Tutorials ▾ Exercises ▾ Services ▾ Sign Up Log in

CSS JAVASCRIPT SQL PYTHON JAVA PHP HOW TO W3.CSS

CSS Flex Container

❮ Previous Next ❯

The CSS Flex Container

Like we specified in the previous chapter, this is a **flex container** (the blue area) with three **flex items**:

1 2 3

The flex container becomes flexible by setting the `display` property to `flex`:

Example

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

Try it Yourself »

The CSS properties we use for the flex container are:

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

The CSS flex-direction Property

The `flex-direction` property specifies the display-direction of flex items in the flex container.

The `flex-direction` property can have one of the following values:

- `row`
- `column`
- `row-reverse`
- `column-reverse`

Example

The `row` value is the default value, and it displays the flex items horizontally (from left to right):

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

Result:

1 2 3

Try it Yourself »

Example

The `column` value displays the flex items vertically (from top to bottom):

```
}
```

Result:

1

2

3

Try it Yourself »

The `row-reverse` value displays the flex items horizontally (but from right to left):

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

Result:

3 2 1

Try it Yourself »

Example

The `column-reverse` value displays the flex items vertically (but from bottom to top):

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

Result:

3

2

Try it Yourself »

The CSS flex-wrap Property

The `flex-wrap` property specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line.

The `flex-wrap` property can have one of the following values:

- `nowrap`
- `wrap`
- `wrap-reverse`

Example

The `nowrap` value specifies that the flex items will not wrap (this is default):

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

Result:

1 2 3 4 5 6 7 8 9

Try it Yourself »

The `wrap` value specifies that the flex items will wrap if necessary:

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

Result:

1 2 3 4 5

6 7 8 9

Try it Yourself »

Example

The `wrap-reverse` value specifies that the flex items will wrap if necessary, in reverse order:

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

Result:

6 7 8 9

Try it Yourself »

The CSS flex-flow Property

The `flex-flow` property is a shorthand property for setting both the `flex-direction` and `flex-wrap` properties.

Example

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

Try it Yourself »

The CSS justify-content Property

The `justify-content` property is used to align the flex items when they do not use all available space on the main-axis (horizontally).

The `justify-content` property can have one of the following values:

- `center`
- `flex-start`
- `flex-end`
- `space-around`
- `space-between`
- `space-evenly`

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

Result:

1 2 3

Try it Yourself »

Example

The `flex-start` value positions the flex items at the beginning of the container (this is default):

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

Result:

1 2 3

Try it Yourself »

The flex-end value positions the flex items at the end of the container:

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

Result:

1 2 3

Try it Yourself »

Example

The space-around value displays the flex items with space around them:

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

Result:

1 2 3

Try it Yourself »

The `space-between` value displays the flex items with space between them:

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

Result:

1 2 3

Try it Yourself »

Example

The `space-evenly` value displays the flex items with equal space around them:

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

Result:

1 2 3

Try it Yourself »

The align-items property is used to align the flex items when they do not use all available space on the cross-axis (vertically).

The align-items property can have one of the following values:

- center
- flex-start
- flex-end
- stretch
- baseline
- normal

In the following examples we use a 200 pixels high container, to better demonstrate the align-items property.

Example

The center value positions the flex items in the middle of the container:

```
.flex-container {
  display: flex;
  height: 200px;
  align-items: center;
}
```

Result:

1 2 3

Try it Yourself »

The `flex-start` value positions the flex items at the top of the container:

```
.flex-container {
  display: flex;
  height: 200px;
  align-items: flex-start;
}
```

Result:

1 2 3

Try it Yourself »

Example

The `flex-end` value positions the flex items at the bottom of the container:

```
.flex-container {
  display: flex;
  height: 200px;
  align-items: flex-end;
}
```

Result:

Try it Yourself »

Example

The `stretch` value stretches the flex items to fill the container (this is equal to "normal" which is default):

```
.flex-container {
  display: flex;
  height: 200px;
  align-items: stretch;
}
```

Result:

1 2 3

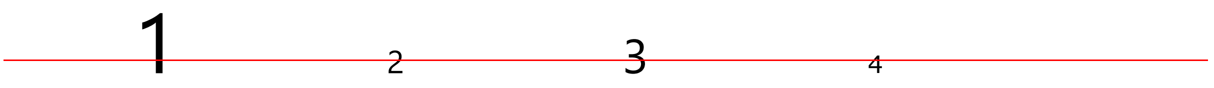
Try it Yourself »

Example

The `baseline` value positions the flex items at the baseline of the container:

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

Note: The example uses different font-size to demonstrate that the items gets aligned by the text baseline:



Try it Yourself »

The CSS align-content Property

The `align-content` property is used to align the flex lines.

