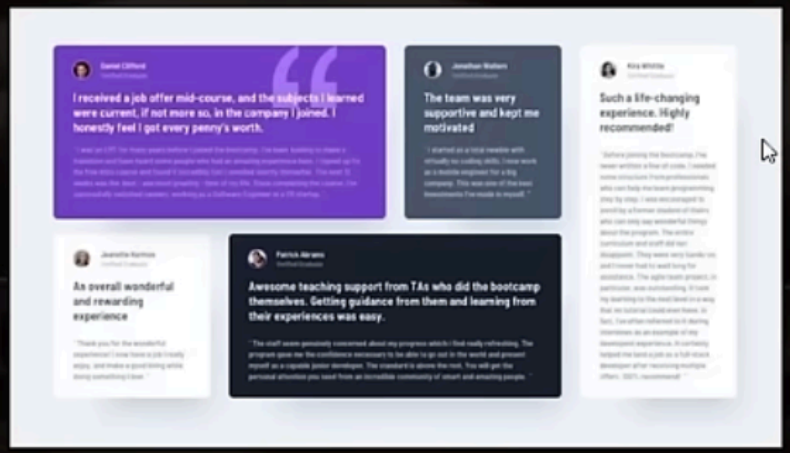
Core Lessons

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)

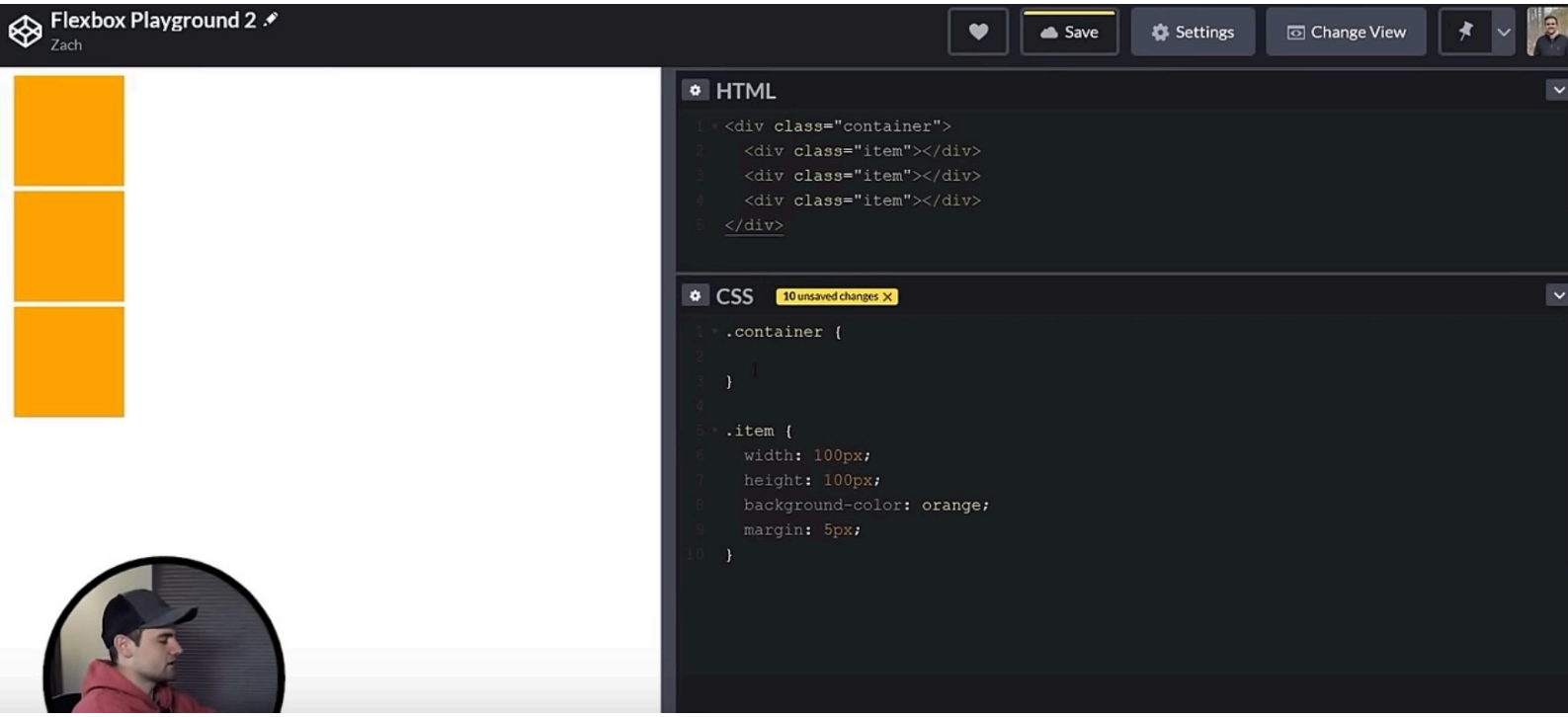
Frontend Mentor Project! (testimonial page)

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 !!!



Flexbox Playground 2

Zach

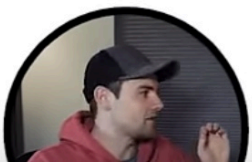
Save | Settings | Change View

HTML

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

CSS 12 unsaved changes

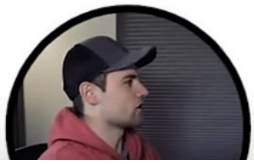
```
.container {
  display: flex;
}

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



Flexbox Playground 2

Zach

Save Settings Change View



HTML 25 unsaved changes

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

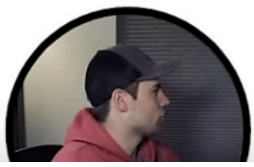
CSS

```
.container {
  display: flex;
}

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

.sub-item {
  width: 15px;
  height: 15px;
```

Flexbox Playground 2

Zach

Save Settings Change View



HTML

```
<div class="container">
  <div class="item"></div>
  <div class="item"></div>
  <div class="item item-3">
    <div class="sub-item"></div>
    <div class="sub-item"></div>
    <div class="sub-item"></div>
  </div>
```

CSS 28 unsaved changes

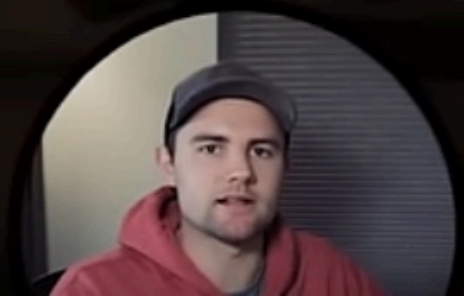
```
.item {
  width: 100px;
  height: 100px;
  background-color: orange;
  margin: 5px;
}

.item-3 {
  display: flex;
}

.sub-item {
  width: 15px;
  height: 15px;
  background-color: blue;
```

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.

while respecting width and height constraints.

`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

Initial value	`normal`
Applies to	all elements
Inherited	no
Computed value	as specified
Animation type	discrete

Formal syntax

```
normal | stretch | <baseline-position> | [ <overflow-position>? <self-position> ]where <baseli
```

Save Setting

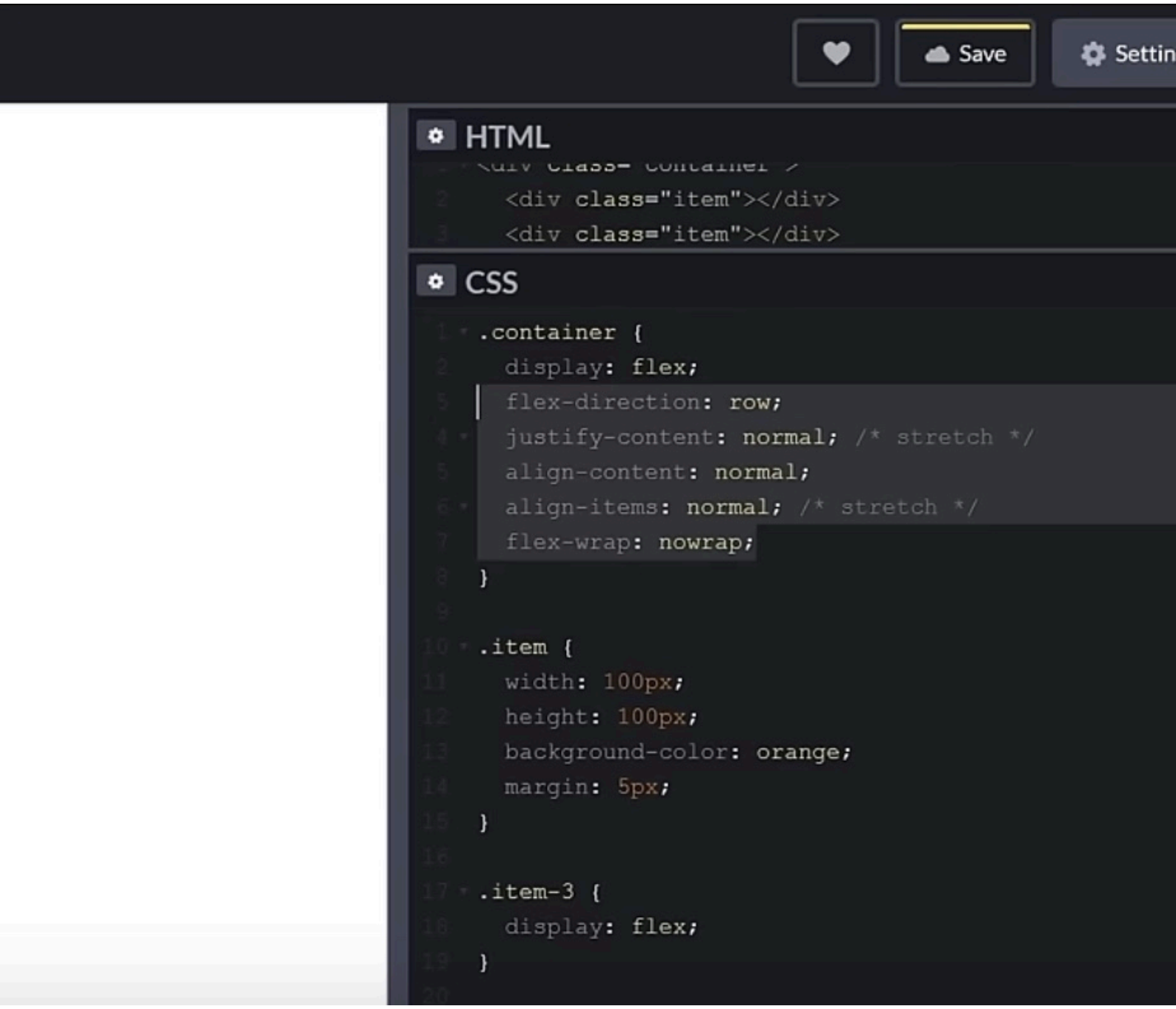
HTML

```
<div class="container">
  <div class="item"></div>
  <div class="item"></div>
  <div class="item item-3">
    <div class="sub-item"></div>
    <div class="sub-item"></div>
```

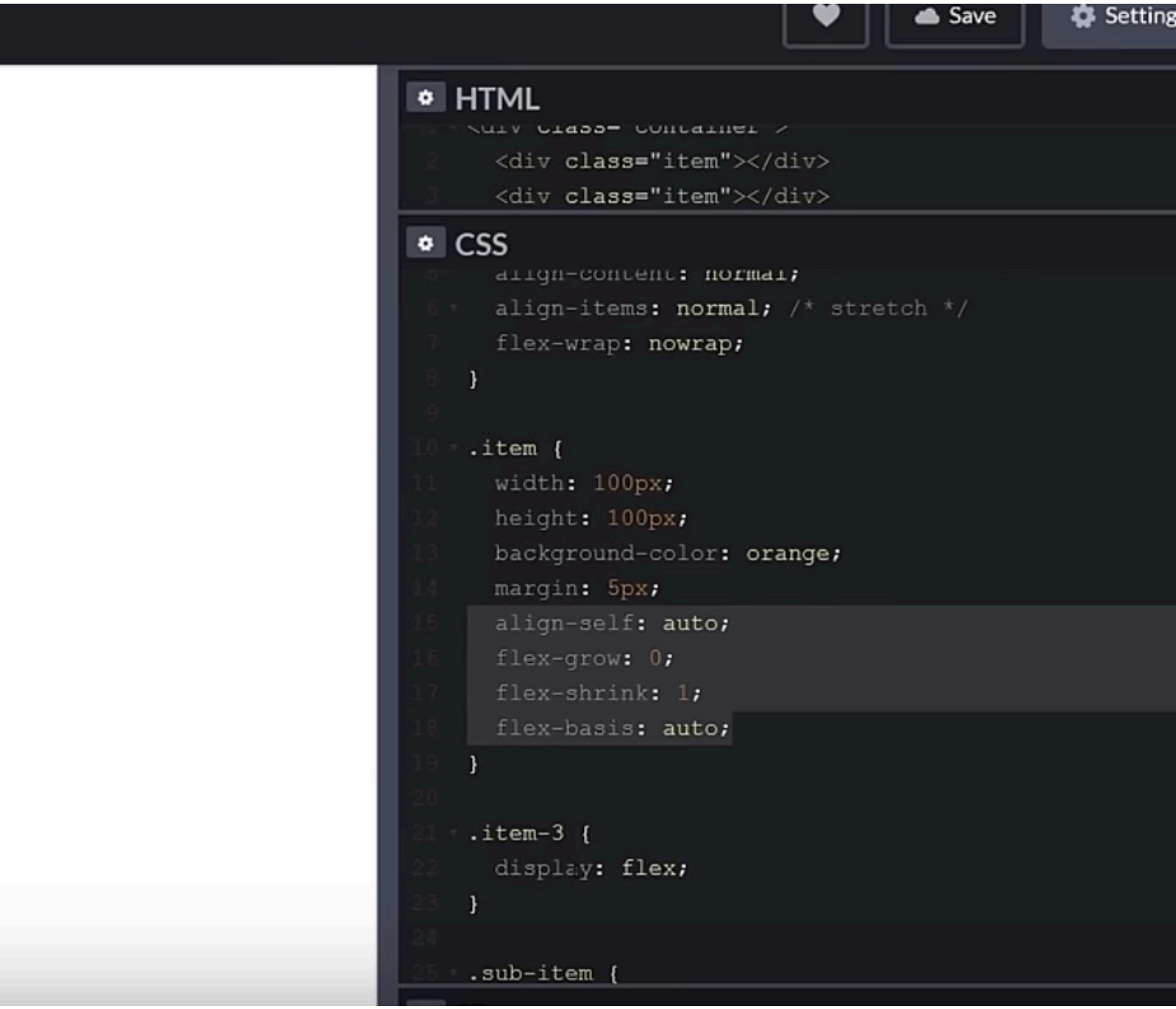
CSS

```
.item {
  width: 100px;
  height: 100px;
  background-color: orange;
  margin: 5px;
  flex: 0 1 auto;
  flex-grow: 0;
  flex-shrink: 1;
  flex-basis: auto;
}

.item-3 {
  display: flex;
}

.sub-item {
```

Default values for flex container:



Default values for flex item:



Row vs. Column

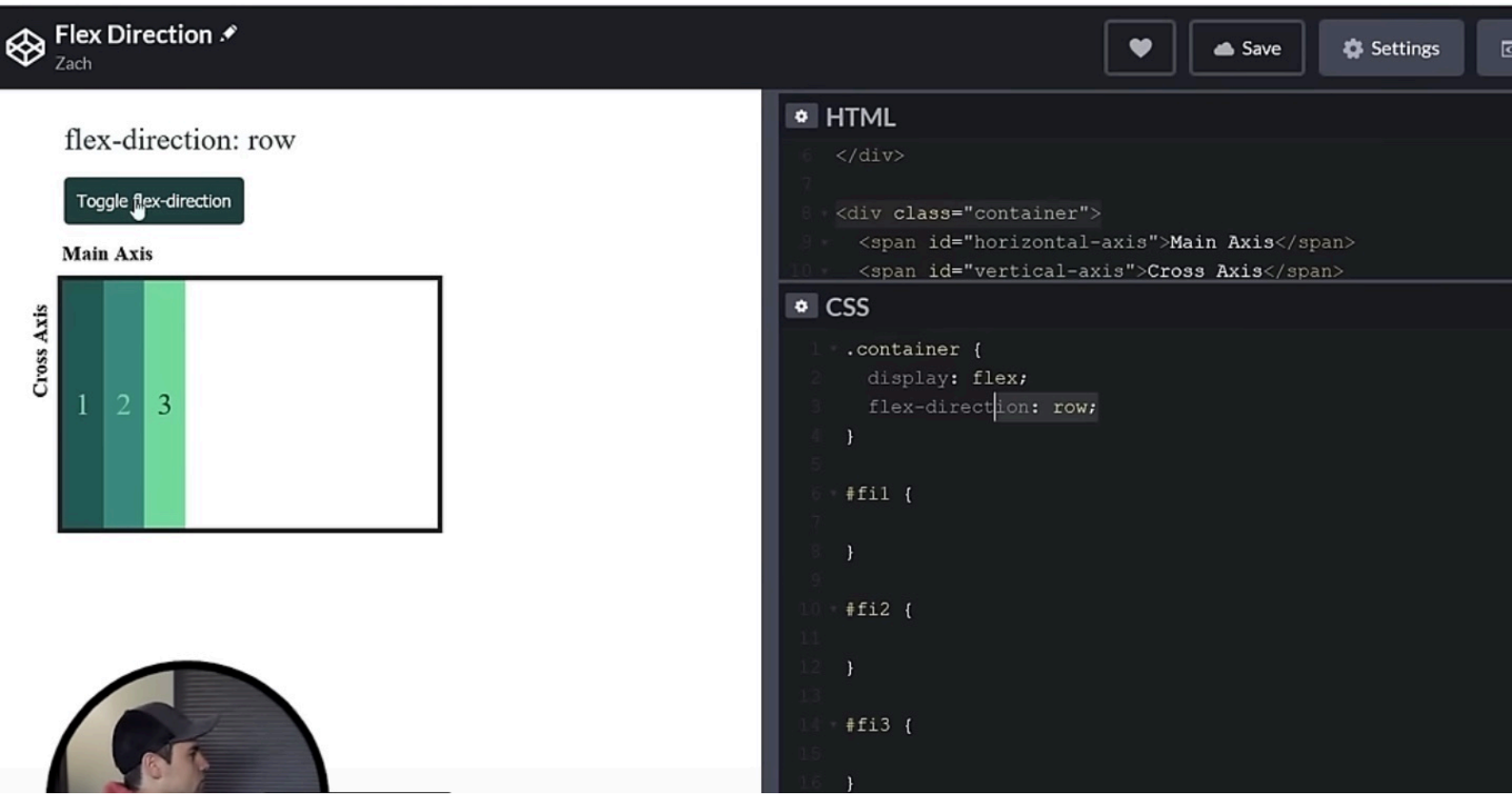
HTML

