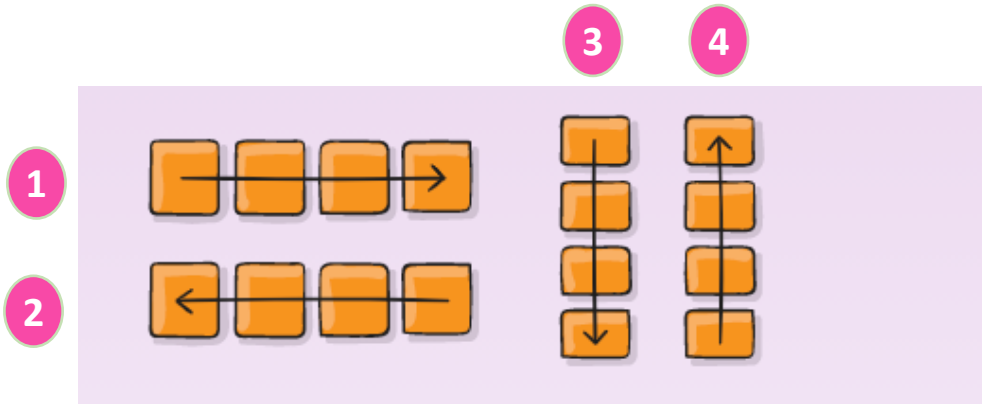
- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content



- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

CSS Flexbox – flex-direction

```
.container {
    flex-direction: row | row-reverse | column | column-reverse;
}
```



1. **row:** left to right – **by default**
2. **row-reverse:** right to left
3. **column:** top to bottom
4. **column-reverse:** bottom to top

Container Properties

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

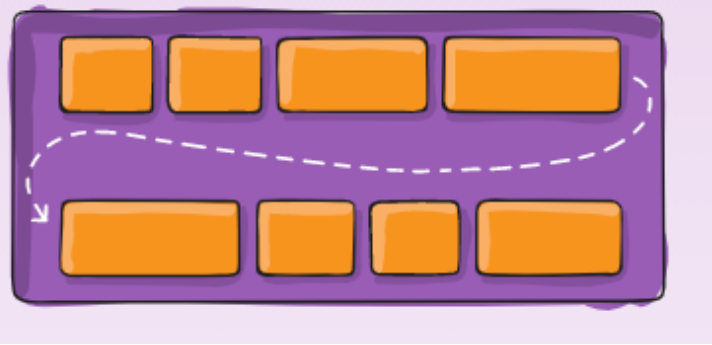
CSS Flexbox – flex-wrap

Container Properties

- flex-direction
- **flex-wrap**
- flex-flow
- justify-content
- align-items
- align-content

- By default, **flex items** will **all** try to fit onto one line.
- You can change that and allow the items to wrap as needed with this property.

```
.container {
    flex-wrap: nowrap | wrap | wrap-reverse;
}
```



- **nowrap:** all flex items will **be on one line** – **by default**
- **wrap:** flex items **will wrap onto multiple lines**, from **top to bottom**.
- **wrap-reverse:** flex items will **wrap onto multiple lines** from **bottom to top.**

CSS Flexbox – flex-flow

- This is a **shorthand** for the **flex-direction** and **flex-wrap** properties.
- The **default** value is **row nowrap**.

flex-flow: flex-direction flex-wrap;

```
flex-flow: row nowrap;
flex-flow: row wrap;
flex-flow: row wrap-reverse;
flex-flow: column nowrap;
flex-flow: column wrap;
flex-flow: column wrap-reverse;
```

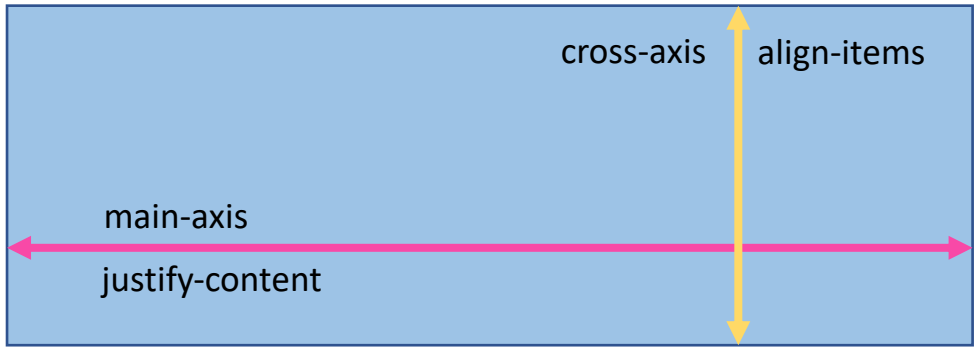
Container Properties

- flex-direction
- flex-wrap
- **flex-flow**
- justify-content
- align-items
- align-content

