

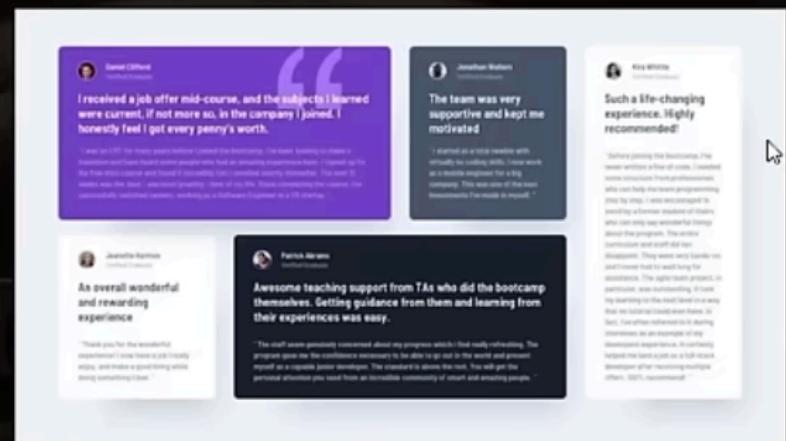
Core Lessons

1. What is Flexbox?
2. What Flexbox can do for us
 - a. Simplify layouts
 - b. Reduce need for media queries
(responsive design)
 - c. Simple example w/pricing card from prior tutorial
3. Basic Flexbox Concepts
 - a. Flex container vs. Flex Items
 - b. Main axis vs. Cross Axis
 - c. Aligning and spacing flex items
 - d. Flex wrap (helps w/responsive design)
 - e. Flex sizing (grow, shrink, basis)

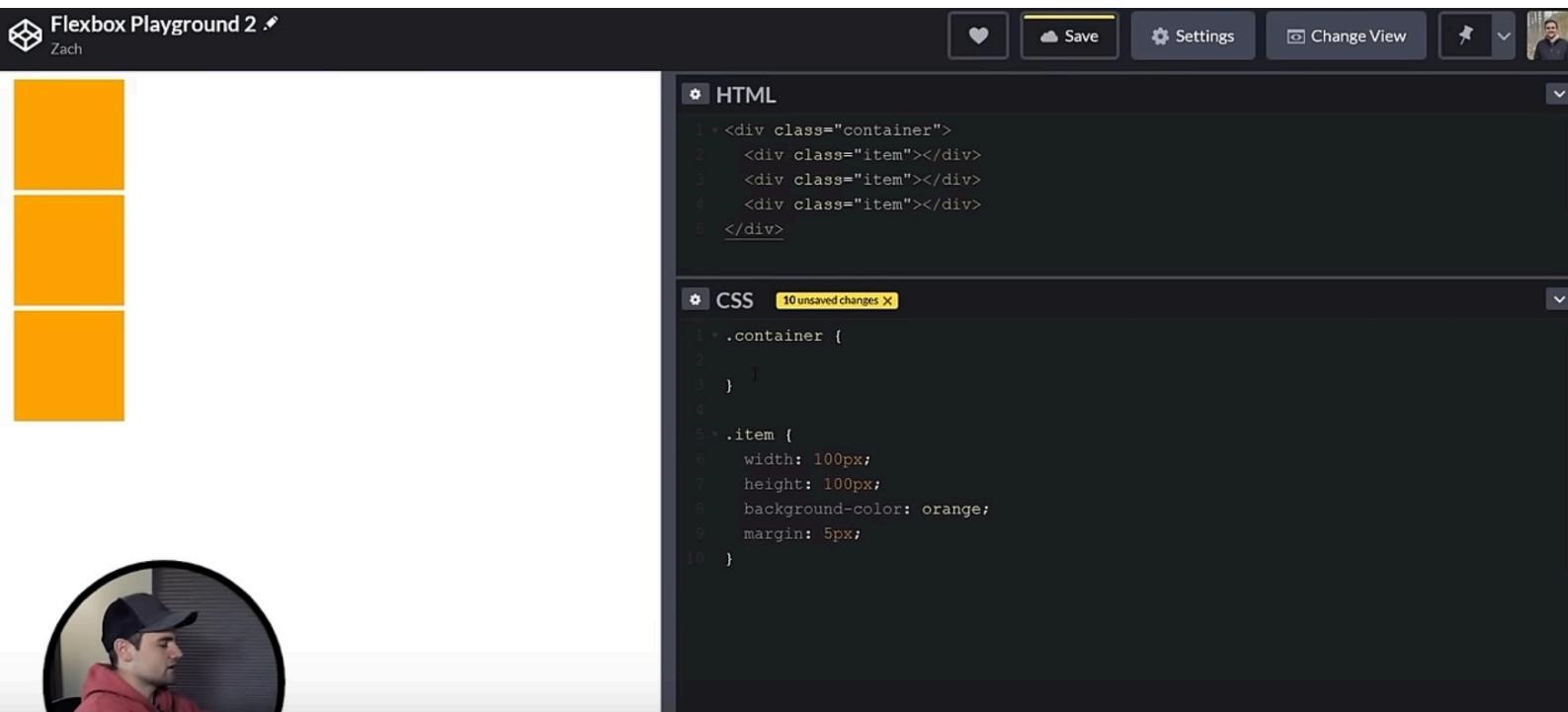
Frontend Mentor Project! (testimonial page)

We will NOT cover...

1. Shorthand Flexbox properties
2. Advanced Flexbox properties



Remember setting a container to display: flex,
only makes its direct children flex items !!!



The screenshot shows the Flexbox Playground 2 interface. On the left, there is a preview area displaying three orange rectangular boxes stacked vertically. To the right of the preview are two code editors: an 'HTML' editor and a 'CSS' editor. The 'HTML' editor contains the following code:

```
<div class="container">
  <div class="item"></div>
  <div class="item"></div>
  <div class="item"></div>
</div>
```

The 'CSS' editor contains the following code:

```
.container {
}
.item {
  width: 100px;
  height: 100px;
  background-color: orange;
  margin: 5px;
}
```

At the top of the playground interface, there are several buttons: a heart icon, a save icon, settings icon, change view icon, and a user profile icon.

The screenshot shows a web-based development environment for learning Flexbox. At the top, there's a navigation bar with icons for heart, save, settings, change view, and user profile. Below the bar, the title "Flexbox Playground 2" and the author "Zach" are displayed. On the left, there's a video player showing a man in a cap and red hoodie. The main area is divided into two tabs: "HTML" and "CSS".

HTML

```
<div class="container">
  <div class="item"></div>
  <div class="item"></div>
  <div class="item"></div>
</div>
```

CSS 12 unsaved changes ×

```
.container {
  display: flex;
}

.item {
  width: 100px;
  height: 100px;
  background-color: orange;
  margin: 5px;
}
```

 Flexbox Playground 2 

Zach



HTML 25 unsaved changes X

```
1 <div class="container">
2   <div class="item"></div>
3   <div class="item"></div>
4   <div class="item">
5     <div class="sub-item"></div>
6     <div class="sub-item"></div>
7     <div class="sub-item"></div>
8   </div>
```

CSS

```
1 .container {
2   display: flex;
3 }
4
5 .item {
6   width: 100px;
7   height: 100px;
8   background-color: orange;
9   margin: 5px;
10 }
11
12 .sub-item {
13   width: 15px;
14   height: 15px;
```



 Flexbox Playground 2

Zach





HTML

```
1 <div class="container">
2   <div class="item"></div>
3   <div class="item"></div>
4   <div class="item item-3">
5     <div class="sub-item"></div>
6     <div class="sub-item"></div>
7     <div class="sub-item"></div>
8   </div>
```

CSS 28 unsaved changes ×

```
0 * .item {
1   width: 100px;
2   height: 100px;
3   background-color: orange;
4   margin: 5px;
5 }
6
7 * .item-3 {
8   display: flex;
9 }
10
11 * .sub-item {
12   width: 15px;
13   height: 15px;
14   background-color: blue;
15 }
```

Flex Container Properties

1. display
2. flex-direction
3. justify-content
4. align-items
5. flex-wrap
6. align-content

Flex Item Properties

1. align-self
2. order
3. flex-grow
4. flex-shrink
5. flex-basis



It's useful to look at the formal definition table which shows the default values some properties are set to.

more respecting main and right containers.

safe

Used alongside an alignment keyword. If the chosen keyword means that the item overflows the alignment container causing data loss, the item is instead aligned as if the alignment mode were `start`.

unsafe

Used alongside an alignment keyword. Regardless of the relative sizes of the item and alignment container and whether overflow which causes data loss might happen, the given alignment value is honored.

