# Refactoring *(the way we talk about)* CSS.

Rachel Andrew @ Nordic.js

# Doing things on the web since 1996

Co-founder **Perch CMS** & **Notist**. Editor in Chief **Smashing Magazine**. Writer of many books. **CSS Working Group** Member representing **Fronteers**. Spec editor **Multicol** and Page Floats. **MDN** tech writer.

I thought that I had teaching CSS layout all figured out

# Talking about CSS as a layout system

CSS can be explained in the same structured way we use for other languages.

- Flow Layout
- Changing the value of display
- Out of flow elements
- Formatting Contexts
- Writing Modes
- Logical, flow-relative properties and values
- Alignment
- Sizing
- Media & Feature Queries

# Understanding display

# Normal Flow

Block and Inline Layout

# Just some HTML and content

Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

Turnip greens yarrow ricebean rutabaga endive cauliflower sea lettuce kohlrabi amaranth water spinach avocado daikon napa cabbage asparagus winter purslane kale.

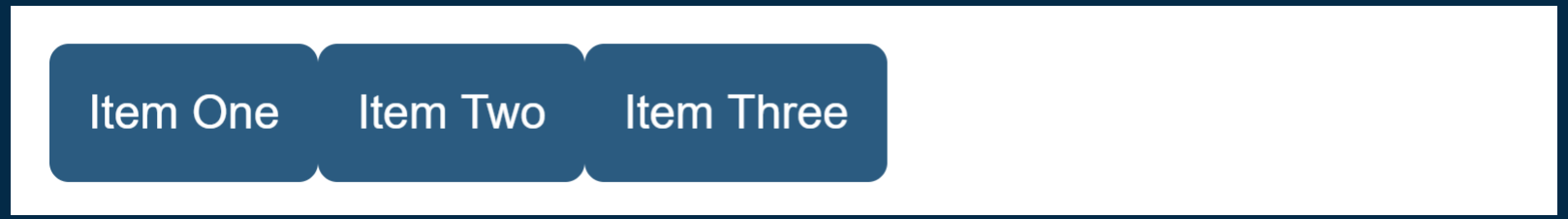
CSS is doing work for us,
before we write any CSS.

Item One

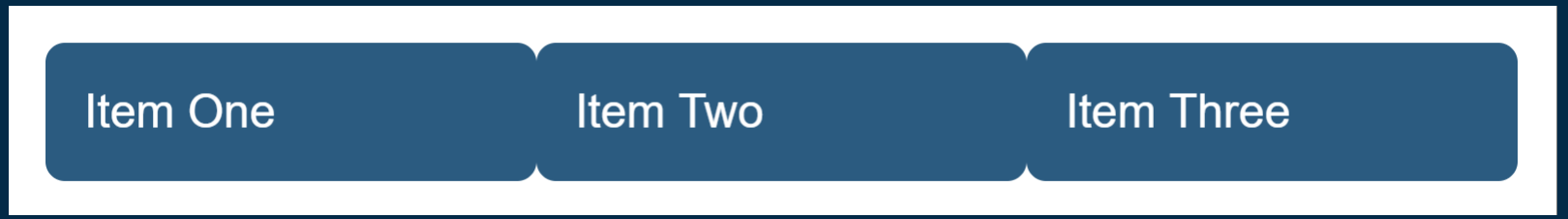
Item Two

Item Three

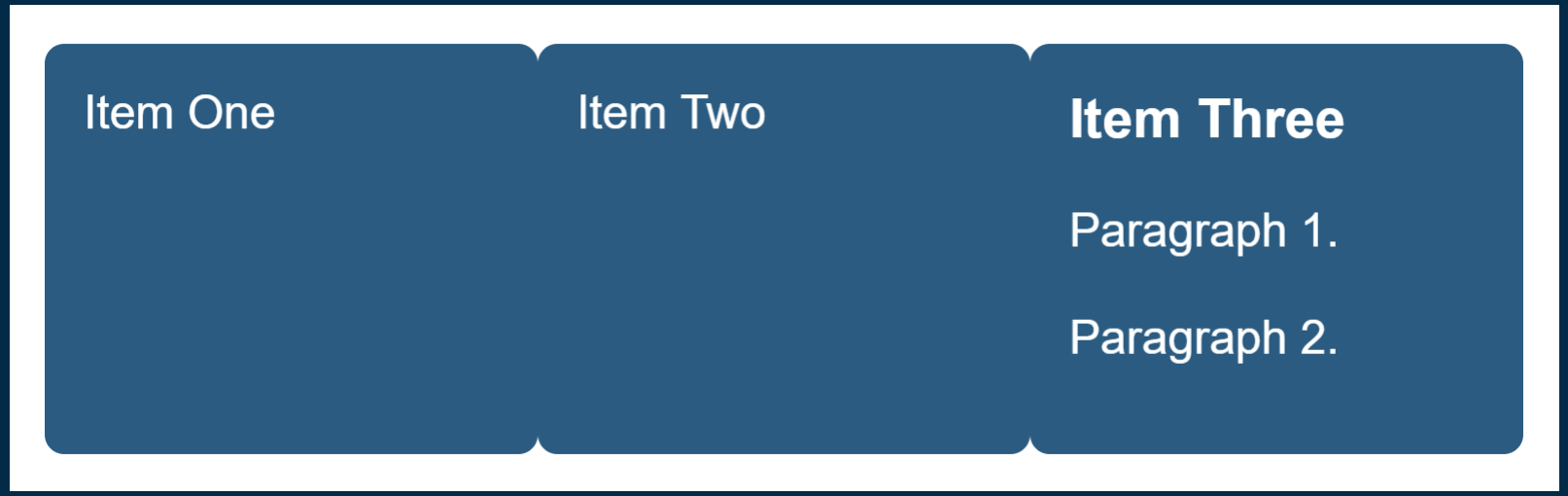
```
.example {
  display: flex;
}
```



```
.example {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
}
```

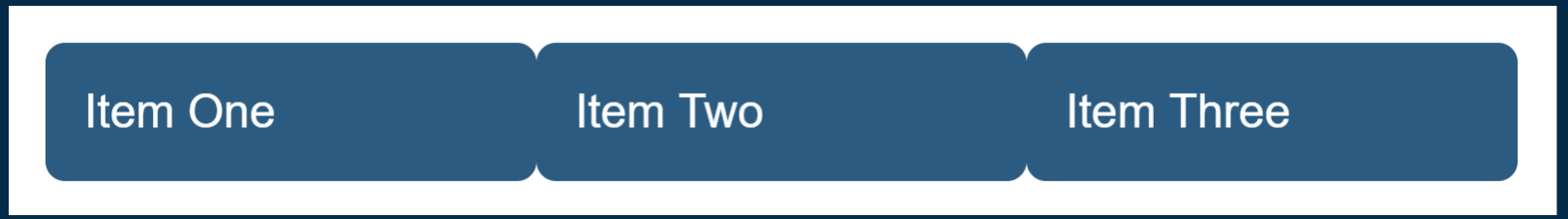


Changing the value of display changes that element and its direct children.

