Flexboxes

Introduction

The Flexible Box Module, usually referred to as flexbox, was designed as a one-dimensional layout model, and as a method that could offer space distribution between items in an interface and powerful alignment capabilities. This article gives an outline of the main features of flexbox, which we will be exploring in more detail in the rest of these guides.

- To declare a container as flexbox, use CSS display property to assign it to ‘flex’

```
.container {
    display: flex;
}
```

HTML Code for Reference

- Examples using this code are going to be reffered across this presentation.
- In addition to this, simple CSS styling such as background-color, font-size, border, margin, height and width have been assigned to both .container and .item selectors in our CSS files. This styling code is not shown here. Only flexbox related code is shown in this presentation.

```
<div class="container">
    <div class="item one">Item 1</div>
    <div class="item two">Item 2</div>
    <div class="item three">Item 3</div>
    <div class="item four">Item 4</div>
</div>
```

Flexbox Container Properties

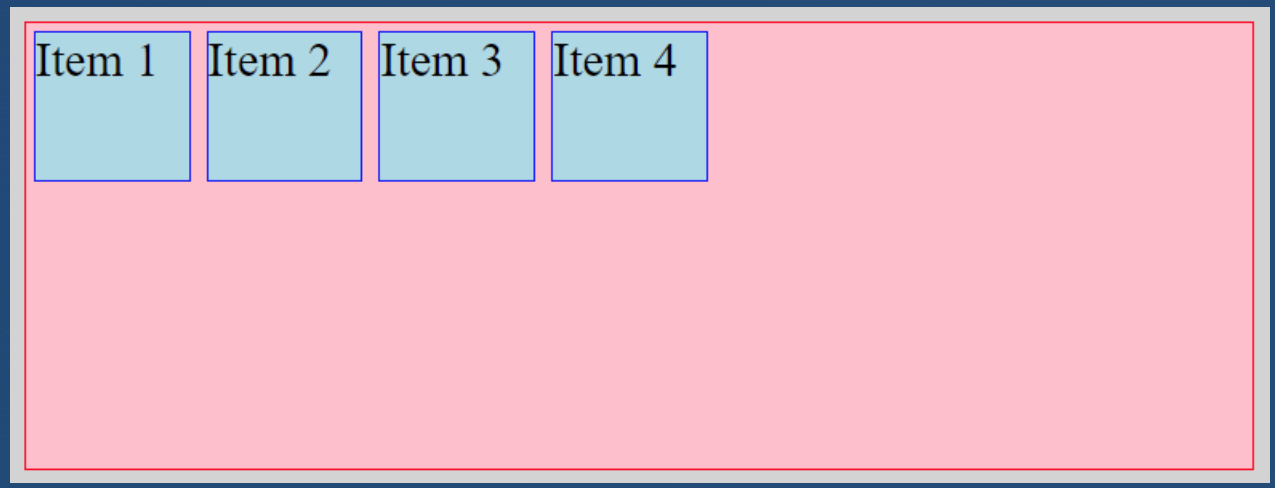
- flex-direction
- flex-wrap
- justify-content
- align-items
- align-content

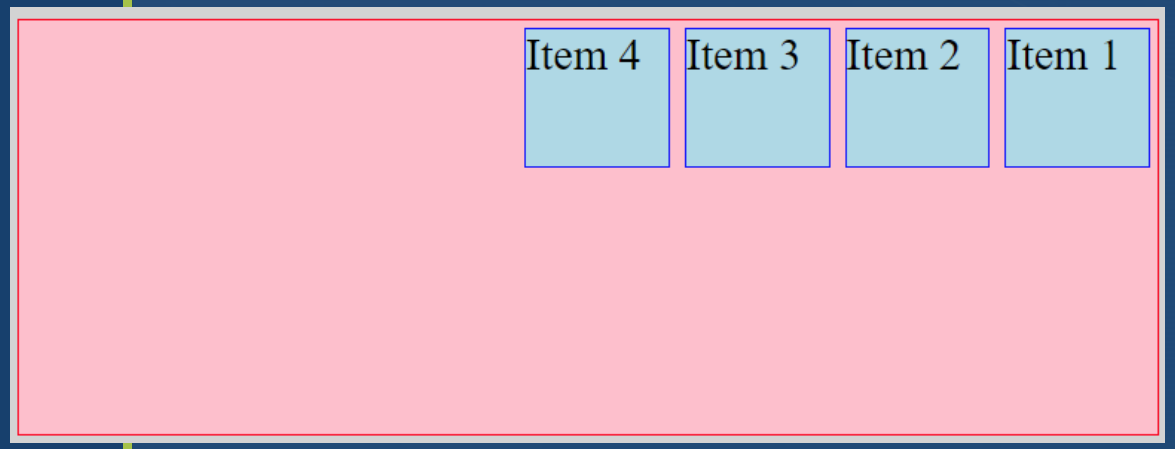
flex-direction

- This property sets how flex items are placed in the flex container defining the main axis and the direction (normal or reversed).
- Valid values – row, column, row-reverse, column-reverse

