5.1 Flexbox

Flexbox container and items

The ***Flexible Box*** or ***flexbox*** is a CSS layout mode that provides an efficient way to lay out elements in a container so the elements behave predictably when the container is resized or viewed on different screen sizes.

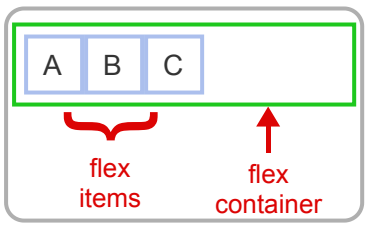
A ***flex container*** is an element that has the CSS property `display` set to `flex` to create a block-level flex container or `inline-flex` to create an inline flex container. Ex: `<div style="display: flex">`. Flex containers hold flex items. A ***flex item*** is a child element of a flex container that is positioned and sized according to various CSS flexbox properties.

PARTICIPATION ACTIVITY 5.1.1: Flexbox example renders three div elements on the same row.

1 2 3 2x speed

```
<div id="container">
   <div>A</div>
   <div>B</div>
   <div>C</div>
</div>
```

```
/* flex container */
#container {
   display: flex;
   border: 1px green solid;
   padding: 5px;
}

/* flex items */
#container > div {
   padding: 10px;
   border: 1px blue solid;
}
```



The flex items have padding and blue borders.

Captions

1. Without any CSS, the A, B, and C div elements display vertically, each filling the browser width.
2. Setting the CSS display property to "flex" makes the outer div the flex container. The flex items now display on the same row.
3. The flex items have padding and blue borders.

Feedback?

PARTICIPATION ACTIVITY 5.1.2: Flexbox container and items.

Refer to the animation above.

1) The flex container has _____ flex item(s).
   - 1
   - 2
   - 3 (selected)

Correct

The flex container has three `<div>` children, which are called "flex items". Any number of flex items may appear in a flex container.

2) Removing _____ from the `#container` rule displays all flex items vertically on separate rows.
   - `display: flex;` (selected)
   - `border: 1px green solid;`
   - `padding: 5px;`

Correct

Removing the `display` property makes the flex items fill the flex container on separate rows. Only when using `display: flex;` does the flex container's items appear side-by-side on the same row.

3) Flex items appear _____ within a flex container by default.
   - left aligned (selected)
   - centered
   - right-aligned

Correct

The three flex items appear on the left side of the flex container. Other CSS properties can align the items to the right or center.

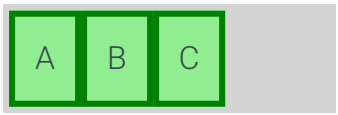
Feedback?

Flex container properties

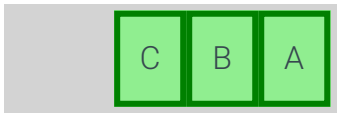
Several CSS properties modify the default behavior of a flex container:

- The ***flex-direction*** property defines the direction of flex items within the container using values:

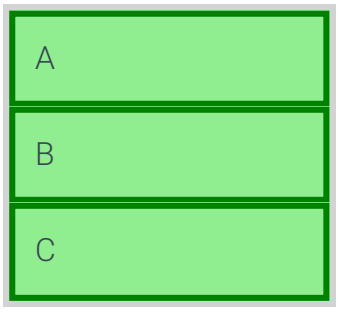
`row` (default)



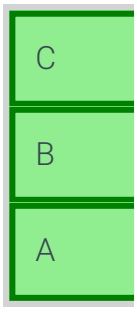
`row-reverse`



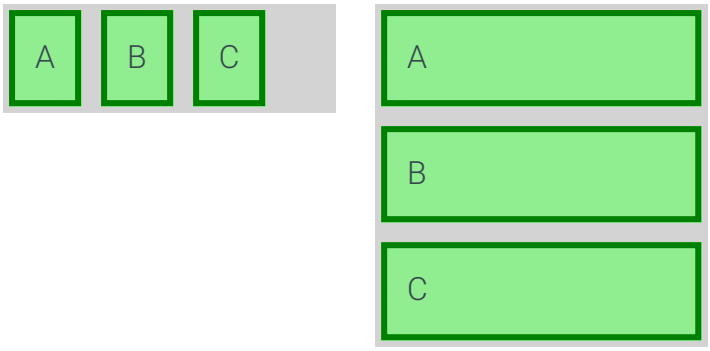
`column`



`column-`



- The ***gap*** property defines the space between flex items. Ex: `gap: 10px;` puts a 10px gap between all items.