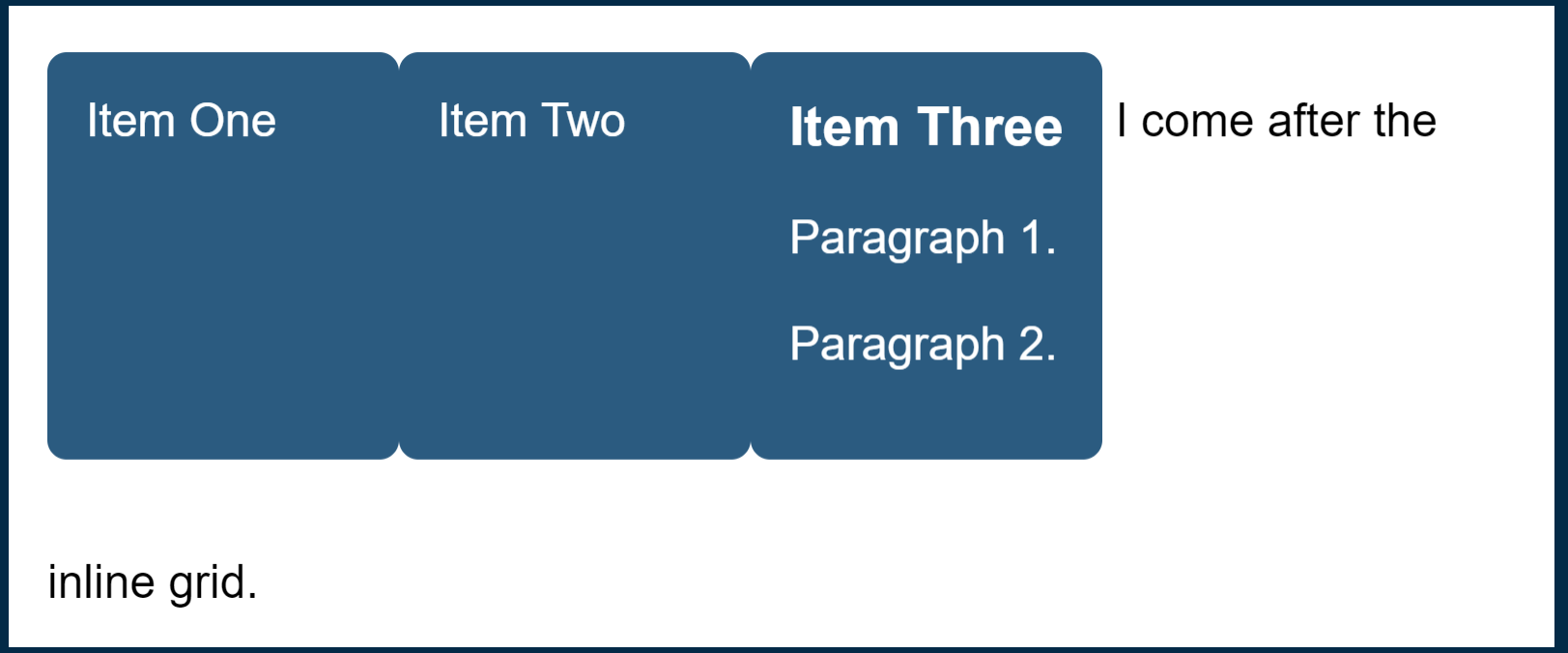

Item One
Item Two
Item Three
Paragraph 1.
Paragraph 2.

# The two values of display

```
.example {
  display: block grid;
  grid-template-columns: 1fr 1fr 1fr;
}
```



```
.example {
  display: inline grid;
  grid-template-columns: 1fr 1fr 1fr;
}
```



# The outer display type

How the box behaves in the layout - block or inline

# The inner display type

The formatting context of the direct children – grid, flex etc.

# Creating a new Block Formatting Context

```
.box {
  background-color: rgb(43,91,128);
}
```


Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

# display: flow-root

```
.box {
  background-color: rgb(43,91,128);
  display: flow-root;
}
```


Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

# Writing Modes

```
writing-mode: horizontal-tb;
```

Inline Dimension

Block Dimension

Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

Turnip greens yarrow ricebean rutabaga endive cauliflower sea lettuce kohlrabi amaranth water spinach avocado daikon napa cabbage asparagus winter purslane kale.

Celery potato scallion desert raisin horseradish spinach carrot soko. Lotus root water spinach fennel kombu maize bamboo shoot green bean swiss chard seakale pumpkin onion chickpea gram corn pea. Brussels sprout coriander water chestnut gourd swiss chard wakame kohlrabi beetroot carrot watercress. Corn amaranth salsify bunya nuts nori azuki bean chickweed potato bell pepper artichoke.

```
writing-mode: vertical-rl;
```

Block Dimension

Inline Dimension

Block Start

Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

Turnip greens yarrow ricebean rutabaga endive cauliflower sea lettuce kohlrabi amaranth water spinach avocado daikon napa cabbage asparagus winter purslane kale.

Celery potato scallion desert raisin horseradish spinach carrot soko. Lotus root water spinach fennel kombu maize bamboo shoot green bean swiss chard seakale pumpkin onion chickpea gram corn pea. Brussels sprout coriander water chestnut gourd swiss chard wakame kohlrabi beetroot carrot watercress. Corn amaranth salsify bunya nuts nori azuki bean chickweed potato bell pepper artichoke.

Block End

Block Start

Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

Turnip greens yarrow ricebean rutabaga endive cauliflower sea lettuce kohlrabi amaranth water spinach avocado daikon napa cabbage asparagus winter purslane kale.

Celery potato scallion desert raisin horseradish spinach carrot soko. Lotus root water spinach fennel kombu maize bamboo shoot green bean swiss chard seakale pumpkin onion chickpea gram corn pea. Brussels sprout coriander water chestnut gourd swiss chard wakame kohlrabi beetroot carrot watercress. Corn amaranth salsify bunya nuts nori azuki bean chickweed potato bell pepper artichoke.

Block End

Inline Start

Inline End

Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

Turnip greens yarrow ricebean rutabaga endive cauliflower sea lettuce kohlrabi amaranth water spinach avocado daikon napa cabbage asparagus winter purslane kale.

Celery potato scallion desert raisin horseradish spinach carrot soko. Lotus root water spinach fennel kombu maize bamboo shoot green bean swiss chard seakale pumpkin onion chickpea gram corn pea. Brussels sprout coriander water chestnut gourd swiss chard wakame kohlrabi beetroot carrot watercress. Corn amaranth salsify bunya nuts nori azuki bean chickweed potato bell pepper artichoke.

Inline Start

Veggies es bonus vobis, proinde vos postulo essum magis kohlrabi welsh onion daikon amaranth tatsoi tomatillo melon azuki bean garlic.

Gumbo beet greens corn soko endive gumbo gourd. Parsley shallot courgette tatsoi pea sprouts fava bean collard greens dandelion okra wakame tomato. Dandelion cucumber earthnut pea peanut soko zucchini.

Turnip greens yarrow ricebean rutabaga endive cauliflower sea lettuce kohlrabi amaranth water spinach avocado daikon napa cabbage asparagus winter purslane kale.

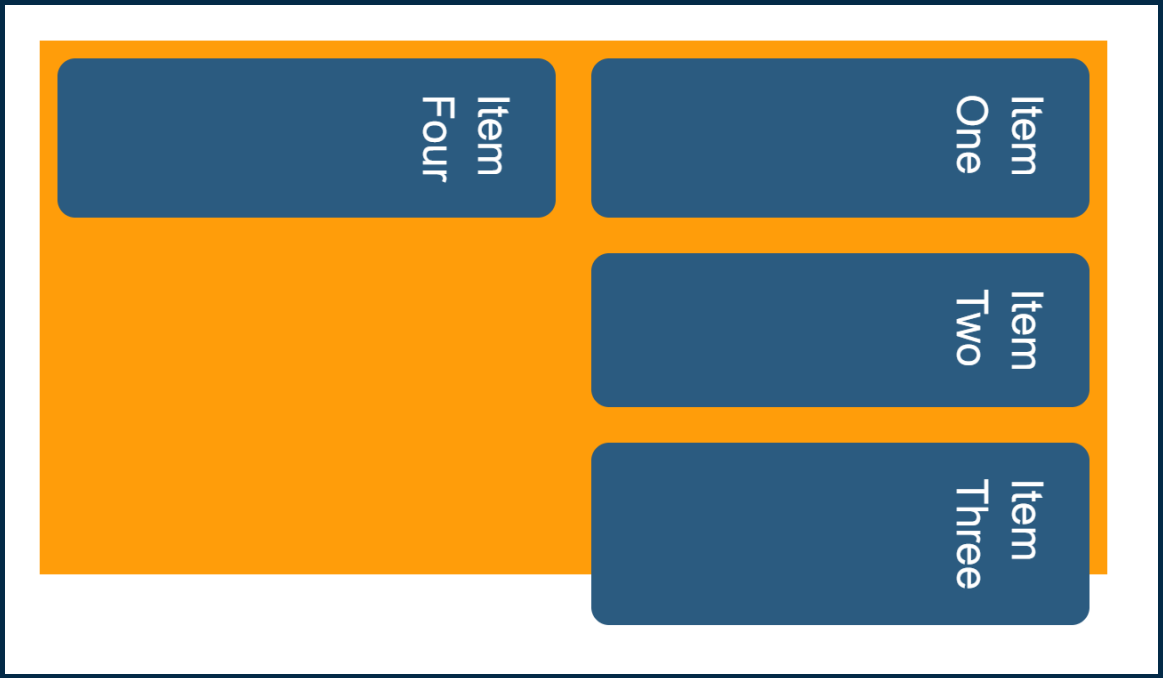
Celery potato scallion desert raisin horseradish spinach carrot soko. Lotus root water spinach fennel kombu maize bamboo shoot green bean swiss chard seakale pumpkin onion chickpea gram corn pea. Brussels sprout coriander water chestnut gourd swiss chard wakame kohlrabi beetroot carrot watercress. Corn amaranth salsify bunya nuts nori azuki bean chickweed potato bell pepper artichoke.