flex-direction

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

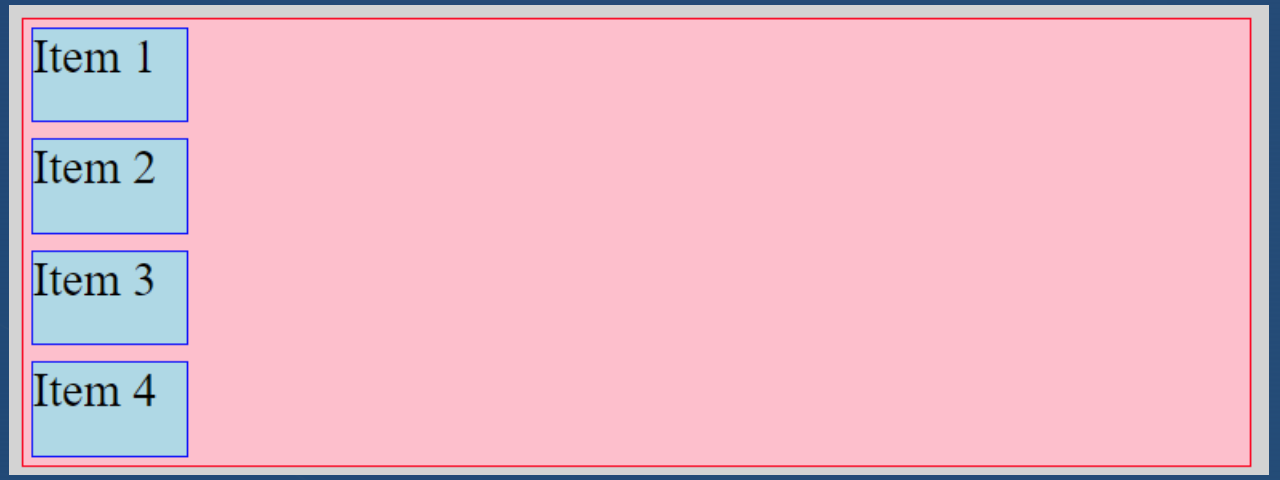


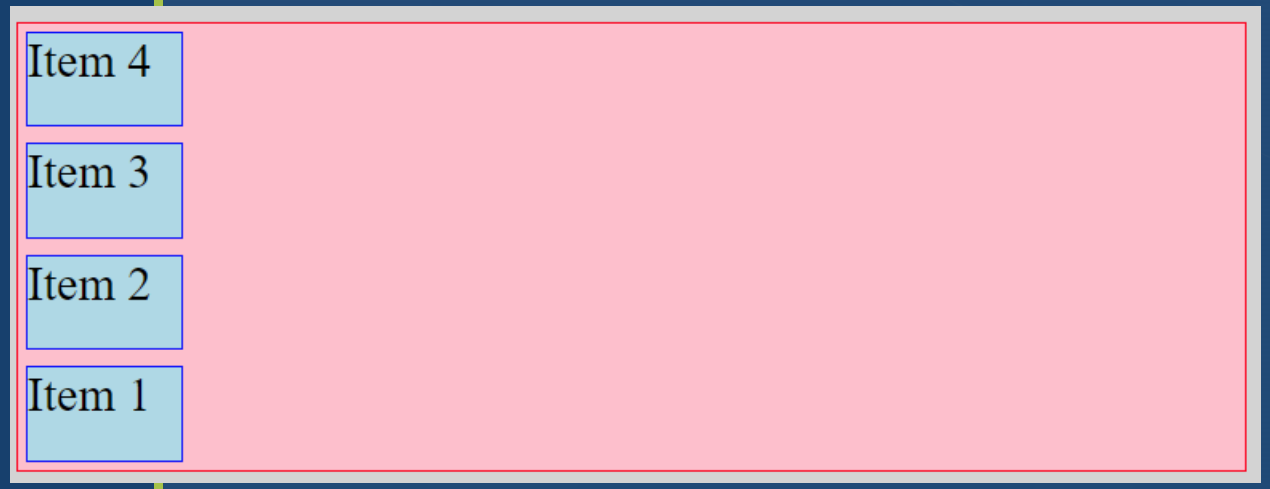



```
.container {
    display: flex;
    flex-direction: row-reverse;
}
```

flex-direction

```
.container {
    display: flex;
    flex-direction: column;
}
```