```
<div class="top-section">
  <div class="current-setting">
    <p><span id="setting">flex-direction: row</span></p>
  </div>
  <button onclick="setFlex()">Toggle flex-direction</button>
</div>

<div class="container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div id="fi1">
    <p>1</p>
  </div>
  <div id="fi2">
    <p>2</p>
  </div>
  <div id="fi3">
```

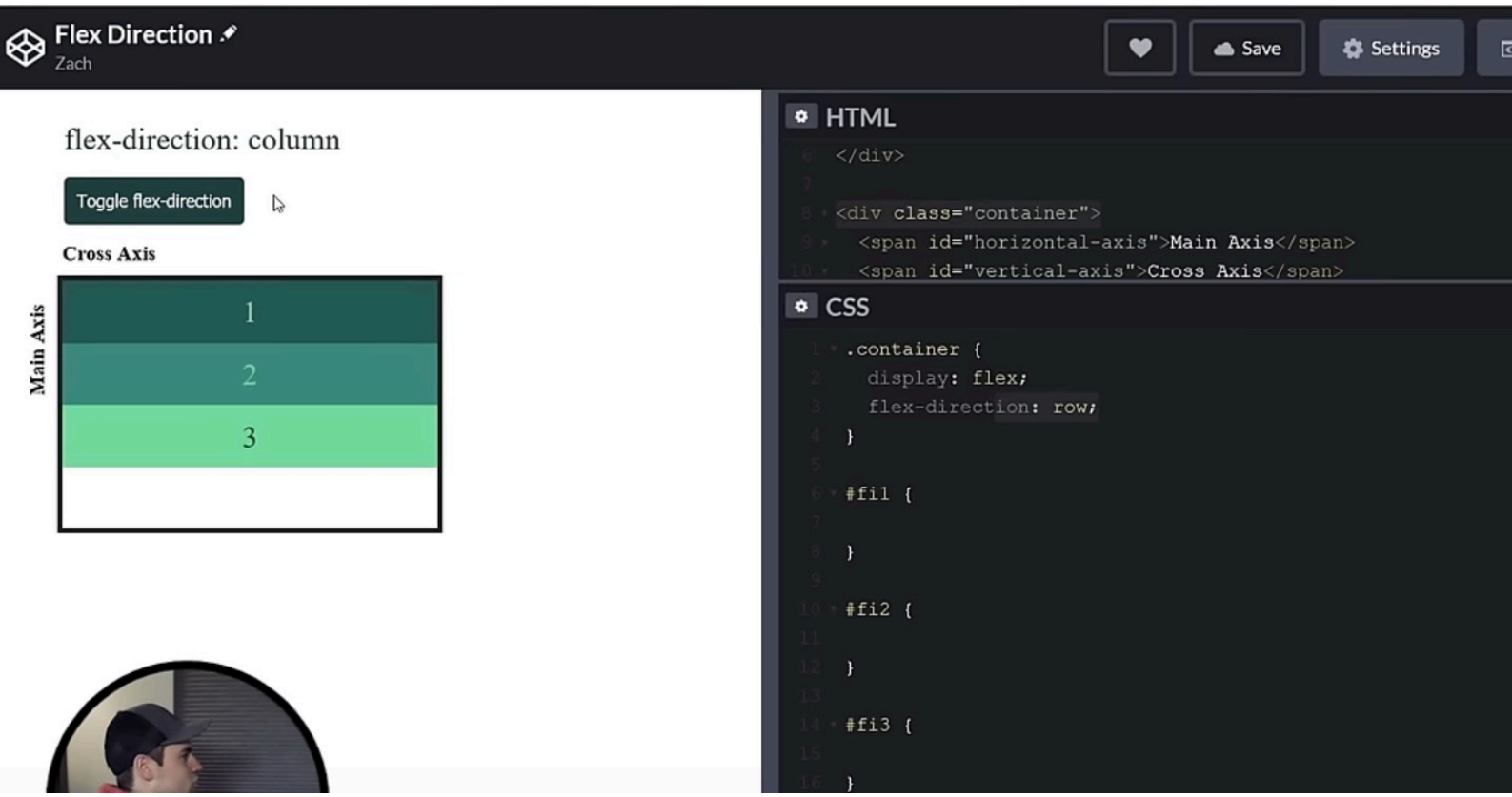
HTML

```
  <button onclick="setFlex()">Toggle flex-direction</button>
</div>

<div class="container">
  <span id="horizontal-axis">Main Axis</span>
  <span id="vertical-axis">Cross Axis</span>
  <div id="fi1">
    <p>1</p>
  </div>
  <div id="fi2">
    <p>2</p>
  </div>
  <div id="fi3">
    <p>3</p>
  </div>
</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.



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.



Flexbox Playground

Zach

Save

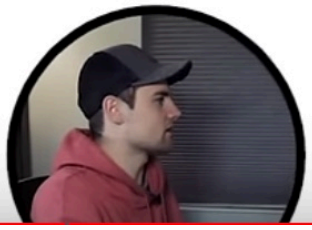
Settings

1

2

3

4

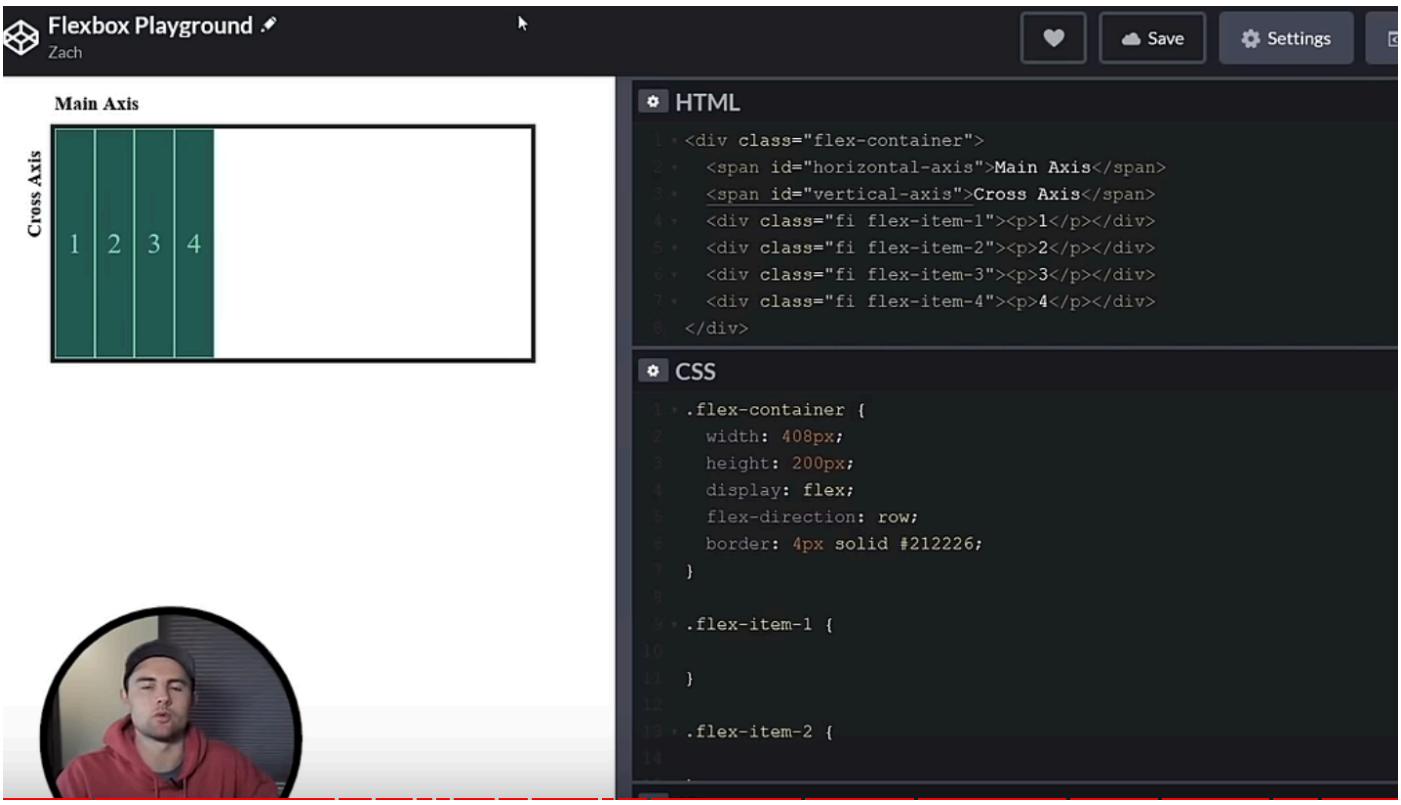


HTML

```
<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>
  <div class="fi flex-item-4"><p>4</p></div>
</div>
```

CSS

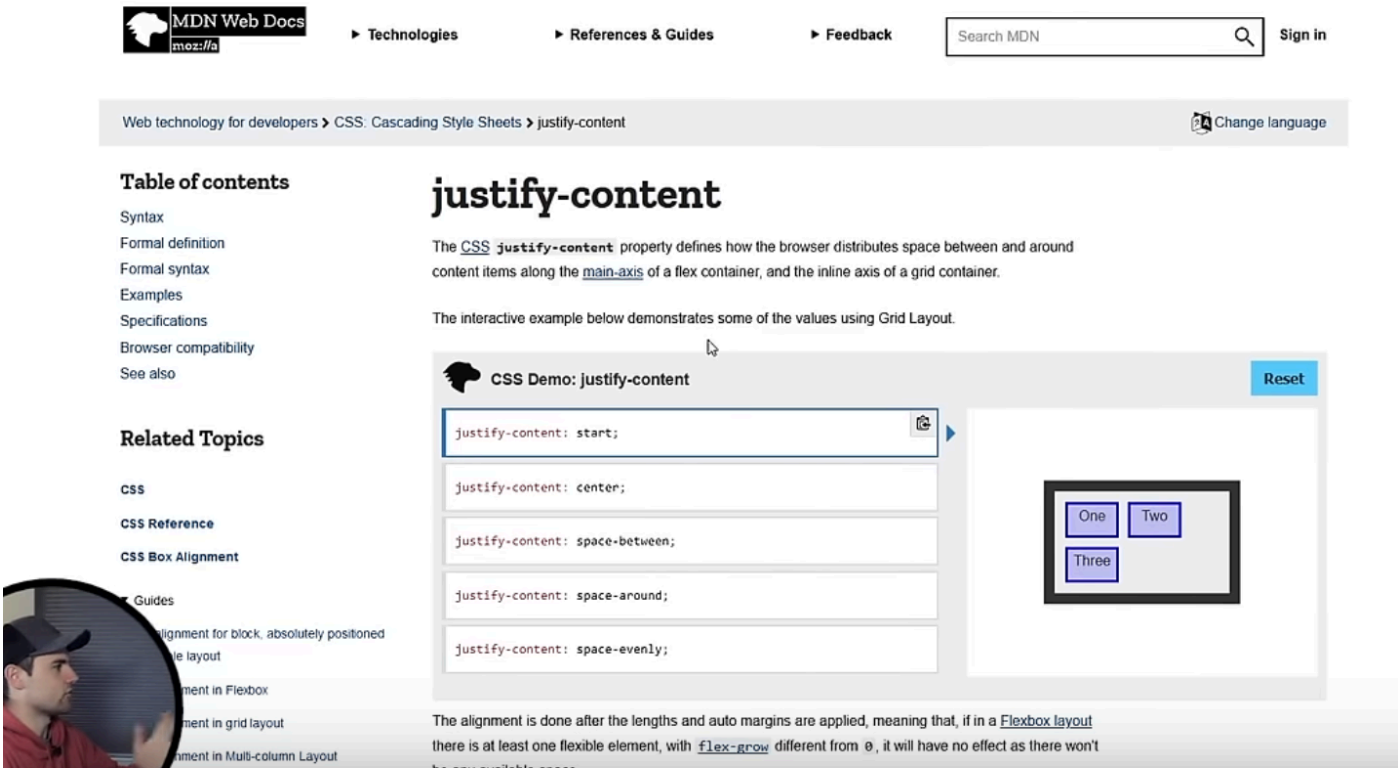
```
.flex-container {
/*     width: 408px; */
/*     height: 200px; */
/*   display: flex; */
/*   flex-direction: row; */
/*   border: 4px solid #212226; */
}

.flex-item-1 {

}

.flex-item-2 {
```

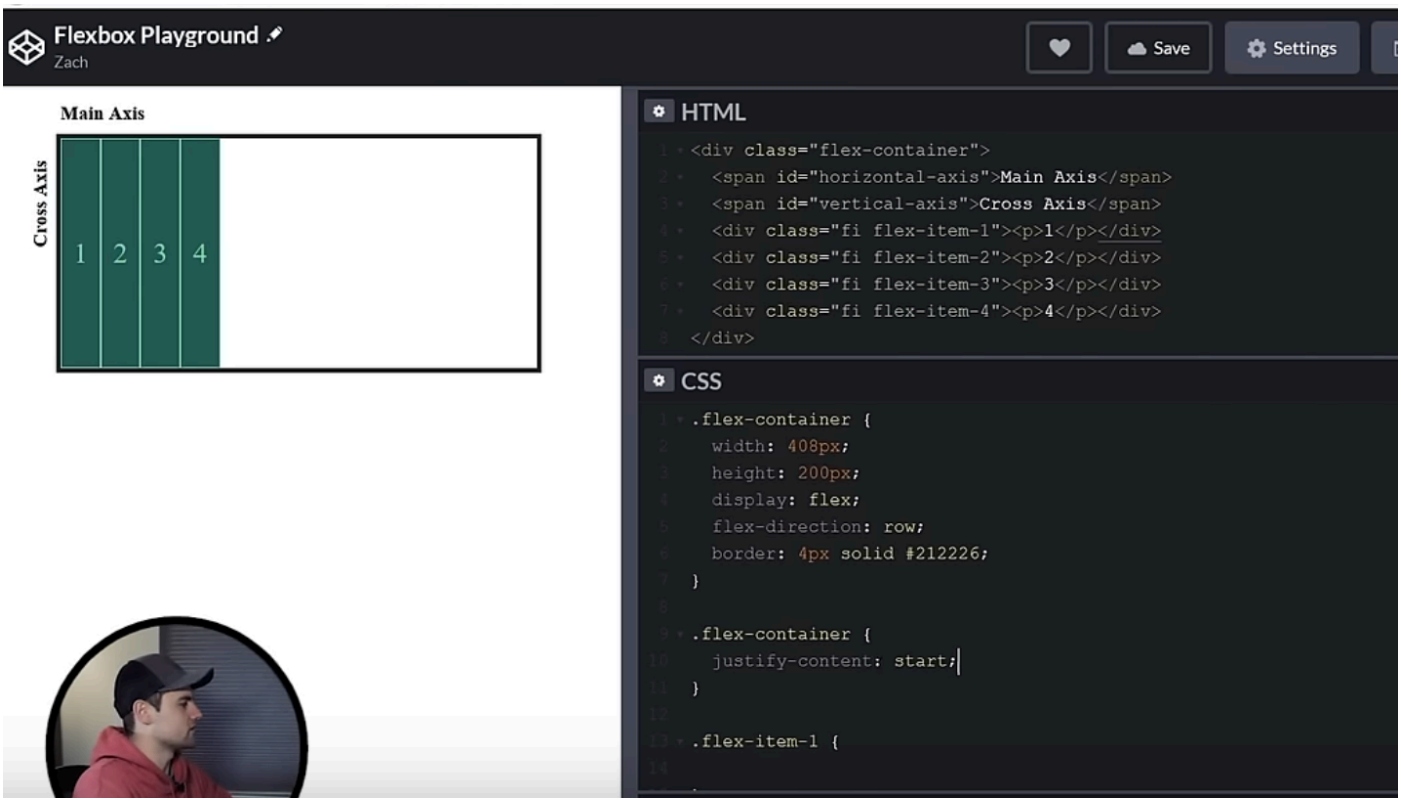

Flexbox Playground
Zach
Save
Settings
Main Axis
Cross Axis
1
2
3
4
HTML

Main Axis
Cross Axis
1
2
3
4

CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
}
.flex-item-1 {
}
.flex-item-2 {