The `align-content` property is similar to `align-items` , but instead of aligning flex items, it aligns the flex lines.

The `align-content` property can have one of the following values:

- `center`
- `stretch`
- `flex-start`
- `flex-end`
- `space-around`
- `space-between`
- `space-evenly`

In the following examples we use a 600 pixels high container, with the `flex-wrap` property set to `wrap` , to better demonstrate the `align-content` property.

Example

With `center` , the flex lines are packed toward the center of the container:

```
.flex-container {
}
```

Result:

1 2 3 4 5

6 7 8 9

Try it Yourself »

Example

With `stretch`, the flex lines stretch to take up the remaining space of the container (this is default):

```
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: stretch;
}
```

Result:

6 7 8 9

Try it Yourself »

Example

With `flex-start`, the flex lines are packed toward the start of the container:

```css
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: flex-start;
}
```

Result:

6 7 8 9

Try it Yourself »

Example

With `flex-end`, the flex lines are packed toward the end of the container:

```
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: flex-end;
}
```

Result:

1 2 3 4 5

6 7 8 9

Try it Yourself »

Example

With `space-between`, the space between the flex lines are equal, but the first item is flush with the start edge of the container, and the last item is flush with the end edge of the container:

```css
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: space-between;
}
```

Result:

6 7 8 9

Try it Yourself »

Example

With `space-around`, the space between the flex lines are equal, but the space before the first item and after the last item is set to half of the space between the flex lines:

```
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: space-around;
}
```

Result:

1 2 3 4 5

6 7 8 9

Try it Yourself »

Example

With `space-evenly` , the flex lines are evenly distributed in the flex container, with equal space on top, bottom and between:

```
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: space-evenly;
}
```

Result:

1 2 3 4 5

6 7 8 9

Try it Yourself »

Perfect Centering

In the following example we will solve a common style problem: perfect centering.

SOLUTION: Set both the `justify-content` and `align-items` properties to `center`, and the flex item will be perfectly centered:

```
.flex-container {
  display: flex;
  height: 300px;
  justify-content: center;
  align-items: center;
}
```

Try it Yourself »

The CSS Flex Container Properties

The following table lists all the CSS Flex Container properties:

Property	Description
align-content	Modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines
align-items	Vertically aligns the flex items when the items do not use all available space on the cross-axis
display	Specifies the display behavior (the type of rendering box) for an element
flex-direction	Specifies the direction of the flex items inside a flex container
flex-flow	A shorthand property for flex-direction and flex-wrap
flex-wrap	Specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line
justify-content	Horizontally aligns the flex items when the items do not use all available space on the main-axis

Tutorials ▾ Exercises ▾ Services ▾ Sign Up Log in

CSS JAVASCRIPT SQL PYTHON JAVA PHP HOW TO W3.CSS

COLOR PICKER

PLUS SPACES GET CERTIFIED

Top Tutorials

Tutorials ▾ Exercises ▾ Services ▾ Sign Up Log in

CSS JAVASCRIPT SQL PYTHON JAVA PHP HOW TO W3.CSS

W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
Java Tutorial
C++ Tutorial
jQuery Tutorial

Top References

HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference
HTML Colors
Java Reference
Angular Reference
jQuery Reference

Top Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
Java Examples
XML Examples
jQuery Examples

Get Certified

HTML Certificate
CSS Certificate
JavaScript Certificate
Front End Certificate
SQL Certificate
Python Certificate
PHP Certificate
jQuery Certificate
Java Certificate
C++ Certificate
C# Certificate
XML Certificate

FORUM ABOUT ACADEMY

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning.
Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness
of all content. While using W3Schools, you agree to have read and accepted our terms of use, cookie and privacy policy.