Inline End

# Web layout was tied to physical dimensions

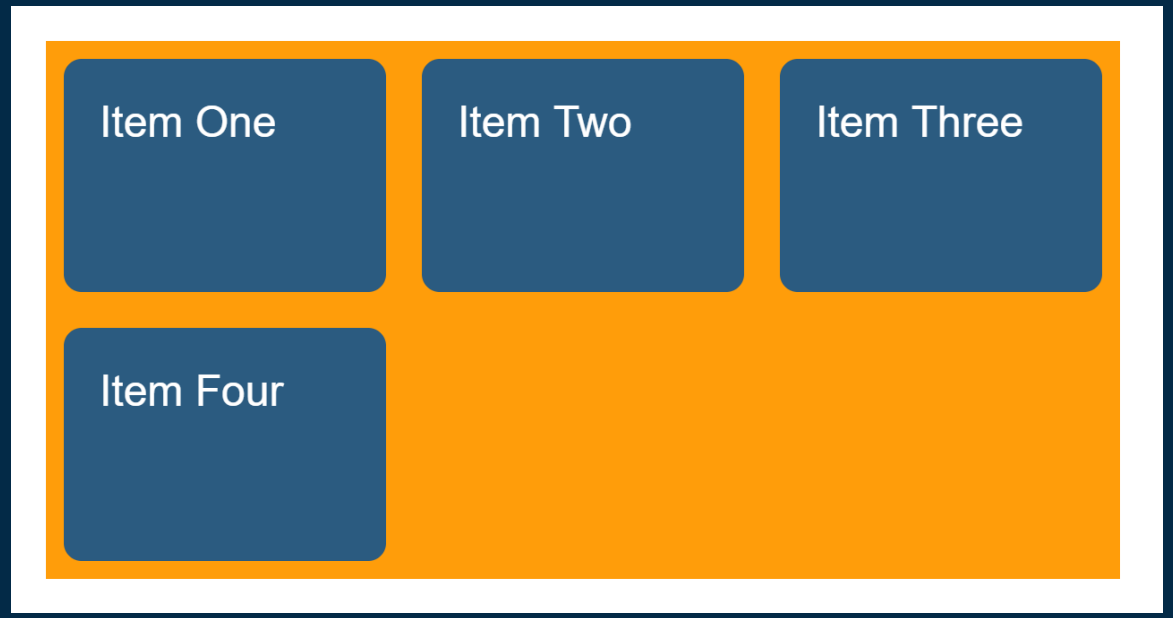
We think in top, right, bottom, left. Or width & height.

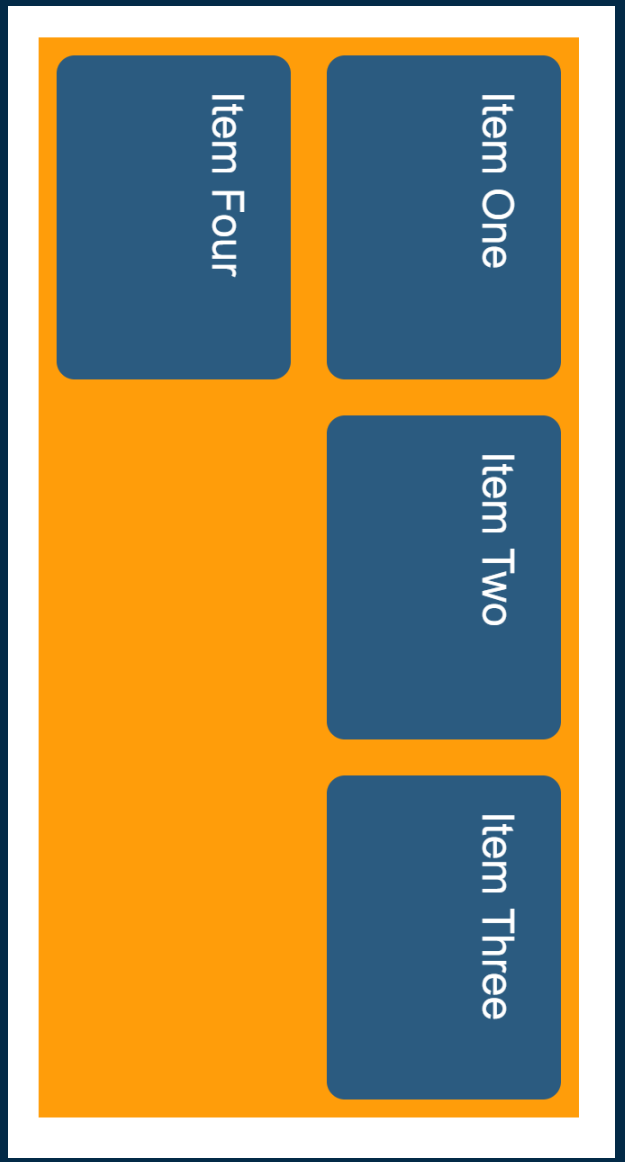
```
.example {
  width: 600px;
  height: 300px;
}
```

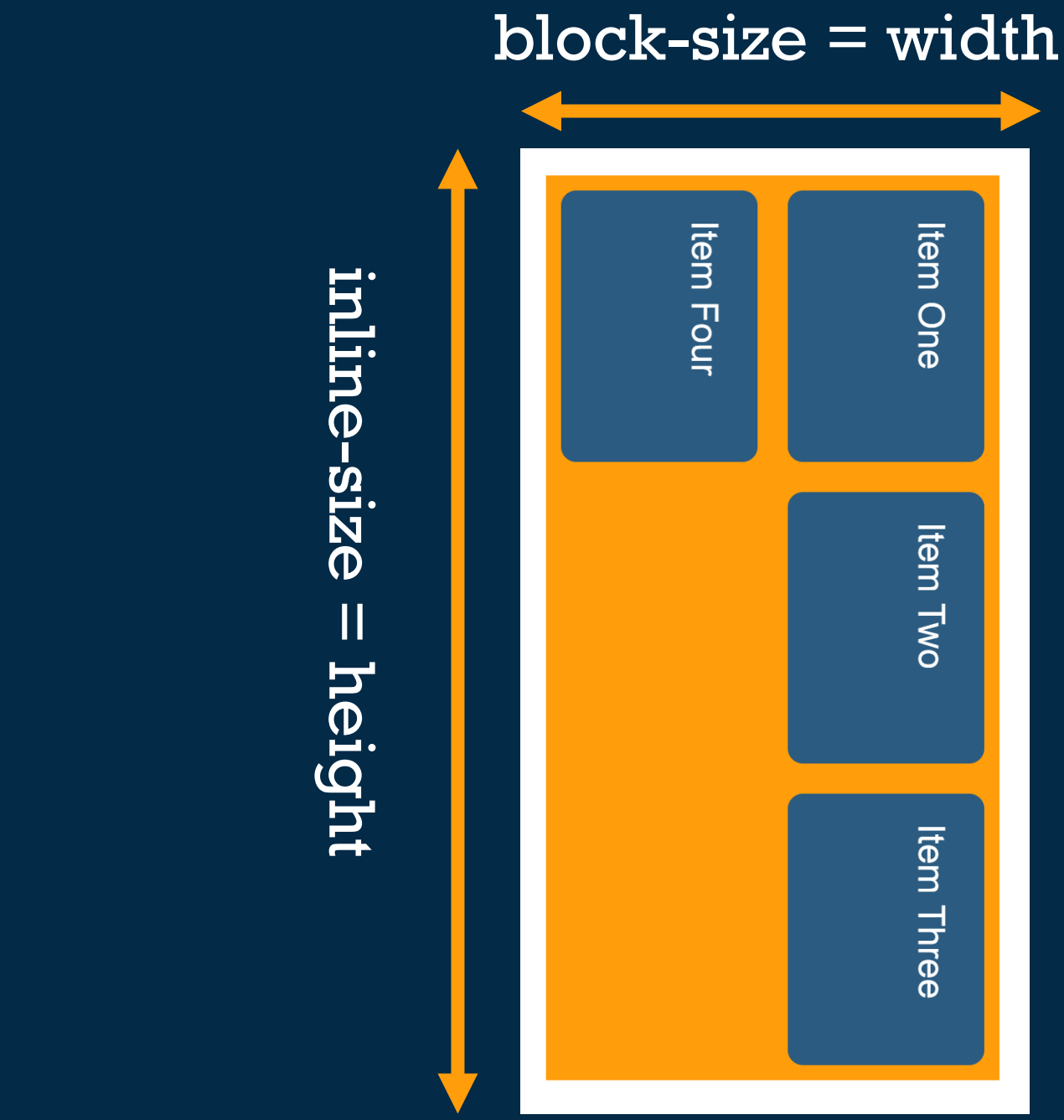
# Logical Properties & Values

```
.example {
  inline-size: 600px;
  block-size: 300px;
}
```