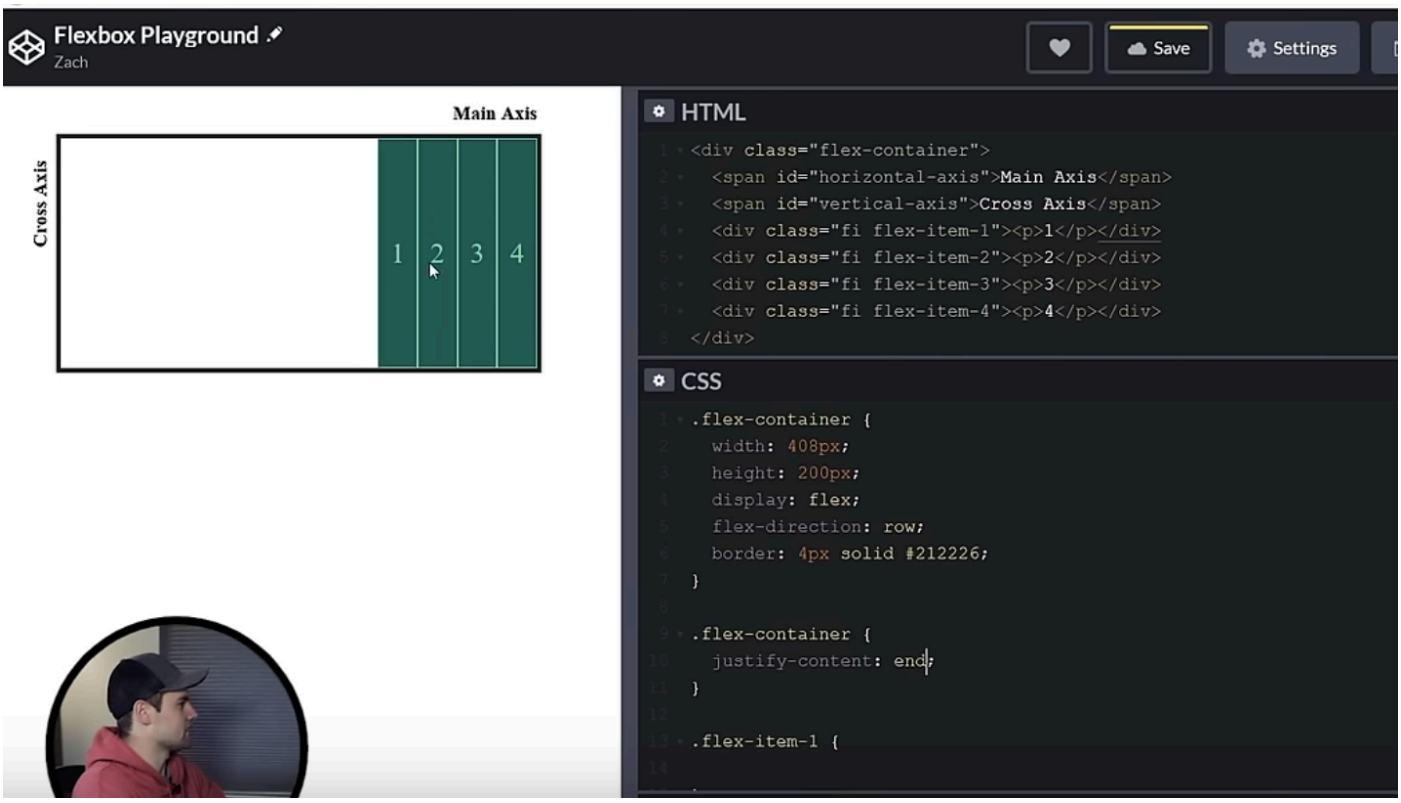
justify-content aligns the group of items along the main axis.




Flexbox Playground
Zach
Save
Settings
Main Axis
Cross Axis
1
2
3
4
HTML

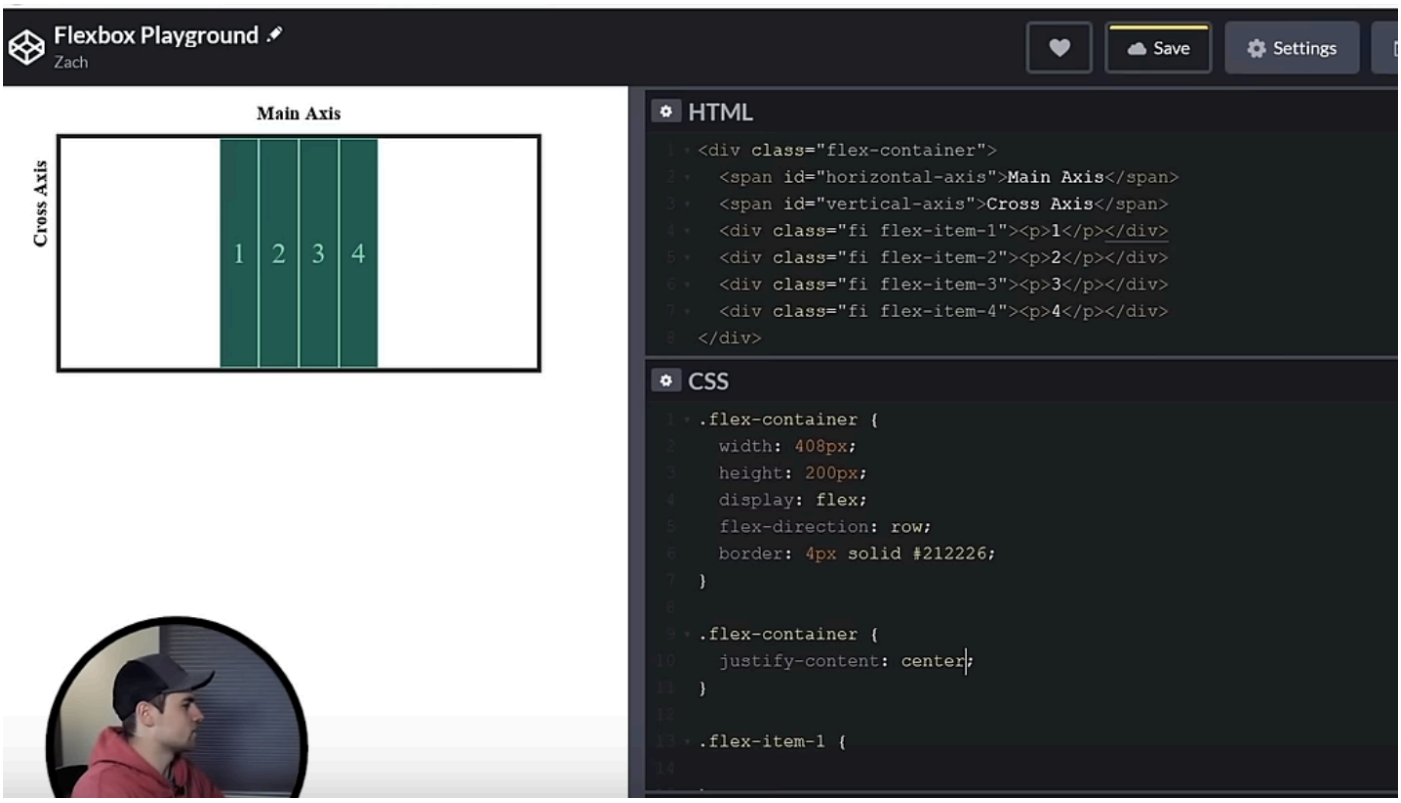
Main Axis
Cross Axis
1
2
3
4

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
Save
Settings
Main Axis
Cross Axis
1
2
3
4
HTML

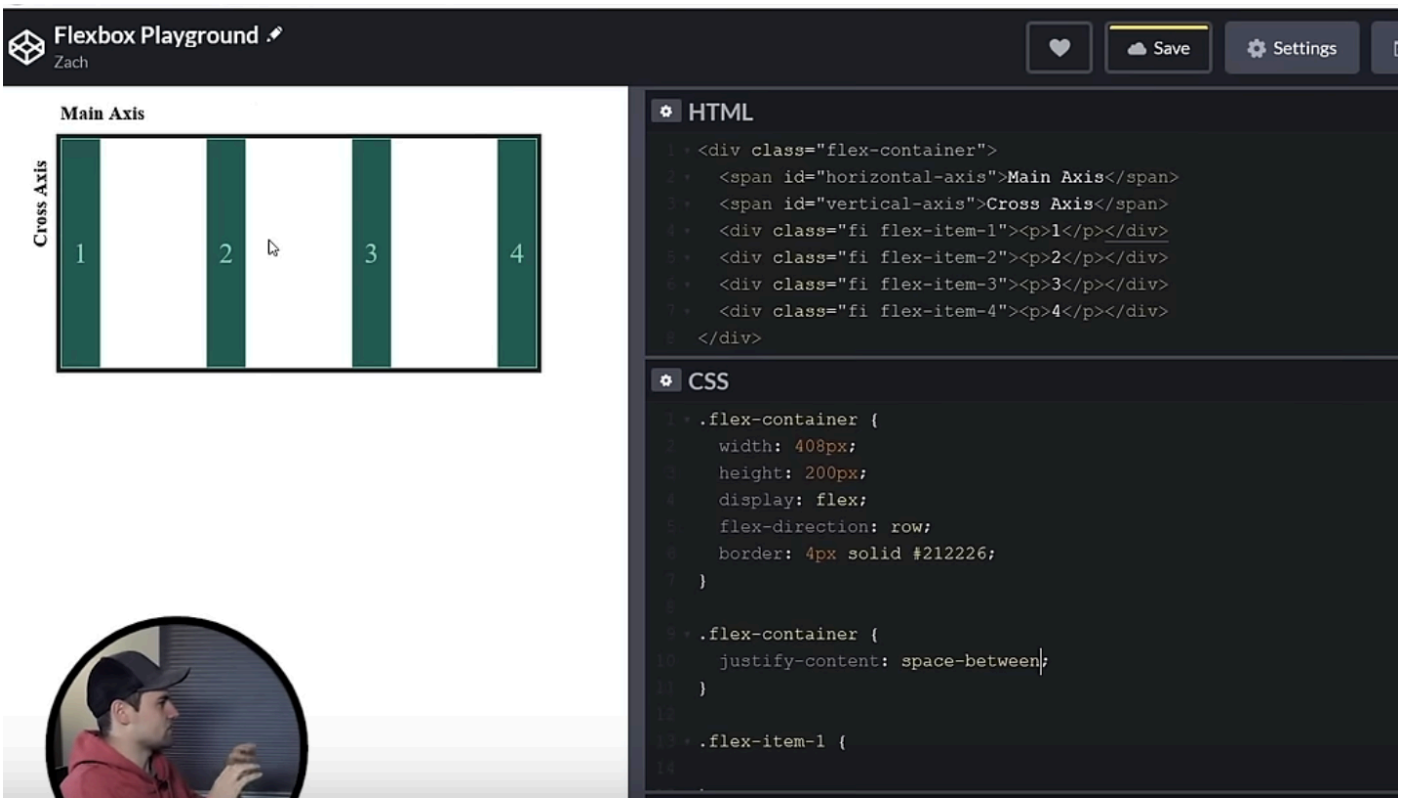
Main Axis
Cross Axis
1
2
3
4

CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
}
.flex-container {
justify-content: end;
}
.flex-item-1 {


Flexbox Playground
Zach
Save
Settings
Main Axis
Cross Axis
1
2
3
4
HTML

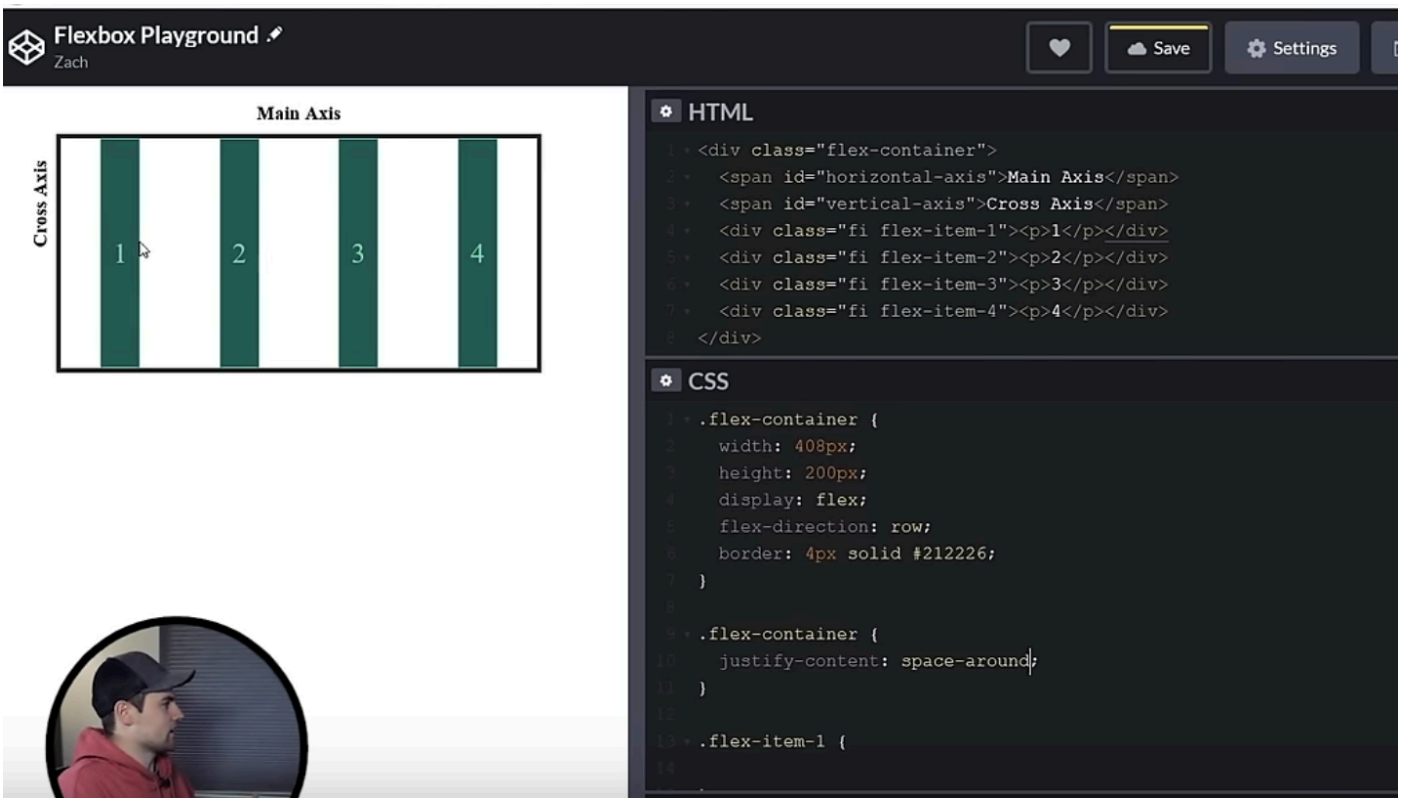
Main Axis
Cross Axis
1
2
3
4

CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
}
.flex-container {
justify-content: center;
}
.flex-item-1 {