Formal definition

<u>Initial value</u>	normal
<u>Applies to</u>	all elements
<u>Inherited</u>	no
<u>Computed value</u>	as specified
<u>Animation type</u>	discrete

Formal syntax

`normal | stretch | baseline-position | flex-grow-position | self-position | none | initial | inherit`



Save

Settings

HTML

```
1 * <div class="container">
2     <div class="item"></div>
3     <div class="item"></div>
4 *     <div class="item item-3">
5         <div class="sub-item"></div>
6         <div class="sub-item"></div>
7         <div class="sub-item"></div>
```

CSS

```
5 * .item {
6     width: 100px;
7     height: 100px;
8     background-color: orange;
9     margin: 5px;
10    flex: 0 1 auto;
11    flex-grow: 0;
12    flex-shrink: 1;
13    flex-basis: auto;
14 }
15
16 * .item-3 {
17     display: flex;
18 }
19
20 * .sub-item {
```

Default values for flex container:

The screenshot shows a code editor interface with a dark theme. At the top right are three buttons: a heart icon, a 'Save' icon, and a 'Settings' gear icon. Below these are two tabs: 'HTML' and 'CSS'. The 'HTML' tab contains the following code:

```
1 <div class="container">
2   <div class="item"></div>
3   <div class="item"></div>
```

The 'CSS' tab contains the following code, which defines the default properties for a flex container:

```
1 .container {
2   display: flex;
3   flex-direction: row;
4   justify-content: normal; /* stretch */
5   align-content: normal;
6   align-items: normal; /* stretch */
7   flex-wrap: nowrap;
8 }
9
10 .item {
11   width: 100px;
12   height: 100px;
13   background-color: orange;
14   margin: 5px;
15 }
16
17 .item-3 {
18   display: flex;
19 }
```

Default values for flex item:

The screenshot shows a code editor interface with two tabs: 'HTML' and 'CSS'. The 'HTML' tab contains the following code:

```
<div class="item"></div>
<div class="item"></div>
```

The 'CSS' tab contains the following code, with line numbers from 1 to 25:

```
1   align-content: normal;
2   align-items: normal; /* stretch */
3   flex-wrap: nowrap;
4 }
5
6 * .item {
7   width: 100px;
8   height: 100px;
9   background-color: orange;
10  margin: 5px;
11  align-self: auto;
12  flex-grow: 0;
13  flex-shrink: 1;
14  flex-basis: auto;
15 }
16
17 * .item-3 {
18   display: flex;
19 }
20
21 * .sub-item {
```

The CSS code highlights several properties: align-content, align-items, flex-wrap, .item selector, align-self, flex-grow, flex-shrink, flex-basis, .item-3 selector, and display. The flex-basis property is highlighted with a light gray box.

Row vs. Column

HTML

```
1 * <div class="top-section">
2 *   <div class="current-setting">
3 *     <p><span id="setting">flex-direction: row</span></p>
4 *   </div>
5 *   <button onclick="setFlex()">Toggle flex-direction</button>
6 * </div>
7 *
8 * <div class="container">
9 *   <span id="horizontal-axis">Main Axis</span>
10 *  <span id="vertical-axis">Cross Axis</span>
11 *  <div id="fi1">
12 *    <p>1</p>
13 *  </div>
14 *  <div id="fi2">
15 *    <p>2</p>
16 *  </div>
17 *  <div id="fi3">
```

HTML

```
1 <button onclick="setflex()>Toggle Flex-direction</button>
2 </div>
3
4 <div class="container">
5   <span id="horizontal-axis">Main Axis</span>
6   <span id="vertical-axis">Cross Axis</span>
7   <div id="fi1">
8     <p>1</p>
9   </div>
10  <div id="fi2">
11    <p>2</p>
12  </div>
13  <div id="fi3">
14    <p>3</p>
15  </div>
16 </div>
```

flex-direction: row align flex items one next to each other and sets the horizontal axis as the main axis and the vertical axis as the cross axis.

The screenshot shows a web-based code editor interface. At the top, there's a navigation bar with icons for file, save, and settings. Below the title "Flex Direction" and author "Zach", there's a button labeled "Toggle flex-direction". A preview area shows a container with three items labeled 1, 2, and 3, with "Main Axis" text above item 1 and "Cross Axis" text to its left. To the right, the code editor displays the following HTML and CSS:

```
HTML
</div>
<div class="container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
</div>

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

#fi1 {
}

#fi2 {
}

#fi3 {
```

flex-direction: column align flex items one on top of each other and sets the vertical axis as the main axis and the horizontal axis as the cross axis.

The screenshot shows a web-based code editor interface. At the top, there's a header with a logo, the title "Flex Direction", a user name "Zach", and buttons for "Save" and "Settings". The main area has two tabs: "HTML" and "CSS", with "HTML" currently selected. The HTML code is:

```
</div>
<div class="container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>

```

The CSS code is:

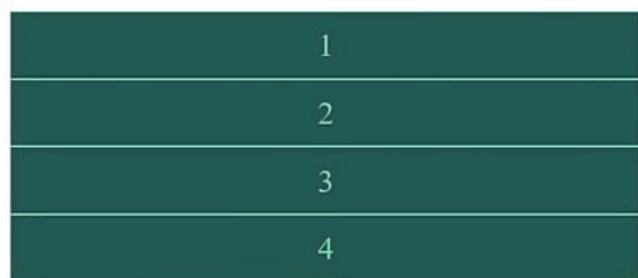
```
.container {
  display: flex;
  flex-direction: row;
}

#fi1 {
}

#fi2 {
}

#fi3 {
}
```

On the left side of the editor, there's a sidebar with the text "Main Axis" and "Cross Axis". Below the sidebar, there's a large preview area containing three green rectangular boxes labeled "1", "2", and "3" from top to bottom. To the right of the preview area, there's a circular profile picture of a person wearing a cap.



HTML

```
1 <div class="flex-container">
2 <!-- <span id="horizontal-axis">Main Axis</span> -->
3 <!-- <span id="vertical-axis">Cross Axis</span> -->
4 <div class="fi flex-item-1"><p>1</p></div>
5 <div class="fi flex-item-2"><p>2</p></div>
6 <div class="fi flex-item-3"><p>3</p></div>
7 <div class="fi flex-item-4"><p>4</p></div>
8 </div>
```

CSS

```
1 .flex-container {
2 /* width: 408px; */
3 /* height: 200px; */
4 /* display: flex; */
5 /* flex-direction: row; */
6 /* border: 4px solid #212226; */
7 }
8
9 .flex-item-1 {
10
11 }
12
13 .flex-item-2 {
14 }
```

Flexbox Playground ·
Zach

Main Axis

Cross Axis

```
HTML
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
7   <div class="fi flex-item-4"><p>4</p></div>
8 </div>
```

```
CSS
1 .flex-container {
2   width: 408px;
3   height: 200px;
4   display: flex;
5   flex-direction: row;
6   border: 4px solid #212226;
7 }
8
9 .flex-item-1 {
10
11 }
12
13 .flex-item-2 {
14 }
```

justify-content aligns the group of items along the main axis.