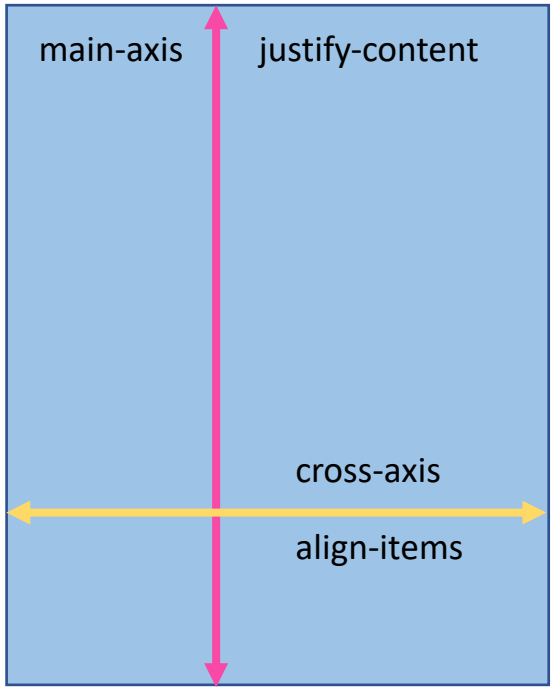
CSS Flexbox – justify-content

- **justify-content** for the main-axis
- **align-items** for the cross-axis

flex-direction : row



flex-direction : column



Container Properties

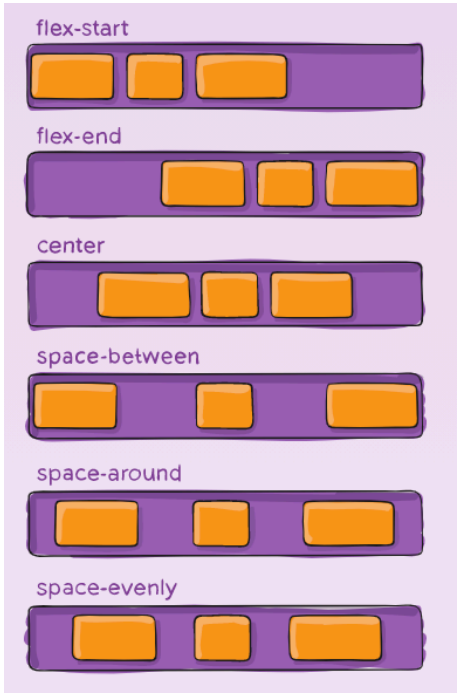
- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