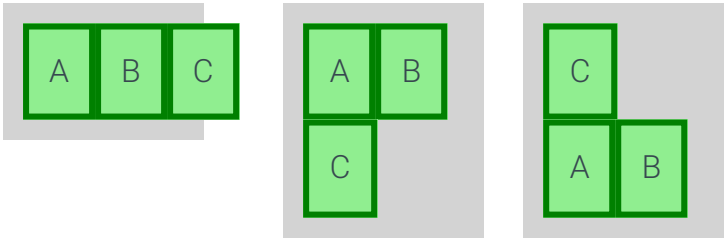
- The ***justify-content*** property justifies the flex items within the container using values:

`flex-start` (default) `flex-end` `center`

`space-between` `space-around` `space-evenly`

- The ***flex-wrap*** property determines if or how flex items wrap onto multiple rows when the container is not wide enough to hold all items, using values:

`nowrap` (default) `wrap` `wrap-reverse`



PARTICIPATION ACTIVITY 5.1.3: Flex container properties for photos.

The webpage below uses a flexbox to display three photos with captions.

Make the following CSS modifications to the flex container. After each modification, render the webpage to verify the modification works properly.

1. Add a `gap` property to the flex container to put a 20px gap between the photos.
2. Add a `flex-direction` property to the flex container to display the photos in reverse.

3. Add a `justify-content` property to the flex container to display the photos centered in the browser window.
4. Add a `flex-wrap` property to the flex container so the photos wrap to the next row when the browser width is reduced.

Note: The window that renders the webpage can be resized using the small handle in the bottom-right corner.

HTML CSS

```
<div id="container">
   <div class="photo">
      <img src="https://resources.zybooks.com/WebProgramming/d
      <p>
         Ducati motorcycle
      </p>
   </div>
   <div class="photo">
      <img src="https://resources.zybooks.com/WebProgramming/k
      <p>
         Japanese restaurant
      </p>
   </div>
   <div class="photo">
      <img src="https://resources.zybooks.com/WebProgramming/l
```

Render webpage Reset code

Your webpage

Ducati motorcycle

Japanese restaurant

Lake in Col

▶ View solution

Feedback?

PARTICIPATION ACTIVITY 5.1.4: Flex container properties.

Refer to the HTML below. Assume a CSS rule exists that makes the `<div>` a flex container.

```
<div id="container">
   <img src="madrid.jpg" alt="Madrid">
   <img src="hanoi.jpg" alt="Hanoi">
   <img src="islamabad.jpg" alt="Islamabad">
</div>
```

1) What `flex-direction` value displays Madrid on the right side of the grid container with all three images on the same row?

- `row`
- `row-reverse` (selected)
- `column-reverse`

Correct

`row-reverse` places the first grid item on the right side, the second item to the left, and so on, all on the same row.

2) What `justify-content` value puts an equal amount of space on either side of the three images but leaves no space between the images?

- `center` (selected)
- `space-between`
- `space-around`

Correct

`center` displays all three images centered in the container with no space in between.

3) What `flex-wrap` value makes the Islamabad image appear below Madrid if the grid container is not wide enough to show all three images on the same row?

- `nowrap`
- `wrap` (selected)
- `wrap-reverse`

Correct

`wrap` makes the third image (Islamabad) wrap under the first image (Madrid) when the grid container is not wide enough.

Feedback?

Flex item properties

A flex item's width is determined by the combination of three CSS properties:

- The ***flex-basis*** property sets the initial length of a flex item. The values can be `auto` (the default), a percentage, or a length unit. The default value `auto` makes the flex item the same initial length as the content.
- The ***flex-grow*** property sets a proportion that determines how much of the available container space should be assigned to the item. The default is 0, meaning the size should be based on the item's content.
- The ***flex-shrink*** property sets a proportion that determines the item's minimum size. The default is 1, meaning the size should shrink at the same rate as other items when the container width shrinks. A value of 0 means the item should not change sizes when the container width shrinks.

PARTICIPATION ACTIVITY 5.1.5: Changing flex item properties.