The screenshot shows the MDN Web Docs page for the CSS property `justify-content`. At the top, there's a navigation bar with links for Technologies, References & Guides, Feedback, and a search bar. Below the header, a breadcrumb trail indicates the page is under 'Web technology for developers > CSS: Cascading Style Sheets > justify-content'. On the left, there's a sidebar with a 'Table of contents' section containing links to Syntax, Formal definition, Formal syntax, Examples, Specifications, Browser compatibility, and See also. Below the sidebar, there's a 'Related Topics' section with links to CSS, CSS Reference, and CSS Box Alignment. A small circular profile picture of a man is visible on the left side of the sidebar. The main content area has a title 'justify-content' and a brief description of what it does. It includes an interactive demo where users can see how different justify-content values (start, center, space-between, space-around, space-evenly) affect the layout of three boxes labeled 'One', 'Two', and 'Three'. A 'Reset' button is at the top right of the demo area. A note below the demo explains that the alignment is done after lengths and auto margins are applied, and that if there's only one flexible element with `flex-grow` set to 0, it won't have any effect.

 Flexbox Playground 

Zach

Main Axis

Cross Axis



HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
  <div class="fi flex-item-4"><p>4</p></div>
</div>
```

CSS

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
}

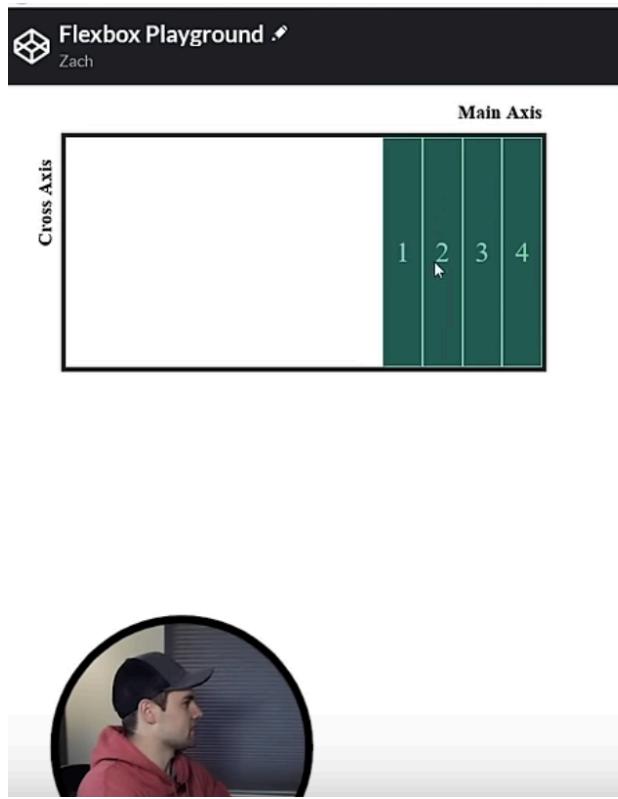
.flex-container {
  justify-content: start;
}

.flex-item-1 {
```

Flexbox Playground 
Zach

Main Axis

Cross Axis



HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
7   <div class="fi flex-item-4"><p>4</p></div>
8 </div>
```

CSS

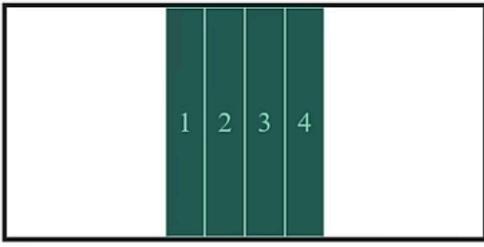
```
1 .flex-container {
2   width: 400px;
3   height: 200px;
4   display: flex;
5   flex-direction: row;
6   border: 4px solid #212226;
7 }
8
9 .flex-container {
10   justify-content: end;
11 }
12
13 .flex-item-1 {
```



Flexbox Playground 
Zach

Main Axis

Cross Axis



HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
7   <div class="fi flex-item-4"><p>4</p></div>
8 </div>
```

CSS

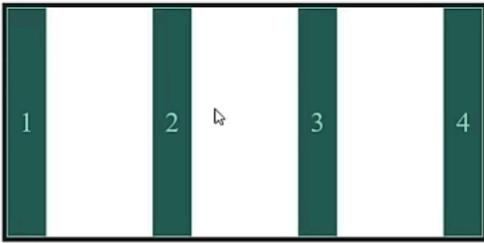
```
1 .flex-container {
2   width: 408px;
3   height: 200px;
4   display: flex;
5   flex-direction: row;
6   border: 4px solid #212226;
7 }
8
9 .flex-container {
10   justify-content: center;
11 }
12
13 .flex-item-1 {
```



Flexbox Playground 
 Zach

Main Axis

Cross Axis



HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
7   <div class="fi flex-item-4"><p>4</p></div>
8 </div>
```

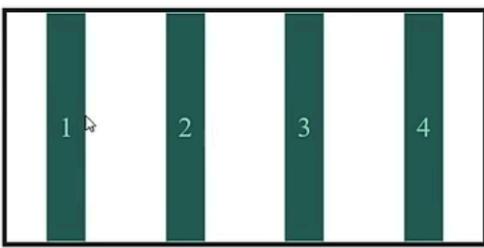
CSS

```
1 .flex-container {
2   width: 408px;
3   height: 200px;
4   display: flex;
5   flex-direction: row;
6   border: 4px solid #212226;
7 }
8
9 .flex-container {
10   justify-content: space-between;
11 }
12
13 .flex-item-1 {
```

Flexbox Playground 
 Zach

Main Axis

Cross Axis



HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
7   <div class="fi flex-item-4"><p>4</p></div>
8 </div>
```

CSS

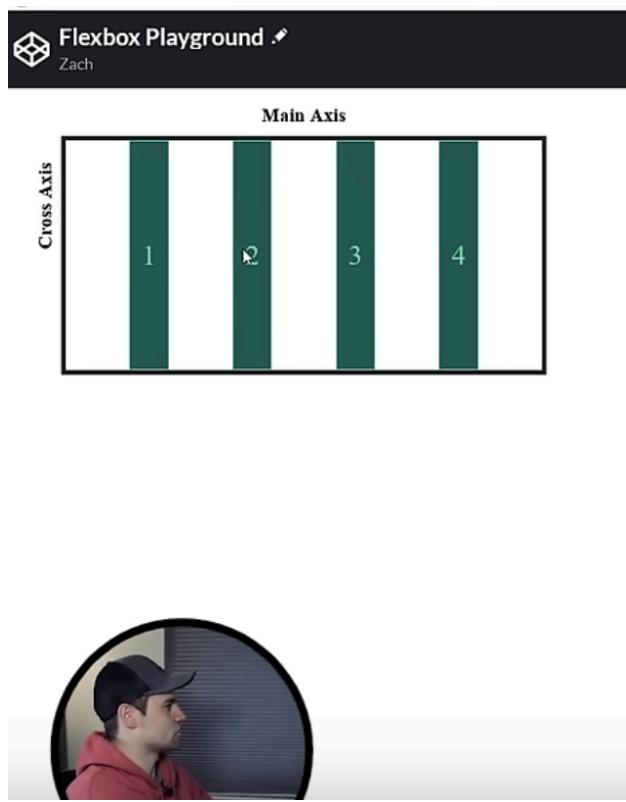
```
1 .flex-container {
2   width: 408px;
3   height: 200px;
4   display: flex;
5   flex-direction: row;
6   border: 4px solid #212226;
7 }
8
9 .flex-container {
10   justify-content: space-around;
11 }
12
13 .flex-item-1 {
```



Flexbox Playground 
Zach

Main Axis

Cross Axis



HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
7   <div class="fi flex-item-4"><p>4</p></div>
8 </div>
```

CSS