CSS Flexbox – justify-content

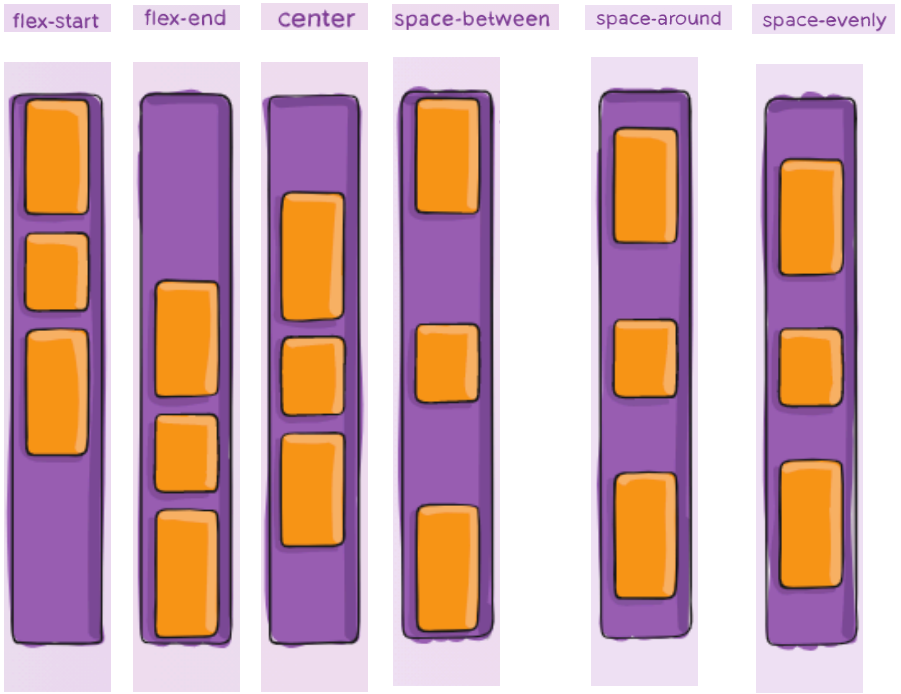
- This defines the alignment along the main axis.

```
.container {
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly;
}
```

main axis : row



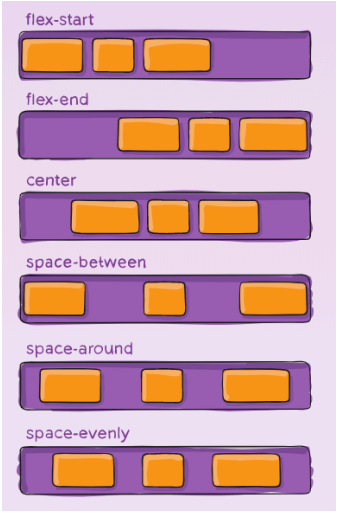
main axis : column



Container Properties

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

CSS Flexbox – justify-content



Container Properties

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

- **flex-start:** items are **packed toward the start of the flex-direction**.
- **flex-end:** items are **packed toward the end of the flex-direction.**
- **center:** items are **centered along the line**
- **space-between:** items are **evenly distributed in the line**; first item is on the start line, last item on the end line.
- **space-around:** items are **evenly distributed in the line with equal space around them**. Note that visually the spaces aren't equal, since all the items have equal space on both sides.
- **space-evenly:** items are distributed so that **the spacing between any two items (and the space to the edges) is equal**.

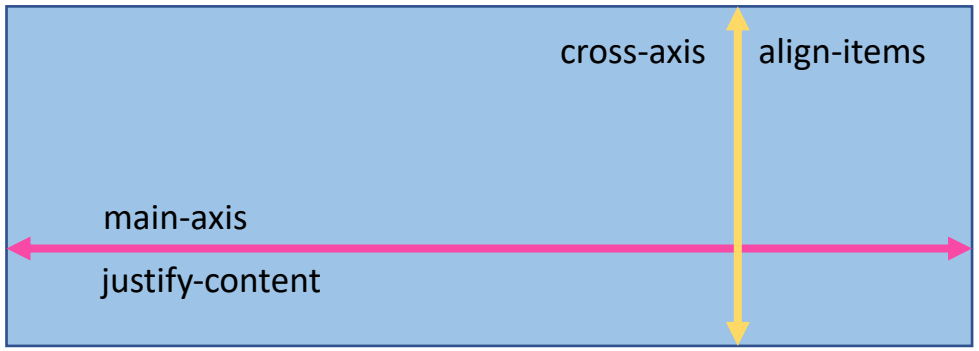
CSS Flexbox – align-items

- This defines the **default behavior for how flex items are laid out along the cross axis** on the current line.
- **justify-content** for the main-axis
- **align-items** for the cross-axis