inline-size = width
block-size = height
Item One
Item Two
Item Three
Item Four

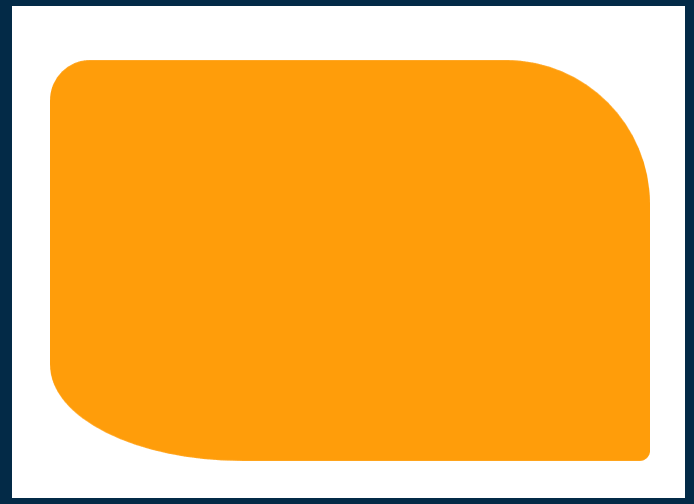

block-size = width
inline-size = height
Item Four
Item One
Item Two
Item Three

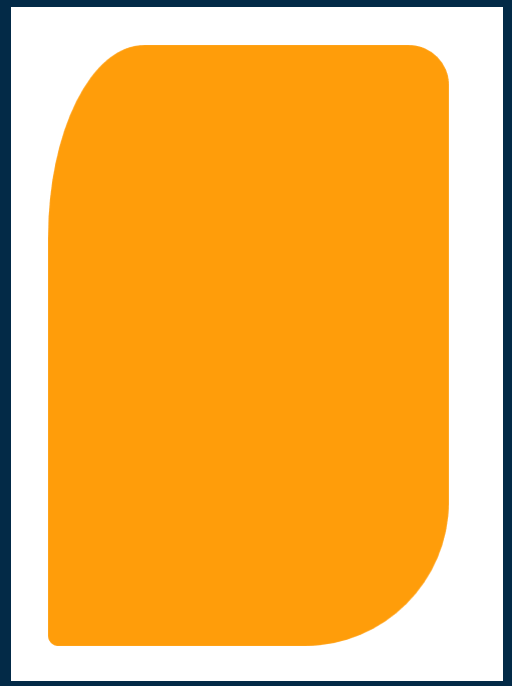
# Physical v. Logical

```
.example {
  padding-top: 10px;
  padding-right: 2em;
  margin-bottom: 2em;
}
```

```
.example {
  padding-block-start: 10px;
  padding-inline-end: 2em;
  margin-block-end: 2em;
  margin-inline: 1em;
}
```

```
.example {
  border-start-start-radius: 20px;
  border-start-end-radius: 3em;
  border-end-start-radius: 2em 4em;
  border-end-end-radius: 5px;
}
```





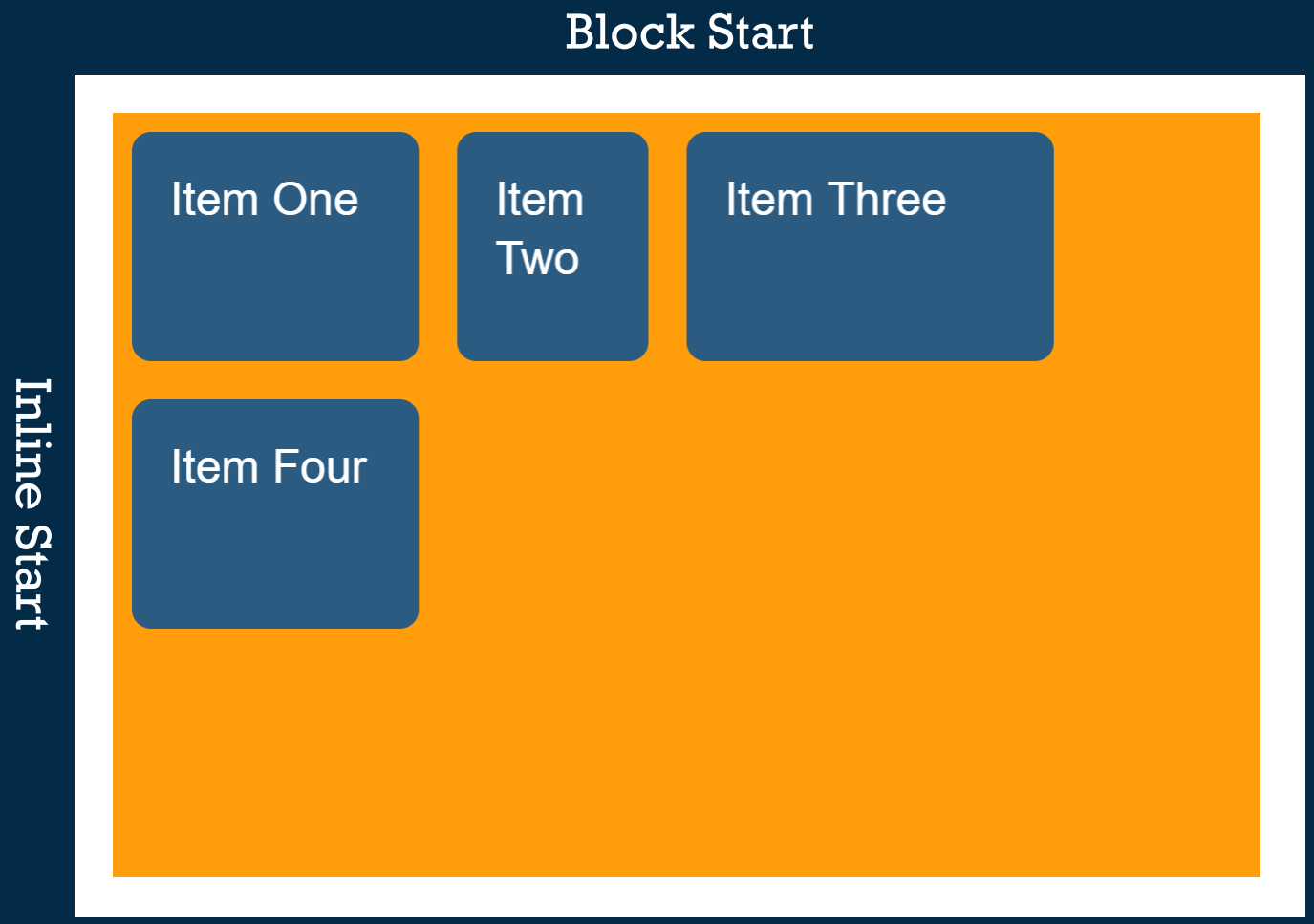
We need to think in terms of this flow-relative, logical world.