```
1 .flex-container {
2   width: 408px;
3   height: 200px;
4   display: flex;
5   flex-direction: row;
6   border: 4px solid #212226;
7 }
8
9 .flex-container {
10   justify-content: space-around;
11 }
12
13 .flex-item-1 {
```

`align-items` aligns the group of items along the cross axis.

MDN Web Docs moz://a

▶ Technologies ▶ References & Guides ▶ Feedback

Search MDN 🔍 Sign in

Web technology for developers ▶ CSS: Cascading Style Sheets ▶ align-items ⓘ Change language

Table of contents

Syntax
Formal definition
Formal syntax
Examples
Specifications
Browser compatibility
See also

Related Topics

CSS
CSS Reference
CSS Box Alignment
Guides
Alignment for block, absolutely positioned, and inline elements
Grid layout
Item alignment in Flexbox
Item alignment in grid layout
Item alignment in Multicolumn layout

align-items

The CSS `align-items` property sets the `align-self` value on all direct children as a group. In Flexbox, it controls the alignment of items on the Cross Axis. In Grid Layout, it controls the alignment of items on the Block Axis within their grid area.

The interactive example below demonstrates some of the values for `align-items` using grid layout.

CSS Demo: align-items Reset

align-items: stretch;

align-items: center;

align-items: start;

align-items: end;

One Two
Three

by default the width of the item is set to auto - occupying the space necessary to display its contents, and its height is the that of the container, unless otherwise specified by width and height properties.

Flexbox Playground 🎨

Zach

Main Axis

Cross Axis

HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><n>1</n></div>
```

CSS

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
}

.flex-container {
  justify-content: start;
  align-items: normal; | |
}

.flex-item-1 {

}

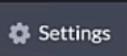
.flex-item-2 {
```

Save

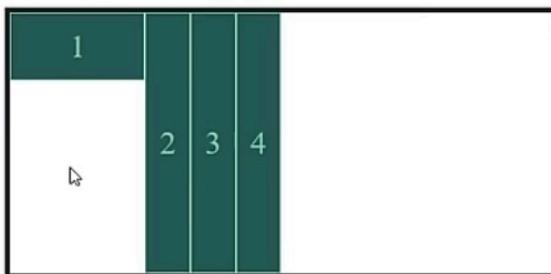
Settings

Flexbox Playground

Zach



Main Axis



Cross Axis



HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="flex-item-1"><n>1</n></div>
```

CSS

```
flex-direction: row;
border: 4px solid #212226;
}

.flex-container {
  justify-content: start;
  align-items: normal;
}

.flex-item-1 {
  width: 100px;
  height: 50px;
}

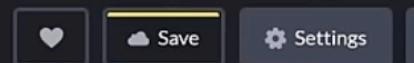
.flex-item-2 {

}

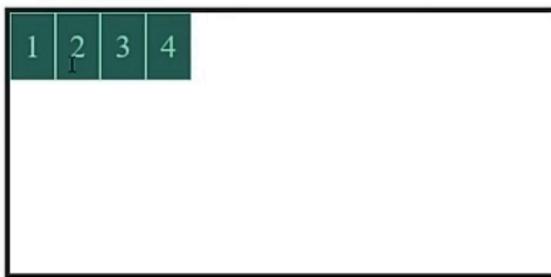
.flex-item-3 {
```

Flexbox Playground

Zach



Main Axis



Cross Axis



HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="flex-item-1"><p>1</p></div>
```

CSS

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
}

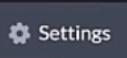
.flex-container {
  justify-content: start;
  align-items: flex-start;
}

.flex-item-1 {
}

.flex-item-2 {
```

Flexbox Playground

Zach



Main Axis



HTML

```
1 * <div class="flex-container">
2 *   <span id="horizontal-axis">Main Axis</span>
3 *   <span id="vertical-axis">Cross Axis</span>
4 *   <div class="fi flex-item-1"><p>1</p></div>
5 *   <div class="fi flex-item-2"><p>2</p></div>
6 *   <div class="fi flex-item-3"><p>3</p></div>
```

CSS

```
66 * .fi p {
67 *   font-size: 1.5rem;
68 *   margin: 0;
69 *   color: #71D99E;
70 * }
71 *
72 * .flex-item-1 p {
73 *   font-size: 3rem;
74 * }
```

Flexbox Playground

Zach



Main Axis



Cross Axis



HTML

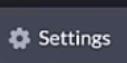
```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
```

CSS

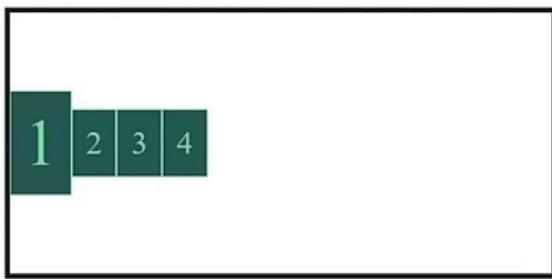
```
2   flex-direction: row;
3   border: 4px solid #212226;
4 }
5
6 .flex-container {
7   justify-content: start;
8   align-items: flex-end;
9 }
10
11 .flex-item-1 {
12 }
13
14 .flex-item-2 {
15 }
16
17 .flex-item-3 {
18 }
19
20 }
```

Flexbox Playground

Zach



Main Axis



Cross Axis



HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
```

CSS

```
1   flex-direction: row;
2   border: 4px solid #212226;
3 }
4
5 .flex-container {
6   justify-content: start;
7   align-items: center;
8 }
9
10 .flex-item-1 {
11 }
12
13 .flex-item-2 {
14 }
15
16 .flex-item-3 {
17 }
```

 Flexbox Playground 

Zach

Main Axis

Cross Axis

1 2 3 4



HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
```

CSS

```
flex-direction: row;
border: 4px solid #212226;
}

.flex-container {
  justify-content: start;
  align-items: baseline;
}

.flex-item-1 {

}

.flex-item-2 {

}
```