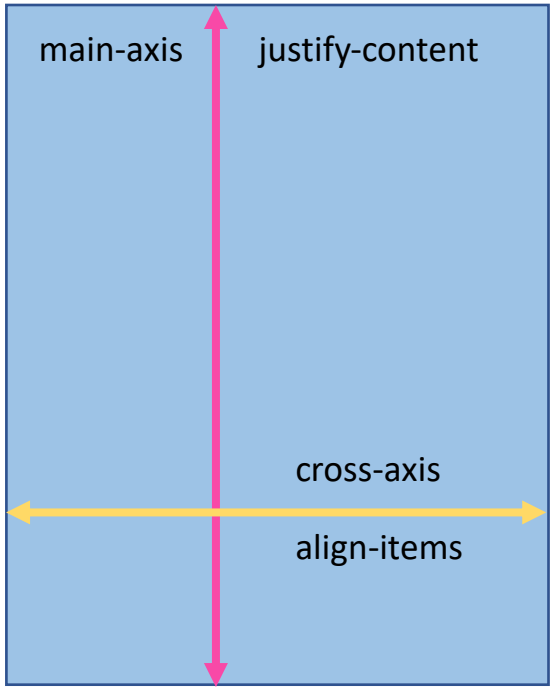
Container Properties

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

flex-direction : row



flex-direction : column



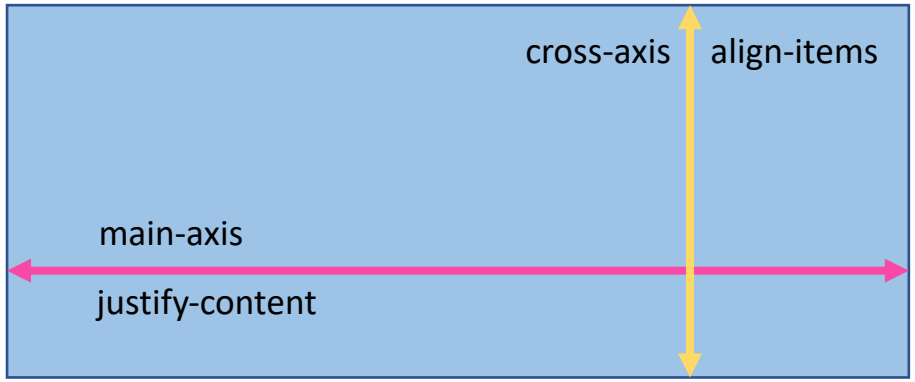
CSS Flexbox – align-items

```
.container {
    align-items: stretch | flex-start | flex-end | center | baseline;
}
```

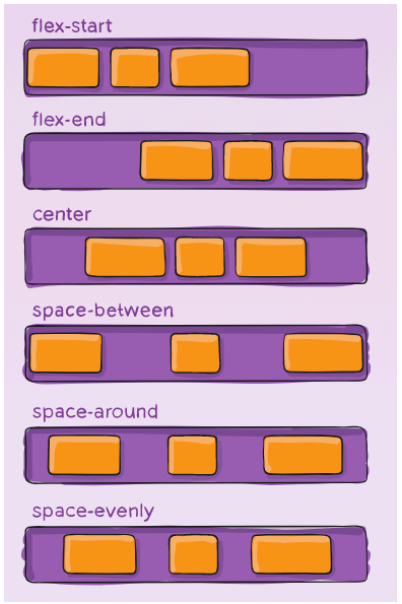
Container Properties

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- **align-items**
- align-content