# Box Alignment

https://drafts.csswg.org/css-align/

# Aligning things in the block and inline dimensions.

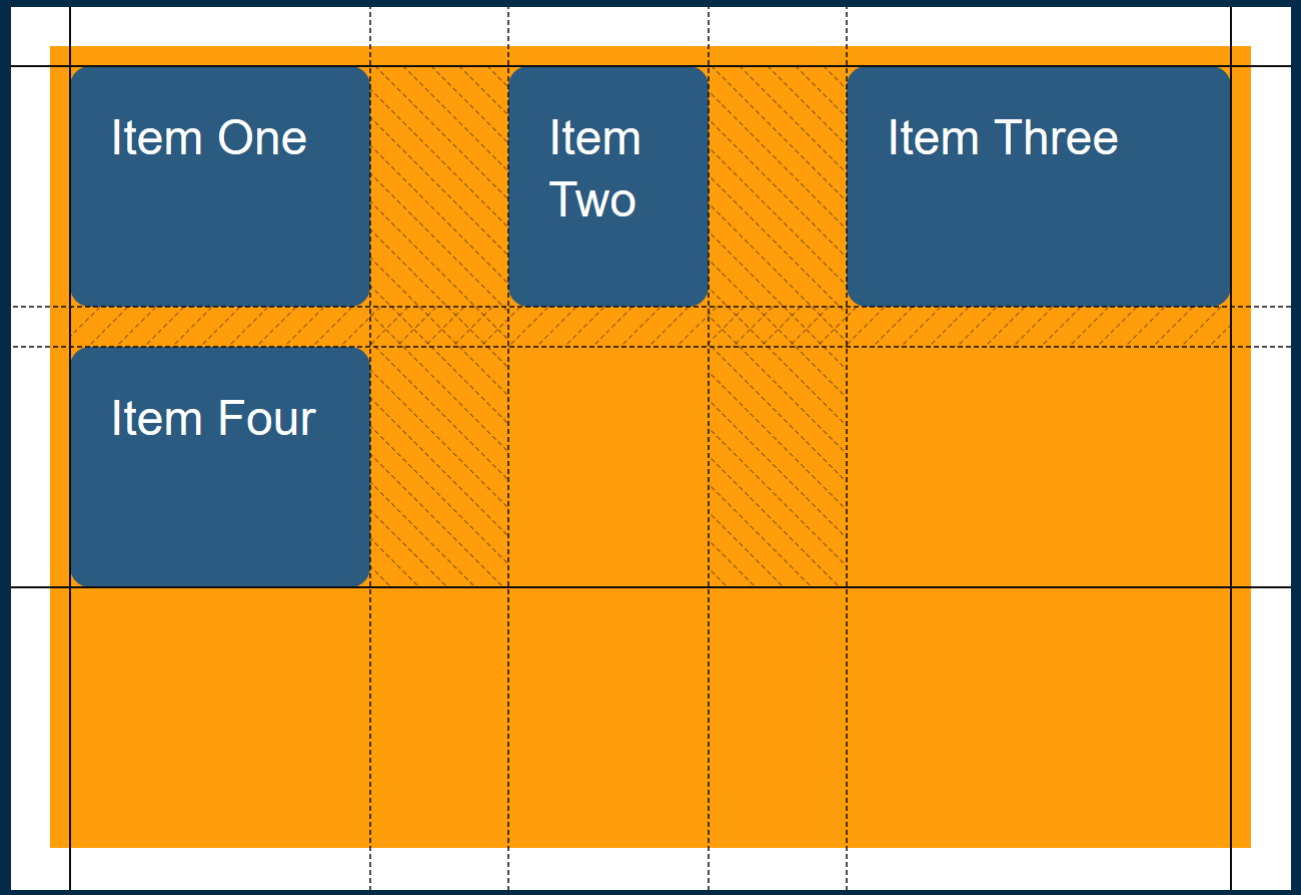
Distribution of space and alignment of items within their space.


Block Start
Inline Start
Item One
Item Two
Item Three
Item Four

# justify-content

In Grid, inline dimension space distribution between tracks

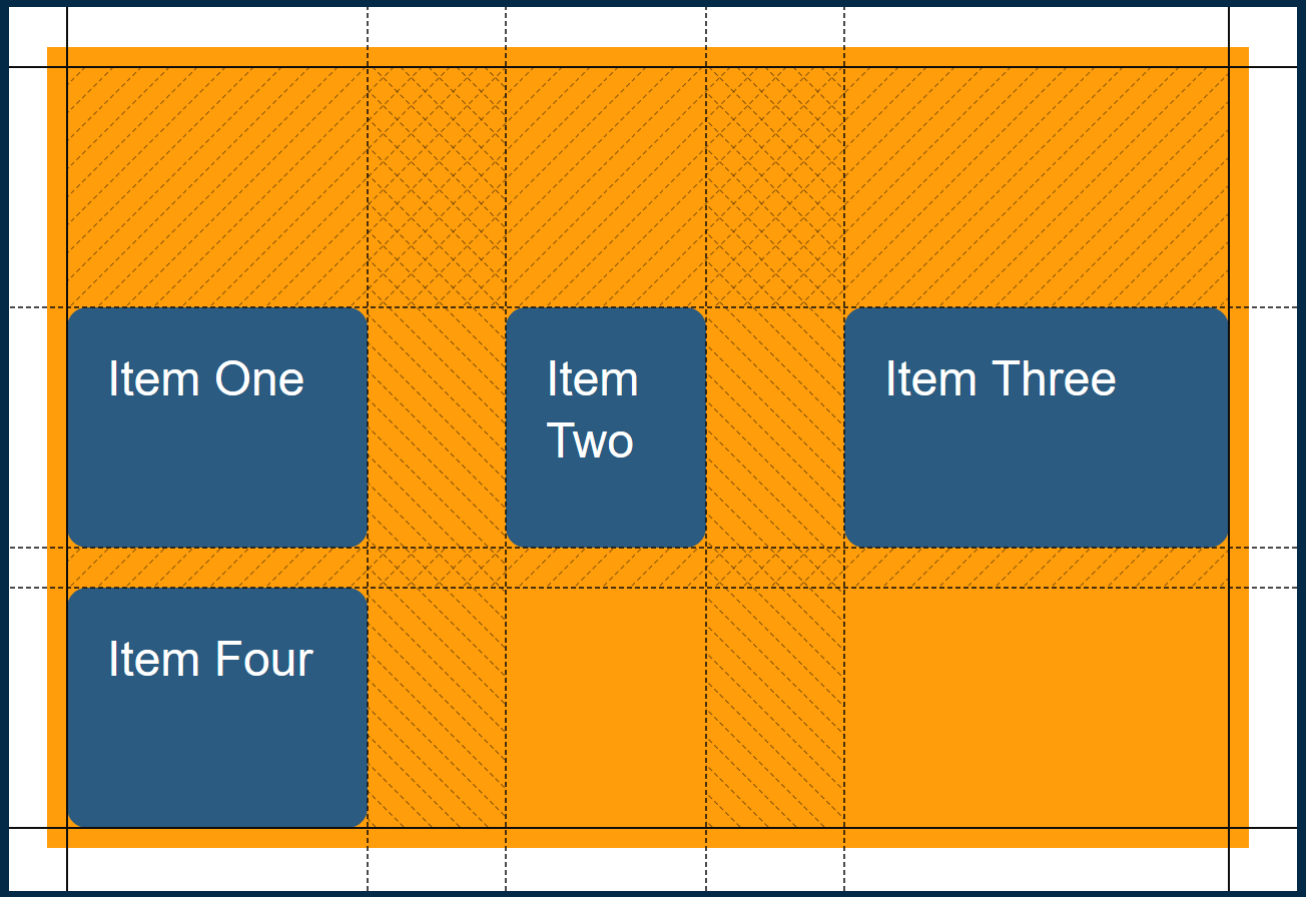
```
.example {
  justify-content: space-between;
}
```



# align-content

In Grid, block dimension space distribution between tracks

```
.example {
  align-content: end;
}
```

