CSS Display Property

CSS Display

The CSS display property is one of the most fundamental and powerful properties in CSS. It defines how elements are displayed in the document flow, determining whether they are block-level, inline, flex, grid, or even hidden. Display is an important property to define layout of our page.

Block Elements

Block-level elements take up the full width of their container and start on a new line.

Example

```
.block-element {
    display: block;
}
```

Inline Elements

Inline elements do not start on a new line and only take up as much width as necessary.

Example

```
.inline-element {
    display: inline;
}
```

CSS Display

Inline-Block Elements

Inline-block elements behave like inline elements but allow setting width and height, which is not possible with purely inline elements

Example

```
.block-element {
    display: inline-block;
}
```

None (Hiding Elements)

Using display: none; completely removes an element from the document flow.

Example

```
.hidden-element {
    display: none;
}
```

CSS Display

Flexbox

Flexbox, short for Flexible Box Layout, is a CSS module designed to arrange elements efficiently in a one-dimensional layout, either as a row or a column. It provides powerful alignment and distribution capabilities, making it ideal for responsive design.

Flexbox consists of two primary components:

- **Flex Container** - The parent element that contains flex items.
- **Flex Items -** The child elements within the flex container that are positioned according to flexbox rules.

Example

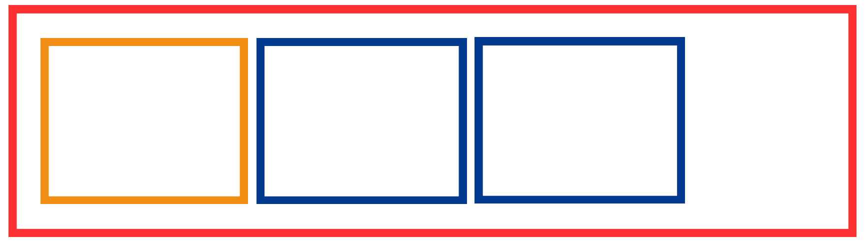
```
.container{
    display: flex;
}
```

CSS Flexbox

Flex Container Properties

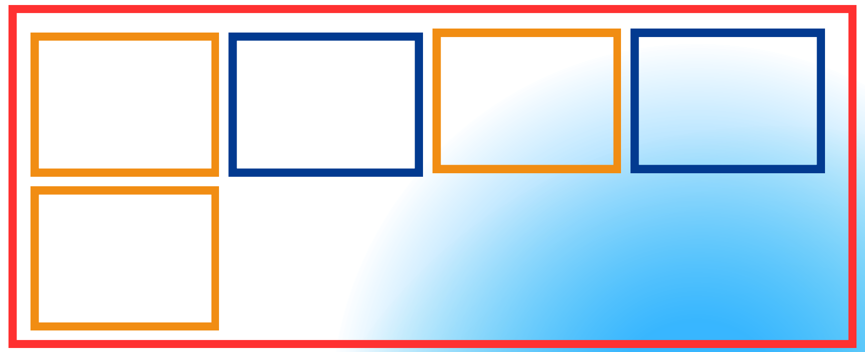
1. flex-direction : Defines the direction in which flex items are placed in the container

```
.container{
    display: flex;
    flex-direction: row;  /* row (Default) | row-reverse | column | column-reverse */
}
```



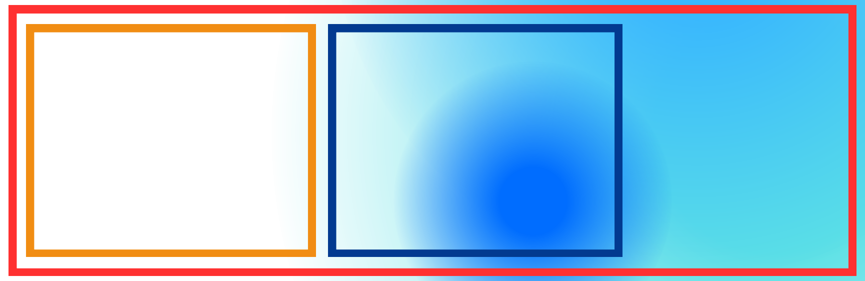
2. flex-wrap : Determines whether items should wrap to the next line when the container is too small.

```
.container{
    display: flex;
    flex-wrap: wrap;  /* nowrap (Default) | wrap | wrap-reverse */
}
```