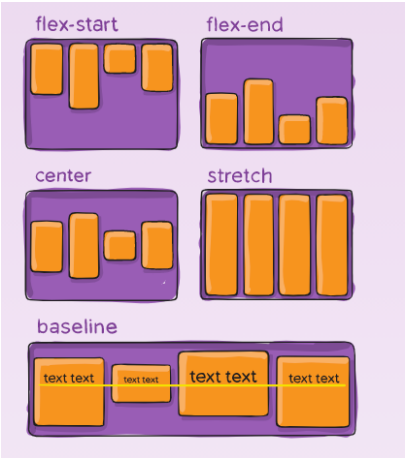
flex-direction : row



justify-content



align-items



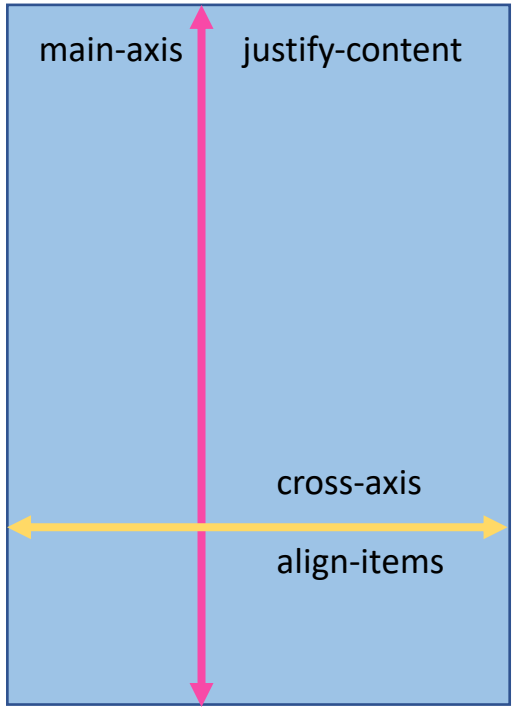
- **stretch:** stretch to fill the container – **by default**
- **flex-start:** items are placed at the start of the cross axis.
- **flex-end:** items are placed at the end of the cross axis.
- **center:** items are centered in the cross-axis.
- **baseline:** items are aligned such as their baselines align.

CSS Flexbox – align-items

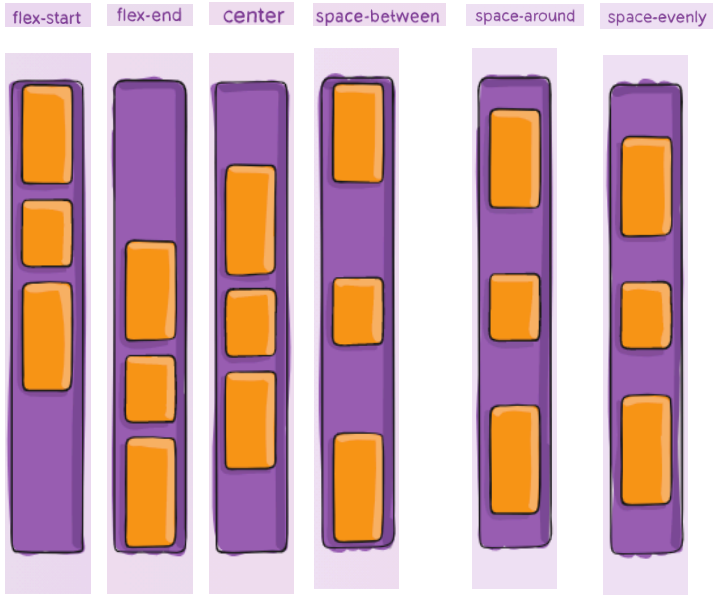
Container Properties

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

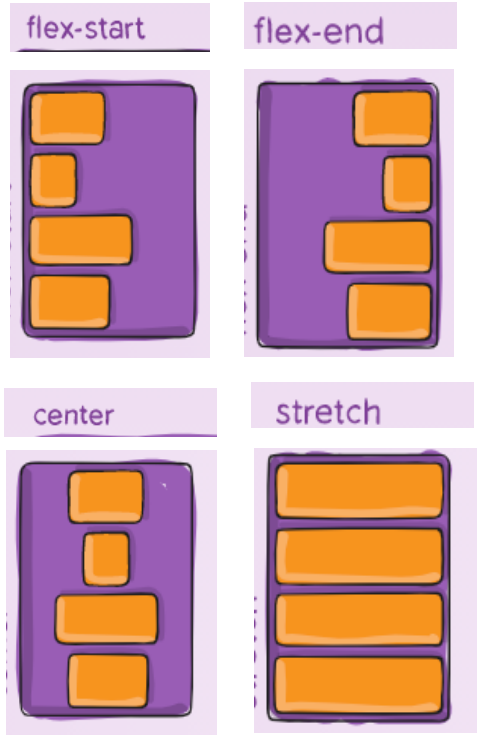
flex-direction : column



justify-content



align-items



CSS Flexbox – align-content

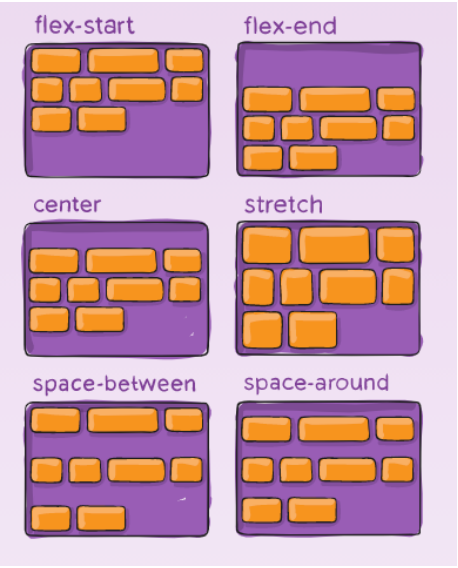
Container Properties

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- **align-content**

- To align elements on the cross-axis plane, when there is a lot of available space we can use the property, ***align-content***.