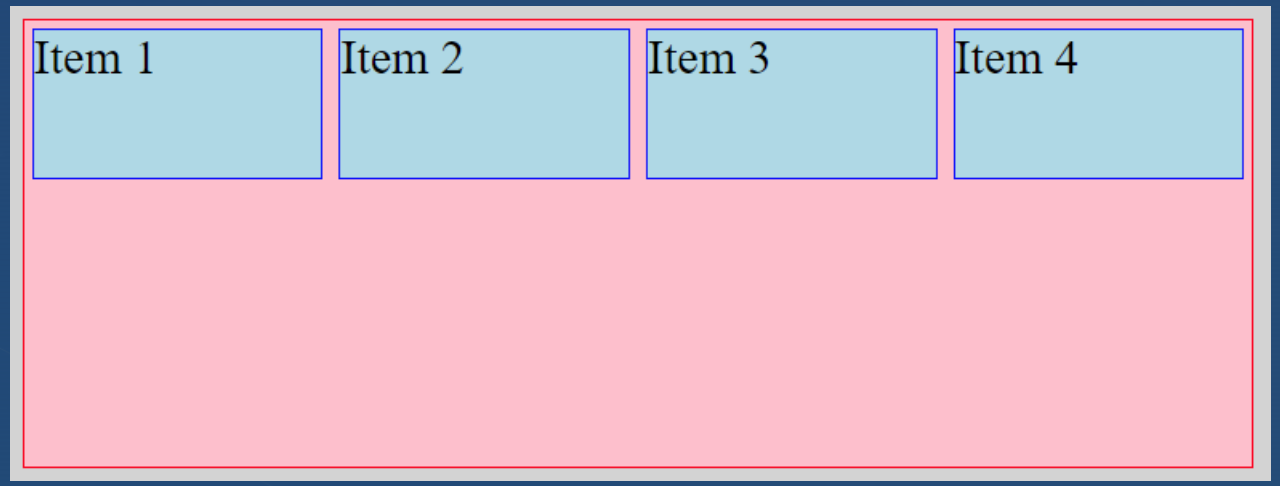

```
.container {
    display: flex;
    flex-direction: column-reverse;
}
```

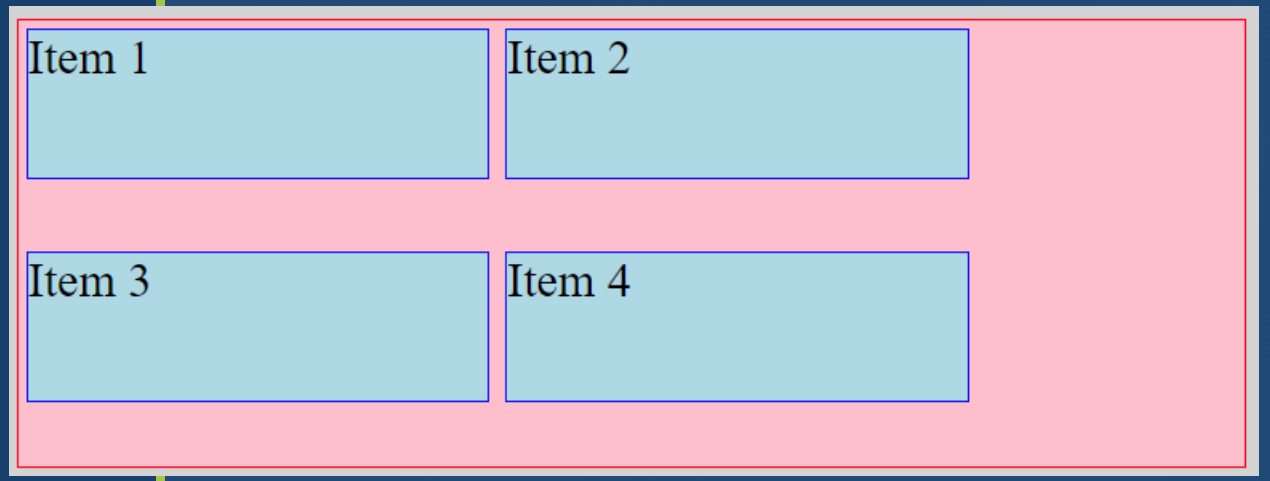
flex-wrap

- This CSS property sets whether flex items are forced onto one line or can wrap onto multiple lines. If wrapping is allowed, it sets the direction that lines are stacked.
- Valid values – nowrap (default) , wrap, wrap-reverse

flex-wrap

```
.container {
    display: flex;
    flex-direction: row;
    flex-wrap: nowrap;
}
```

