COESIONE ITALIA 21-27 PIEMONTE

Cofinanziato dall'Unione europea

REGIONE PIEMONTE

Web Developer

HTML, CSS e Strumenti di Digital Marketing (SEO, SEM, SEA)

Docente: Shadi Lahham

Intervento realizzato da ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Flexbox

Modern layout

Shadi Lahham - Web development

Using Flexbox

What is Flexbox

- A web layout model
- Consists of responsive items, flex items, within a container
- Items can be ordered and arranged in various ways
- Items can grow and shrink based on container size
- Content can flow in various directions

Float vs Flexbox

- Float
  - Complex
  - Requires the use of float, margin, width and clear to work properly
  - Works on all browsers, even very old ones

- Flexbox
  - Easy and intuitive
  - Can do a lot more than float
  - Works on all modern browsers

Flexbox in a nutshell

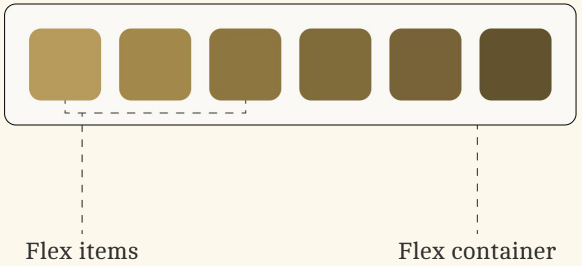
- Container and items
  - Flex items appear within a container
  - Flex determines the layout which is easily responsive
- Direction and wrap
  - In what sequence the items appear
  - What happens when space runs out
- Justify and align
  - Spacing between items on the main axis
  - Alignment of items with respect to each other
- Grow, shrink, basis
  - How items grow or shrink based on the changing space of the container

Flex container

Flex container
an element with *display: flex* or *display: inline-flex* that establishes a flex formatting context for its direct children

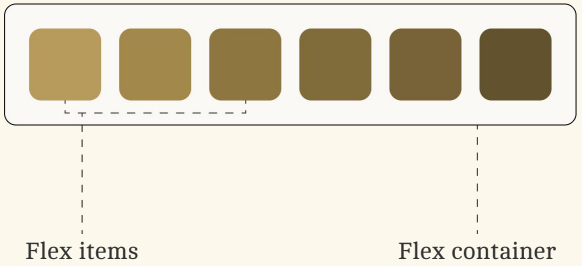
display: flex
makes the container a block-level flex container

display: inline-flex
makes the container an inline-level flex container



Flex items

Flex items
the direct children of a flex container laid out according to the rules of the flexbox layout model



Flex container & items

html

```html
<div class="flex-container">
  <div class="flex-item">item 1</div>
  <div class="flex-item">item 2</div>
  <div class="flex-item">item 3</div>
  <div class="flex-item">item 4</div>
  <div class="flex-item">item 5</div>
</div>
```