Flexbox Playground
Zach
Save
Settings
Main Axis
Cross Axis
1
2
3
4
HTML

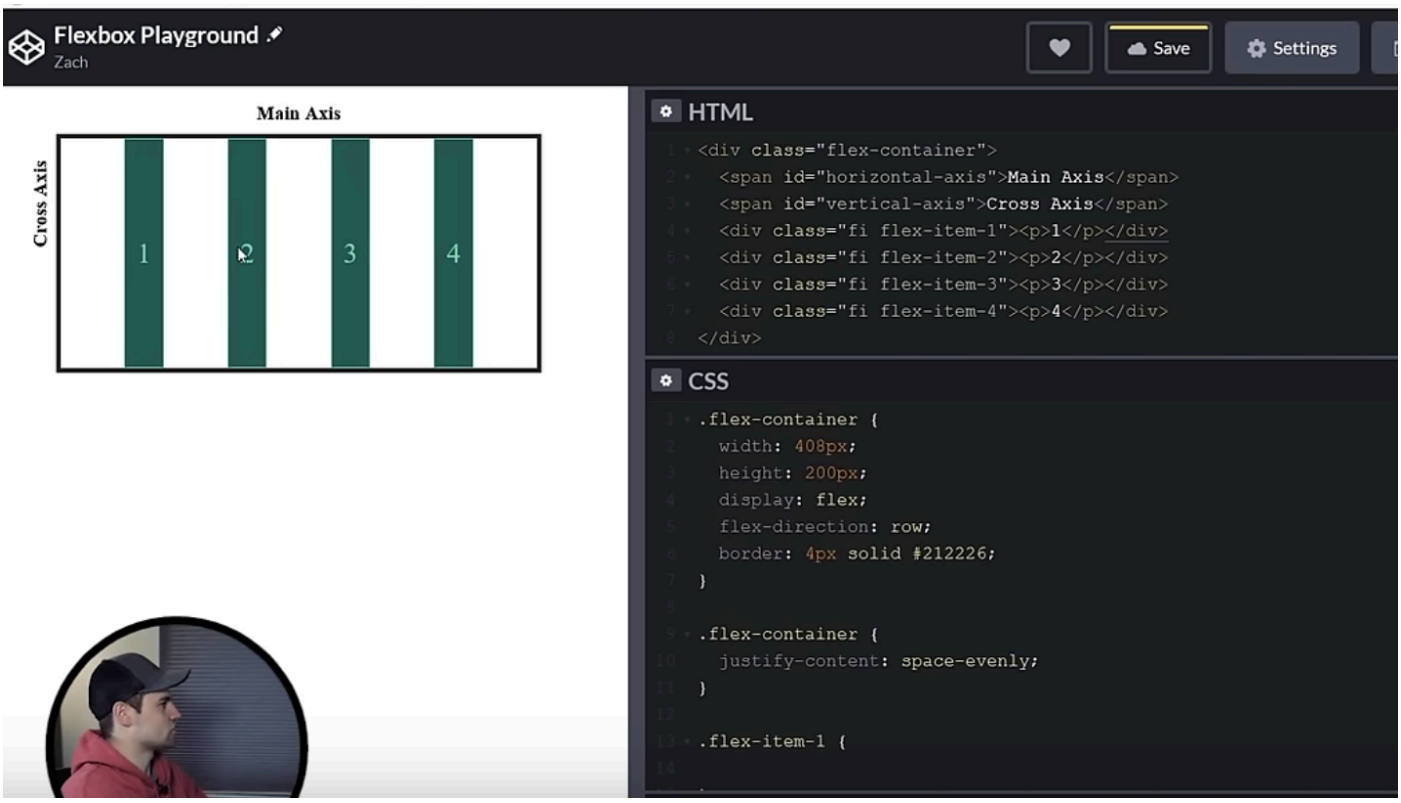
Main Axis
Cross Axis
1
2
3
4

CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
}
.flex-container {
justify-content: space-between;
}
.flex-item-1 {


Flexbox Playground
Zach
Save
Settings
Main Axis
Cross Axis
1
2
3
4
HTML

Main Axis
Cross Axis
1
2
3
4

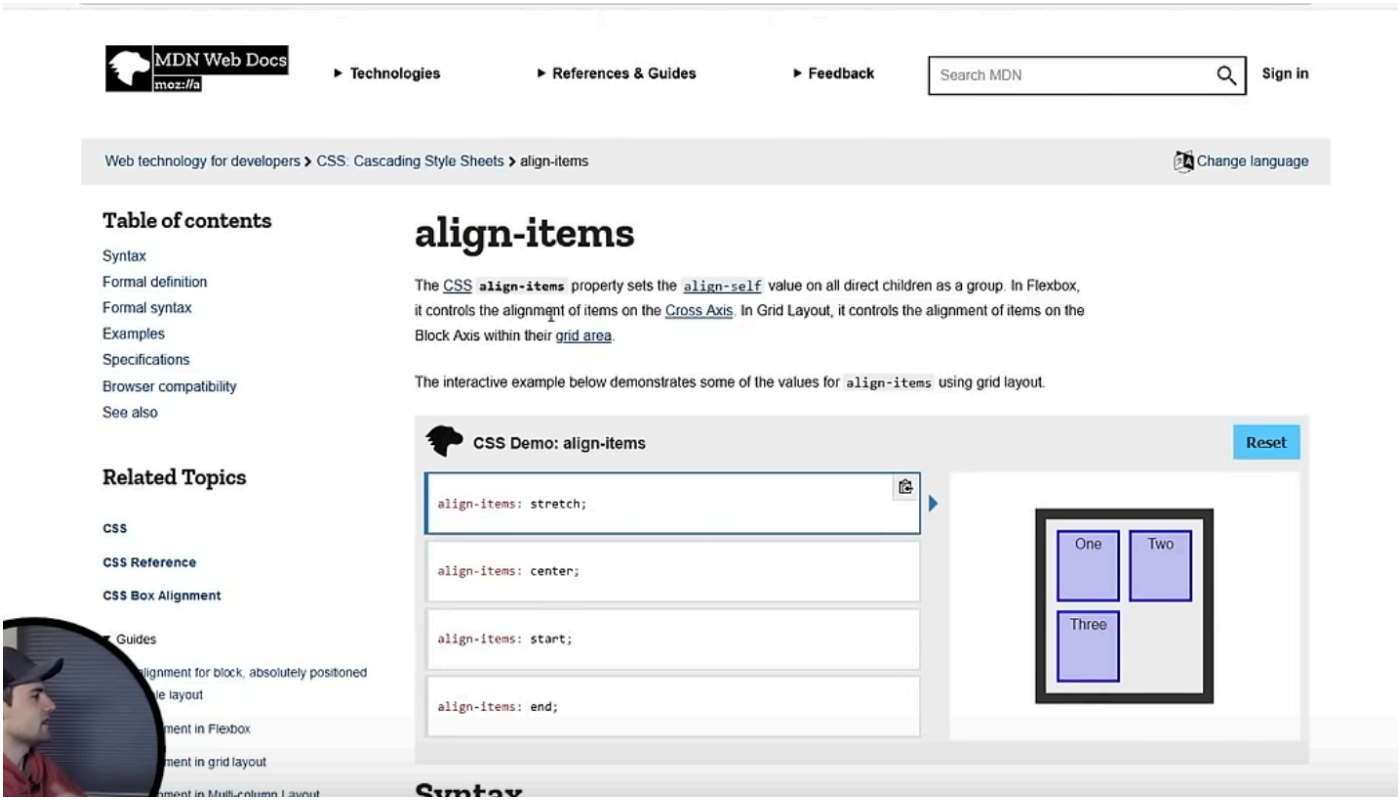
CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
}
.flex-container {
justify-content: space-around;
}
.flex-item-1 {


Flexbox Playground
Zach
Save
Settings
Main Axis
Cross Axis
1
2
3
4
HTML

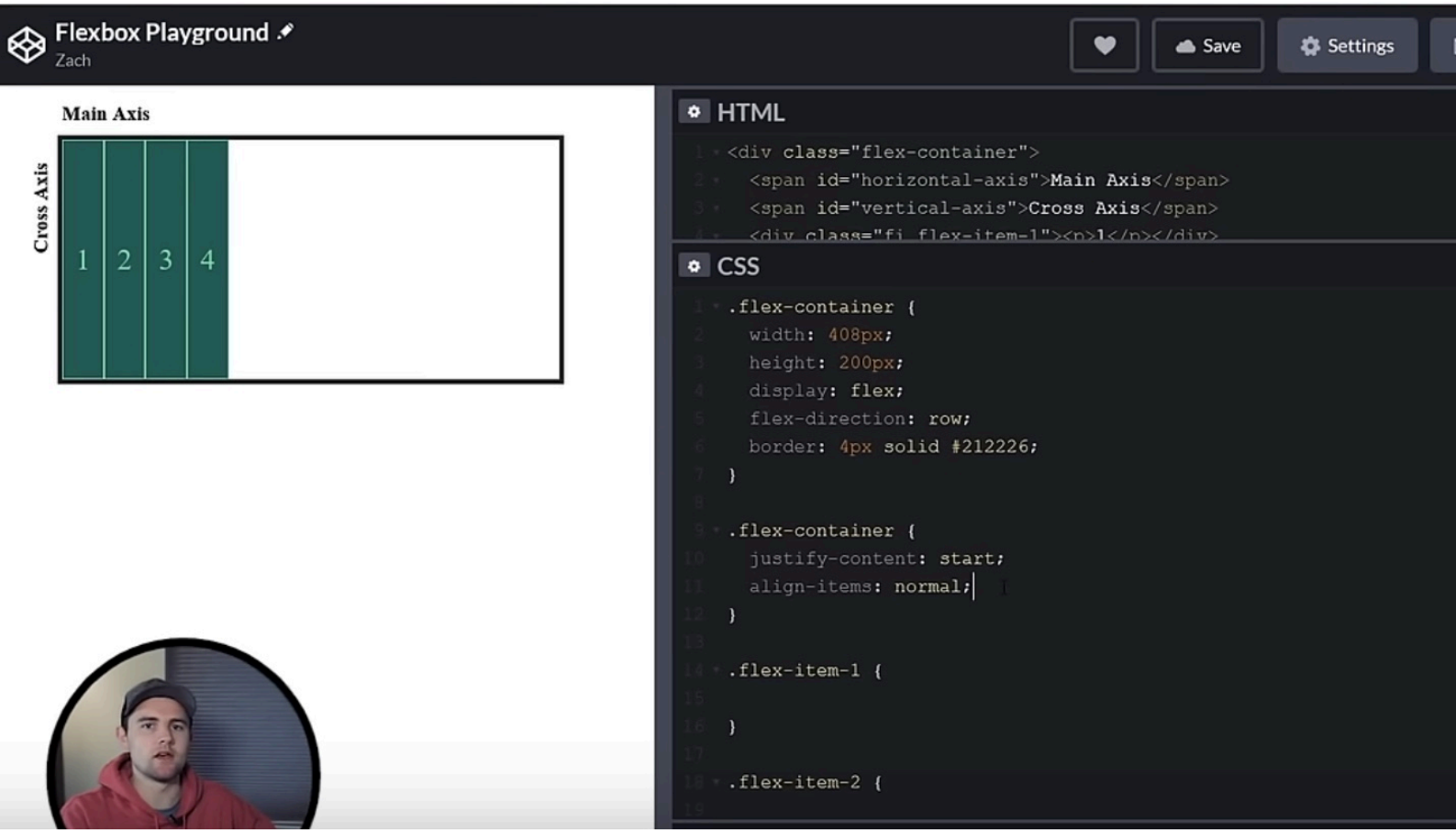
Main Axis
Cross Axis
1
2
3
4

CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
}
.flex-container {
justify-content: space-evenly;
}
.flex-item-1 {

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



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.



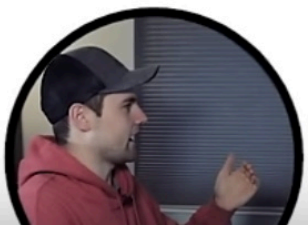
Flexbox Playground
Zach

Save Settings

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>
```

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 {
```

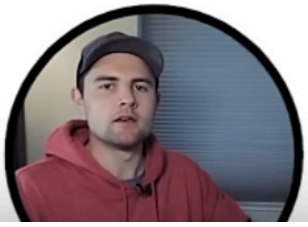
Flexbox Playground
Zach

Save | Settings

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>
```

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

Save | Settings

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

```
.fi p {
  font-size: 1.5rem;
  margin: 0;
  color: #71D99E;
}

.flex-item-1 p {
  font-size: 3rem;
}
```

Flexbox Playground

Zach

Save

Settings

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: flex-end;
}

.flex-item-1 {

}

.flex-item-2 {

}
```

Flexbox Playground

Zach

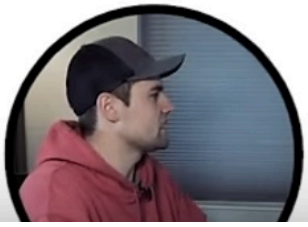
Save

Settings

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: center;
}

.flex-item-1 {

}

.flex-item-2 {

}
```

Flexbox Playground
Zach

Save Settings

Main Axis

Cross Axis

1 2 3 4

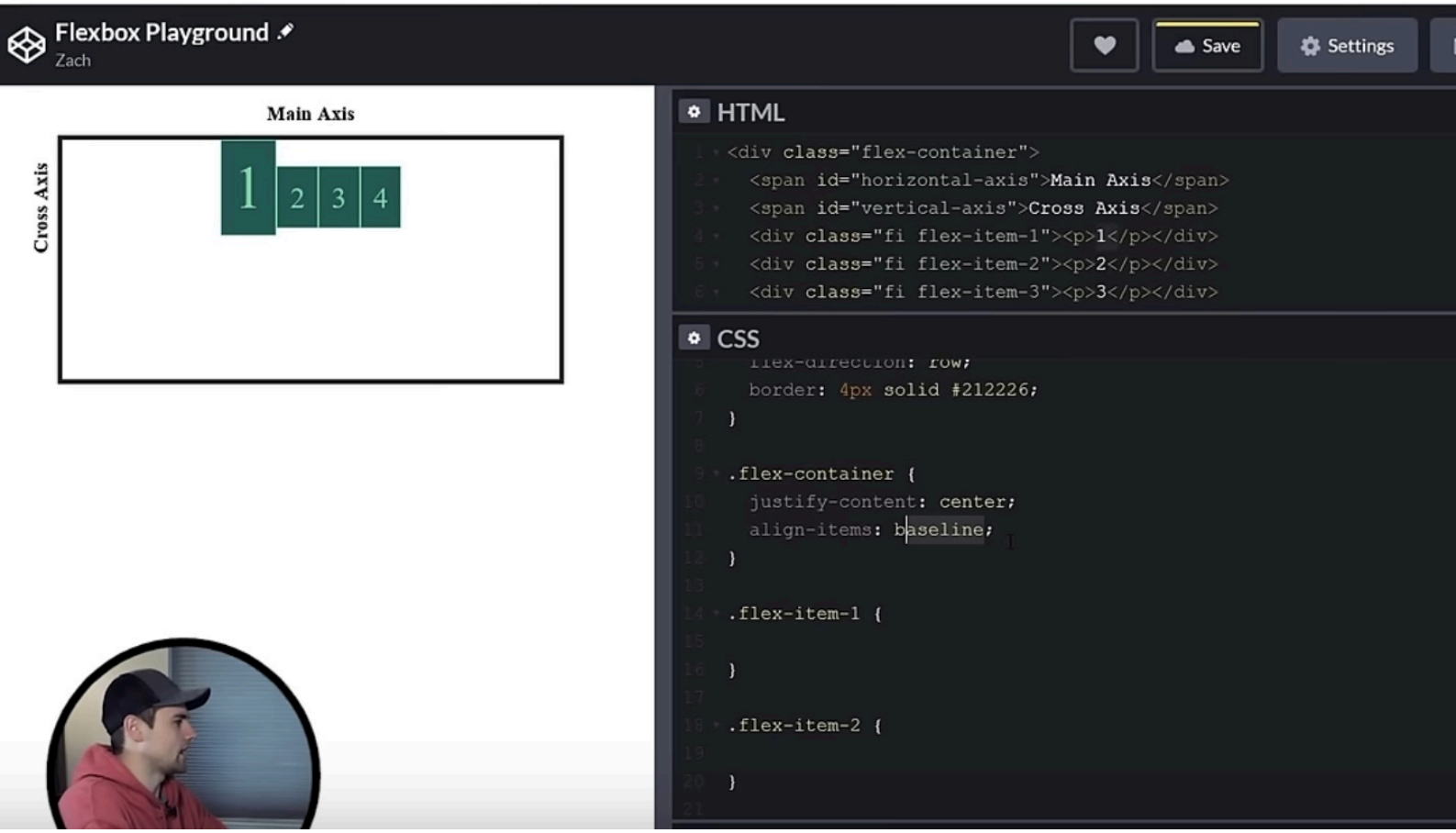
HTML

```
<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

Save Settings

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: center;
  align-items: center;
}