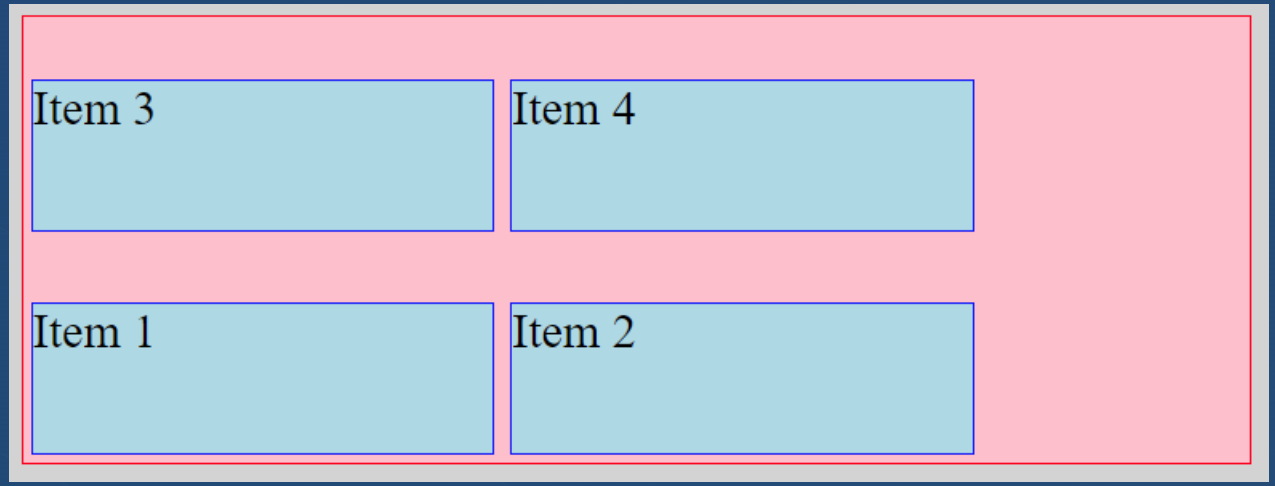





```
.container {
    display: flex;
    flex-direction: row;
    flex-wrap: wrap;
}
```

flex-wrap

```
.container {
    display: flex;
    flex-direction: row;
    flex-wrap: wrap-reverse;
}
```

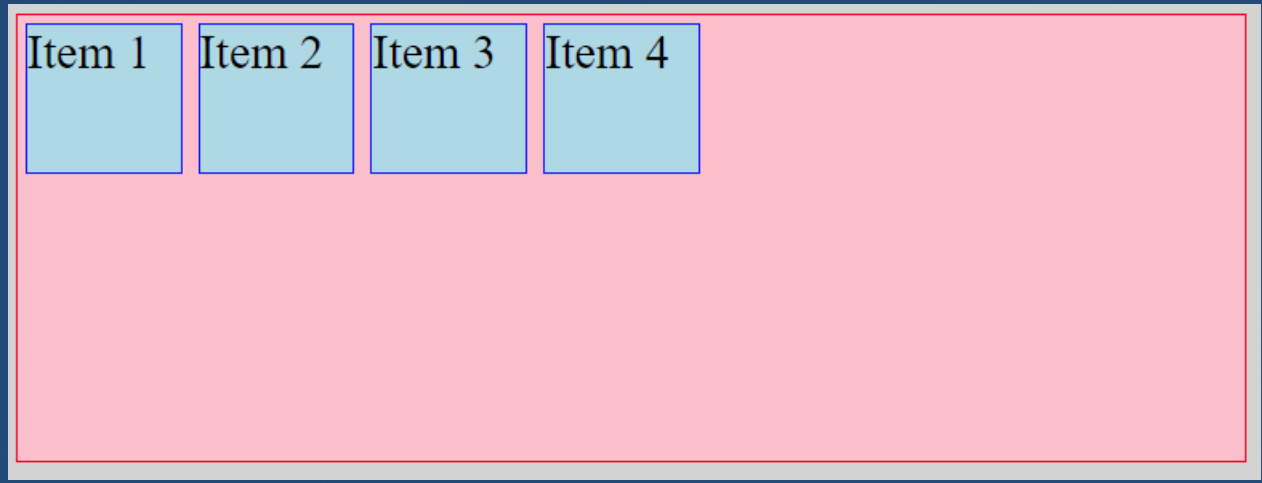


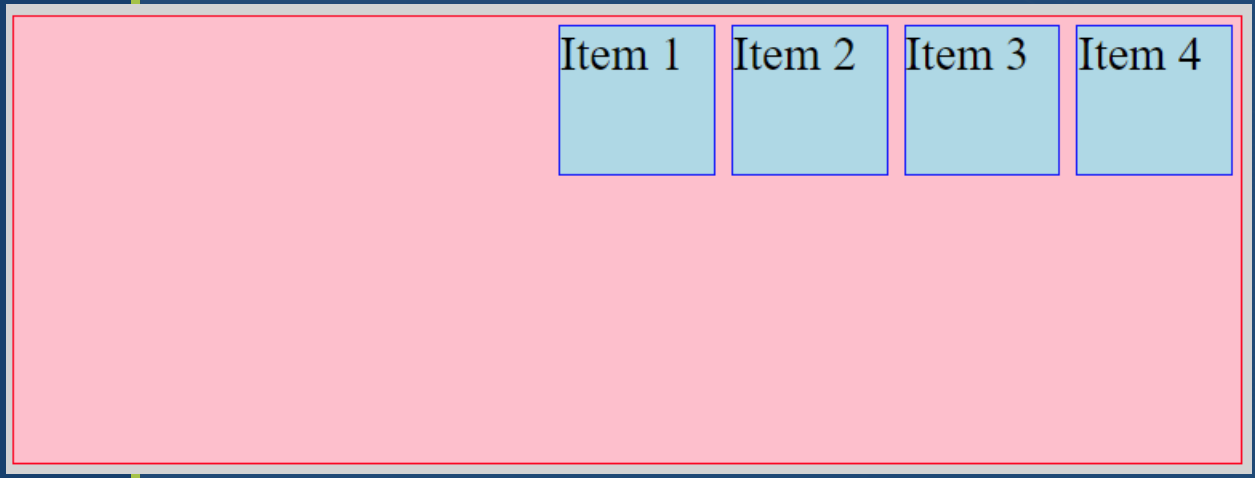
justify-content

- This property defines how the browser distributes space between and around content items along the main-axis of a flex container, and the inline axis of a grid container.
- Valid values – flex-start, flex-end, center, space-between, space-around, space-evenly