css

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

.flex-item {
  background: #d6b561;
}
```

Main and cross axes

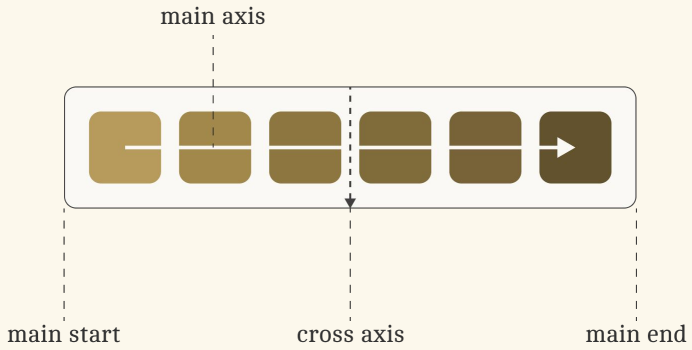
The main axis

the primary axis along which flex items are laid out in a flex container is determined by the flex-direction property

the direction of the main axis can be either horizontal or vertical based on the specified value

The cross axis

perpendicular to the main axis is the cross axis, which runs across it and has a direction that depends on the orientation of the main axis



Direction

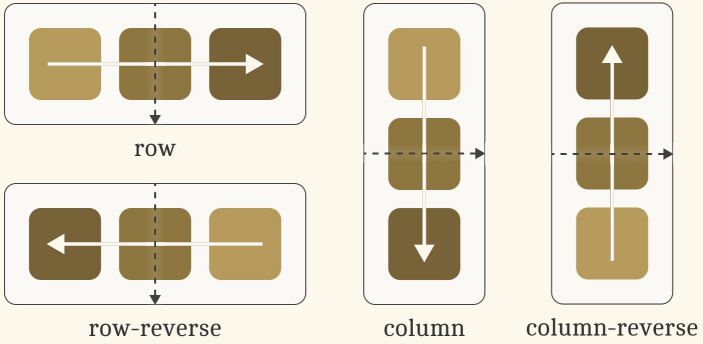
flex-direction
the property defines the main axis, while the cross axis runs perpendicular to it

row
flex items laid out horizontally, left to right

row-reverse
flex items laid out horizontally, right to left

column
flex items laid out vertically, top to bottom

column-reverse
flex items laid out vertically, bottom to top



Wrap

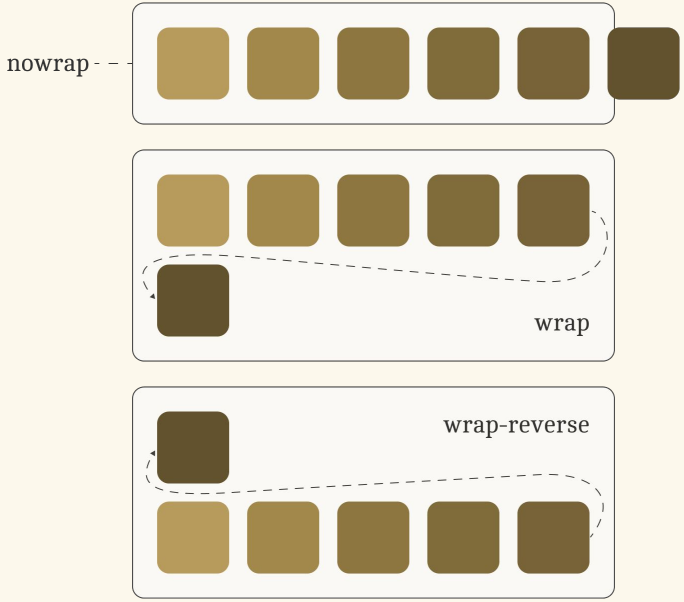
flex-wrap
controls if the flex container is single or multi-line and sets the line direction

flex-wrap: nowrap (default)
items are laid out in a single line and won't wrap, even if they overflow the container

flex-wrap: wrap
items wrap onto multiple lines, top to bottom

flex-wrap: wrap-reverse
items wrap onto multiple lines, bottom to top

flex-flow
Shorthand for *flex-direction* and *flex-wrap*
e.g. flex-flow: column wrap-reverse;



Justify

spacing between items on the main axis using the property justify-content

flex-start
items are packed toward the start of the main axis

flex-end
Items are packed toward the end of the main axis

center
items are centered along the main axis

difference between alignment values: start, flex-start, and self-start?

flex-start

flex-end

center

Justify

spacing between items on the main axis using the property justify-content

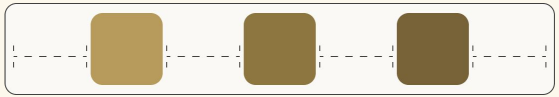
space-between
items are evenly distributed with space between them

space-around
items are evenly distributed with space around them

space-evenly
items are evenly distributed with equal space around them

space-between

space-around



space-evenly

Justify

```
.flex-container {
 display: flex;
  /* flex-flow: row wrap; */
 flex-direction: row;
 flex-wrap: wrap;
 justify-content: space-around;
 border: 1px solid black;
}