3. justify-content : Aligns items along the flex-direction (horizontally for row, vertically for column).

```
.container{
    display: flex;
    justify-content: flex-start;  /* flex-start (Default) | flex-end | center | space-between | space-around | space-evenly */
}
```

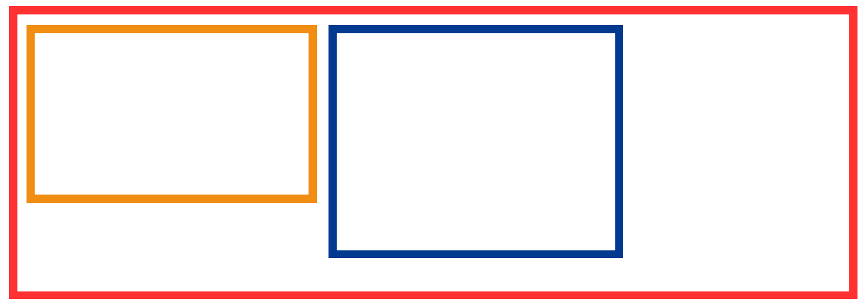


CSS Flexbox

Flex Container Properties

4. align-items : Aligns items along the cross axis (perpendicular to the flex-direction).

```
.container{
    display: flex;
    align-items: flex-start;  /* flex-start | flex-end | center | column-reverse | baseline | stretch  (Default) */
}
```

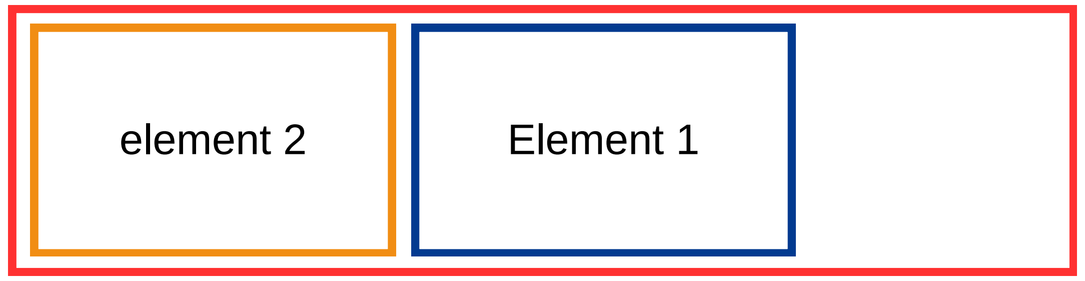


CSS Flexbox

Flex Item Properties

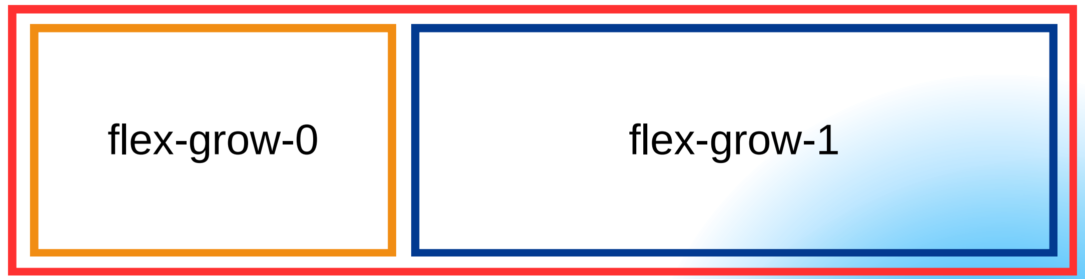
1. order : Defines the order in which items appear in the flex container.

```
.flex-item {
  order: 1; /* Default is 0 */
}
```



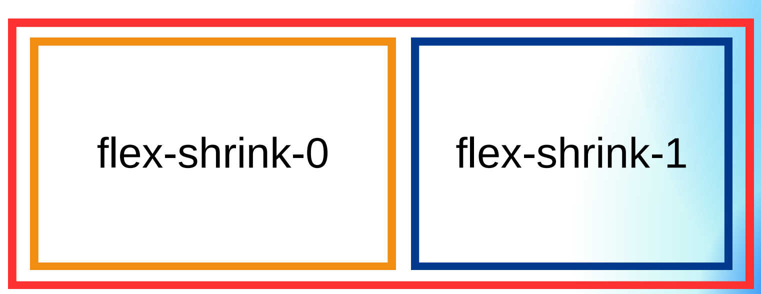
2. flex-grow : Defines if an item should grow relative to others.

```
.flex-item {
  flex-grow: 1; /* Default is 0 */
}
```



