Introducing CSS Flexbox Layout

PRESENTED BY

Matthew Ellison, UA Europe

MATTHEW ELLISON

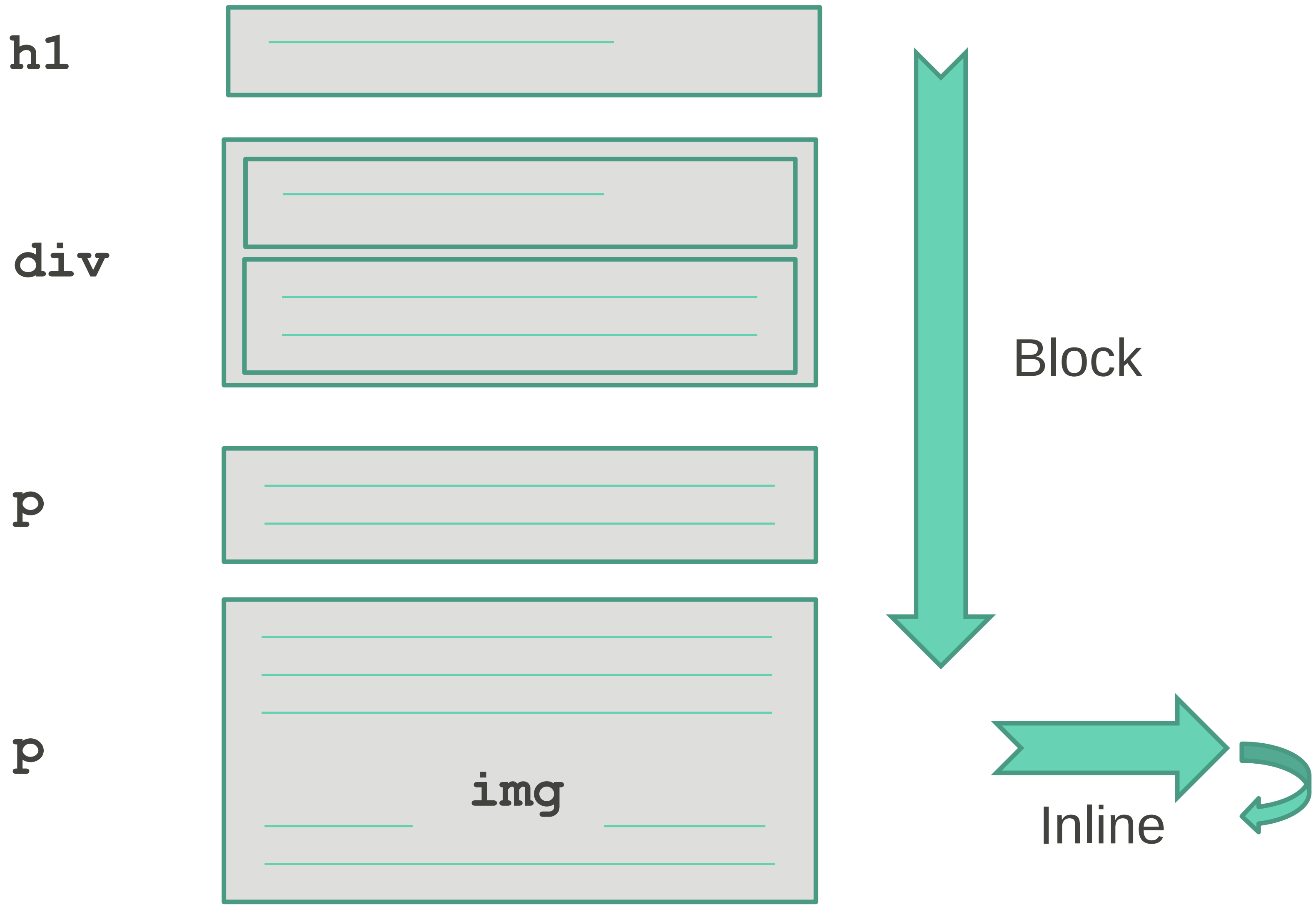
- Consultant and trainer for User Assistance tools and technologies since 1993
- User of MadCap products since 2006
- Certified Flare Instructor
- Based in the UK

madcap™
FLARE
CERTIFIED INSTRUCTOR

WHAT WE’LL COVER THIS AFTERNOON

- The limitations of html’s block model
- Responsive layout with Media Queries
- Introducing Flexbox
- Scenarios where Flexbox is useful
- Flexbox properties:
  building a solution from the ground up
- Browser support
- Implementing CSS Flexbox in Flare