.flex-item {
 background: #d6b561;
}
```

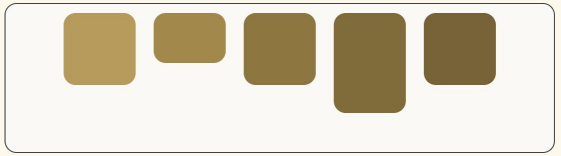
Align

alignment of items with respect to each other
along the cross axis using align-items

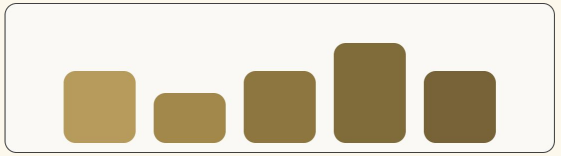
flex-start
items are aligned to the start of the cross axis

flex-end
items are aligned to the end of the cross axis

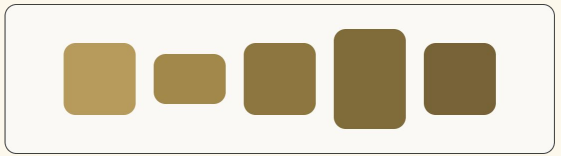
center
items are centered along the cross axis



flex-start



flex-end



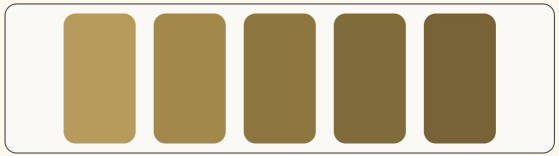
center

Align

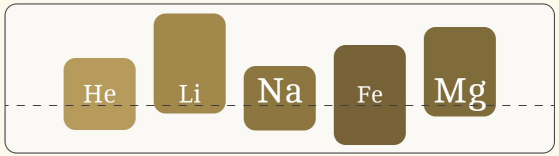
alignment of items with respect to each other
along the cross axis using align-items

baseline
items are aligned such that their baselines align

stretch
items are stretched to fill the container along
the cross axis, taking up any remaining space



stretch



baseline

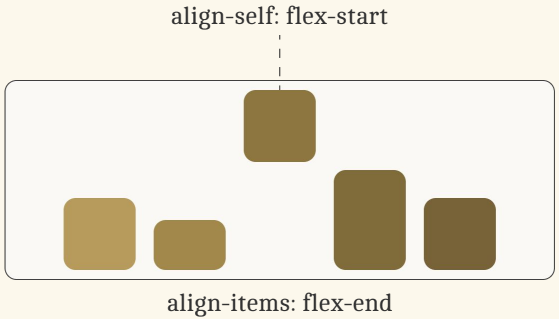
aligning Items in a Flex Container

Align self

the CSS property align-self overrides the align-items property for individual flex items within a flex container

the align-items property sets the alignment of all flex items along the cross axis whereas align-self adjusts the alignment of a single flex item along the cross axis

this provides more precise control over the layout of individual items within the flex container



Align self

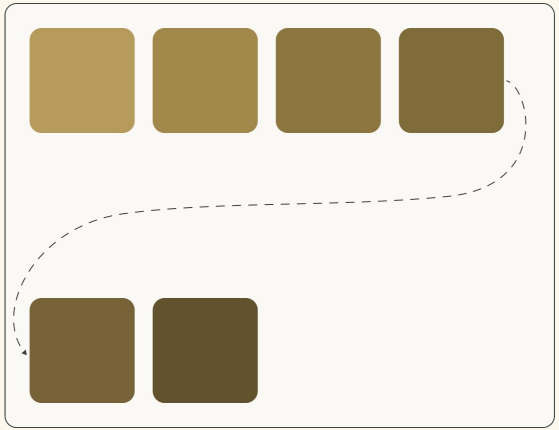
```
.flex-container {
 display: flex;
 height: 150px;
 align-items: flex-end;
 border: 1px solid black;
}

.flex-item {
 background: #d6b561;
 flex: 0 0 100px;
 height: 50px;
}

.flex-item.special {
 align-self: flex-start;
}
```

Align content

the align-content property in CSS Flexbox controls spacing between lines in a multi-line flex container along the cross axis

it only takes effect when flex-wrap is set to wrap or wrap-reverse and there are multiple lines of items. If there's only one line align-content won't have any visible effect

values include flex-start, flex-end, center, space-between, space-around, space-evenly, and stretch



align-content: space-between

Align content

```
.flex-container {
 display: flex;
 flex-direction: row;
 flex-wrap: wrap;
 align-content: space-between;
 height: 400px;
 border: 1px solid black;
}

.flex-item {
 background: #d6b561;
 flex: 0 0 200px;
 height: 100px;
}
```

Grow, shrink, basis

how items grow or shrink based on the changing space of the container

flex-grow
determines if the element can take up additional space or not. 0 means the item will not grow

flex-shrink
determines how much an element is allowed to shrink. 0 means the item will not shrink

flex-basis
sets the default size of a node along the main axis. It can be specified in different units such as px, %, em, etc. The default value auto means size is based on content size or width and height

flex-grow

flex-grow

flex-grow

Grow, shrink, basis

flex
shorthand property that combines flex-grow, flex-shrink, and flex-basis into a single declaration

example

```
flex: 1 0 200px;
```

default

```
flex: 0 1 auto;
```

flex-grow is **0** so the item won't grow
flex-shrink is **1** so the item will shrink
flex-basis is **auto** so the item's initial size is based on its content

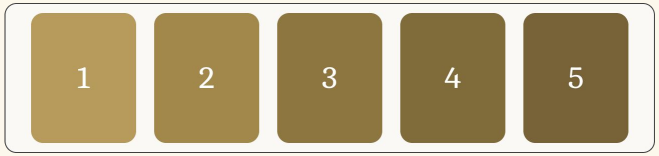
Order

the order of items within a flex container can be customized using the order property

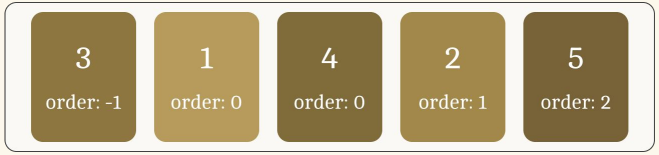
items are arranged by ascending order value, then source code order

default order value is 0 if not specified

negative values can position items earlier in the sequence



original order



order

order | CSS-Tricks
ordering flex items | MDN

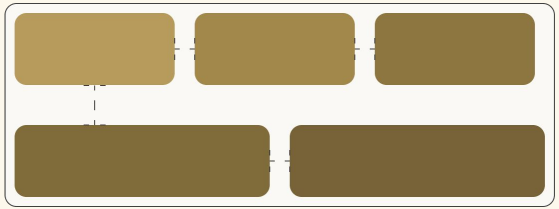
Order property