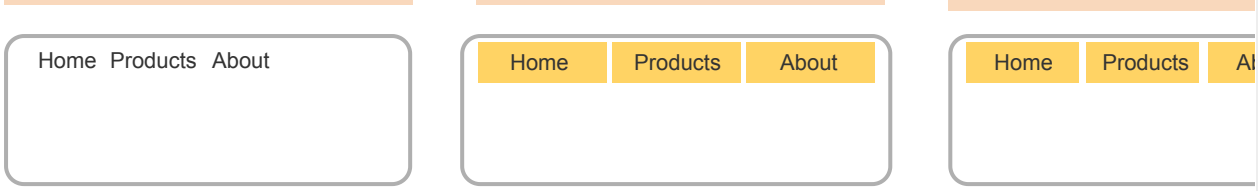
1 2 3 4 5 6 7 2x speed

```
<nav>
  <ul>
    <li><a href="index.html">Home</a></li>
    <li><a href="products.html">Products</a></li>
    <li><a href="about.html">About</a></li>
  </ul>
</nav>
```

```
nav ul {
  display: flex;
  list-style-type: none;
  padding: 0;
}
```

```
nav li {
  flex-grow: 1;
  background-color: gold;
  text-align: center;
}
```

```
nav li {
  flex-basis: 100px;
  flex-shrink: 0;
  background-color:
  text-align: center
}
```



Changing flex-shrink from the default 1 to 0 prevents the li elements from shrinking when the b resized.

Captions

1. A website's navigation links are displayed in an unordered list.
2. Making the ul element a flex container places the nav links on the same row.
3. By default, the li elements have flex-basis:auto and flex-grow:0, so li elements are only as wide as the item's content.
4. Changing flex-grow from the default 0 to 1 gives all li elements the same proportion. The elements fill the flex container.
5. Replacing "flex-grow:1" with "flex-basis:100px" makes each li element 100px wide.
6. Resizing the browser changes the container size. When the container shrinks, the li elements shrink to fill the available space.
7. Changing flex-shrink from the default 1 to 0 prevents the li elements from shrinking when the browser is resized.

Feedback?

PARTICIPATION ACTIVITY 5.1.6: Flex item properties.

Refer to the webpage below.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>ACME Widgets</title>
   <style>
      body {
         font-family: Arial;
      }

      header {
         display: flex;
         justify-content: space-between;
      }

      header h1 {
         margin: 6px;
      }

      nav ul {
         display: flex;
         list-style-type: none;
         padding: 0;
         margin: 0;
         width: 500px;
      }

      nav li {
         background-color: gold;
         margin: 5px;
         padding: 10px;
         text-align: center;
      }

      nav a {
         color: black;
         text-decoration: none;
      }
   </style>
</head>
<body>
   <header>
      <h1>ACME Widgets</h1>
      <nav>
         <ul>
            <li><a href="index.html">Home</a></li>
            <li><a href="products.html">Products</a></li>
            <li><a href="services.html">Services</a></li>
            <li><a href="about.html">About</a></li>
         </ul>
      </nav>
   </header>
</body>
</html>
```

1) How many flex containers exist in the webpage?

- 0
- 1
- 2 (selected)

Correct

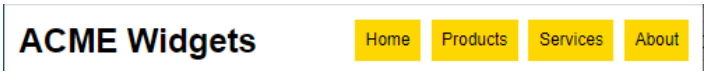
The CSS uses `display:flex` for the header and ul elements.

2) The heading "ACME Widgets" appears _____ the navigation links in the browser.

- to the left of (selected)
- to the right of
- above

Correct

The header element is a flex container that places all flex items (the heading and the nav links) on the same row. By default, the items appear on the row in the same order the items appear in the HTML.



3) What is the `flex-basis` for the `<nav>` flex item?

- `auto` (selected)
- 0
- 1

Correct

Since no `flex-basis` is specified in the CSS, the nav element uses the default `flex-basis` value `auto`.

4) How wide is the `<nav>` flex item?

- Just wide enough to hold all the navigation links (selected)
- Half the width of the webpage
- Entire width of the webpage

Correct

Since the nav element uses the default `flex-basis` value `auto`, the flex item is just wide enough to hold the item's contents (the nav links).

5) In addition to the CSS below, ____ must be added to the `nav li` rule so the li elements fill half the row.