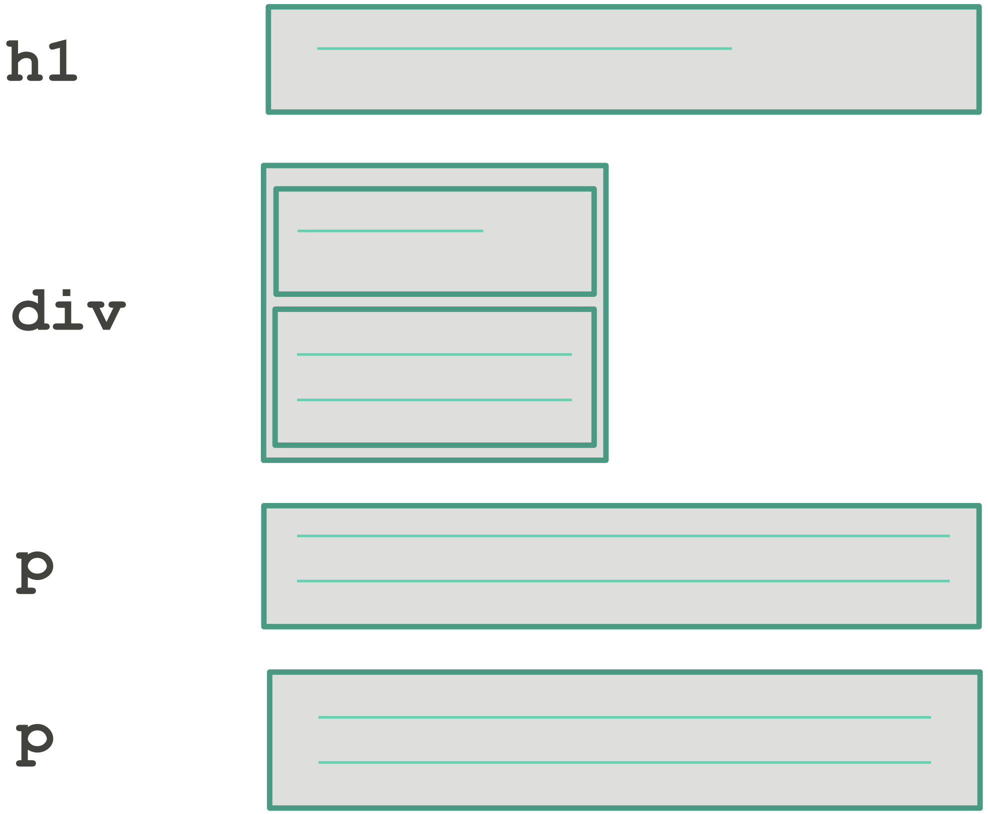
THE LIMITATIONS OF HTML’S BLOCK MODEL