Working together with justify-content and align-items

Flexbox Playground 🎨

Zach

Main Axis

Cross Axis

HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
```

CSS

```
flex-direction: row;
border: 4px solid #212226;
}

.flex-container {
  justify-content: center;
  align-items: baseline;
}

.flex-item-1 {

}

.flex-item-2 {

}

.flex-item-3 {
```

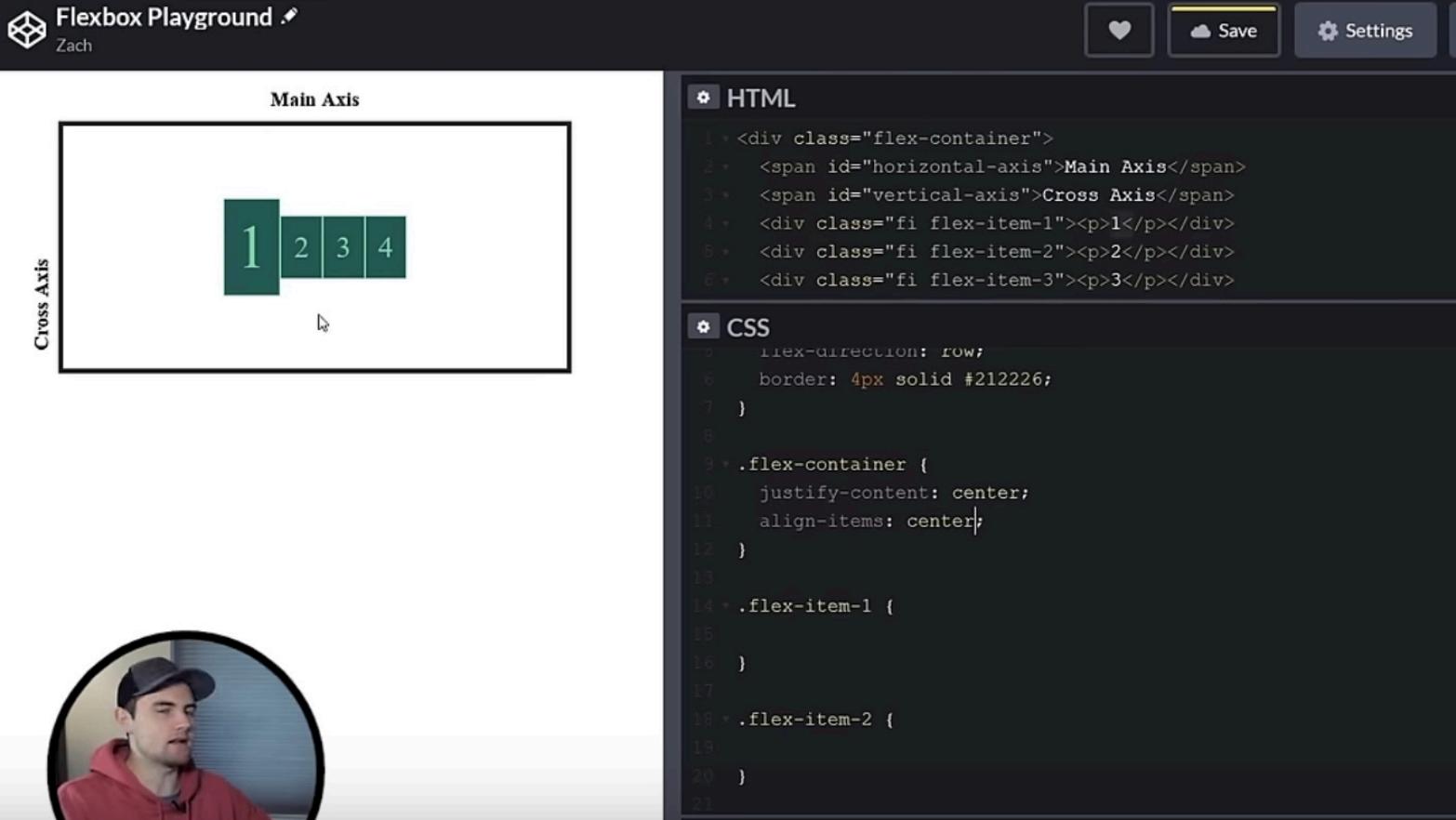
A circular portrait of a man with dark hair and a beard, wearing a black baseball cap and a red hoodie, looking towards the camera.

 Flexbox Playground 

Zach

Main Axis

Cross Axis



The Flexbox Playground interface consists of several sections:

- Top Bar:** Includes a logo, the title "Flexbox Playground", a pencil icon for editing, and user profile buttons for "Save" and "Settings".
- Section Headers:** "Main Axis" and "Cross Axis" are displayed above the visual representation.
- Visual Representation:** A large rectangular container labeled "flex-container" contains four items labeled 1, 2, 3, and 4. Item 1 is a large green box, while items 2, 3, and 4 are smaller green boxes arranged horizontally to its right. A small circular arrow icon is located below item 4.
- Code Editor:** Contains two tabs: "HTML" and "CSS".
- HTML Tab:** Displays the following code:

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
```
- CSS Tab:** Displays the following code:

```
flex-direction: row;
border: 4px solid #212226;
}

.flex-container {
  justify-content: center;
  align-items: center;
}

.flex-item-1 {

}

.flex-item-2 {
```

 Flexbox Playground  Zach

Main Axis

Cross Axis



 Save 

HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
```

CSS

```
1  flex-direction: row;
2  border: 4px solid #212226;
3  }
4
5 .flex-container {
6   justify-content: end;
7   align-items: flex-end;
8 }
9
10 .flex-item-1 {
11 }
12
13 .flex-item-2 {
14 }
15
16 .flex-item-3 {
17 }
18
19 .flex-item-4 {
20 }
21
```

How to deal with overflow: items outside the limits of their container.

Flexbox Playground ↗
Zach

Main Axis

Cross Axis

HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
  <div class="fi flex-item-4"><p>4</p></div>
  <div class="fi flex-item-5"><p>5</p></div>
  <div class="fi flex-item-6"><p>6</p></div>
  <div class="fi flex-item-7"><p>7</p></div>
  <div class="fi flex-item-8"><p>8</p></div>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
  <div class="fi flex-item-4"><p>4</p></div>
  <div class="fi flex-item-5"><p>5</p></div>
  <div class="fi flex-item-6"><p>6</p></div>
  <div class="fi flex-item-7"><p>7</p></div>
  <div class="fi flex-item-8"><p>8</p></div>
</div>
```

CSS

```
.flex-container {
  width: 408px;
```



Flex items were set to width and height of 50px (container is 400px wide) clearly their size has changes - use dev tools on firefox to check in fact that the items were shrunk.

The screenshot shows the Firefox Developer Tools Inspector panel over a Flexbox Playground example. The playground displays a grid of 16 boxes labeled 1 through 8 arranged in two rows of eight. A circular profile picture is positioned on the left side of the first row. The Main Axis is horizontal, and the Cross Axis is vertical. The CSS code in the left pane includes classes for horizontal and vertical axes, and individual flex item classes (fi) for each box. The Inspector's Layout tab is selected, showing the `div.flex-container` with a `flex-direction: row;` rule. The right pane lists the Flex Items, each with its class name and a small preview icon.

Flexbox Playground - Zach

HTML CSS JS Result

```
<div class="flex-container">
<span id="horizontal-axis">Main Axis</span>
<span id="vertical-axis">Cross Axis</span>
<div class="fi flex-item-1"><p>1</p></div>
<div class="fi flex-item-2"><p>2</p></div>
<div class="fi flex-item-3"><p>3</p></div>
<div class="fi flex-item-4"><p>4</p></div>
<div class="fi flex-item-5"><p>5</p></div>
<div class="fi flex-item-6"><p>6</p></div>
<div class="fi flex-item-7"><p>7</p></div>
<div class="fi flex-item-8"><p>8</p></div>
<div class="fi flex-item-9"><p>1</p></div>
<div class="fi flex-item-10"><p>2</p></div>
<div class="fi flex-item-11"><p>3</p></div>
<div class="fi flex-item-12"><p>4</p></div>
<div class="fi flex-item-13"><p>5</p></div>
<div class="fi flex-item-14"><p>6</p></div>
<div class="fi flex-item-15"><p>7</p></div>
<div class="fi flex-item-16"><p>8</p></div>
</div>
```