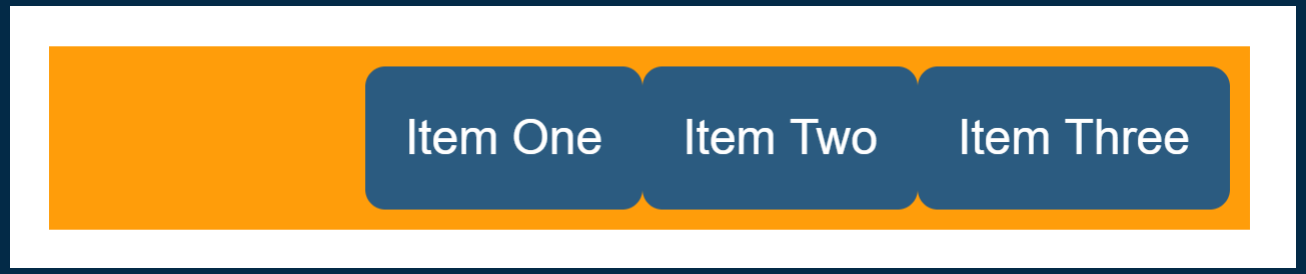


In flexbox,
we justify on the main axis and
align on the cross axis

# justify-content

In Flex, main axis space distribution between flex items

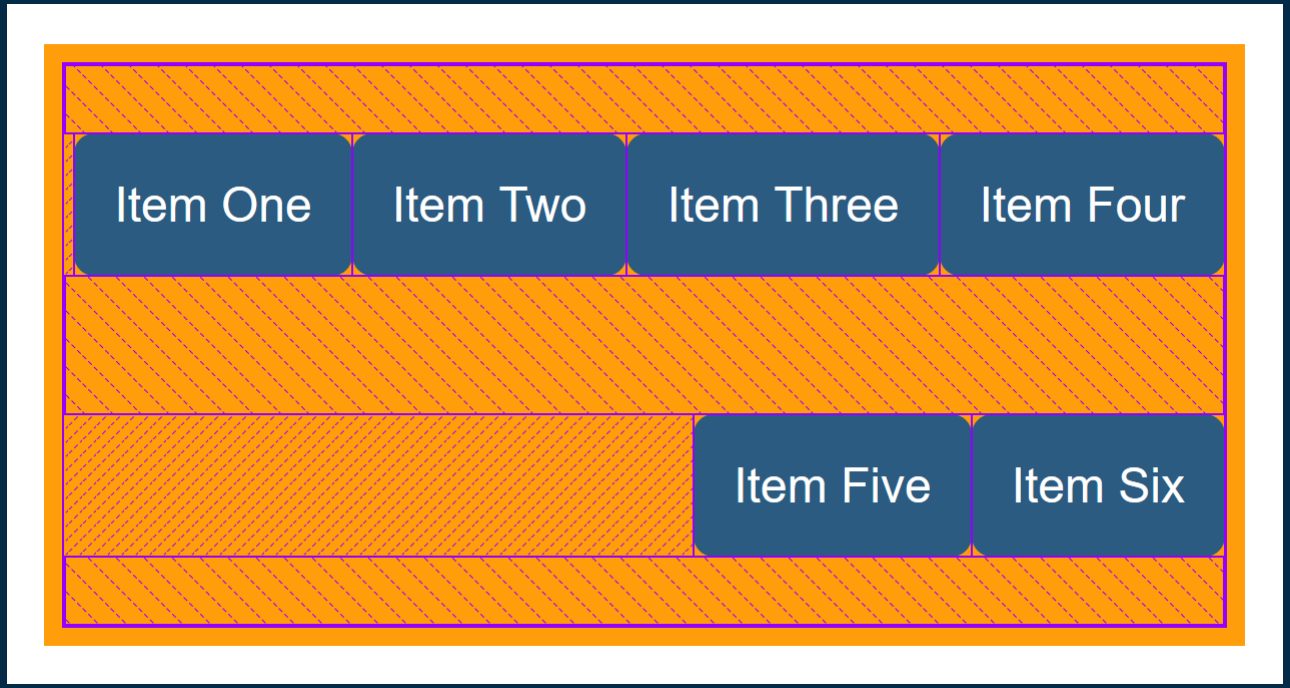
```
.example {
  justify-content: flex-end;
}
```



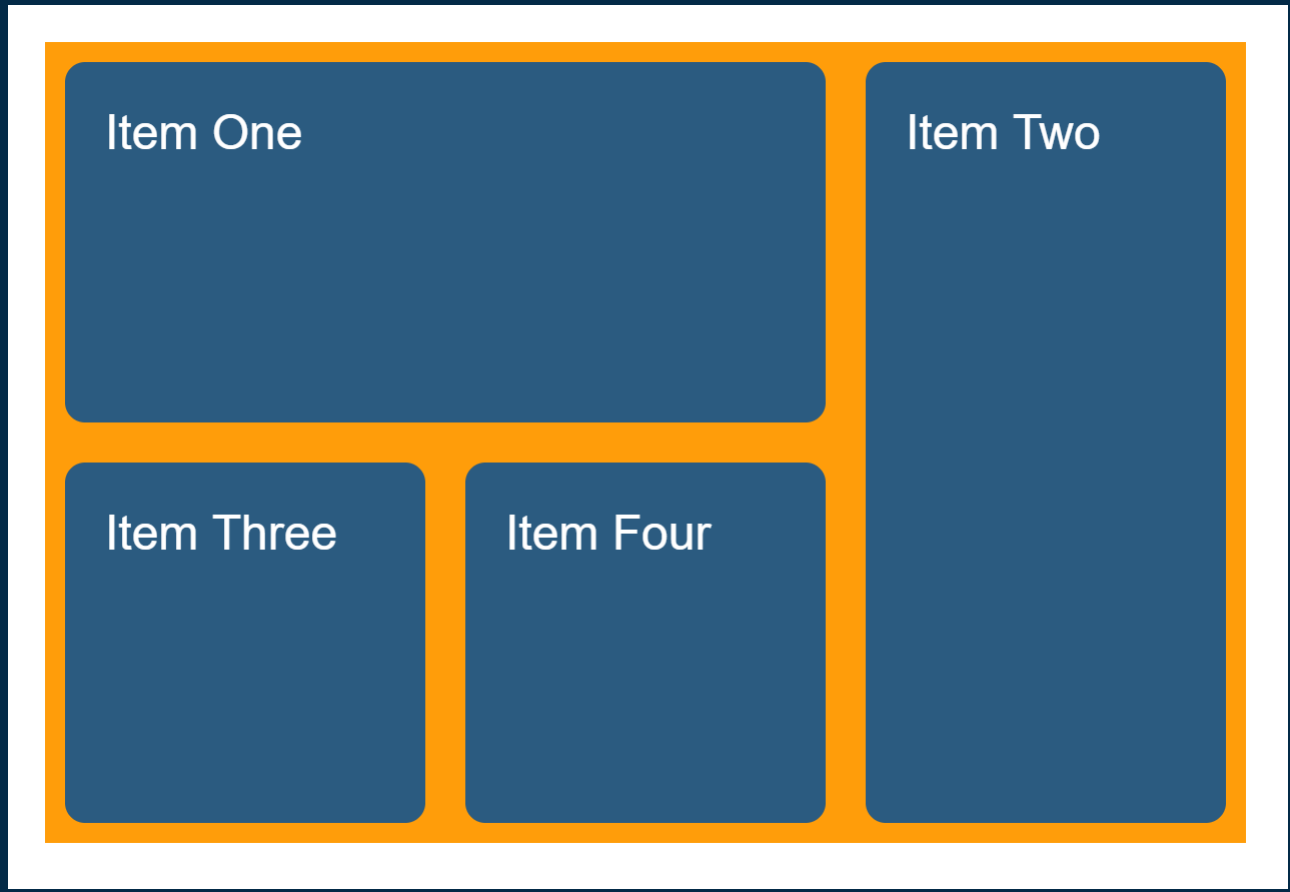
# align-content

In Flex, cross axis space distribution between flex lines

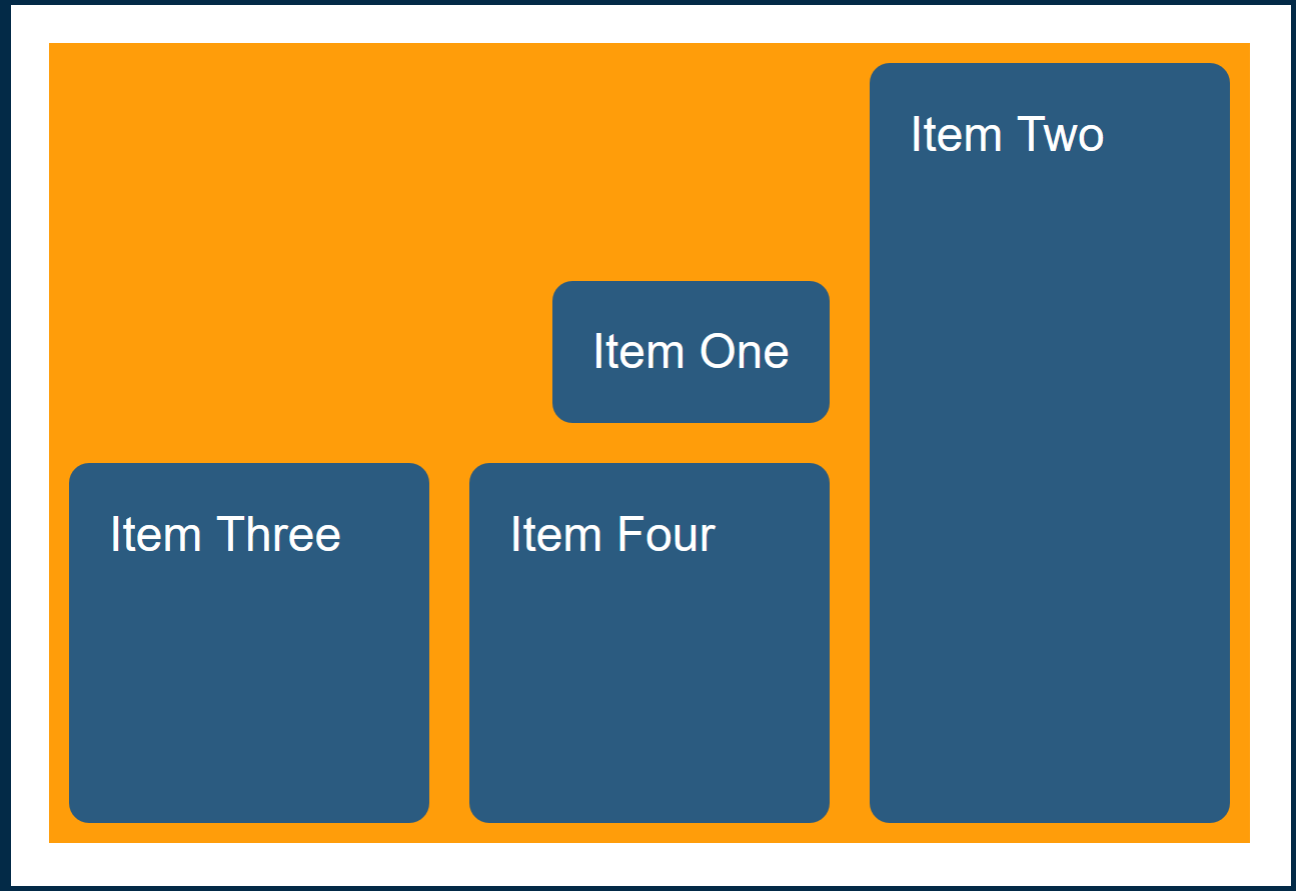
```
.example {
  align-content: space-around;
}
```