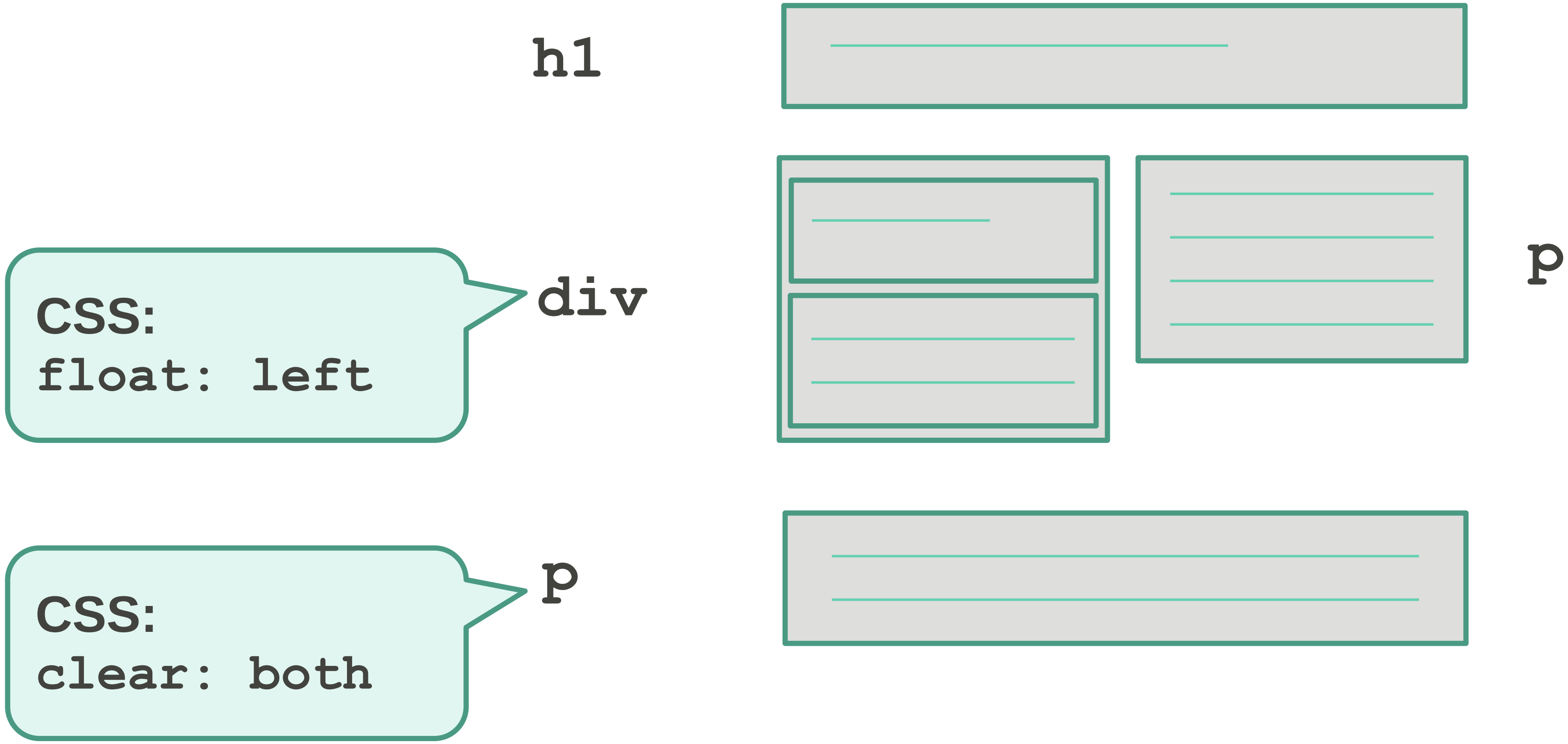
SIDE-BY-SIDE LAYOUT TRADITIONALLY SOLVED BY…