```
nav {
   flex-basis:
50%;
}
```

- `flex-grow: 0;`
- `flex-grow: 1;` ✓
- `flex-grow: 50%;`

Correct

Giving all li elements the same `flex-grow` value makes all elements grow to fill the flex container.

6) Adding ____ to the `header h1` rule prevents the heading from shrinking when the browser is resized.

- `flex-shrink: none;`
- `flex-shrink: 1;`
- `flex-shrink: 0;` ✓

Correct

0 stops the flex item from shrinking. "ACME" and "Widgets" always remain on the same line, even when the browser width narrows the size of the flex container.

Feedback?

The flex property

The shorthand property ***flex*** specifies `flex-grow`, `flex-shrink`, and `flex-basis` together. Ex: `flex: 0 1 auto;` is the same as `flex-grow: 0; flex-shrink: 1; flex-basis: auto;`.

PARTICIPATION ACTIVITY 5.1.7: Flexbox layout using the flex property.

1 2 3 4 2x speed

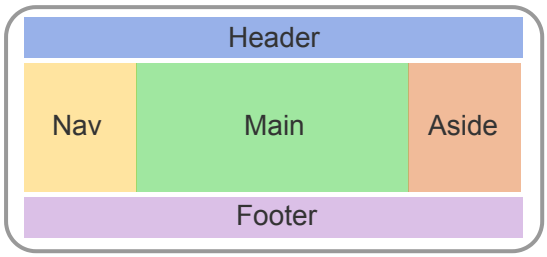
```
<body>
   <header>Header</header>

   <!-- Flexbox layout -->
```

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

```
    <div id="container">
      <nav>Nav</nav>
      <main>Main</main>
      <aside>Aside</aside>
    </div>

    <footer>Footer</footer>
</body>
```

```
nav {
    flex: 0 1 20%;
}

main {
    flex: 0 1 60%;
}

aside {
    flex: 0 1 20%;
}
```

flex-grow flex-basis flex-shrink



<nav> occupies 20% of the row, <main> occupies 60%, and <aside> occupies 20%. 20% + 60% + 20% = 100% of the row.

Captions

1. <header> and <footer> span the entire width of <body>, but the is a flex container that displays the flex items on the same row.
2. <nav>, <main>, and <aside> all have flex-grow = 0, so all three flex items' width should be based on each item's content.
3. <nav>, <main>, and <aside> all have flex-shrink = 1, so all three flex items shrink at the same rate when the browser is resized.
4. <nav> occupies 20% of the row, <main> occupies 60%, and <aside> occupies 20%. 20% + 60% + 20% = 100% of the row.

Feedback?

PARTICIPATION ACTIVITY 5.1.8: Flex item properties.

Refer to the animation above.

1) Which property makes the Main flex item wider than Nav and Aside?

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

Correct

Main is wider because the `flex-basis` value of 60% is larger than nav and aside's `flex-basis` value of 20%.

2) How do the changes below affect the webpage?

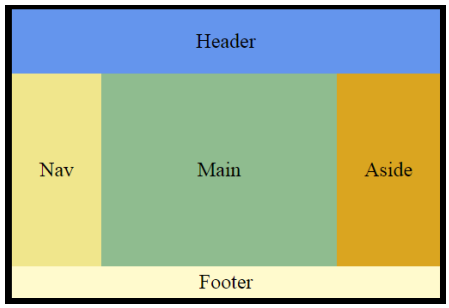
Correct

Since Main's `flex-grow` = 3.5 and Nav and Aside's `flex-grow` = 1, Main is 3.5 times the width of Nav and Aside. 1 + 3.5 + 1 = 5.5 total units exist, so the width of

```
nav   { flex: 1
1 auto; }
main  { flex: 3.5
1 auto; }
aside { flex: 1
1 auto; }
```

- All three flex items are equal width.
- Nav and Aside are wider than Main.
- The layout will be nearly the same as before.

each unit is 100% / 5.5 = 18.2%. Nav and Aside are therefore 18.2% wide, and Main is 18.2% × 3.5 = 63.7% wide. The original `flex-grow` values are Main = 60%, Nav and Aside = 20%, which are nearly the same as the calculated percentages.



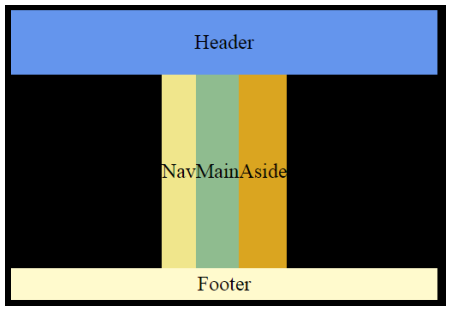
3) How do the changes below affect the webpage?