Main Axis

Cross Axis

Inspector

Search HTML

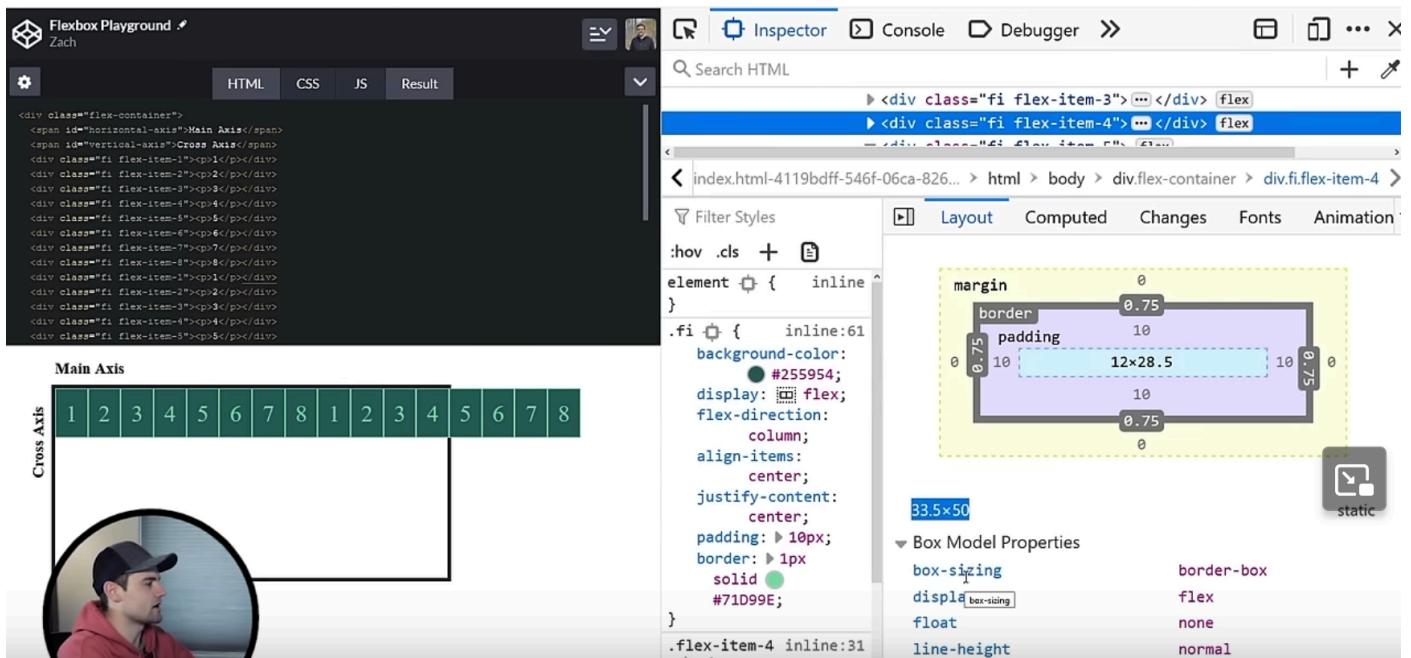
Layout Computed Changes Fonts Animation

Flex Container

div.flex-container

Flex Items

- 1 div.fi.flex-item-1
- 2 div.fi.flex-item-2
- 3 div.fi.flex-item-3
- 4 div.fi.flex-item-4
- 5 div.fi.flex-item-5
- 6 div.fi.flex-item-6
- 7 div.fi.flex-item-7
- 8 div.fi.flex-item-8



We can deal with this using the overflow property (not specific to flex box)

The screenshot shows a web-based Flexbox playground interface. On the left, there is a preview area displaying a grid of 16 items arranged in a 4x4 pattern. The items are numbered 1 through 16. Below the grid is a horizontal scrollbar. On the right, there are two main sections: 'HTML' and 'CSS'. The 'HTML' section contains the following code:

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
```

The 'CSS' section contains the following code:

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
  overflow: auto;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
}

.flex-item-1 {
```

A circular profile picture of a man wearing a cap is visible on the far left of the interface.

It's better to use the `flex-wrap` property, since we saw with the previous one we don't really get what we want.

The screenshot shows the Flexbox Playground interface. On the left, there is a preview area with two rows of four items each, labeled 1 through 8. The top row is labeled "Main Axis" and the bottom row "Cross Axis". To the right of the preview are two tabs: "HTML" and "CSS". The "HTML" tab contains the following code:

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><br>1</br></div>
```

The "CSS" tab contains the following code:

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
  flex-wrap: wrap;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
}

.flex-item-1 {
```

If we try to use align-items to align the second row below the first row, it won't work, since now flex-wrap is set to wrap.

The screenshot shows a Flexbox Playground interface. On the left, there's a preview area with two rows of four items each, labeled 1 through 8. The first row is aligned to the start (flex-start), and the second row is also aligned to the start. On the right, the code editor shows the HTML and CSS. The HTML includes a container with a border, a horizontal axis, a vertical axis, and a single flex item. The CSS defines the container with a width of 408px, height of 200px, display as flex, row direction, a 4px solid border, wrap flex-wrap, and align-items as flex-start. The .fi class defines the item width and height as 50px each.

```
HTML
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><n>1</n></div>
</div>

CSS
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
  flex-wrap: wrap;
  align-items: flex-start;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
```

 Flexbox Playground ·
Zach

Main Axis

Cross Axis

1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8



HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><n>1</n></div>
```

CSS

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
  flex-wrap: nowrap;
  align-items: flex-start;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
```

We need to use a new property called align-content.

The screenshot shows the Flexbox Playground interface. On the left, there is a 2x8 grid with the first row labeled "Main Axis" and the second row labeled "Cross Axis". The grid contains numbers from 1 to 8. On the right, the playground displays the generated HTML and CSS code.

HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
```

CSS

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
  flex-wrap: wrap;
  align-content: flex-start;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
```

 Flexbox Playground ·
Zach

Main Axis

1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8

Cross Axis



Save Settings Change View

HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
```

CSS

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
  flex-wrap: wrap;
  align-content: flex-end;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
```

 Flexbox Playground ·
Zach

Main Axis

1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8
Cross Axis							

Cross Axis



HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><r>1</r></div>
```

CSS

```
.flex-container {
  width: 408px;
  height: 200px;
  display: flex;
  flex-direction: row;
  border: 4px solid #212226;
  flex-wrap: wrap;
  align-content: center;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
```

When we align from the container we are moving the items as a group, but we can also align single items, selecting them and setting their alignment properties separately.

The screenshot shows the Flexbox Playground interface. On the left, there's a visual representation of a flex container with 7 items labeled 1 through 7. Item 1 is at the top left, and items 2 through 7 are arranged horizontally below it. A vertical line on the left is labeled "Cross Axis" and a horizontal line at the top is labeled "Main Axis". On the right, there are two tabs: "HTML" and "CSS". The "HTML" tab shows the following code:

```
<div>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
  <div class="fi flex-item-4"><p>4</p></div>
  <div class="fi flex-item-5"><p>5</p></div>
  <div class="fi flex-item-6"><p>6</p></div>
  <div class="fi flex-item-7"><p>7</p></div>
</div>
```

The "CSS" tab shows the following code:

```
.flex-container {
}

.flex-item-1 {
  align-self: flex-start;
}

.flex-item-2 {

}

.flex-item-3 {

}

.flex-item-4 {
```

At the bottom left, there is a circular profile picture of a man with a beard and a cap.

FLEXBOX Playground · Zach

Main Axis

Cross Axis

```
<div class="flex-item-2"><p>2</p></div>
<div class="flex-item-3"><p>3</p></div>
<!-->
<div class="flex-item-4"><p>4</p></div>
<div class="flex-item-5"><p>5</p></div>
<div class="flex-item-6"><p>6</p></div>
```

```
.flex-item-2 {
  align-self: flex-end;
}

.flex-item-3 {
  align-self: center;
}

.flex-item-4 {
  align-self: flex-end;
}

/* Don't worry about anything below this. It is not important
   for the example */
```

HTML

CSS 16 unsaved changes

JS

FLEXBOX Playground · Zach

Main Axis

Cross Axis

```
HTML
<div class="flex-container">
  <div class="flex-item-1"><p>1</p></div>
  <div class="flex-item-2"><p>2</p></div>
  <div class="flex-item-3"><p>3</p></div>
  <div class="flex-item-4"><p>4</p></div>
</div>
```

```
CSS
* {
  width: 50px;
  height: 50px;
}

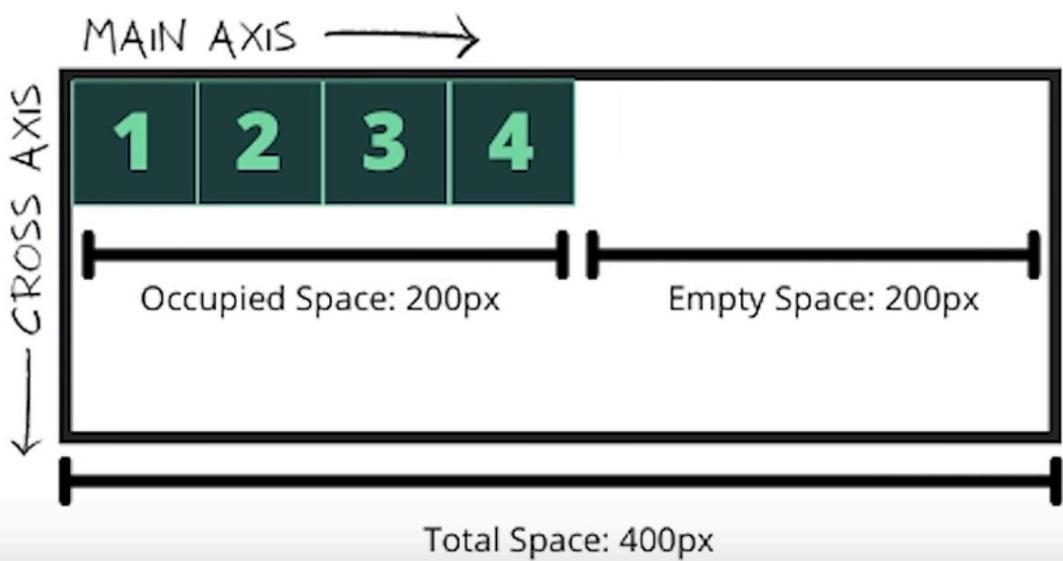
.flex-container {
  justify-content: center;
}

