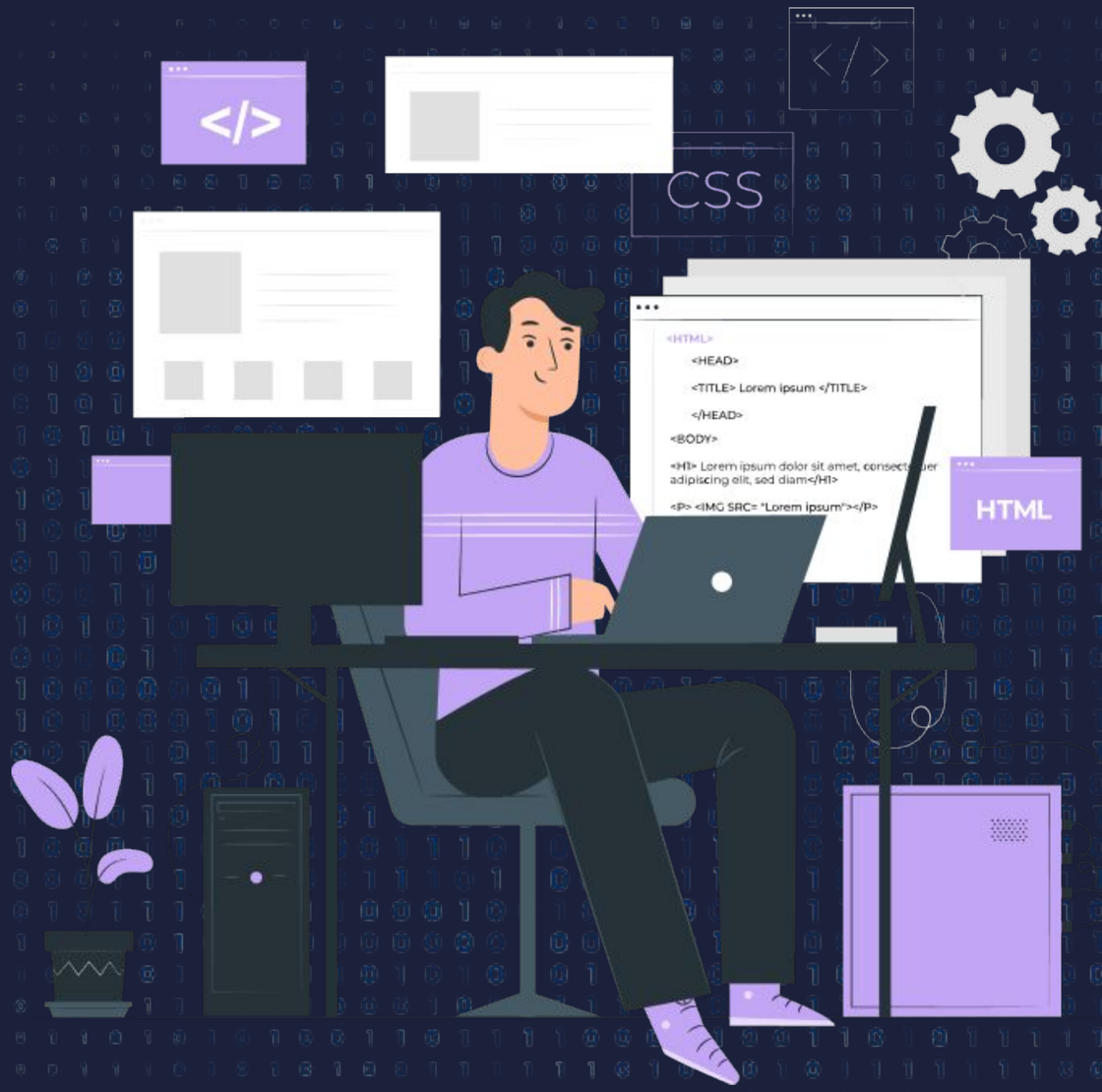
Flex item properties



Topics

- What are Flexbox item properties
- The different types of Flexbox properties, with the example

What are Flexbox item properties?

Flexbox provides several properties that are applied to individual flex items within a Flexbox layout.

Some commonly used properties are as follows

- **order(0)** - specifies the order of items appear in the container
- **align-self (auto)** - used align a single flex item along the cross-axis
- **flex-grow(0)** - specifies how flex item should grow relative to the other items in the container.
- **flex-shrink (1)** - specifies how the flex items should shrink relative to the items in the container.
- **flex-basis(auto)** - specifies the width of the item present inside the flex container.
- **flex(auto)** - shorthand for **flex-grow, flex-shrink** and **flex-basis**

Flex Item properties, with the example

HTML

```
<!— This html is used for all the example   —> —>
 <body>

    <div class="container">
      <div class="item item-1">1box</div>
      <div class="item item-2">2box</div>
      <div class="item item-3">3box</div>
      <div class="item item-4">4box</div>
    </div>
  </body>
```