```
#container {
display: flex;

flex-direction:
row;

justify-content:
center; }
nav   { flex: 0 1
auto; }
main  { flex: 0 1
auto; }
aside { flex: 0 1
auto; }
```

- All three flex items are default width and centered horizontally in the flex container.
- Nav and aside are wider than Main.
- All three flex items will be default width, aligned to the left.

Correct

The "auto" `flex-basis` value gives all three flex items default width since the `flex-grow` values are equal (0) for all three items. `justify-content` centers the flex items.



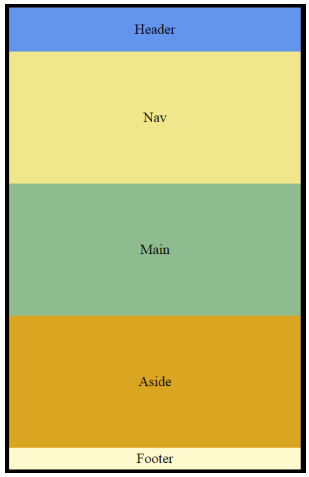
4) How does the change below affect the webpage?

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

- ○ No changes.
- ◉ All three flex items are stacked on top of one another.
- ○ The direction of the flex items is reversed.

Correct

The container lays out the flex items in a single column, an ideal layout for mobile devices.



Feedback?

PARTICIPATION ACTIVITY 5.1.9: Flexbox practice.

The webpage displays three years of Grammy Award nominations for Album of the Year. Each year's nomination is placed in a `<section>` element.

Alter the webpage to display the three sections in a single row:

1. Add a single `<div>` element that surrounds all three sections.
2. Add the proper CSS so the `<div>` becomes a flex container and displays each section on a single row.
3. Add a `flex` property to the `section` rule that sets flex-grow to 0, flex-shrink to 1, and flex-basis to 250px.
4. Add a `flex-wrap` property that makes the sections wrap to the next row when the browser is resized.

After adding the HTML and CSS above, render the webpage. Resize the rendered webpage's window and verify that the sections wrap to the next row when the window isn't wide enough to display the sections side-by-side.

HTML CSS

```
      <li><cite>Cuz I Love You (Deluxe)</cite> by Lizzo</li>
      <li><cite>Father Of The Bride</cite> by Vampire Weekend
      <li><cite>I,I</cite> by Bon Iver</li>
      <li><cite>Norman Rockwell!</cite> by Lana Del Rey</li>
   </ul>
</section>
<section>
   <h2>2018</h2>
   <ul>
      <li><cite>Golden Hour</cite> by Kacey Musgraves</li>
      <li><cite>Invasion Of Privacy</cite> by Cardi B</li>
      <li><cite>By The Way, I Forgive You</cite> by Brandi Car
      <li><cite>Scorpion</cite> by DrakeX</li>
      <li><cite>H.E.R.</cite> by H.E.R.</li>
      <li><cite>Beerbongs & Bentleys</cite> by Post Malone</li
      <li><cite>Dirty Computer</cite> by Janelle Monáe</li>
```

Render webpage Reset code

Your webpage

Grammy Nominees for Album of the Year

2020

- *Folklore* by Taylor Swift
- *Future Nostalgia* by Dua Lipa
- *Hollywood's Bleeding* by Post Malone
- *Chilombo* by Jhené Aiko
- *Black Pumas (Deluxe Edition)* by Black Pumas
- *Everyday Life* by Coldplay
- *Djesse Vol.3* by Jacob Collier
- *Women In Music Pt. III* by HAIM

2019

▶ View solution

Feedback?

CHALLENGE ACTIVITY 5.1.1: Flexbox.

530096.4000608.qx3zqy7

Jump to level 1

Using flex-basis, set element with class first to 30%, class second to be 20%, and class third to be 50%. **SHOW EXPECTED**

CSS HTML

```
.second {

}
.third {

}

.container {
   display: flex;
   width: 120px;
}
.container > div {
   background-color: lightblue;
   border: 2px solid blue;
   height: 25px;
}
```

1 2 3

Check Next

View your last submission

```
.first {
   flex-basis: 50%;
}
.second {
   flex-basis: 20%;
}
.third {
   flex-basis: 30%;
}
```

Feedback?

Exploring further:

- HTML Layouts from W3Schools
- CSS Flexbox from W3Schools

How was this section?

Provide section feedback