- Tables (deprecated for layout)
- **`Float`** property

ELEMENTS RENDERED IN NORMAL FLOW



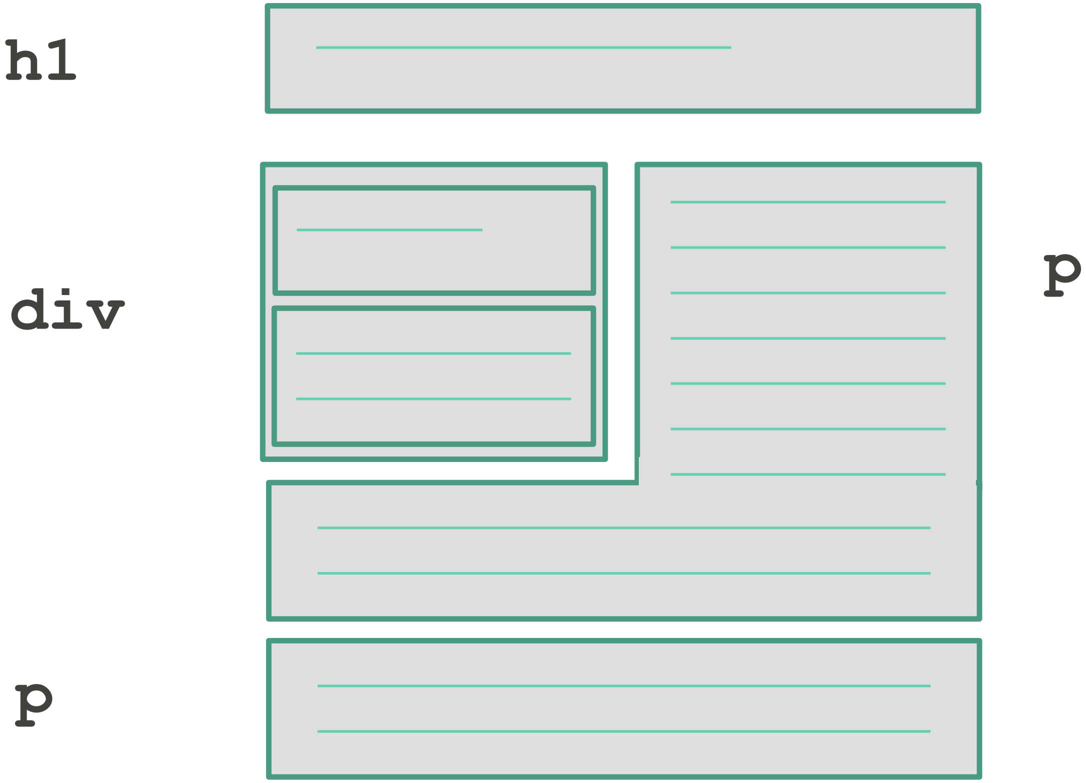
FLOAT PROPERTY



CSS:
float: left

CSS:
clear: both

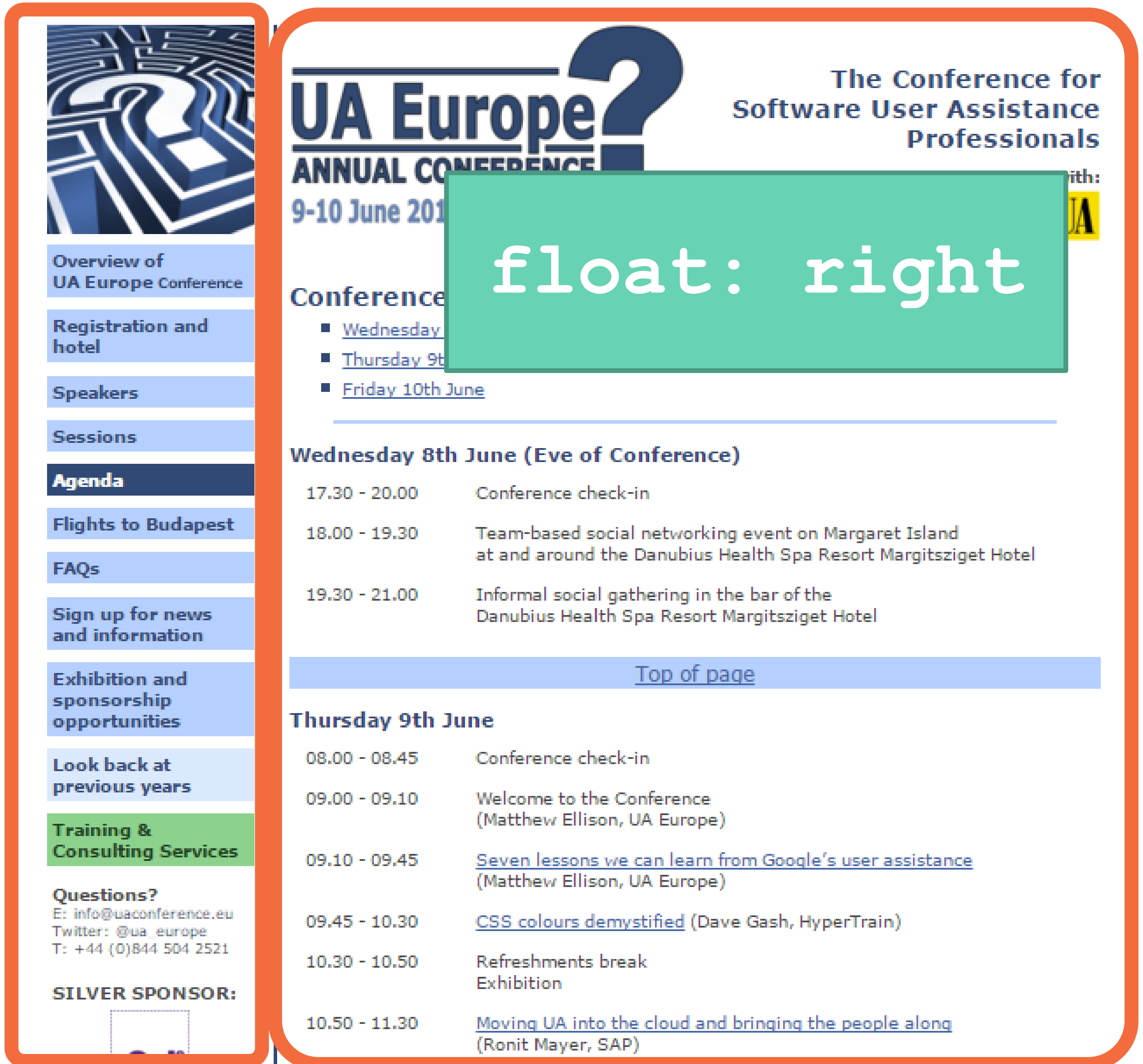
FLOAT PROPERTY



CSS:
`float: left`

CSS:
`clear: both`

EXAMPLE OF FLOAT-BASED LAYOUT



Overview of UA Europe Conference

Registration and hotel

Speakers

Sessions

Agenda

Flights to Budapest

FAQs

Sign up for news and information

Exhibition and sponsorship opportunities

Look back at previous years

Training & Consulting Services

Questions?
E: info@uaconference.eu
Twitter: @ua_europe
T: +44 (0)844 504 2521

SILVER SPONSOR:

UA Europe
ANNUAL CONFERENCE
9-10 June 201

The Conference for
Software User Assistance
Professionals

ith:

float: right

Conference

- Wednesday
- Thursday 9t
- Friday 10th June

Wednesday 8th June (Eve of Conference)

17.30 - 20.00	Conference check-in
18.00 - 19.30	Team-based social networking event on Margaret Island at and around the Danubius Health Spa Resort Margitsziget Hotel
19.30 - 21.00	Informal social gathering in the bar of the Danubius Health Spa Resort Margitsziget Hotel