.flex-item-1 {

}

.flex-item-2 {

}
```

Flexbox Playground
Zach

Save Settings

Main Axis

Cross Axis

1 2 3 4

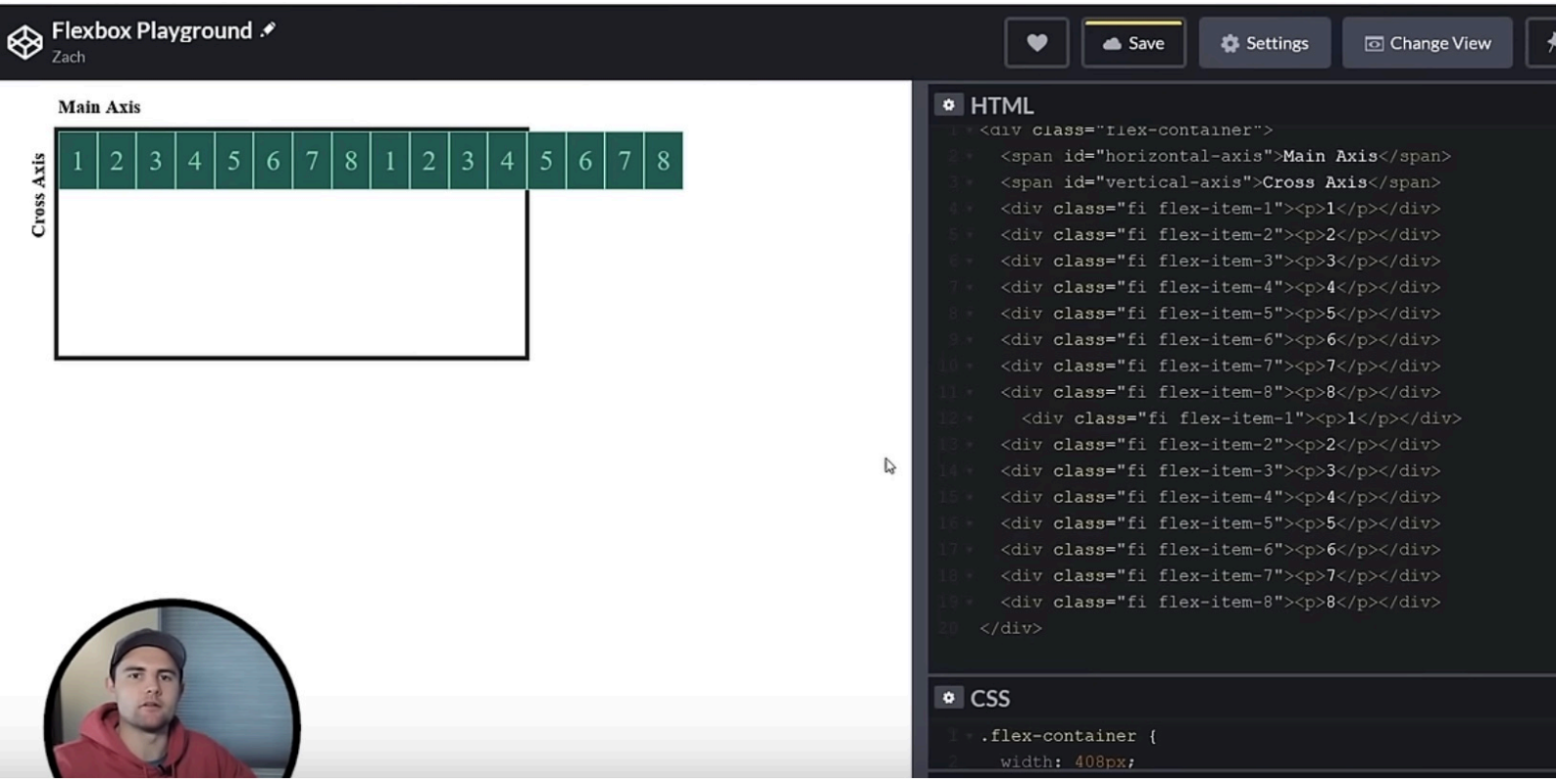
HTML

```
<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>
```

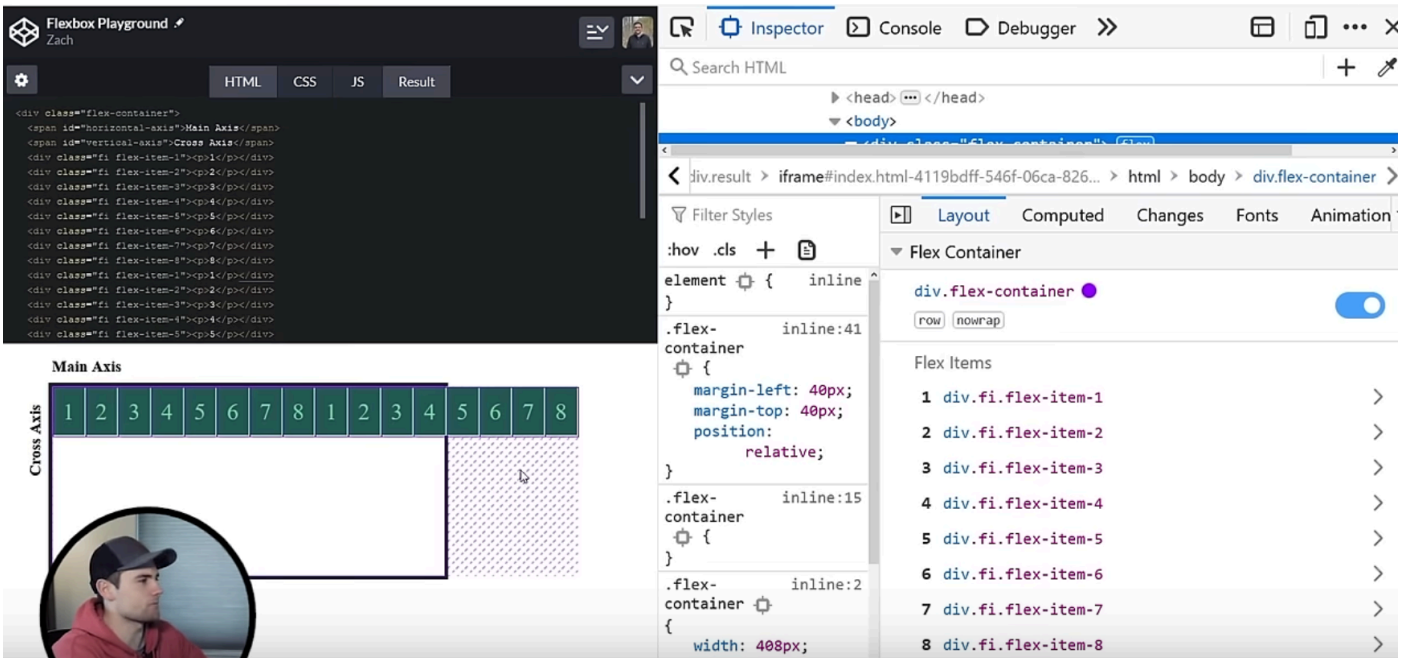
CSS

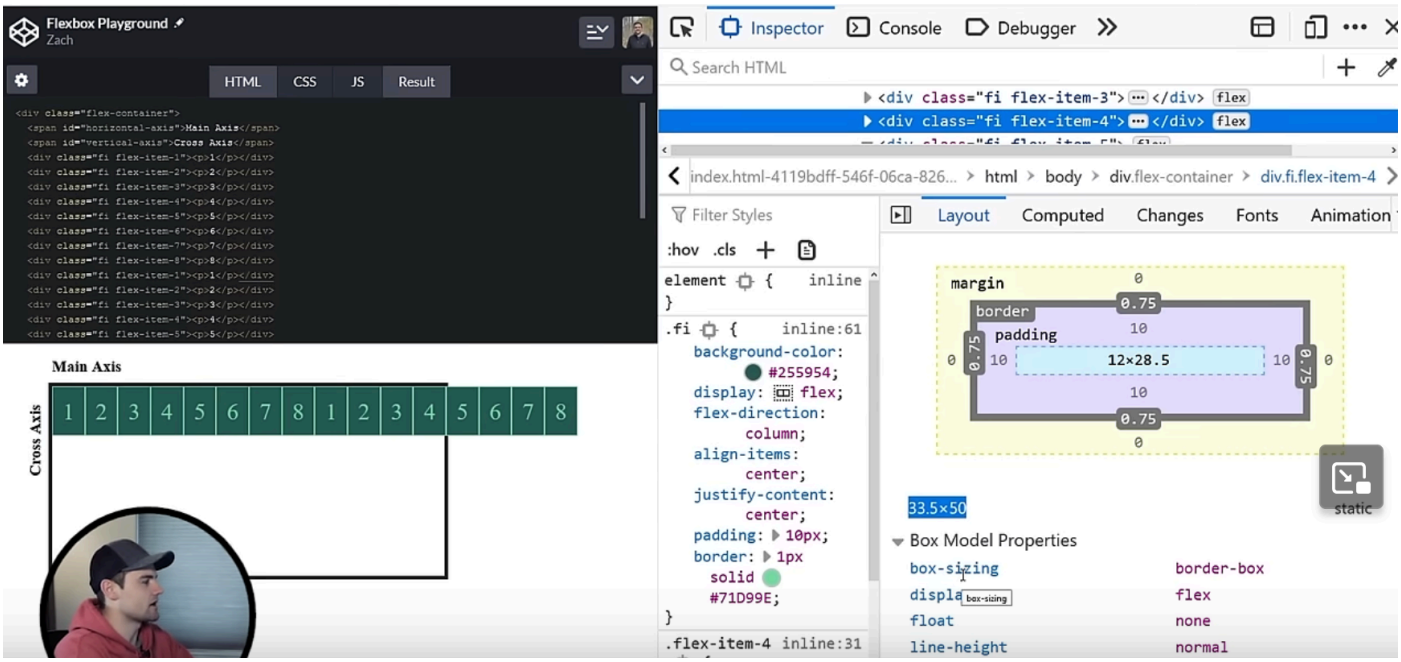
```
  flex-direction: row;
  border: 4px solid #212226;
}