For –content properties to do anything, you must have spare space to distribute!


Item One
Item Two
Item Three
Item Four

```
.item {
  justify-self: end;
  align-self: end;
}
```

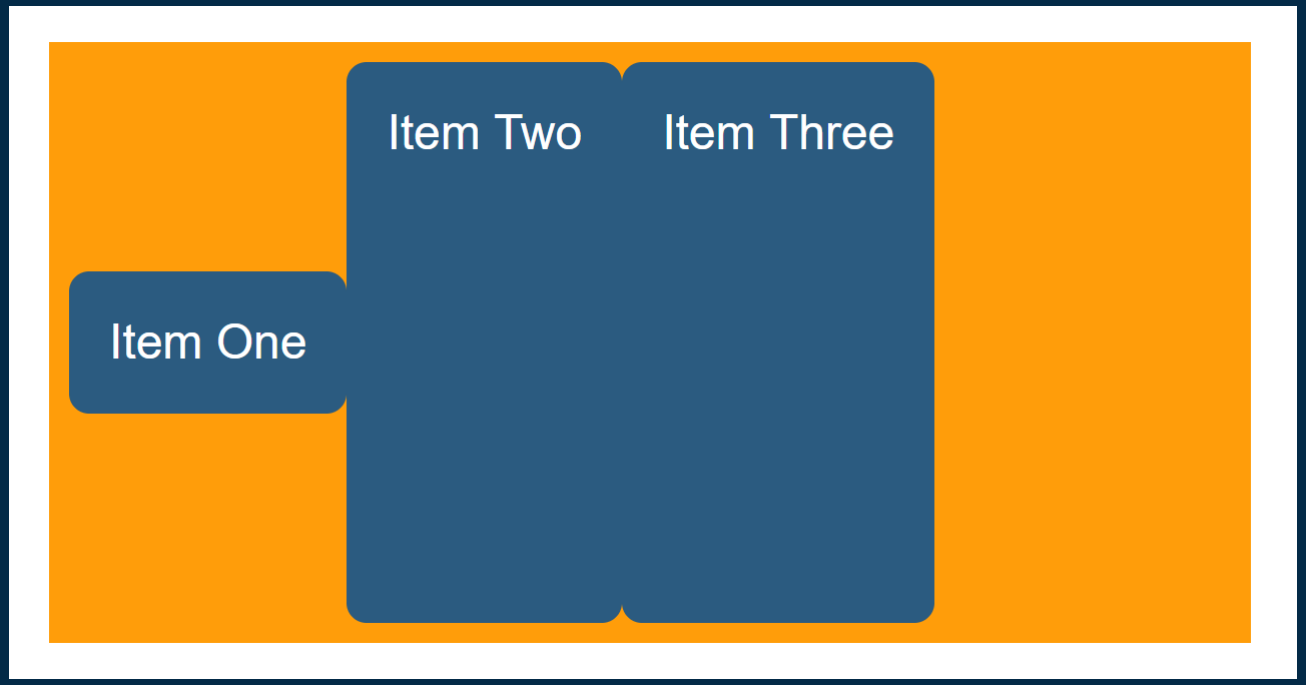


```
.example {
  justify-items: end;
  align-items: end;
}
```

“[justify-content] does not apply to flex items, because there is more than one item in the main axis.”

https://drafts.csswg.org/css-align/#justify-flex

```
.item {
  align-self: center;
}
```



"Prior to alignment via justify-content and align-self, any positive free space is distributed to auto margins in that dimension."

https://www.w3.org/TR/css-flexbox-1/#auto-margins

```
.example div:last-child {
  margin-left: auto;
}
```

# Let’s stop calling stuff that is in the spec a CSS ‘hack’

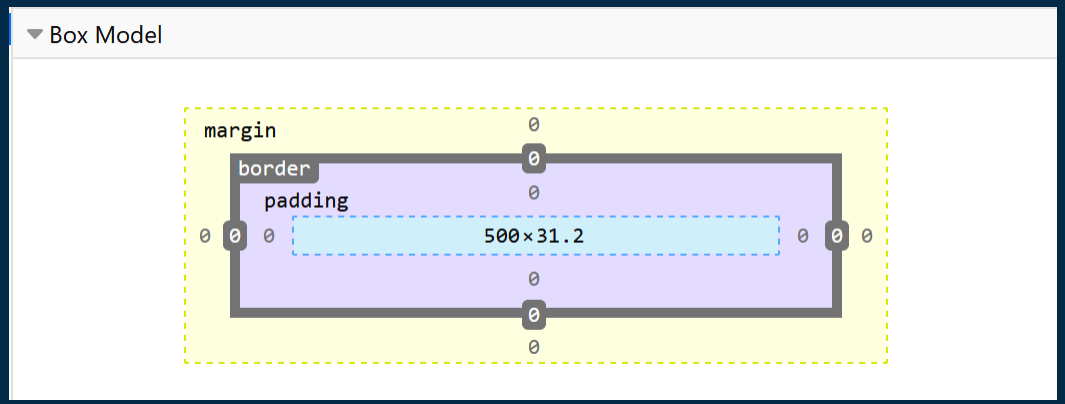
# What about the Box Model?

When we had to control the size of each item in a layout, the Box Model was key.


I am a box with some content.

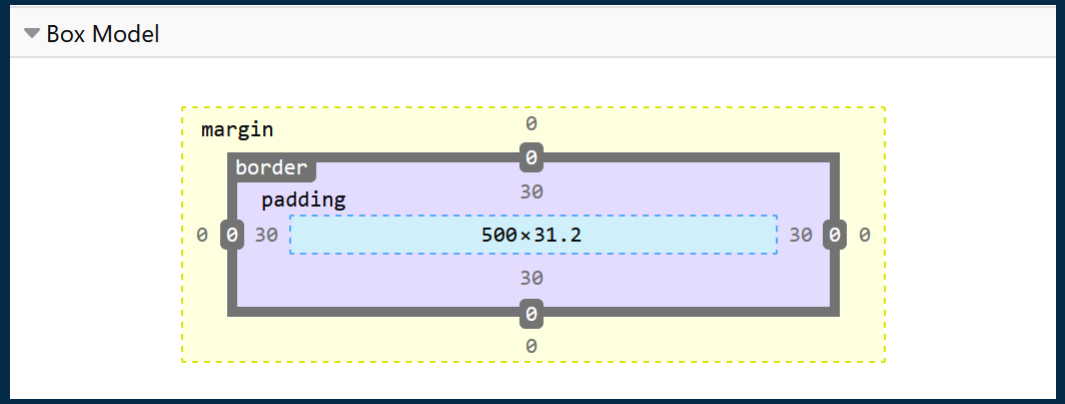
500px

I am a box with some content.