.flex-item-1 {
  align-self: flex-end;
}

.flex-item-2 {
```

JS

```
justify-content: start
```



flex-grow-units:

flex-grow is by default 0 for all flex-items.

what flex-grow does is distribute the remaining empty among the items for which we defined it.

For example, say we have 300px of empty space and 2 flex-items in total:

if we defined flex-grow-1 for each item, each will grow by 1 unit —>
300px/total of 2 units = 150px per unit
each item will grow by 150px.

if we defined flex-grow-1 for one item and flex-grow-2 for the other —>
300px/total of 3 units = 100px per unit
the item with flex-grow-1 will grow by 100px and
the item with flex-grow-2 will grow by 200px.

The screenshot shows the Flexbox Playground interface. On the left, there's a visual representation of a flex container with four items labeled 1, 2, 3, and 4. Item 1 is in a separate row above items 2, 3, and 4. Labels "Main Axis" and "Cross Axis" are positioned to the left of the container. On the right, there's a code editor with three tabs: HTML, CSS, and JS. The HTML tab shows the DOM structure with spans for the axes and divs for the flex items. The CSS tab contains the following code:

```
width: 50px;
height: 50px;
}

.flex-container {
}

.flex-item-1 {
  flex-grow: 1;
}

.flex-item-2 {
```

The JS tab is currently empty.

Flexbox Playground

Zach



Settings

Change View



Main Axis

1	2	3	4

Cross Axis



HTML

```
<span id="horizontal-axis">Main Axis</span>
Cross Axis

<p>1</p>



<p>2</p>



<p>3</p>


```

CSS

```
width: 50px;
height: 50px;

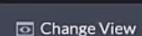
.flex-container {
}

.flex-item-1 {
    flex-grow: 1;
}

.flex-item-2 {
    flex-grow: 1;
}
```

Flexbox Playground

Zach



Main Axis

1	2	3	4

Cross Axis



HTML

```
<span id="horizontal-axis">Main Axis</span>
Cross Axis

<p>1</p>



<p>2</p>



<p>3</p>



<p>4</p>


```

CSS

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

.flex-item-2 {
  flex-grow: 1;
}

.flex-item-3 {
  flex-grow: 1;
}

.flex-item-4 {
  flex-grow: 1;
}
```

 Flexbox Playground ·
Zach

Main Axis

Cross Axis

1 | 2 | 3 | 4

Main Axis

Cross Axis

1 <p>1</p>

2 <p>2</p>

3 <p>3</p>

4 <p>4</p>

HTML

```
<span id="horizontal-axis">Main Axis</span>
<span id="vertical-axis">Cross Axis</span>


CSS 34 unsaved changes X



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

.flex-item-2 {
 flex-grow: 3;
}

.flex-item-3 {
 flex-grow: 1;
}

.flex-item-4 {
 flex-grow: 1;
}
```





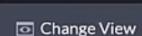


Save Settings Change View


```

Flexbox Playground

Zach



Main Axis

1	2	3	4

Cross Axis



HTML

```
<span id="horizontal-axis">Main Axis</span>
Cross Axis

<p>1</p>



<p>2</p>



<p>3</p>


```

CSS

```
.flex-container {
}

/* 1 3 1 1 = 6 units - 200px empty space / 6 units = 33.333px /
unit */

.flex-item-1 {
    flex-grow: 1;
}
.flex-item-2 {
    flex-grow: 3;
}
.flex-item-3 {
    flex-grow: 1;
}
```



`flex-shrink-units:`
by default `flex-shrink-1`.

`shrink` is works similarly to `flex-grow` but instead of distributing the empty space, it distributes the over-flow !

Attention: content will spill over the container when items cannot be shrunk beyond its minimum content space.

The screenshot shows a Flexbox Playground interface. On the left, there's a circular video player window showing a man with a beard and a cap. The main area displays a 3x3 grid of items labeled 1 through 9. The first row is dark green and labeled "Main Axis". The second row is light green and labeled "Cross Axis". In the bottom right corner of the playground, there's a "Subtitles/closed captions" button. The top navigation bar includes icons for heart, save, settings, and change view, along with a user profile picture.

```
span id="horizontal-axis">Main Axis</span>
<span id="vertical-axis">Cross Axis</span>
```