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

.flex-item-1 {

}

.flex-item-2 {

}
```

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

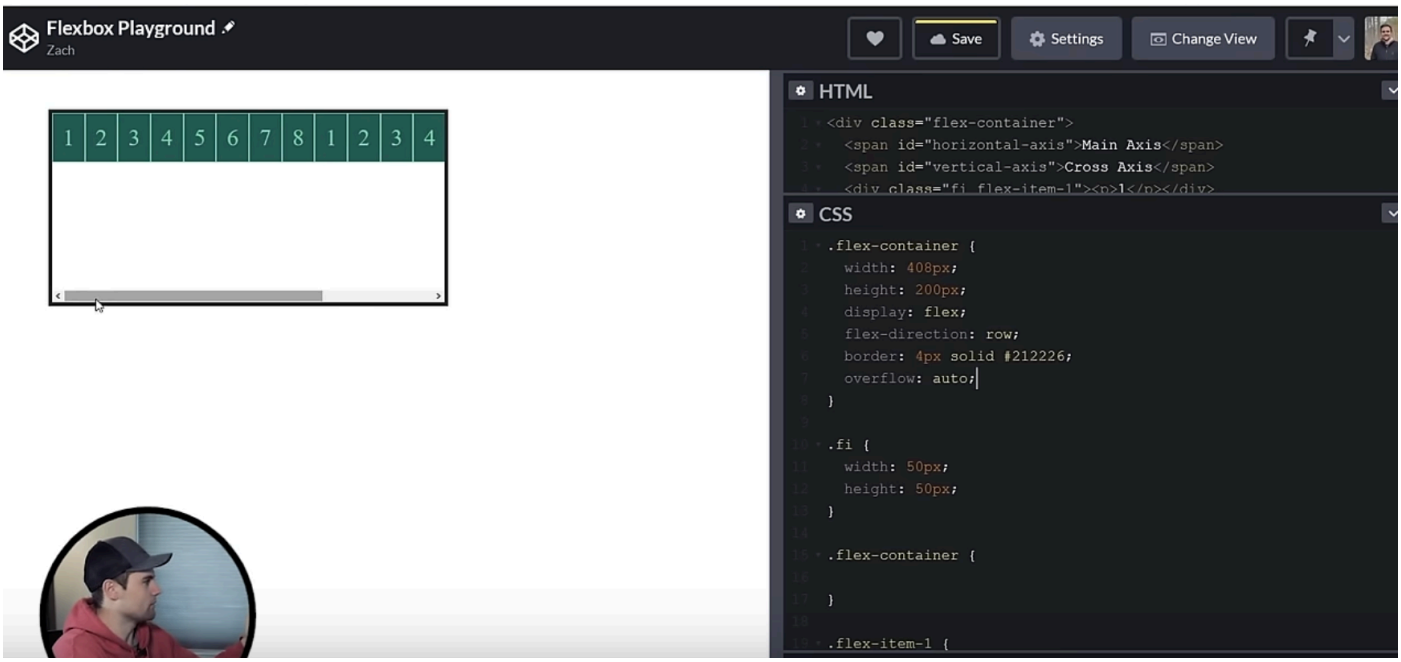


Flex items were set to width and height of 50px (container is 400px wide) clearly their size has changes - use dev tools on firefox to check in fact that the items were shrunk.

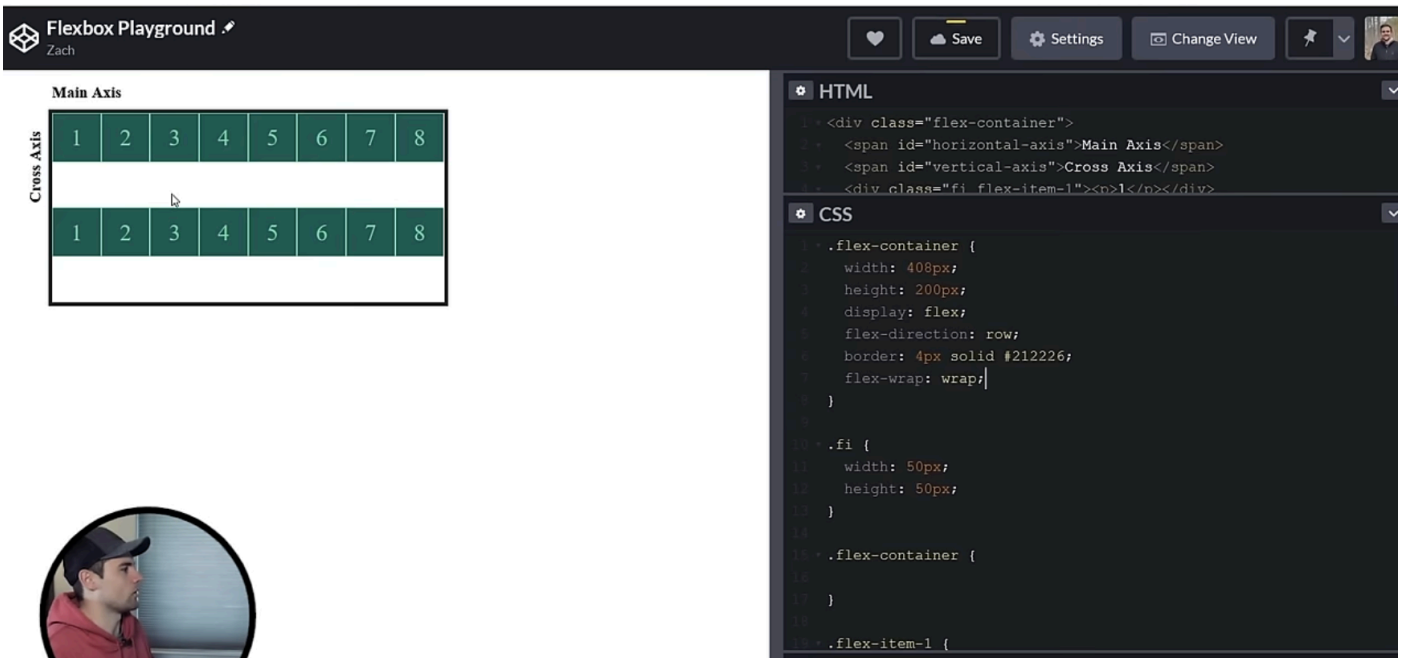



Flexbox Playground
Zach
HTML
CSS
JS
Result
Main Axis
Cross Axis
1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
Inspector
Console
Debugger
Search HTML
  flex
  flex
index.html-4119bdff-546f-06ca-826... > html > body > div.flex-container > div.fi.flex-item-4
Filter Styles
:hov .cls
element { inline }
.fi { inline:61
background-color: #255954;
display: flex;
flex-direction: column;
align-items: center;
justify-content: center;
padding: 10px;
border: 1px solid #71D99E;
}
.flex-item-4 inline:31
Layout
Computed
Changes
Fonts
Animation
margin
border
padding
12×28.5
0
0.75
10
33.5×50
static
Box Model Properties
box-sizing border-box
display flex
float none
line-height normal

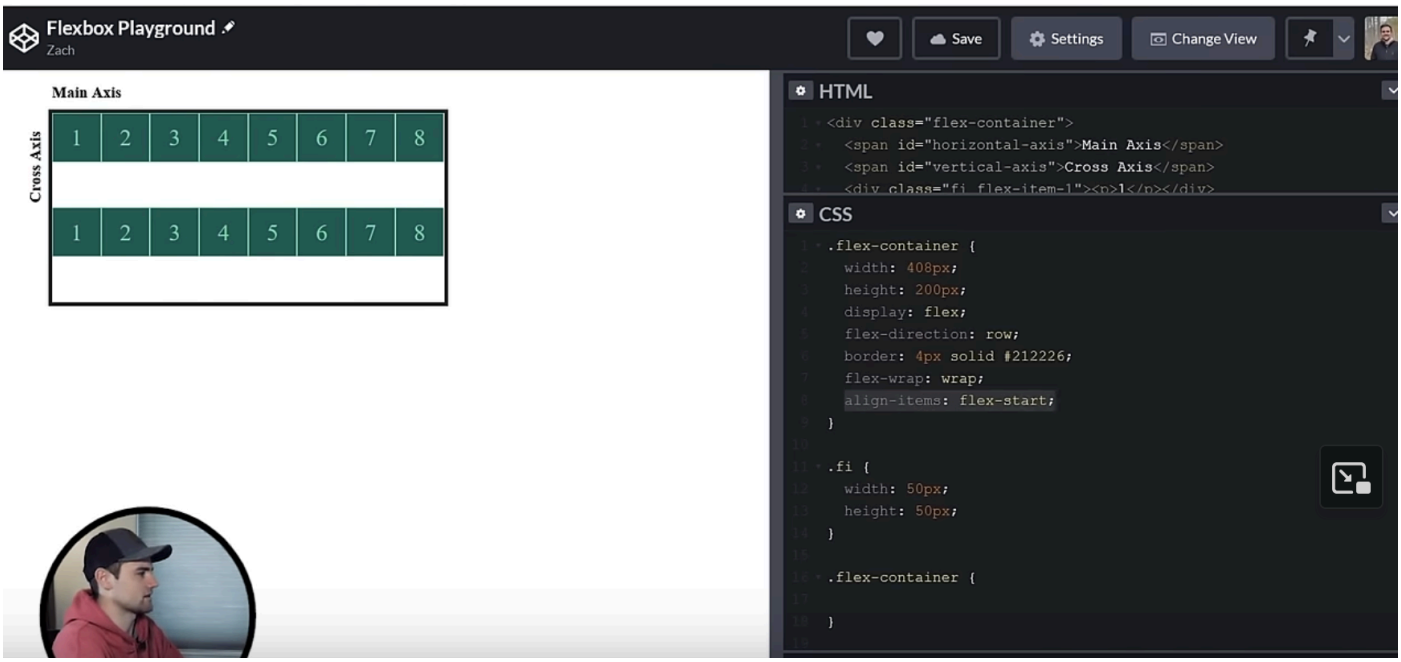
We can deal with this using the overflow property (not specific to flex box)

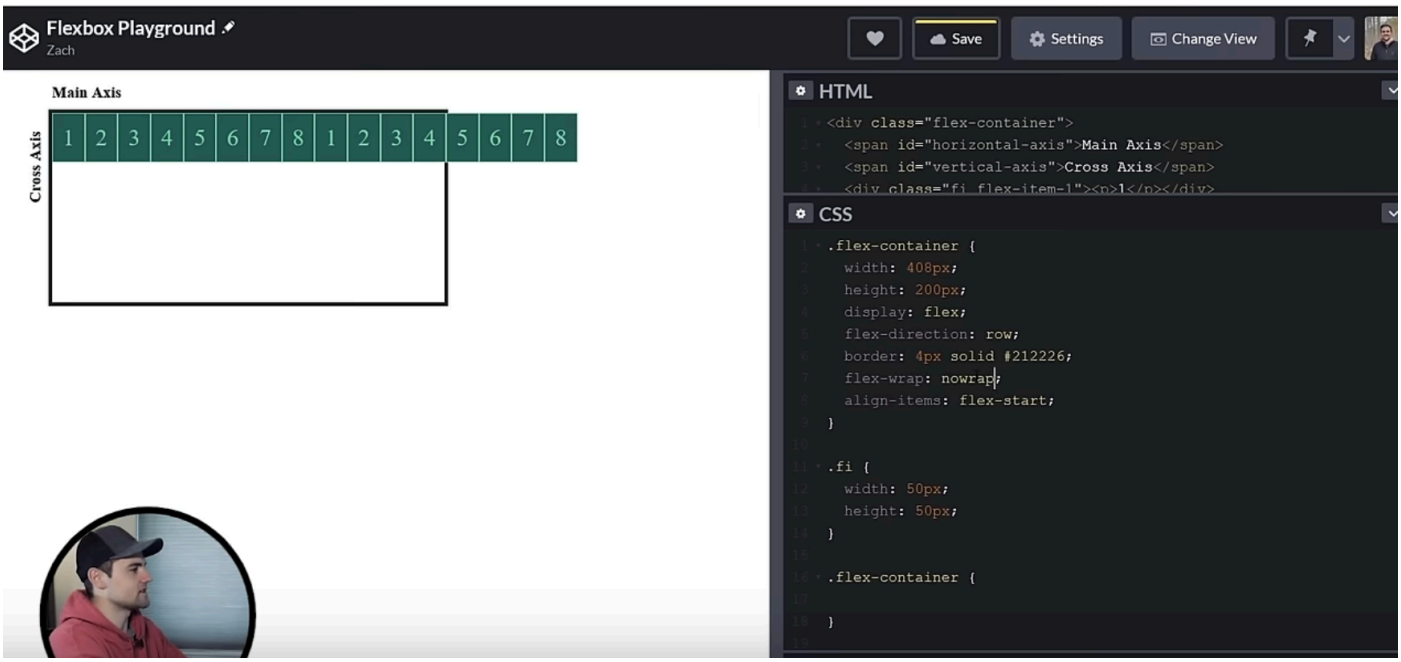


It’s better to use the flex-wrap property, since we saw with the previous one we don’t really get what we want.



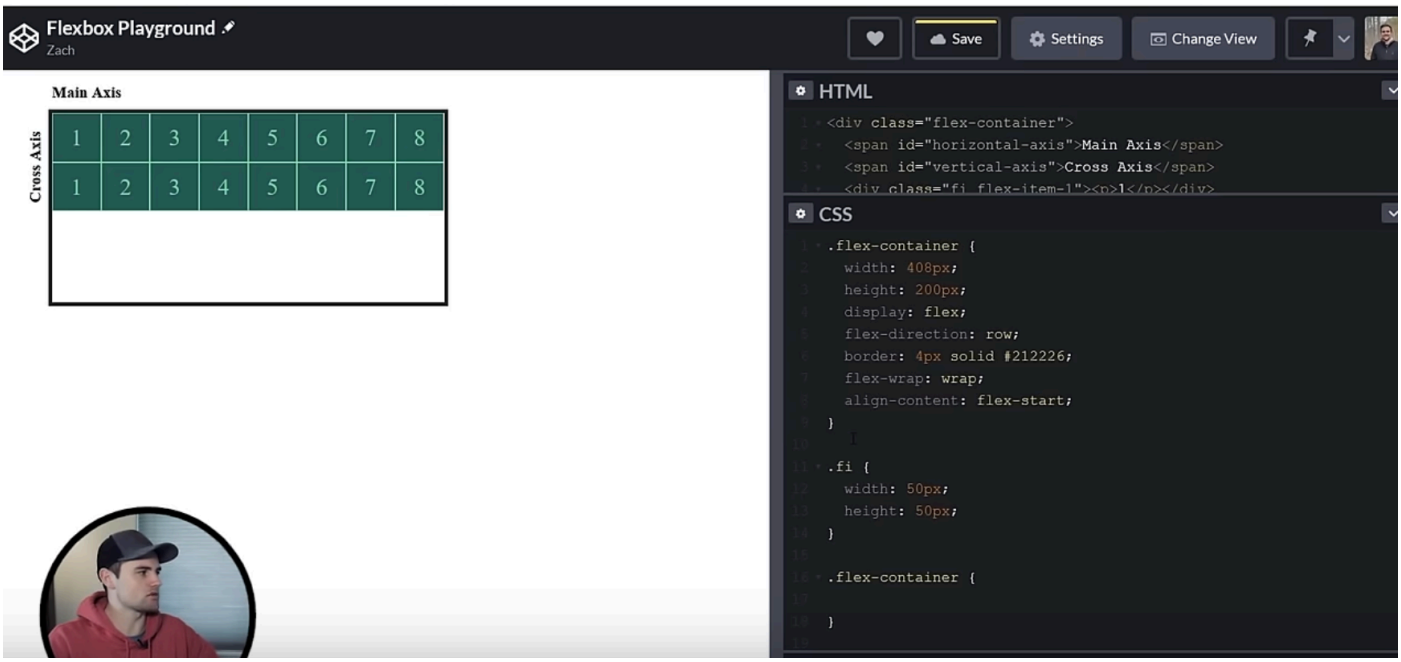
If we try to use align-items to align the second row below the first row, it won't work, since now flex-wrap is set to wrap.

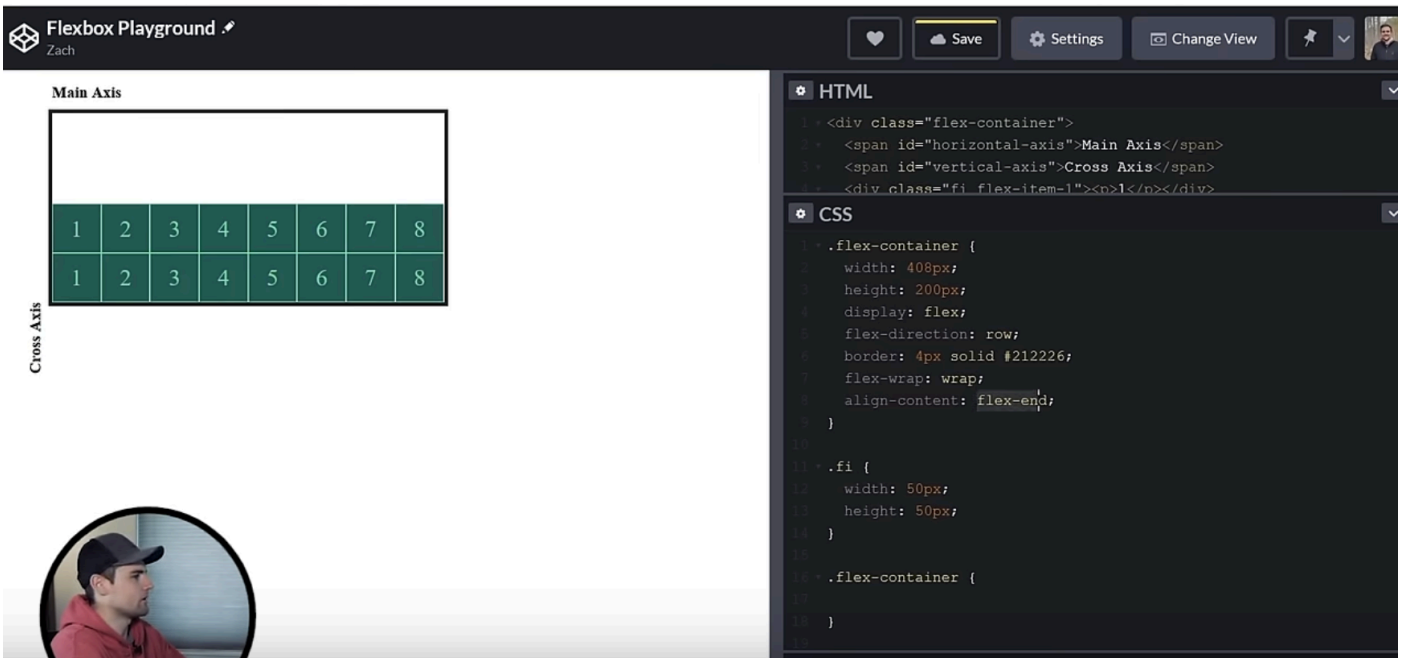



Flexbox Playground
Zach
Save
Settings
Change View
Main Axis
Cross Axis
1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
HTML

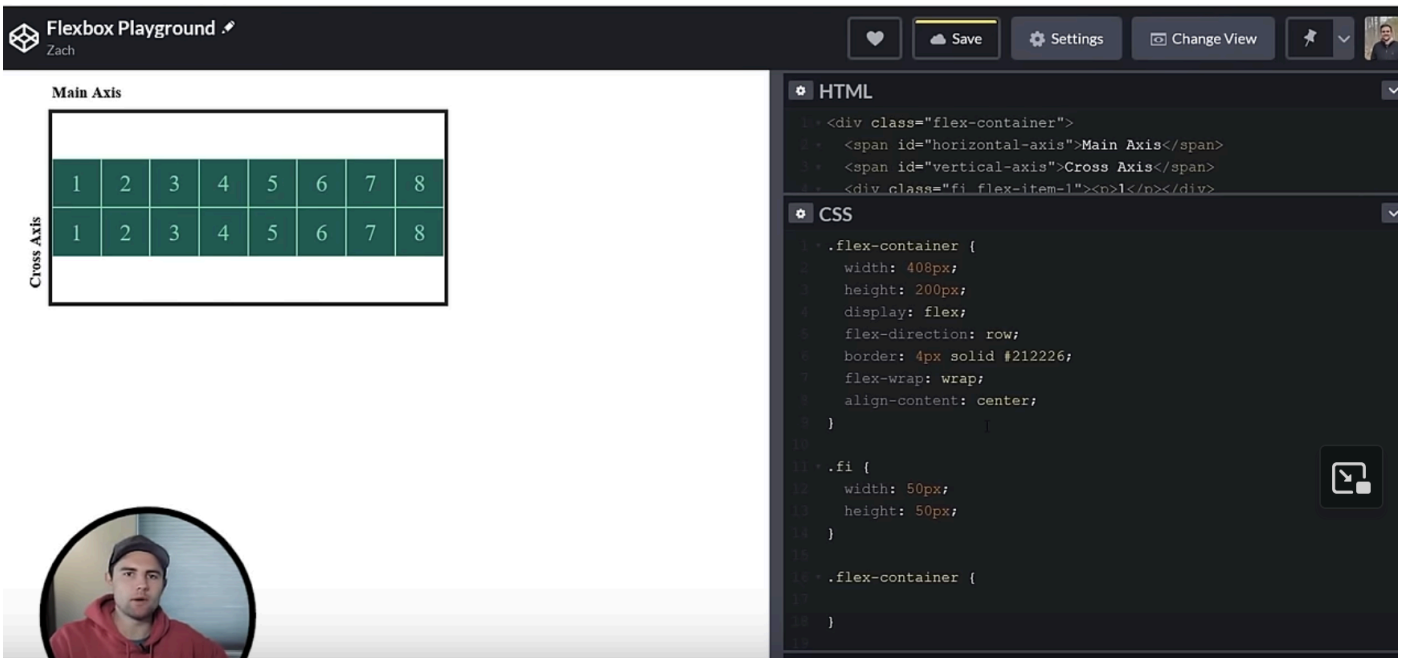
Main Axis
Cross Axis
CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
flex-wrap: nowrap;
align-items: flex-start;
}
.fi {
width: 50px;
height: 50px;
}
.flex-container {
}

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



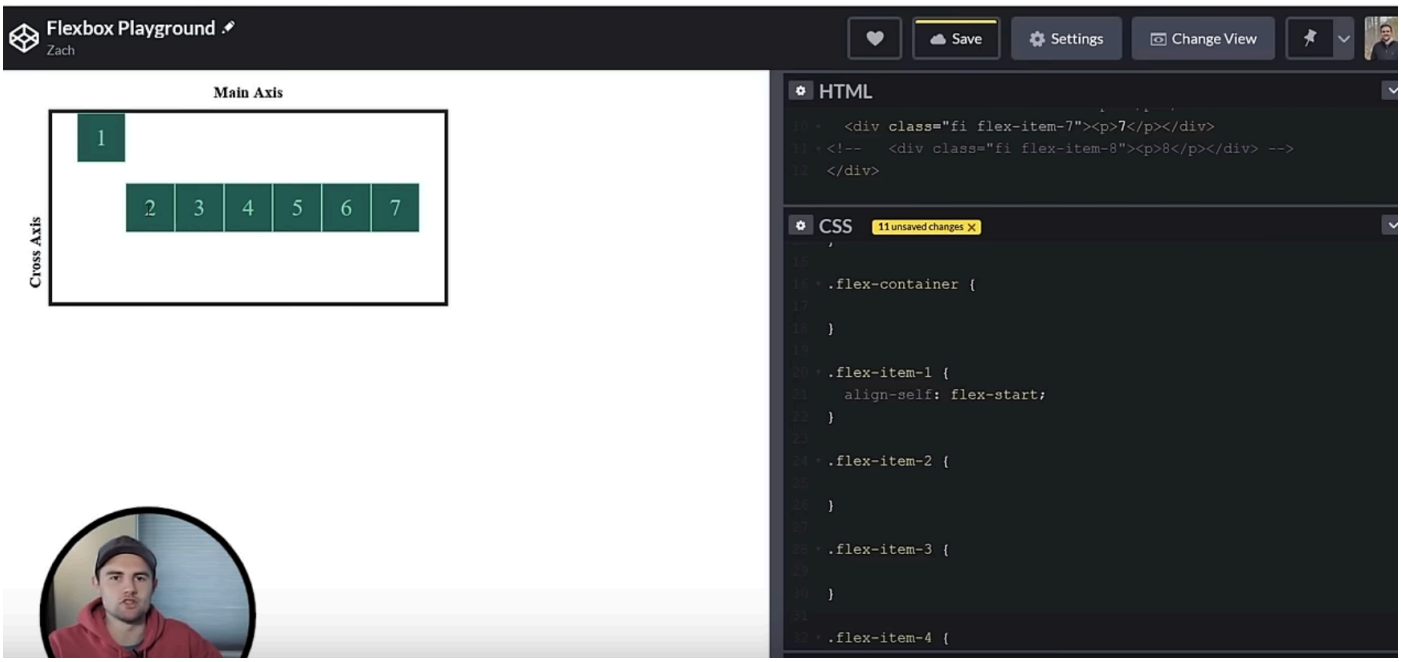

Flexbox Playground
Zach
Save
Settings
Change View
Main Axis
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8
Cross Axis
HTML

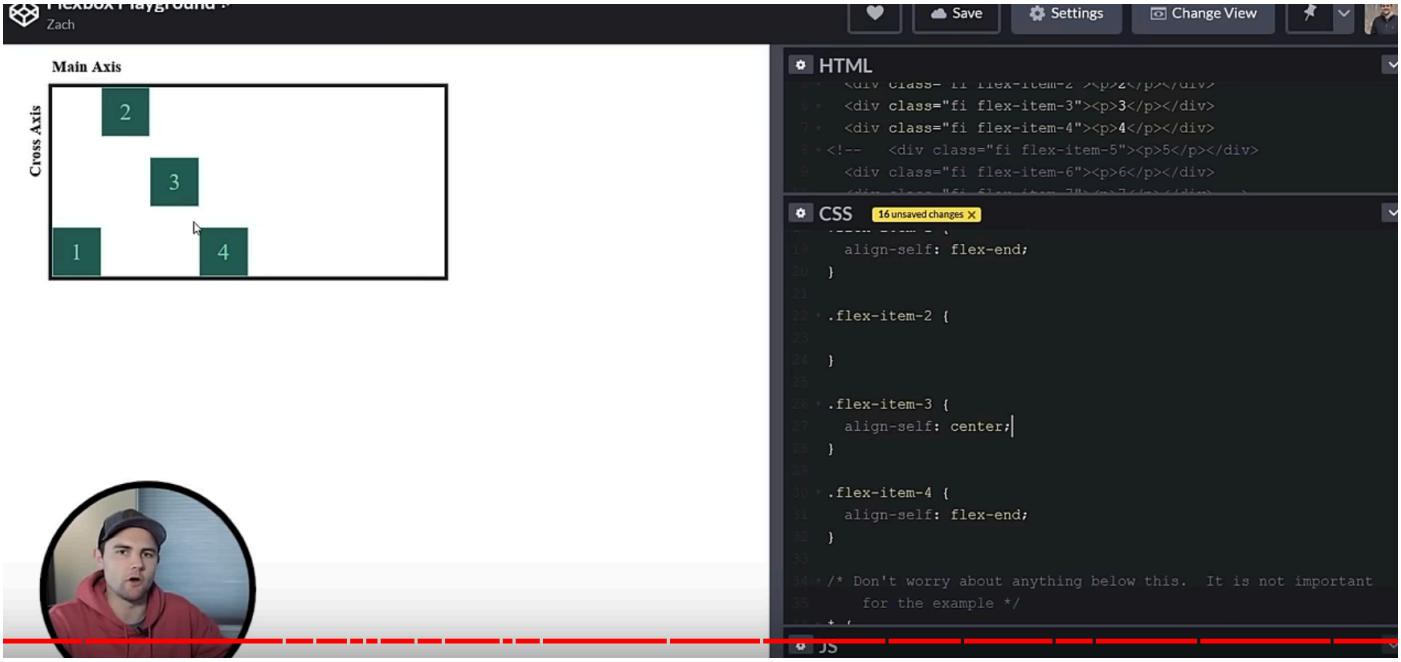
Main Axis
Cross Axis
CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
flex-wrap: wrap;
align-content: flex-end;
}
.fi {
width: 50px;
height: 50px;
}
.flex-container {
}

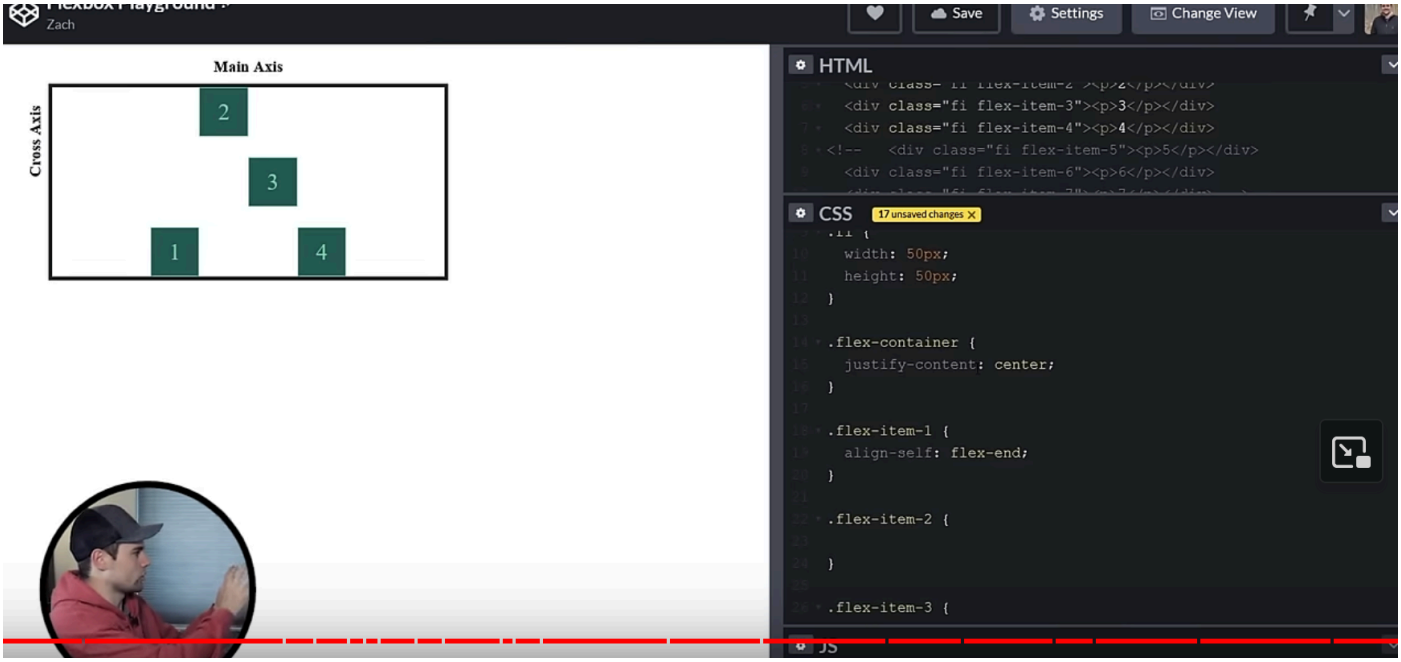

Flexbox Playground
Zach
Save
Settings
Change View
Main Axis
Cross Axis
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8
HTML

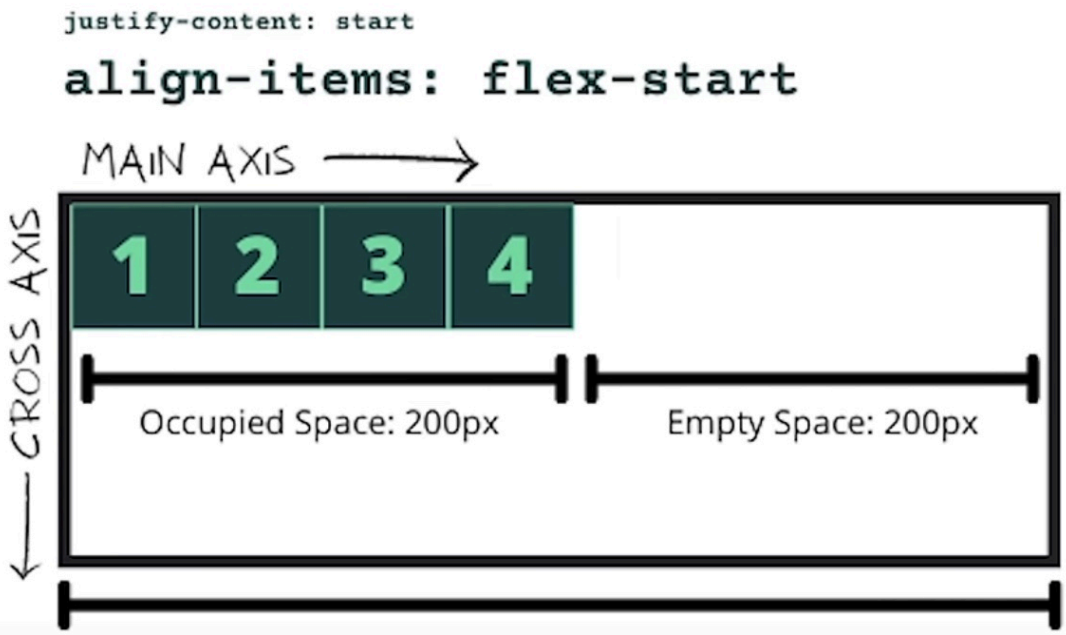
Main Axis
Cross Axis
CSS
.flex-container {
width: 408px;
height: 200px;
display: flex;
flex-direction: row;
border: 4px solid #212226;
flex-wrap: wrap;
align-content: center;
}
.fi {
width: 50px;
height: 50px;
}
.flex-container {
}

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.




Flexbox Playground
Zach
Save
Settings
Change View
Main Axis
Cross Axis
2
3
1
4
HTML
3
4
<!-- 5
6
CSS
16 unsaved changes
align-self: flex-end;
}
.flex-item-2 {
}
.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 */
JS

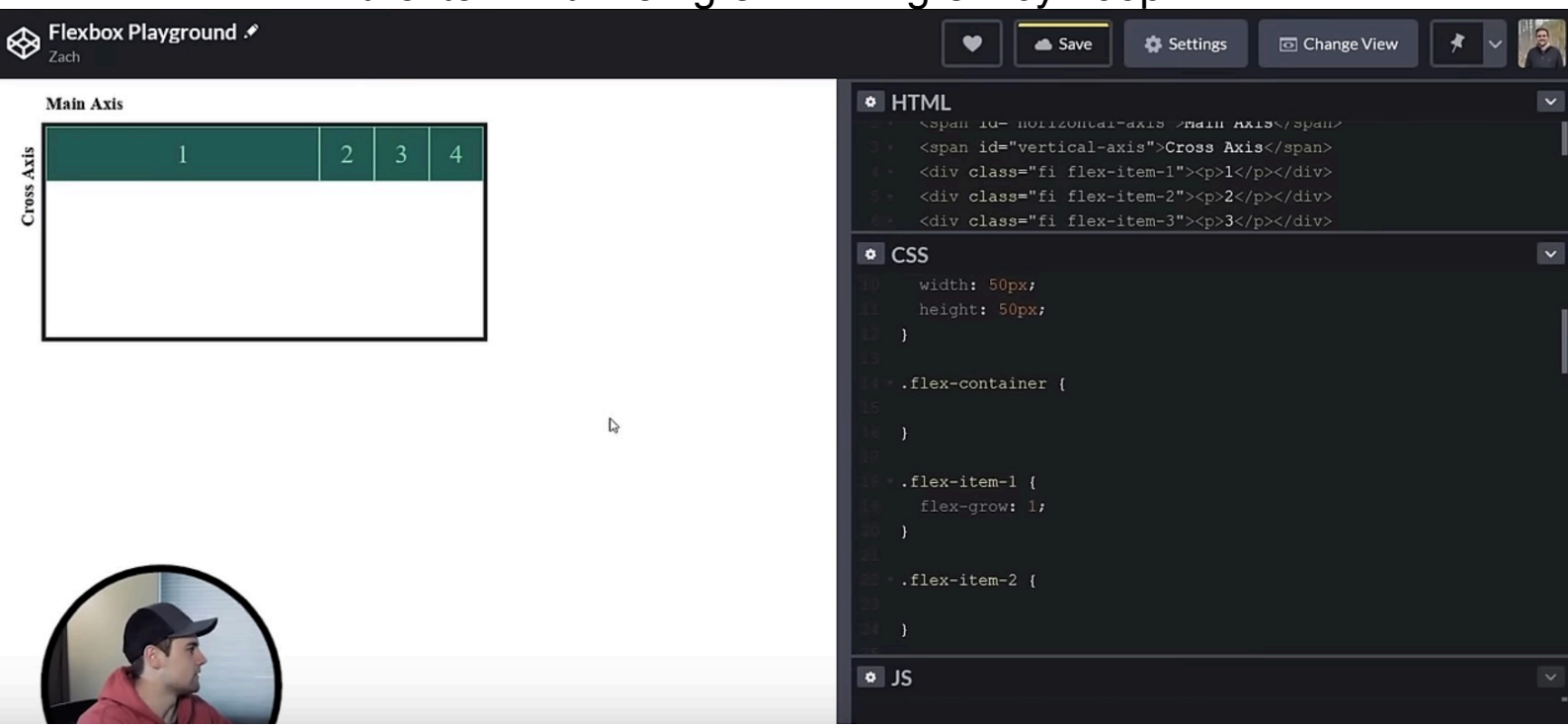

Zach
Save
Settings
Change View
Main Axis
Cross Axis
2
3
1
4
HTML
3
4
<!-- 5
6
CSS
17 unsaved changes
width: 50px;
height: 50px;
}
.flex-container {
justify-content: center;
}
.flex-item-1 {
align-self: flex-end;
}
.flex-item-2 {
}
.flex-item-3 {
JS


justify-content: start
align-items: flex-start
MAIN AXIS
CROSS AXIS
1
2
3
4
Occupied Space: 200px
Empty Space: 200px
Total Space: 400px

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-1for 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.



Flexbox Playground

Zach

Save | Settings | Change View

Main Axis

Cross Axis

1 2 3 4

HTML