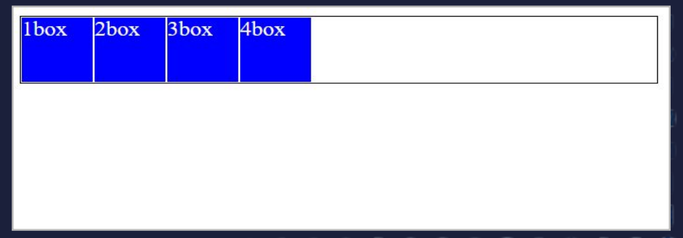
order

CSS

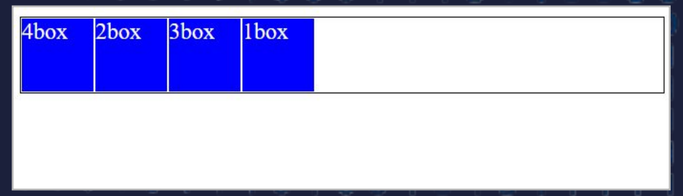
```
.container {
    display: flex;
    flex-direction: row;
    border: 1px solid black;
    color: white;
  }
  .item {
    width: 50px;
    height: 50px;
    background-color: blue;
    border: 1px solid;
  }
/* adding the properties */
  .item-1 {
    order: 4;
  }
  .item-2 {
    order: 2;
  }
  .item-3 {
    order: 3;
  }
  .item-4 {
    order: 1;
  }
```

Output

Before order



After order

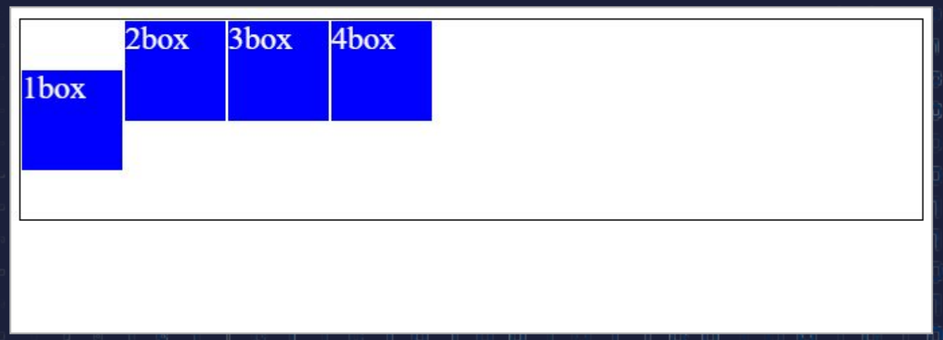


align-self

CSS

```
.container {
  display: flex;
  flex-direction: row;
  border: 1px solid black;
  color: white;
  height: 100px;
}
.item {
  width: 50px;
  height: 50px;
  background-color: blue;
  border: 1px solid;
}
/* aligning the item 1 to center
*/
.item-1 {
  align-self: center;
}
```

Output



flex-grow

CSS

```
.container {
  display: flex;
  flex-direction: row;
  border: 1px solid black;
  color: white;
  height: 100px;
}
.item {
  width: 50px;
  height: 50px;
  background-color: blue;
  border: 1px solid;
}
/* flex grow item properties */
.item-1 {
  flex-grow: 1;
}
```

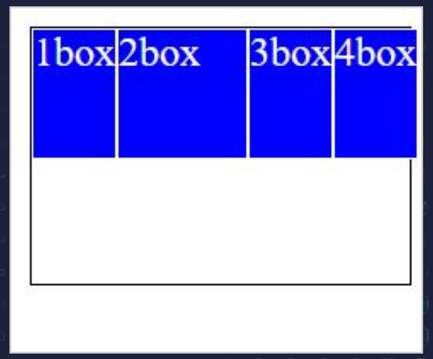
Output

flex-shrink

CSS

```
.container {
  display: flex;
  flex-direction: row;
  border: 1px solid black;
  color: white;
  height: 100px;
}
.item {
  width: 50px;
  height: 50px;
  background-color: blue;
  border: 1px solid;
}
/* 0 shrink value indicate it
will not shrink
   so, item-2 will not shrink.
 */
.item-2 {
  flex-shrink: 0;
}
```

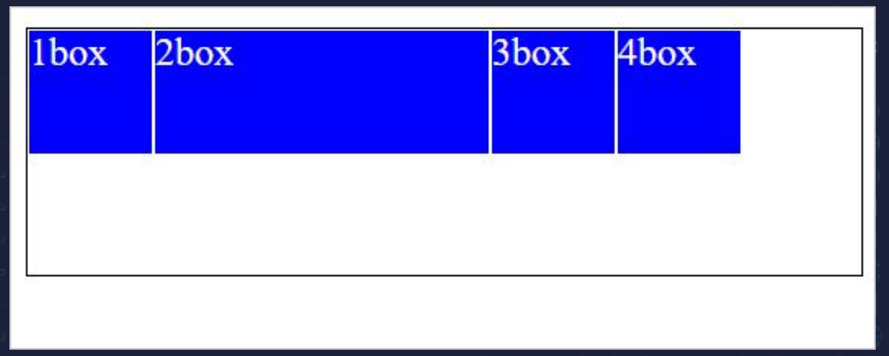
Output



flex-basis

```
.container {
  display: flex;
  flex-direction: row;
  border: 1px solid black;
  color: white;
  height: 100px;
}
.item {
  width: 50px;
  height: 50px;
  background-color: blue;
  border: 1px solid;
}
/* change the width of the 2nd
item*/
.item-2 {
  flex-basis: 40%;
}
```

Output



flex

Syntax -

```
flex: <flex-grow> <flex-shrink> <flex-basis>
```

```
.container {
    display: flex;
    flex-direction: row;
    border: 1px solid black;
    color: white;
  }
  .item {
    width: 50px;
    height: 50px;
    background-color: blue;
    border: 1px solid;
  }
/* flex - flex grow flex-shrink flex-basis */
  .item-1 {
    flex: 1 0 200px;
  }
  .item-2 {
    flex: 1 0 400px;
  }
```

Output


SKILLS
THANK YOU