- This property should be *flex-wrap*.

```
.container {
    align-content: flex-start | flex-end | center | space-between | space-around | space-evenly;
}
```



- **flex-start:** items packed to the **start of the container**.
- **flex-end:** items packed to the **end of the container**.
- **center:** items **centered** in the container
- **space-between:** items **evenly distributed**; the first line is at the start of the container while the last one is at the end
- **space-around:** items **evenly distributed with equal space around each line**
- **space-evenly:** items are **evenly distributed with equal space around them**
- **stretch:** lines **stretch to take up the remaining space** – by default

CSS Flexbox – order

Items Properties

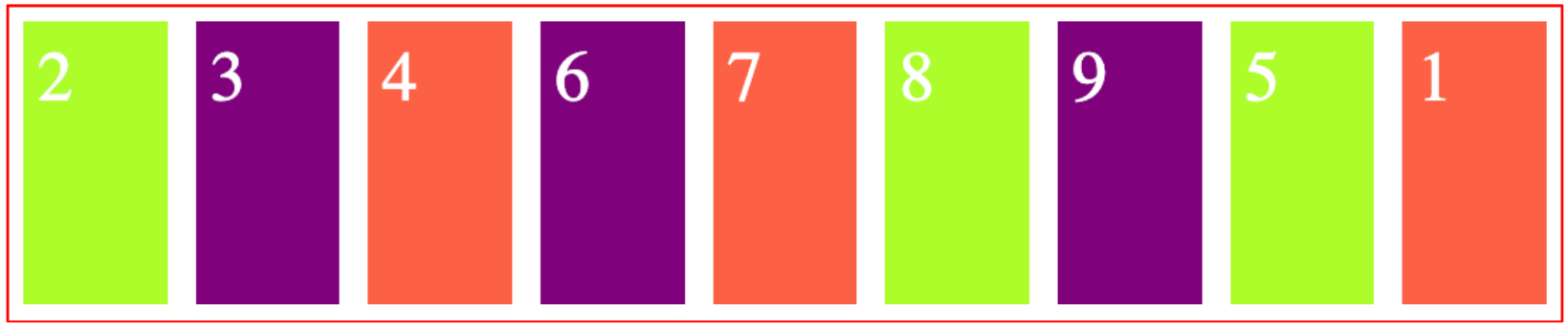
- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

- By default, flex items are **laid out in the source order**.
- However, the **order** property controls the order in which they appear in the flex container.

```css
.one {
    order: 5;
}

.five {
    order: 1;
}
```

```html
<div class="container">
    <div class="item tomato one">1</div>
    <div class="item green two">2</div>
    <div class="item purple three">3</div>
    <div class="item tomato four">4</div>
    <div class="item green five">5</div>
    <div class="item purple six">6</div>
    <div class="item tomato seven">7</div>
    <div class="item green eight">8</div>
    <div class="item purple nine">9</div>
</div>
```



NOTE : By default **order: 0;**

CSS Flexbox – flex-grow

Items Properties

- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

- **Flex-grow**, when applied to an item will scale it relative to the sum of the size of all other items on the same row, which are automatically adjusted according the the value that was specified.

```
.one {
    flex-grow: 8;
}
.two {
    flex-grow: 2;
}
.three {
    flex-grow: 2;
}
```


```
<div class="container">
    <div class="item tomato one">1</div>
    <div class="item green two">2</div>
    <div class="item purple three">3</div>
</div>
```