```
.item-1 { order: 0; /* The second item to be displayed */ }
.item-2 { order: 1; /* The fourth item to be displayed */ }
.item-3 { order: -1; /* The first item to be displayed */ }
.item-4 { order: 0; /* The third item to be displayed */ }
.item-5 { order: 2; /* The fifth item to be displayed */ }
```

Gap

defines the spacing between flex items, creating consistent spacing without using margins

```css
.flex-container {
  display: flex;
  flex-wrap: wrap;
  /* gap: 20px 10px; */
  row-gap: 20px;
  column-gap: 10px;
  max-width: 300px;
}

.flex-item {
  flex: 1 0 100px;
  background: #d6b561;
}
```

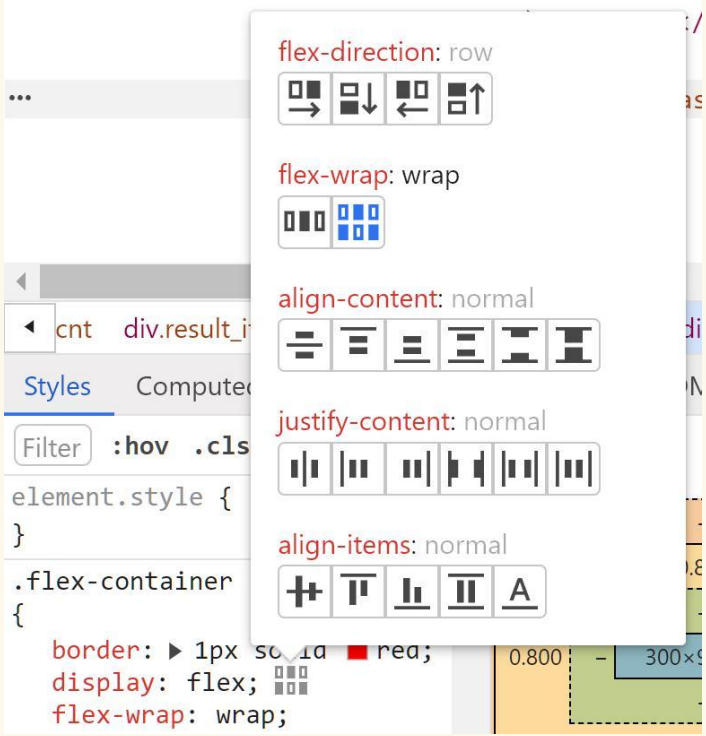


gap: 20px 10px

gap | MDN, gap | Can I use, CSS Gap

Chrome dev tools

Chrome devtools support flexbox and have a friendly UI for editing flex properties



[Flexbox debugging In DevTools]

Your turn

1.Presentation

Study all the links in references and use them to create a short presentation

- Presentation should be 10-15 slides
- Explain in detail how flexbox works
- Include your own code in the presentation
- Show at least 10 different code examples
- Include references to sites that have interesting usages of flex
- Include your **html** and **css** files in the delivery

2.Flex games

Complete all levels of the following flex games for learning CSS flexbox

Flexbox Froggy

Flexbox Defense

Submit an .md file with the answer for each level

Bonus: Flex Box Adventure

Submit an .md file with the answer for each level

3.Columns and nations

Repeat the following exercises from a previous unit using flexbox to structure the page

1. Columns columns
2. United Nations

In a readme.md file, compare the two approaches in terms of:

- Result
- Code quality
- Future flexibility

4.To flex or not to flex

Read the included article from The Blog of Karen Menezes

It's an old article but still contains some useful hints on when to use Flex and when to use basic CSS properties

References

A Complete Guide to Flexbox
Basic concepts of flexbox
CSS Flexbox Container
Flexbox
Even more about how Flexbox works - animated examples

Flexbox playground:

Build with Flexbox - nice playground

References

Solved by Flexbox — Cleaner, hack-free CSS
Flexbox Patterns
Flexbox debugging In DevTools

Very clear and complete tutorial:
Flexbox Tutorial | HTML & CSS Is Hard

Excellent guide:
An Interactive Guide to Flexbox in CSS