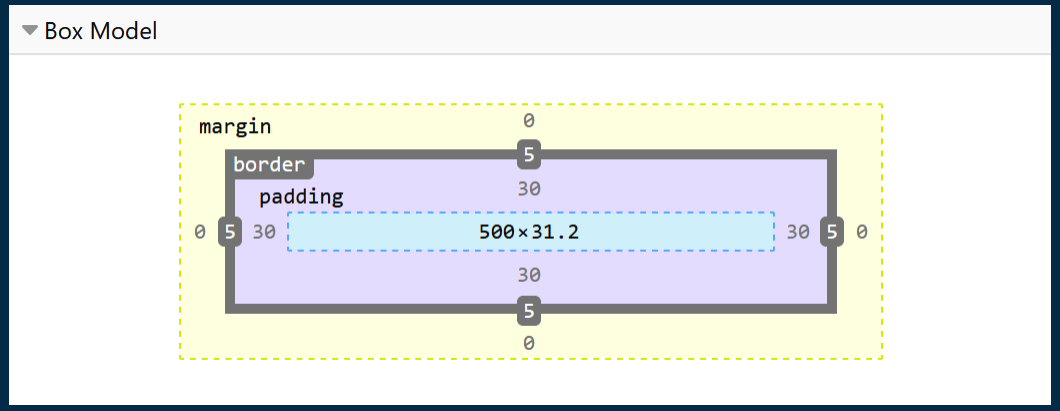
30px + 500px + 30px

I am a box with some content.



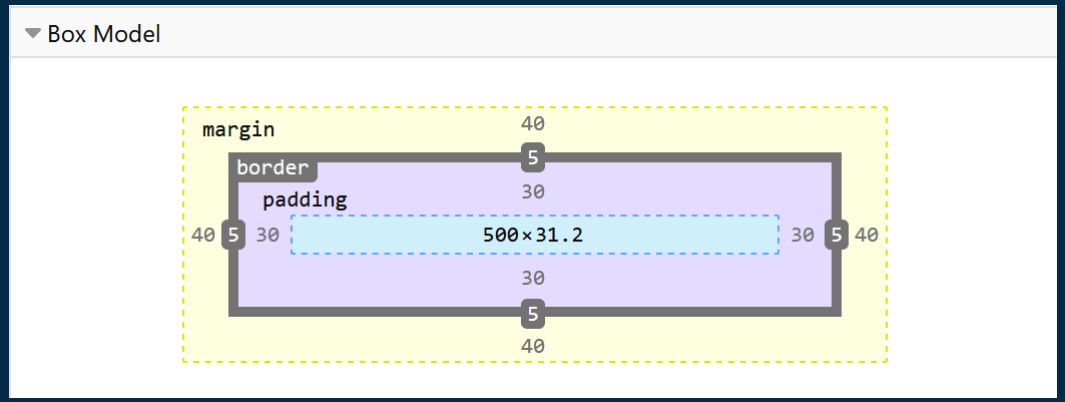
5px + 30px + 500px + 30px + 5px

I am a box with some content.



40px + 5px + 30px + 500px + 30px + 5px + 40px

I am a box with some content.



# What is the inline-size or width of the box?

By default, the **content-box**

# If you want the specified width to include padding and border

Set the box-sizing property to **border-box**.

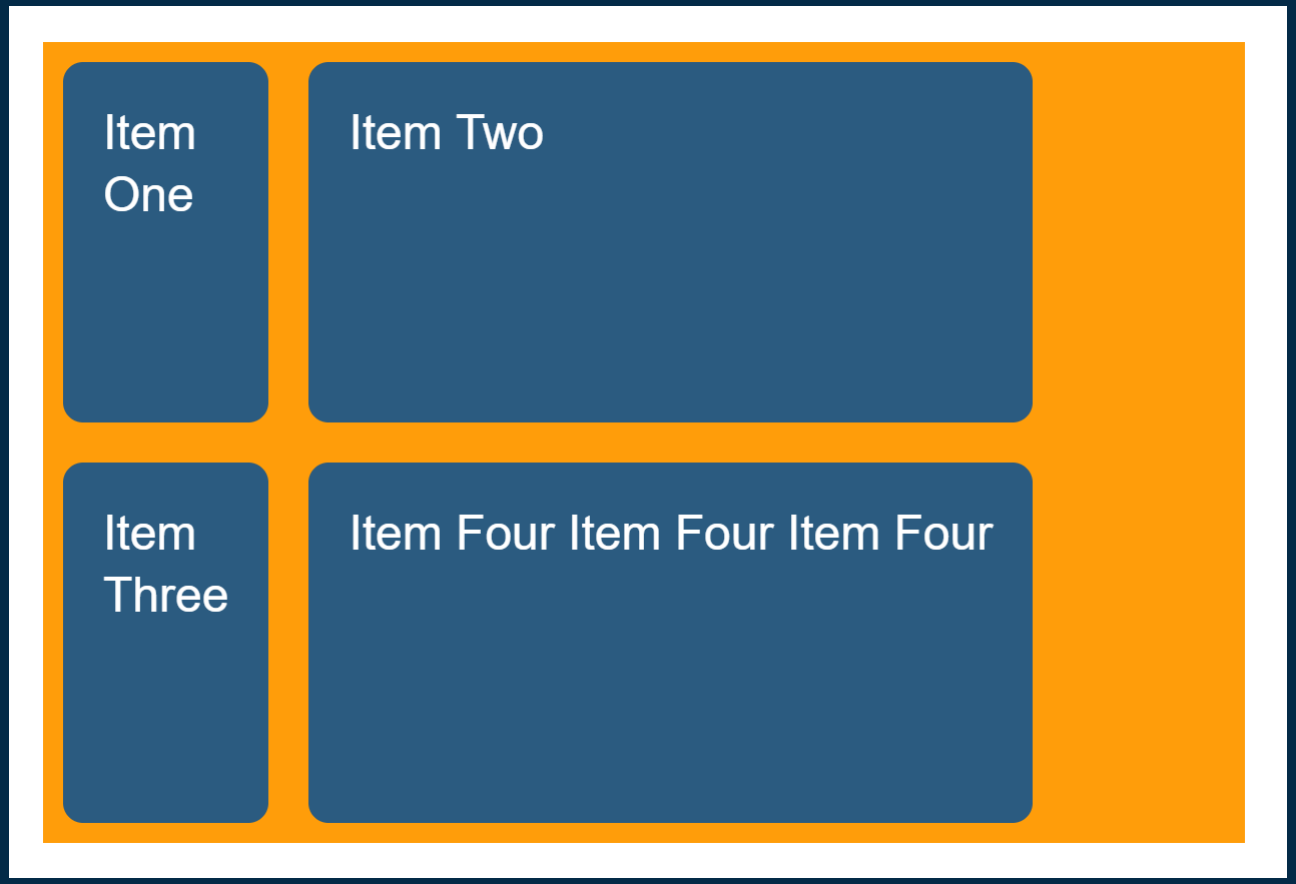
```
.example {
  box-sizing: border-box;
}
```

# How big is that box?

In the past everything was a length or a percentage.

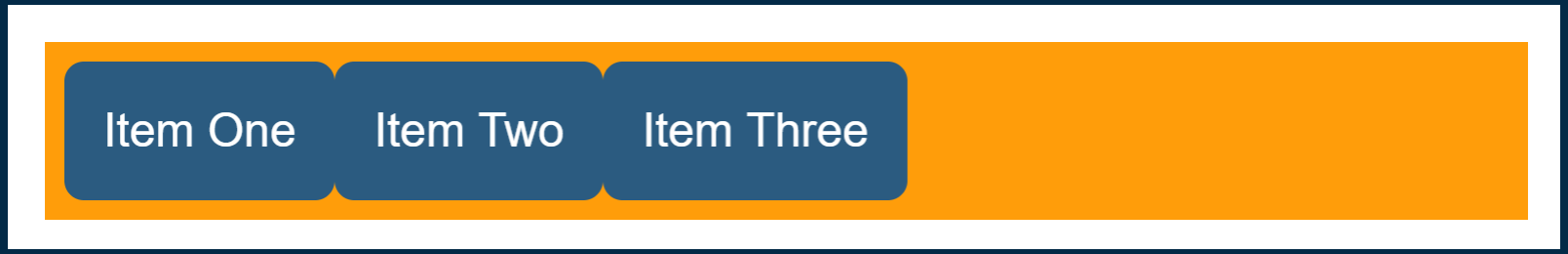
What is the minimum and maximum size of this thing?

```
.example {
  grid-template-columns: min-content max-content;
}
```



Any content-based sizing is worked out based on these min and max content sizes.

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



```
.example > * {
  flex: auto;
}
```

```
.example > * {
  flex: auto;
}
```

```
.example > * {
  flex: 1;
}
```

# Old browsers. They exist.

We have a specification. Some of it isn't implemented yet.

Lack of support is very different to the buggy support of the past.

# Media & Feature Queries

How big is my viewport? Is this a touchscreen? Does this browser support Grid? Respond based on the answers.

We need to stop talking about CSS as a weird and quirky thing.

CSS is different.
Not wrong, not impossible to understand as a system.

# Learn and teach CSS as it is today.

# Thank you!

@rachelandrew