```
.flex-item-1 {
  flex-shrink: 1;
}

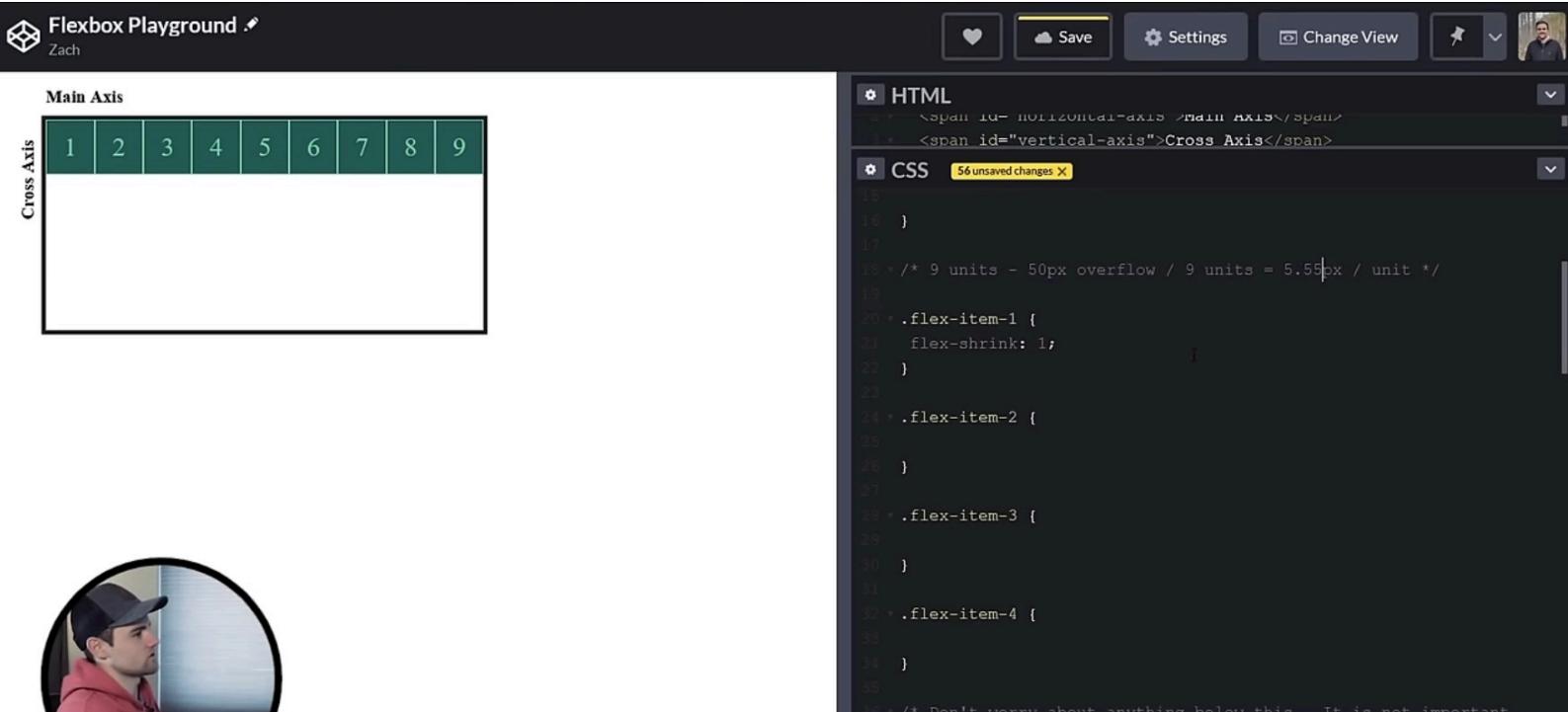
.flex-item-2 {

}

.flex-item-3 {

}

.flex-item-4 {
```



Flexbox Playground

Zach

Main Axis

Cross Axis

HTML

```
<div class="fi flex-item-2"><p>2</p></div>
<div class="fi flex-item-3"><p>3</p></div>
<div class="fi flex-item-4"><p>4</p></div>
<div class="fi flex-item-5"><p>5</p></div>
<div class="fi flex-item-6"><p>6</p></div>
<div class="fi flex-item-7"><p>7</p></div>
<div class="fi flex-item-8"><p>8</p></div>
<div class="fi flex-item-9"><p>9</p></div>
<div class="fi flex-item-9"><p>9</p></div>
<div class="fi flex-item-9"><p>9</p></div>
<div class="fi flex-item-9"><p>9</p></div>
```

CSS

```
}

/* 9 units - 50px overflow / 9 units = 5.55px / unit */

.flex-item-1 {
  flex-shrink: 1;
}
```

Sizing flex-items with flex-basis:
auto will make the item big enough to withhold its content.
otherwise we use percentages to resize.
the percentage refers to the percentage of empty space. (make sure its true...)

The screenshot shows the Flexbox Playground interface. On the left, there's a preview area with a grid labeled "Main Axis" and "Cross Axis". The grid contains four items labeled 1, 2, 3, and 4. Item 1 is highlighted with a dark green background. On the right, there are two tabs: "HTML" and "CSS". The "HTML" tab displays the following code:

```
<div class="fi flex-item-2"><p>2</p></div>
<div class="fi flex-item-3"><p>3</p></div>
<div class="fi flex-item-4"><p>4</p></div>
<!-- &lt;div class="fi flex-item-5"&gt;&lt;p&gt;5&lt;/p&gt;&lt;/div&gt;
&lt;div class="fi flex-item-6"&gt;&lt;p&gt;6&lt;/p&gt;&lt;/div&gt;</pre>

The "CSS" tab displays the following code:



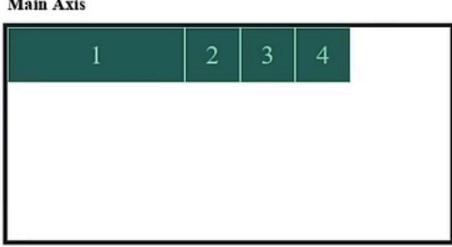
```
15
16 }
17
18 .flex-item-1 {
19 flex-basis: auto;
20 }
21
22 .flex-item-2 {
23 }
24
25 .flex-item-3 {
26 }
27
28 }
29
30 .flex-item-4 {
```


```

 Flexbox Playground ·
Zach

Main Axis

Cross Axis



HTML

```
<div class="fi flex-item-2"><p>2</p></div>
<div class="fi flex-item-3"><p>3</p></div>
<div class="fi flex-item-4"><p>4</p></div>
<!-- <div class="fi flex-item-5"><p>5</p></div>
<div class="fi flex-item-6"><p>6</p></div>
```

CSS

```
flex-direction: row;
border: 4px solid #212226;
}

.fi {
  width: 50px;
  height: 50px;
}

.flex-container {
}

.flex-item-1 {
  flex-basis: 40%;
}

.flex-item-2 {
```





Settings

Change View



Main Axis

1	2	3	4

Cross Axis



HTML

```
1 <div class="flex-container">
2   <span id="horizontal-axis">Main Axis</span>
3   <span id="vertical-axis">Cross Axis</span>
4   <div class="fi flex-item-1"><p>1</p></div>
5   <div class="fi flex-item-2"><p>2</p></div>
6   <div class="fi flex-item-3"><p>3</p></div>
7   <div class="fi flex-item-4"><p>4</p></div>
8   <!-- <div class="fi flex-item-5"><p>5</p></div>
9   <div class="fi flex-item-6"><p>6</p></div>
10  <div class="fi flex-item-7"><p>7</p></div>
11  <div class="fi flex-item-8"><p>8</p></div>
12  <div class="fi flex-item-9"><p>9</p></div> -->
13 </div>
```

CSS

```
.flex-container {
15
16 }
17
18 .flex-item-1 {
19   flex-basis: 25%;
20 }
```

Reordering

The screenshot shows the Flexbox Playground interface. On the left, there's a preview area with a grid labeled "Main Axis" (horizontal) and "Cross Axis" (vertical). The grid contains four items labeled 2, 1, 3, and 4. Item 2 is at the top-left, item 1 is below it, item 3 is to the right of item 1, and item 4 is to the right of item 3. To the left of the preview is a circular profile picture of a man wearing a cap. On the right side of the interface, there are two code editors. The top one is labeled "HTML" and contains the following code:

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div class="fi flex-item-1"><p>1</p></div>
  <div class="fi flex-item-2"><p>2</p></div>
  <div class="fi flex-item-3"><p>3</p></div>
  <div class="fi flex-item-4"><p>4</p></div>
  <!-- <div class="fi flex-item-5"><p>5</p></div>
  <div class="fi flex-item-6"><p>6</p></div>
  <div class="fi flex-item-7"><p>7</p></div>
  <div class="fi flex-item-8"><p>8</p></div>
  <div class="fi flex-item-9"><p>9</p></div> -->
</div>
```

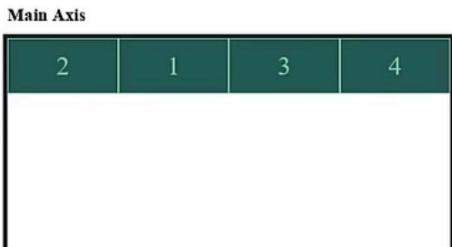
The bottom code editor is labeled "CSS" and contains the following styles:

```
.flex-container {
}
.flex-item-1 {
  flex-basis: 25%;
}
```

 Flexbox Playground ·
Zach

Main Axis

Cross Axis





HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
```

CSS

```
flex-basis: 25%;  
order: 2;  
}  
  
.flex-item-2 {  
  flex-basis: 25%;  
  order: 1;  
}  
  
.flex-item-3 {  
  flex-basis: 25%;  
  order: 3;  
}  
  
.flex-item-4 {  
  flex-basis: 25%;  
  order: 4; }  
  
/* Don't worry about anything below this. It is not important  
for the example */
```

Flexbox Playground

Zach

Main Axis

Cross Axis

HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
```

CSS

```
.flex-item-3 {
  flex-basis: 25%;
  order: 3;
}

.flex-item-4 {
  flex-basis: 25%;
  order: 4;
}

@media (min-width: 600px) {
  .flex-item-4 {
    order: 1;
  }

  .flex-item-1 {
    order: 4;
  }
}
```

Flexbox Playground

Zach

Main Axis

Cross Axis



HTML

```
<div class="flex-container">
  <span id="horizontal-axis">Main Axis</span>
```

CSS

```
.flex-item-3 {
  flex-basis: 25%;
  order: 3;
}

.flex-item-4 {
  flex-basis: 25%;
  order: 4;
}

@media (min-width: 600px) {
  .flex-item-4 {
    order: 1;
  }

  .flex-item-1 {
    order: 4;
  }
}
```

shorthanded commands:
flex: flex-grow flex-shrink flex-basis

```
17  
18 * .flex-item-1 {  
19   flex-basis: 25%;  
20   order: 2;  
21   flex: 1 1 auto;|  
22 }  
23
```