justify-content

```
.container {
    display: flex;
    flex-direction: row;
    justify-content: flex-start;
}
```

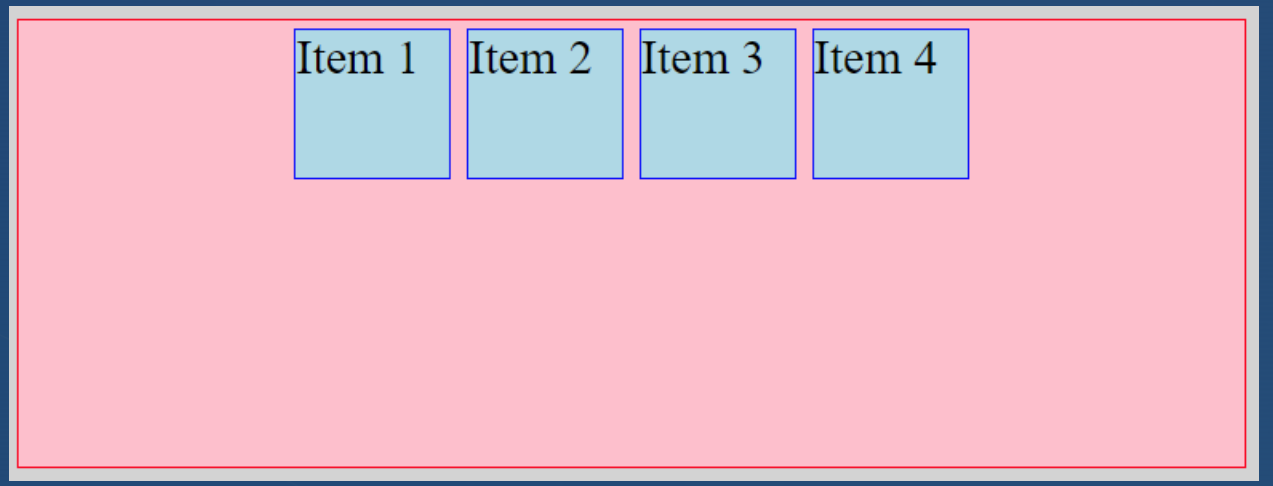


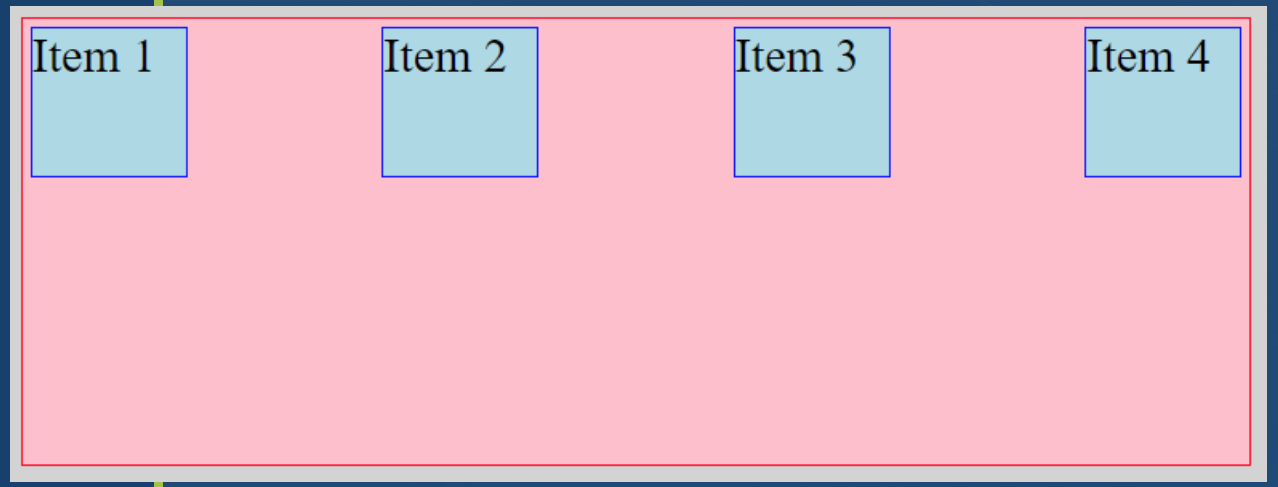



```
.container {
    display: flex;
    flex-direction: row;
    justify-content: flex-end;
}
```

justify-content

```
.container {
    display: flex;
    flex-direction: row;
    justify-content: center;
}
```