CSS Flexbox – flex-shrink

Items Properties

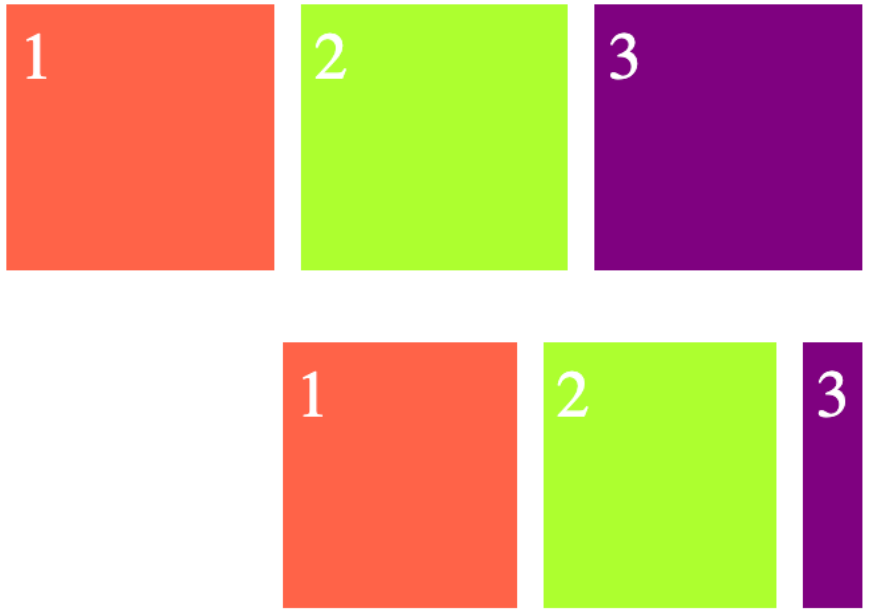
- order
- flex-grow
- **flex-shrink**
- flex-basis
- flex
- align-self

- This property is used to determine the rate at which an item will shrink relative to all items in the container when less space is provided.
- This defines the ability for a flex item to shrink if necessary.

```
.item {
    margin: 10px;
    padding: 10px;
    width: 200px;
    height: 200px;
    font-size: 50px;
    color: white;
    flex-grow: 1;
}
```

```
.three {
    flex-shrink: 7;
}
```

```
<div class="item tomato one">1</div>
<div class="item green two">2</div>
<div class="item purple three">3</div>
```



NOTE : flex-wrap: nowrap; should be

CSS Flexbox – flex-basis

Items Properties

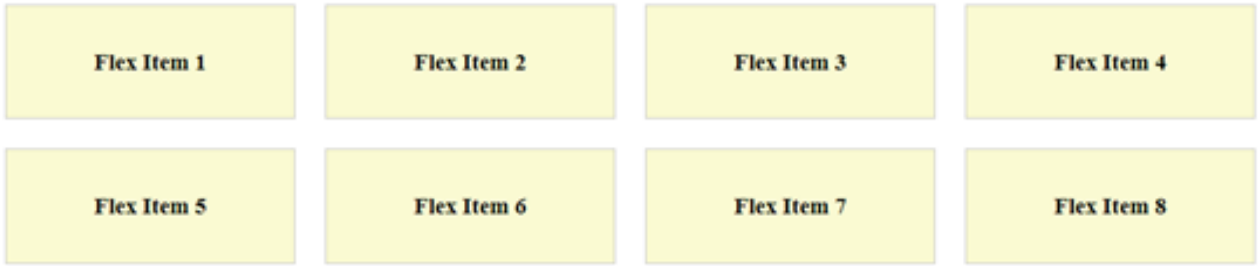
- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

This defines the default size of an element before the remaining space is distributed.

Instead of flex and width properties we can use flex-basis as well.

```
.item {
    flex-basis:  length | auto; /* default auto */
}
```

```
.item {
    background: lightgoldenrodyellow;
    border: #ccc solid 1px;
    padding: 1rem;
    text-align: center;
    /* flex: 1; */
    flex-basis: 200px;
    margin: 0.75rem;
}
```



NOTE : If both *flex-basis* and *width* are defined, *width* will be ignored.

```
width < flex-basis < max/min-width
```

CSS Flexbox – flex-basis

Items Properties

- order
- flex-grow
- flex-shrink
- **flex-basis**
- flex
- align-self

What is the reason to use flex-basis over simply defining the width ?

it is more dynamic in nature. If the available space (parent width) is less than the required width, it automatically applies ***flex-shrink*** and makes the items fit the flex container's width.