```
<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 6 unsaved changes

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

.flex-container {

}

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

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

Flexbox Playground

Zach

Save | Settings | Change View

Main Axis

Cross Axis

1 2 3 4

HTML

```
<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 33 unsaved changes

```
*/

.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

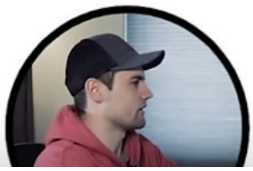
Zach

Save | Settings | Change View

Main Axis

Cross Axis

1 2 3 4



HTML

```
<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 34 unsaved changes

```
*/

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

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

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

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

Flexbox Playground

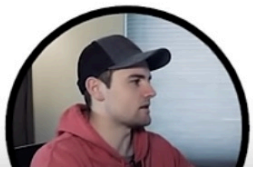
Zach

Save | Settings | Change View

Main Axis

Cross Axis

1 2 3 4



HTML

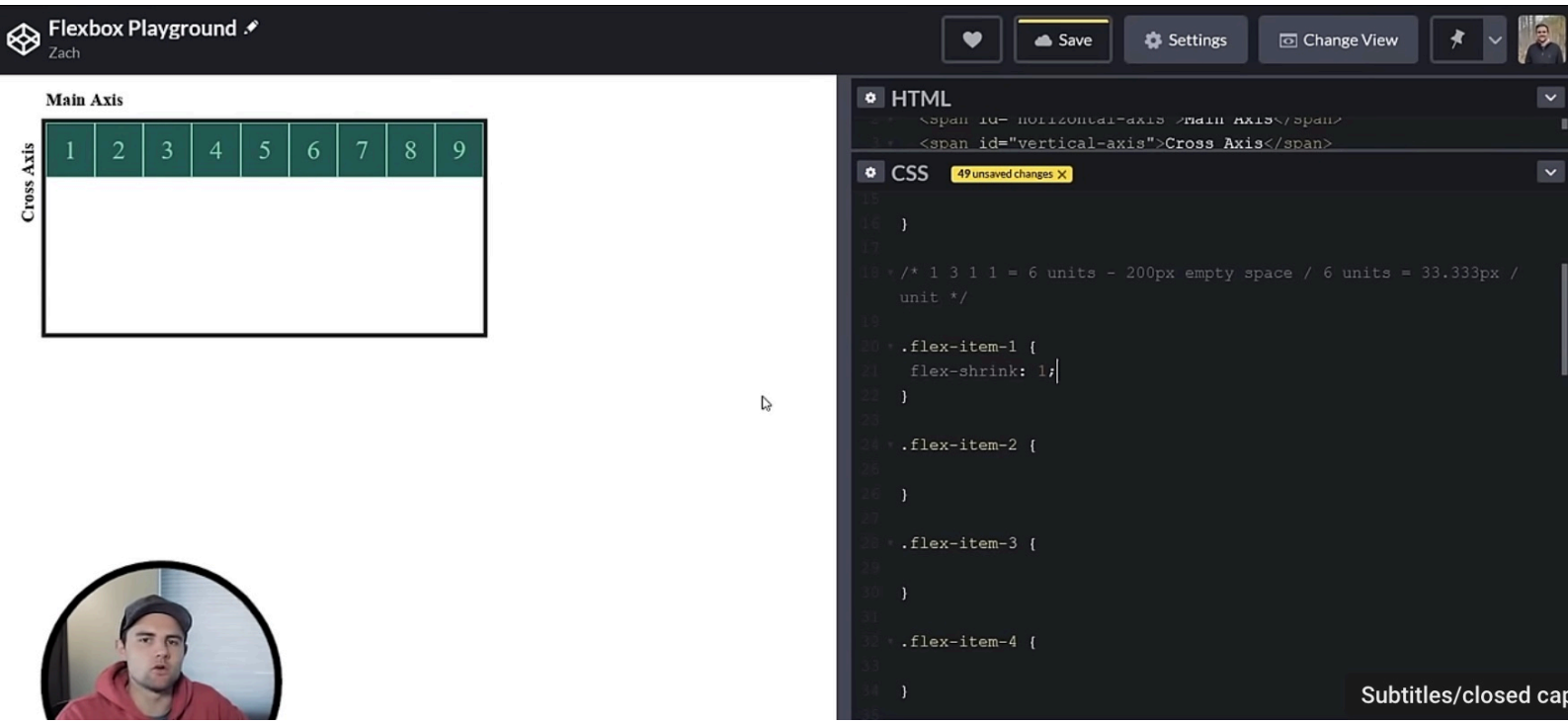
```
<span id="horizontal-axis">Main Axis</span>
<span id="vertical-axis">Cross Axis</span>
<div class="fi flex-item-1"><p>1</p></div>
<div class="fi flex-item-2"><p>2</p></div>
<div class="fi flex-item-3"><p>3</p></div>
```