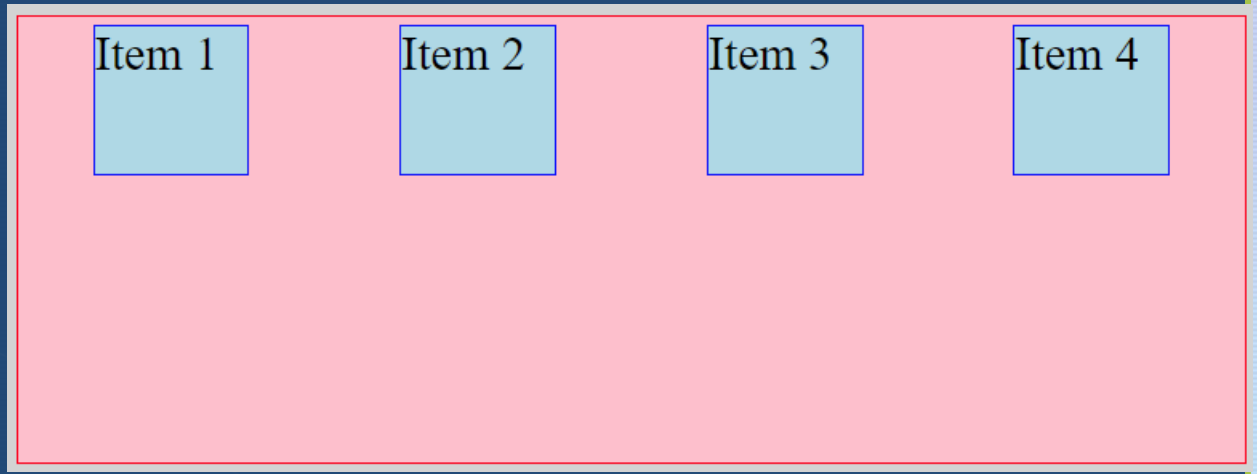


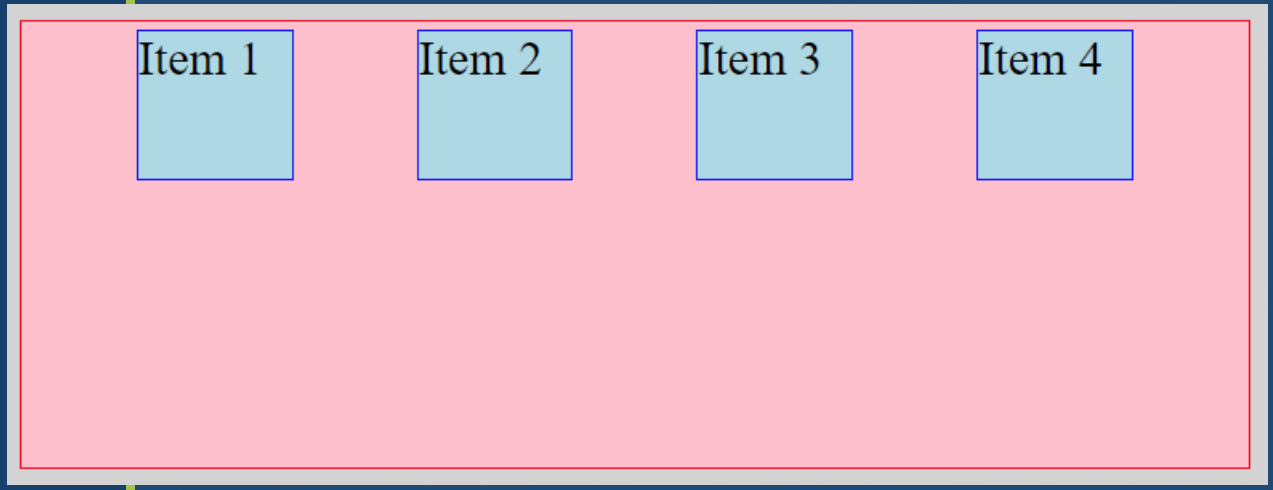



```
.container {
    display: flex;
    flex-direction: row;
    justify-content: space-between;
}
```

justify-content

```
.container {
    display: flex;
    flex-direction: row;
    justify-content: space-around;
}
```