3. flex-shrink : Defines if an item should shrink relative to others.

```
.flex-item {
  flex-shrink: 1; /* Default is 1 */
}
```

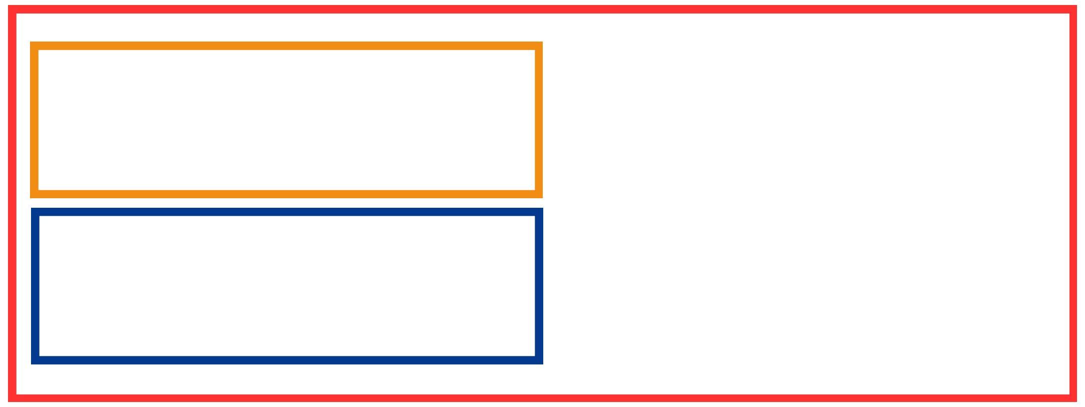


CSS Flexbox

Flex Item Properties

4. flex-basis : Defines the minimum size of an item before remaining space is distributed.

```
.flex-item{
  width: 40%;
  flex-basis: 230px;
}
```

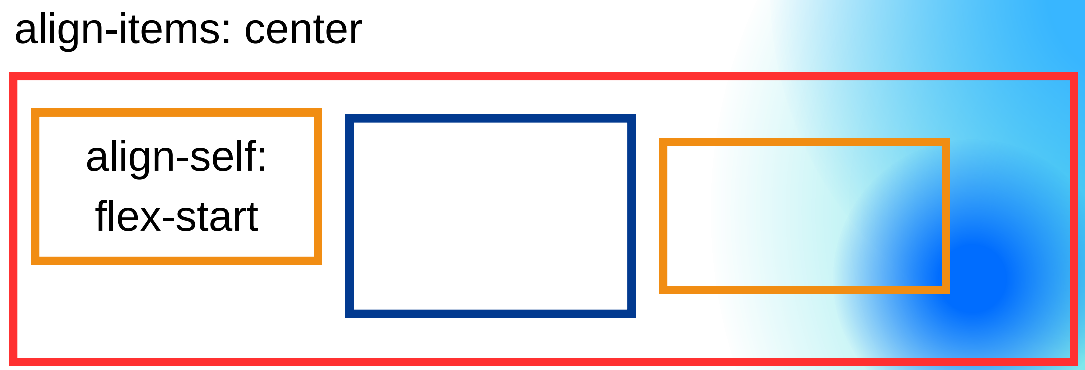


5. flex: It's a shorthand for flex-grow, flex-shrink and flex-basis

```
.flex-item {
  flex: 0 1 auto ; /* Equivalent to flex-grow: 0; flex-shrink: 1; flex-basis: auto; */
}
```

6. align-self : It allows individual items to override align-items property for itself

```
.flex-item {
  align-self: auto; /* Options: auto (Default) | flex-start | flex-end | center | baseline | stretch */
}
```



indixpert

THANK YOU

PHONE NUMBER

(+91) 778 899 2897

WEBSITE

www.indixpert.com