CSS 39 unsaved changes

```
.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.



Flexbox Playground

Zach

Save | Settings | Change View

Main Axis

Cross Axis

1	2	3	4	5	6	7	8	9

HTML

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

CSS

56 unsaved changes

```
}

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

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

.flex-item-2 {

}

.flex-item-3 {

}

.flex-item-4 {

}

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

Flexbox Playground

Zach

Save | Settings | Change View

Main Axis

Cross Axis

1 2 3 4 5 6 7 8 9 9 9 9 9

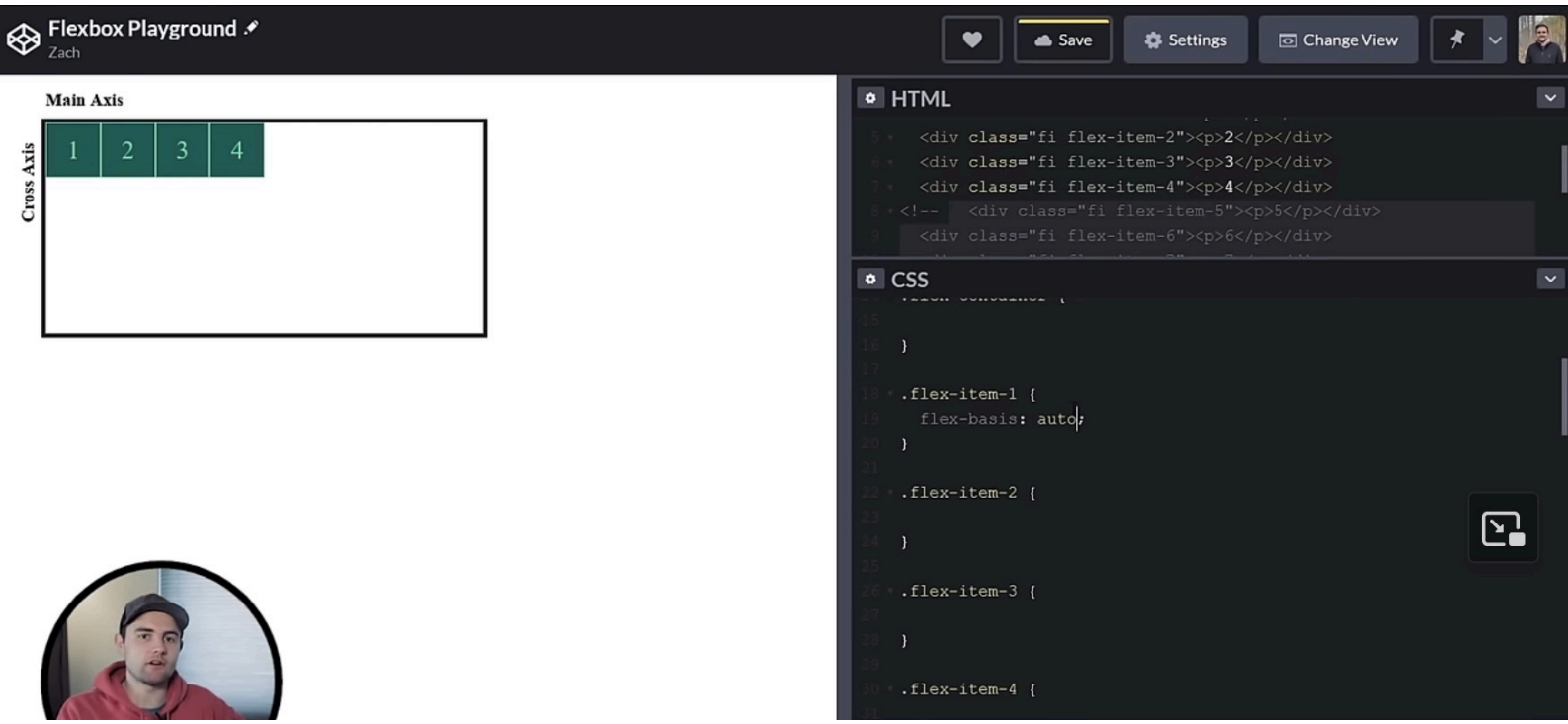
HTML

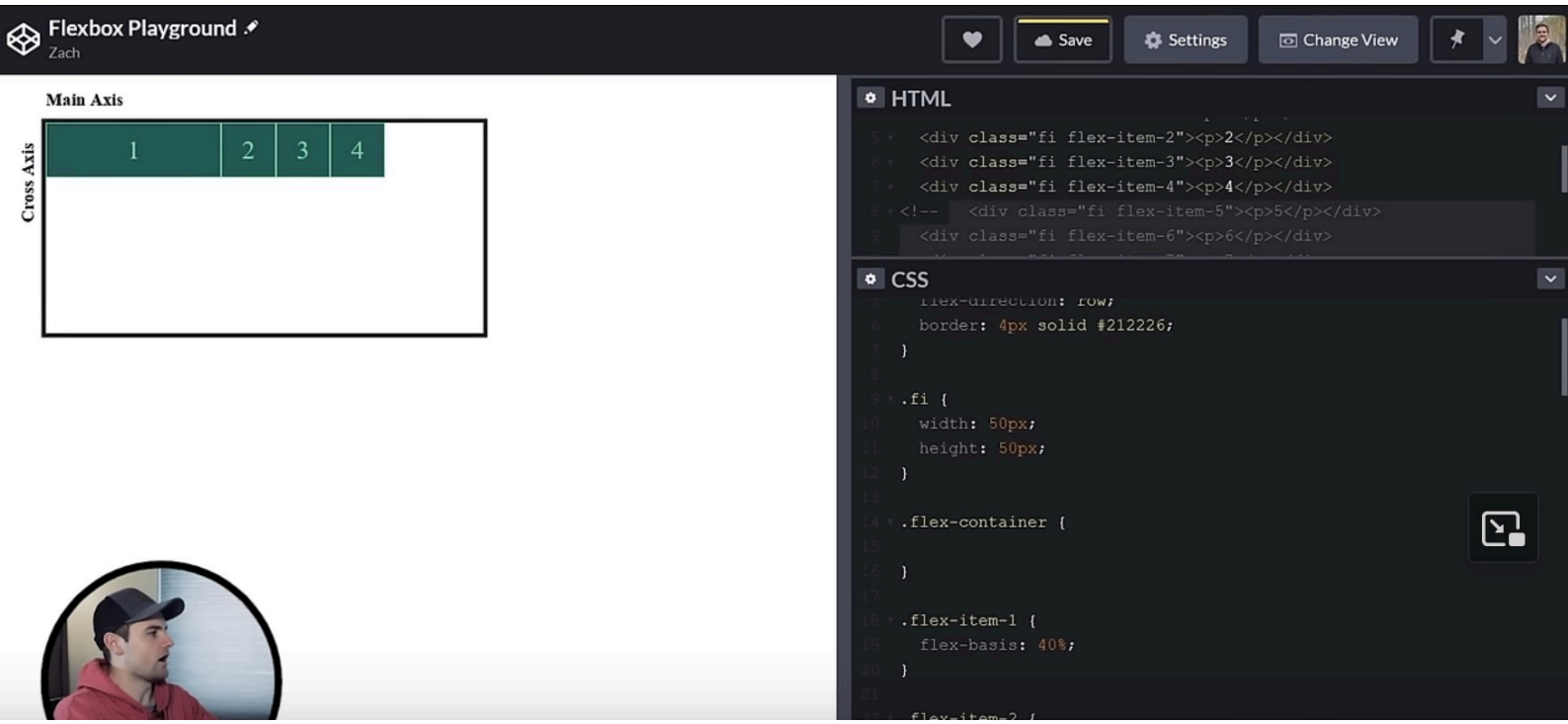
```
<div class="fi flex-item-2"><p>2</p></div>
<div class="fi flex-item-3"><p>3</p></div>
<div class="fi flex-item-4"><p>4</p></div>
<div class="fi flex-item-5"><p>5</p></div>
<div class="fi flex-item-6"><p>6</p></div>
<div class="fi flex-item-7"><p>7</p></div>
<div class="fi flex-item-8"><p>8</p></div>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




Flexbox Playground
Zach
Save
Settings
Change View
Main Axis
Cross Axis
1
2
3
4
HTML
2
3
4
<!-- 5
6
CSS
flex-direction: row;
border: 4px solid #212226;
}
.fi {
width: 50px;
height: 50px;
}
.flex-container {
}
.flex-item-1 {
flex-basis: 40%;
}

Flexbox Playground

Zach

Save | Settings | Change View

Main Axis

Cross Axis

1 2 3 4

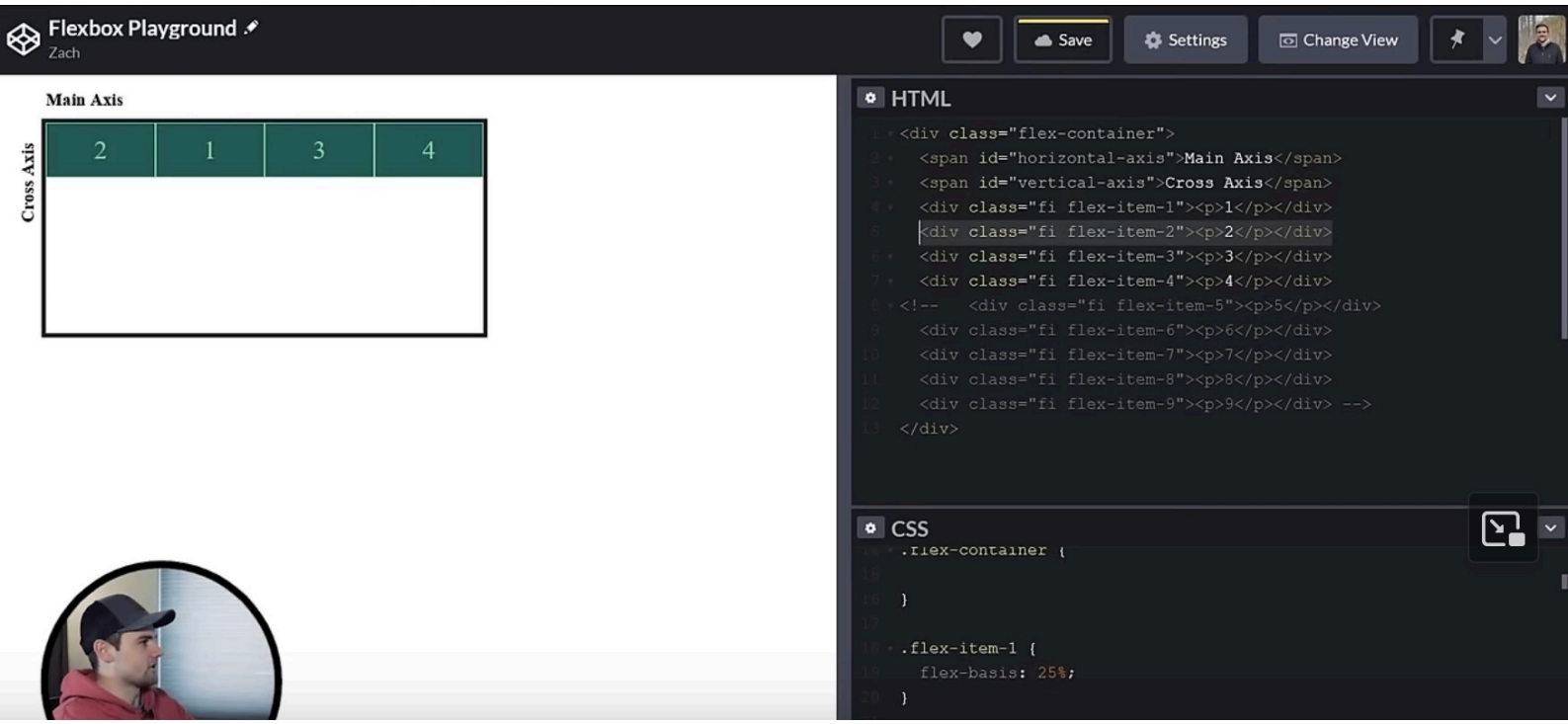
HTML

```
<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>
  <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>
```

CSS

```
.flex-container {

}

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

Reordering


Flexbox Playground
Zach
Save
Settings
Change View
Main Axis
Cross Axis
2
1
3
4
HTML

Main Axis
Cross Axis
1
2
3
4
<!-- 5
6
7
8
9 -->

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

Flexbox Playground

Zach

Save | Settings | Change View

Main Axis

Cross Axis

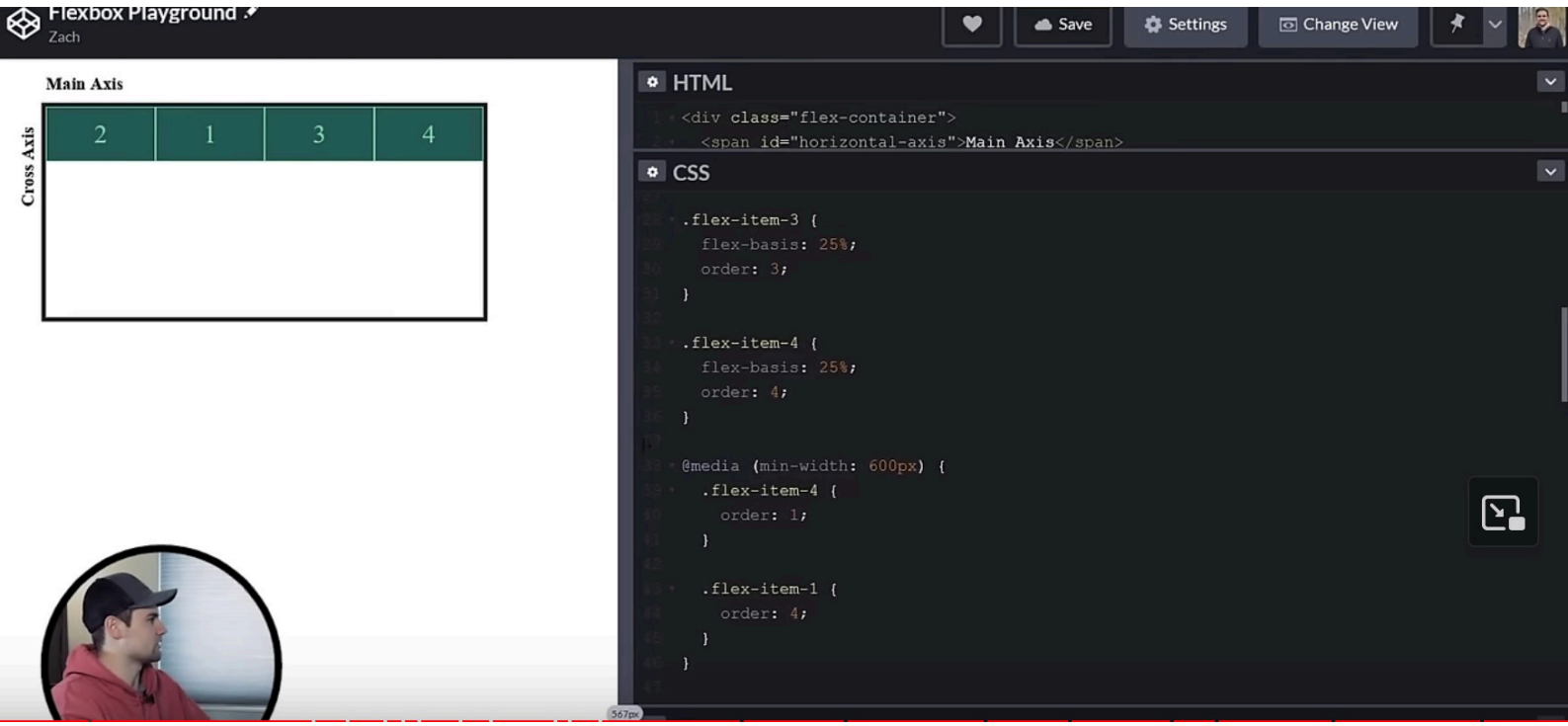
2 1 3 4

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
Save
Settings
Change View
Main Axis
Cross Axis
2
1
3
4
HTML

Main Axis
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;
}
}
567px

Flexbox Playground
Zach

Save | Settings | Change View

Main Axis

Cross Axis

2 4 3 1

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

```
.flex-item-1 {
  flex-basis: 25%;
  order: 2;
  flex: 1 1 auto;
}
```