Lesson:

Flex item properties

PW SKILLS

Topics Covered

- 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. Here are some of the most commonly used properties -

1. **order (0) -** specifies the order in which flex items appear in the container, without affecting their position or size. It takes an integer value and is applied to individual flex items.
2. **align-self (auto) -** used to align a single flex item along the cross-axis, which is perpendicular to the main axis. It takes the same values as the align-items property, and is applied to individual flex items.
3. **flex-grow (0) -** specifies how much a flex item should grow relative to the other items in the container.
4. **flex-shrink (1) -** specifies how much a flex item should shrink relative to the other items in the container.
5. **flex-basis (auto) -** specifies the initial size of a flex item before any growing or shrinking occurs. The value can be a length, a percentage, or the keyword auto.
6. **flex (auto) -** shorthand property that combines the **flex-grow, flex-shrink,** and **flex-basis** properties into a single declaration.

The different types of Flexbox properties, with the example

order

Items are ordered based on the order they are assigned. By default, every item has order 0 and the appearance in the HTML determines the final order. This can be overridden by using the order flex property on each item.

Example-

Index.html (This HTML code will be to all the possible examples)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title>Flex demo</title>
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet" href="style.css" />
  </head>
  <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>
</html>
```

style.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;
  }
```

Browser output -

After adding the order properties it will change to -

align-self

An item can choose to override the container align-items setting, using **align-self**, which has the same 5 possible values of **align-items:**

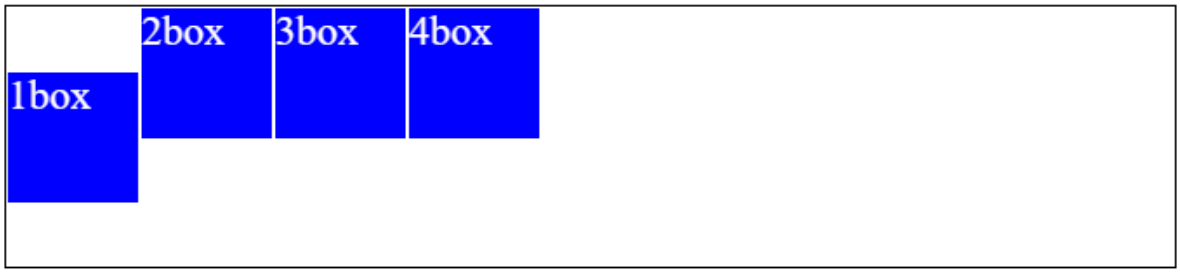
- flex-start OR start - align to the top of the container.
- flex-end OR end - align to the bottom of the container.
- center - align to the center of the container.
- baseline - display at the baseline of the container.
- stretch - items are stretched to fit the container.

Example

style.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;
}
```

Browser output -



flex-grow

It specifies how much a flex item will grow relative to the rest of the flex items. The default for any item of Flexbox is 0, if all items are defined as 1 and one is defined as 2, the bigger element will take the space of two "1" items

Example -
style

```
.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;
}
```

Browser output -

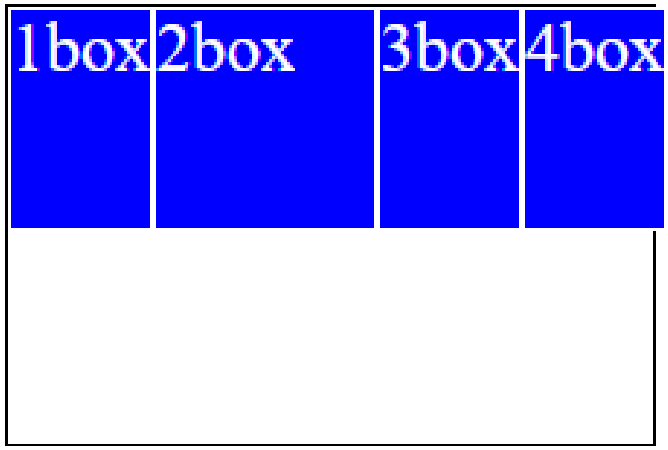
flex-shrink

It specifies how much a flex item will shrink relative to the rest of the flex items. The default for any item of Flexbox is 1, If all items are defined as 1 and one is defined as 3, the bigger element will shrink 3x of the other ones. When less space is available, it will take 3x less space.

Example -

```
.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;
}
```

Browser output -



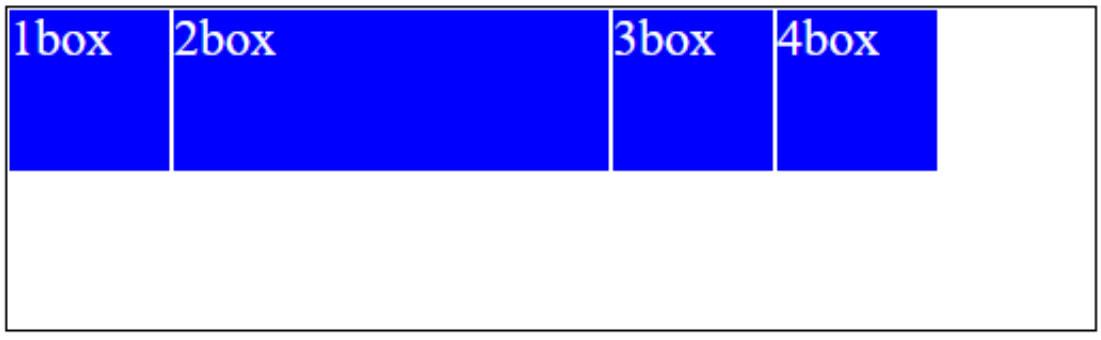
flex-basis

It specifies the initial length of a flex item. If set to auto, it sizes an item according to its width or height and adds extra space based on the flex-grow property.
If set to 0, it does not add any extra space for the item when calculating the layout.
If you specify a pixel number value, it will use that as the length value (width or height depends if it's a row or a column item)

Example -
style.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;
}
/* change the width of the 2nd item*/
.item-2 {
  flex-basis: 40%;
}
```

Browser output -



flex

This property combines the above 3 properties -

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

Syntax -

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

Example -

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

Browser output -