NOTE : If both *flex-basis* and *width* are defined, *width* will be ignored.

```
width < flex-basis < max/min-width
```

CSS Flexbox – flex

Items Properties

- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

This is the shorthand for flex-grow, flex-shrink and flex-basis combined.

```
flex:  flex-grow  flex-shrink  flex-basis;  // by default 0 1 auto;
```

required optional optional

```css
.item {
    margin: 10px;
    padding: 10px;
    height: 200px;
    font-size: 50px;
    color: white;
    flex: 1;
}
```

```html
<div class="item tomato one">1</div>
<div class="item green two">2</div>
<div class="item purple three">3</div>
```

- flex:1; will give, each child equal width and fill up the container width.
- This also makes the items more dynamic, they will automatically adjust the item's width in respect to the browsers width.

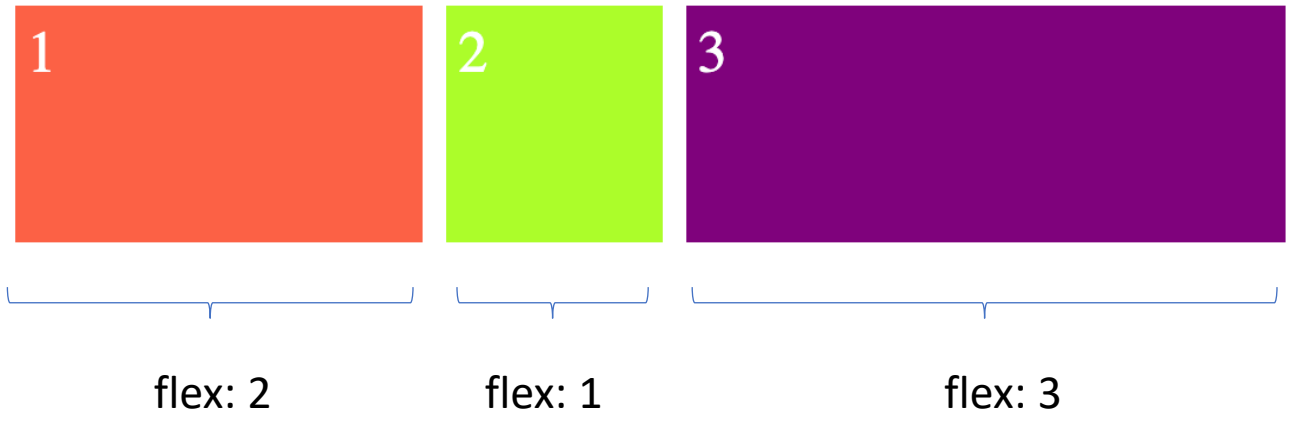
CSS Flexbox – flex

Items Properties

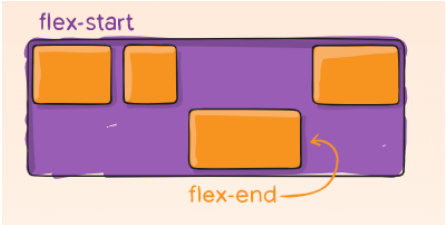
- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

```css
.item {
    margin: 10px;
    padding: 10px;
    height: 200px;
    font-size: 50px;
    color: white;
    flex: 1;
}
.one {
    flex: 2;
}
.three {
    flex: 3;
}
```

```html
<div class="item tomato one">1</div>
<div class="item green two">2</div>
<div class="item purple three">3</div>
```



CSS Flexbox – align-self



Items Properties

- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- **align-self**

This allows the default alignment (or the one specified by align-items) to **be overridden for individual flex items.**

```css
.container {
    width: 100%;
    border: 1px solid red;
    display: flex;
}
```

```css
.item {
    margin: 10px;
    padding: 10px;
    height: 200px;
    font-size: 50px;
    color: white;
    flex: 1;
}
.two {
    align-self: center;
}
.three {
    align-self: flex-end;
}
```

```html
<div class="item tomato one">1</div>
<div class="item green two">2</div>
<div class="item purple three">3</div>
```