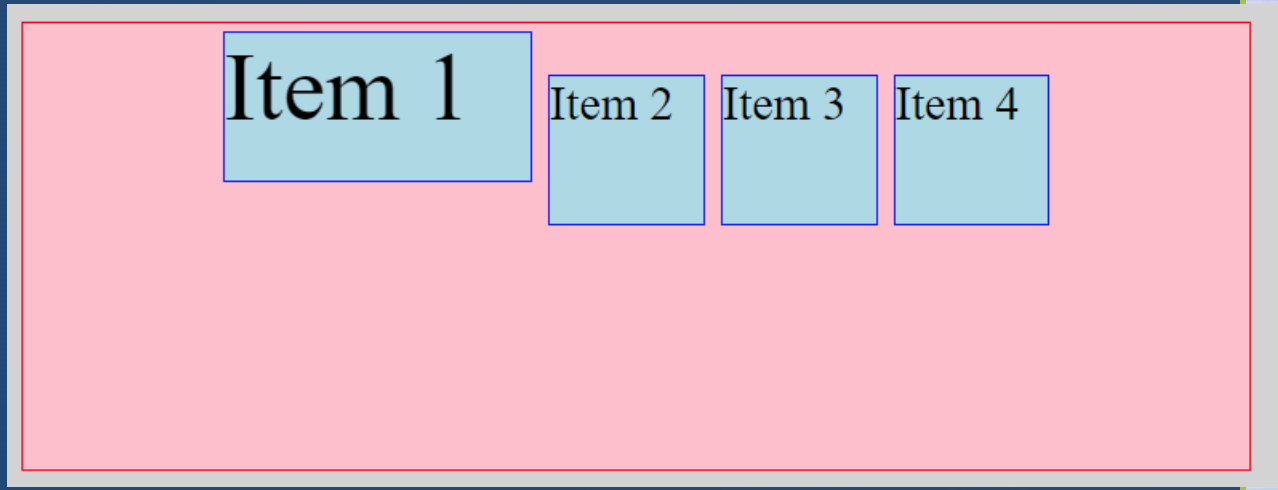



```
.container {
    display: flex;
    flex-direction: row;
    justify-content: space-evenly;
}
```

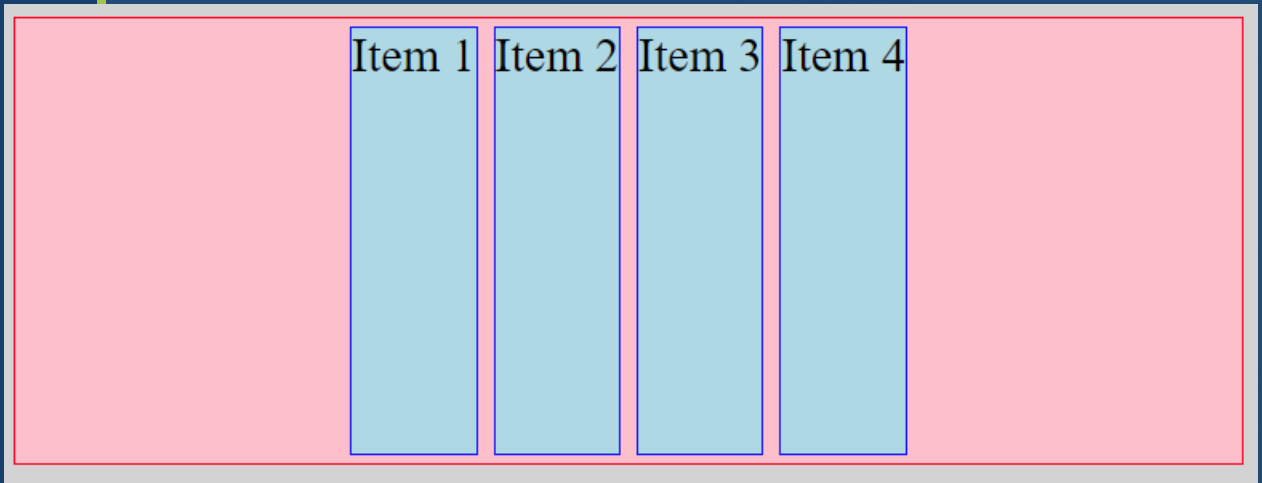
align-items

- This defines the default behavior for how flex items are laid out along the cross axis on the current line. Think of it as the justify-content version for the cross-axis (perpendicular to the main-axis).
- Valid values : flex-start, flex-end, center, baseline, stretch
- (Only baseline and stretch examples are given in following slides as the other three are same as justify-content but with only a different axis).

```
.container {
    display: flex;
    flex-direction: row;
    justify-content: center;
    align-items: baseline;
}
```



Font size and box size of item 1 box was enlarged separately for above example.



Height and width of all items was removed to apply the stretch property. Both height and width were defined with some value for all previous examples.

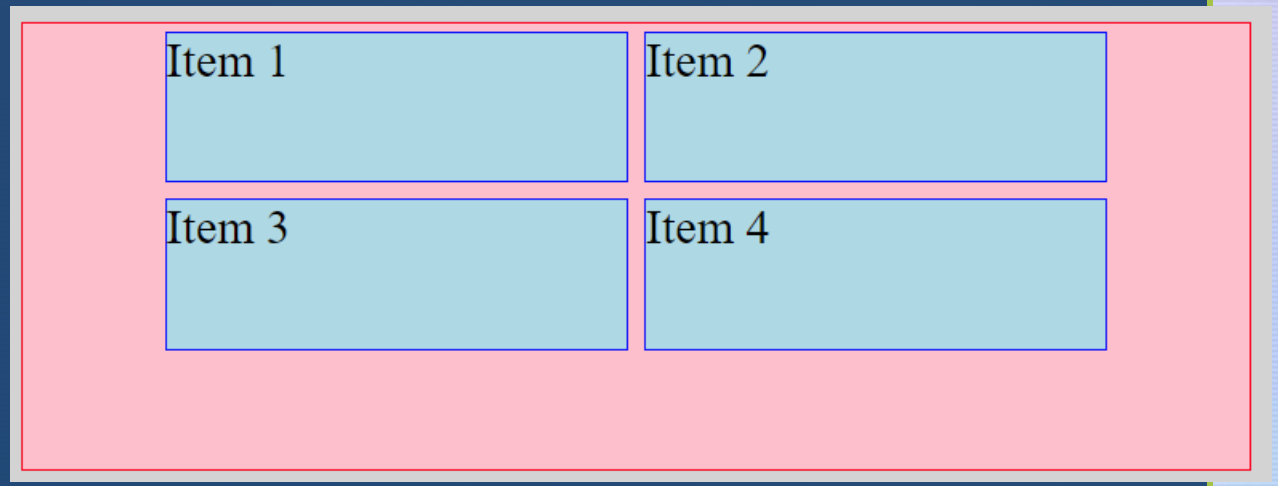
```
.container {
    display: flex;
    flex-direction: row;
    justify-content: center;
    align-items: stretch;
}
```

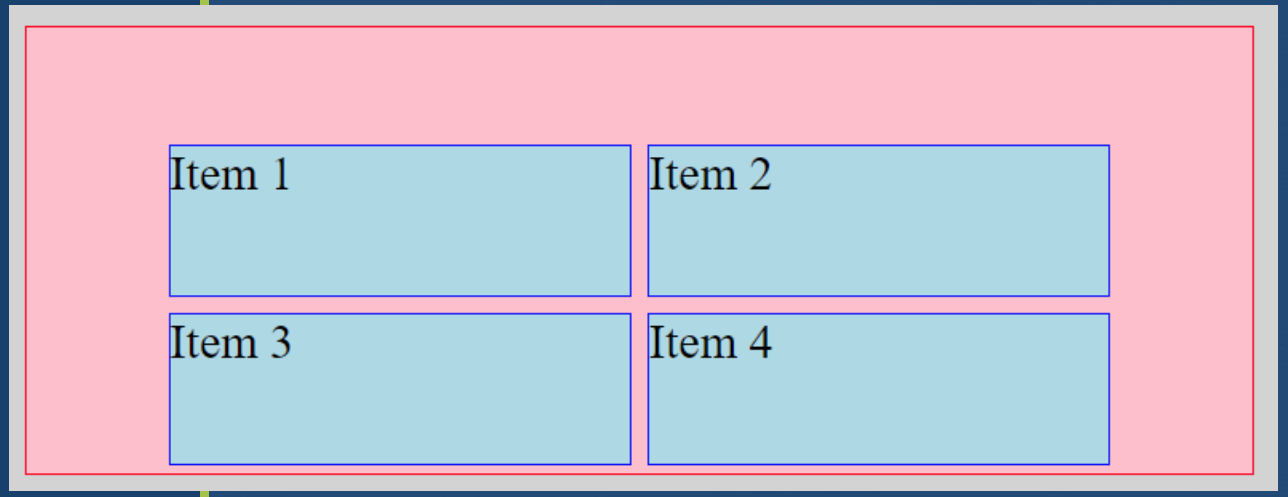
This property has no effect on single line flex containers (i.e. ones with `flex-wrap: nowrap`).

align-content

- This aligns a flex container’s lines within when there is extra space in the cross-axis, similar to how justify-content aligns individual items within the main-axis.
- This property has no effect on single line flex containers (i.e. ones with flex-wrap: nowrap).
- Valid values : flex-start, flex-end, center, stretch, space-between, space-around

```
.container {
    display: flex;
    flex-direction: row;
    justify-content: center;
    align-items: stretch;
    flex-wrap: wrap;
    align-content: flex-start;
}
```






```
.container {
    display: flex;
    flex-direction: row;
    justify-content: center;
    align-items: stretch;
    flex-wrap: wrap;
    align-content: flex-end;
}
```

Only 2 examples are done to show it works as the possible values are same as justify-content.

Flex Items properties

- order
- flex-grow
- flex-shrink
- flex-basis

Code snippets and sample screenshots could not be generated for these properties due to their mostly dynamic nature. Refer to the code to better understand these properties.

order

- 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.
- Defined 0 by default.

flex-grow

- This defines the ability for a flex item to grow if necessary. It accepts a unitless value that serves as a proportion. It dictates what amount of the available space inside the flex container the item should take up.

flex-shrink

- This defines the ability for a flex item to shrink if necessary.

flex-basis

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