Top of page

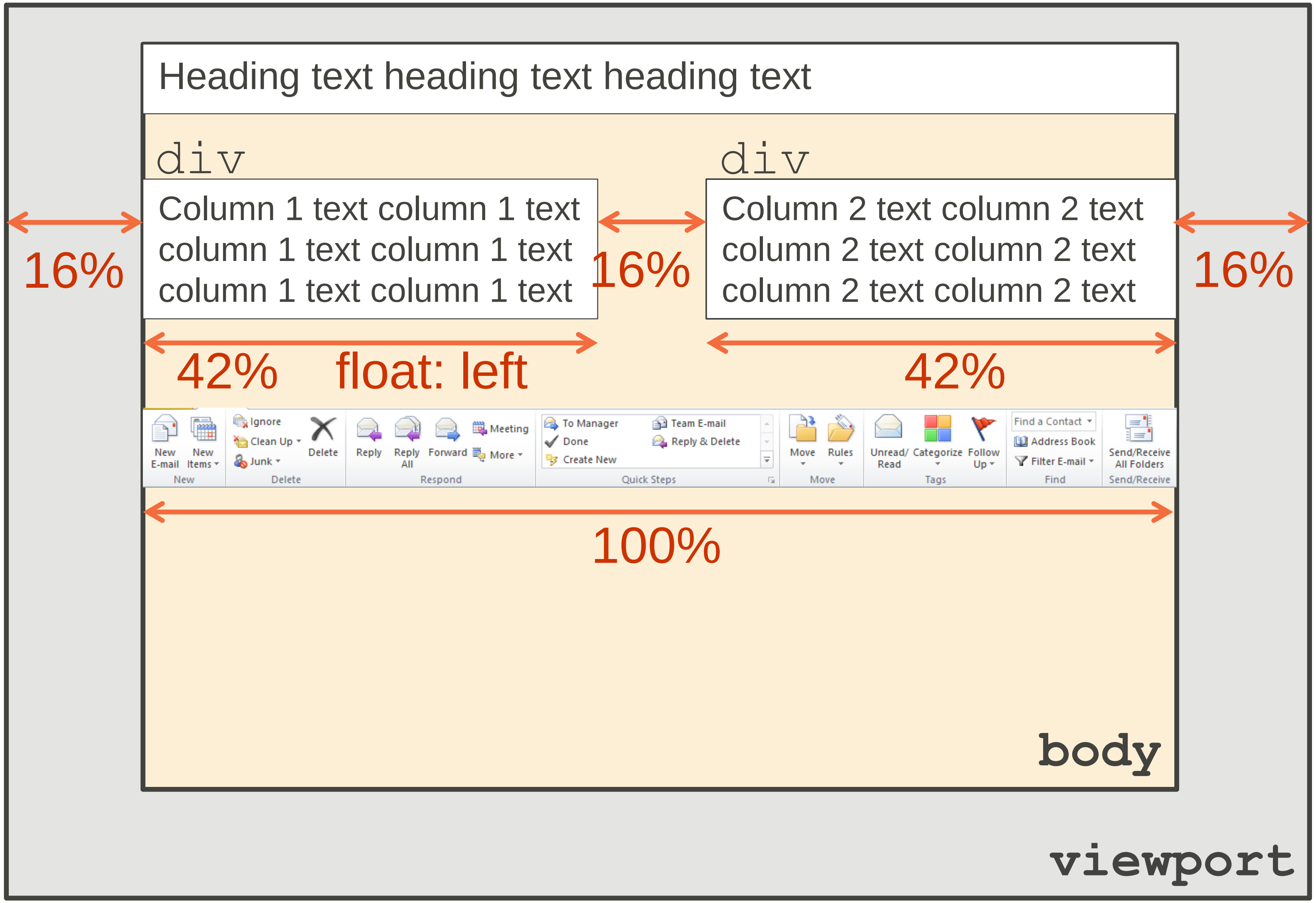
Thursday 9th June

08.00 - 08.45	Conference check-in
09.00 - 09.10	Welcome to the Conference (Matthew Ellison, UA Europe)
09.10 - 09.45	Seven lessons we can learn from Google's user assistance (Matthew Ellison, UA Europe)
09.45 - 10.30	CSS colours demystified (Dave Gash, HyperTrain)
10.30 - 10.50	Refreshments break Exhibition
10.50 - 11.30	Moving UA into the cloud and bringing the people along (Ronit Mayer, SAP)

WHAT ABOUT CSS POSITIONING?

- Introduced in CSS2
- `position` property:
  - `static`
  - `relative`
  - `absolute`
  - `fixed`
- Not easy to understand or work with

RESPONSIVE LAYOUT WITH MEDIA QUERIES



Heading text heading text heading text

div

Column 1 text column 1 text column 1 text column 1 text column 1 text column 1 text

div

Column 2 text column 2 text column 2 text column 2 text column 2 text column 2 text

16%

16%

16%

42% float: left

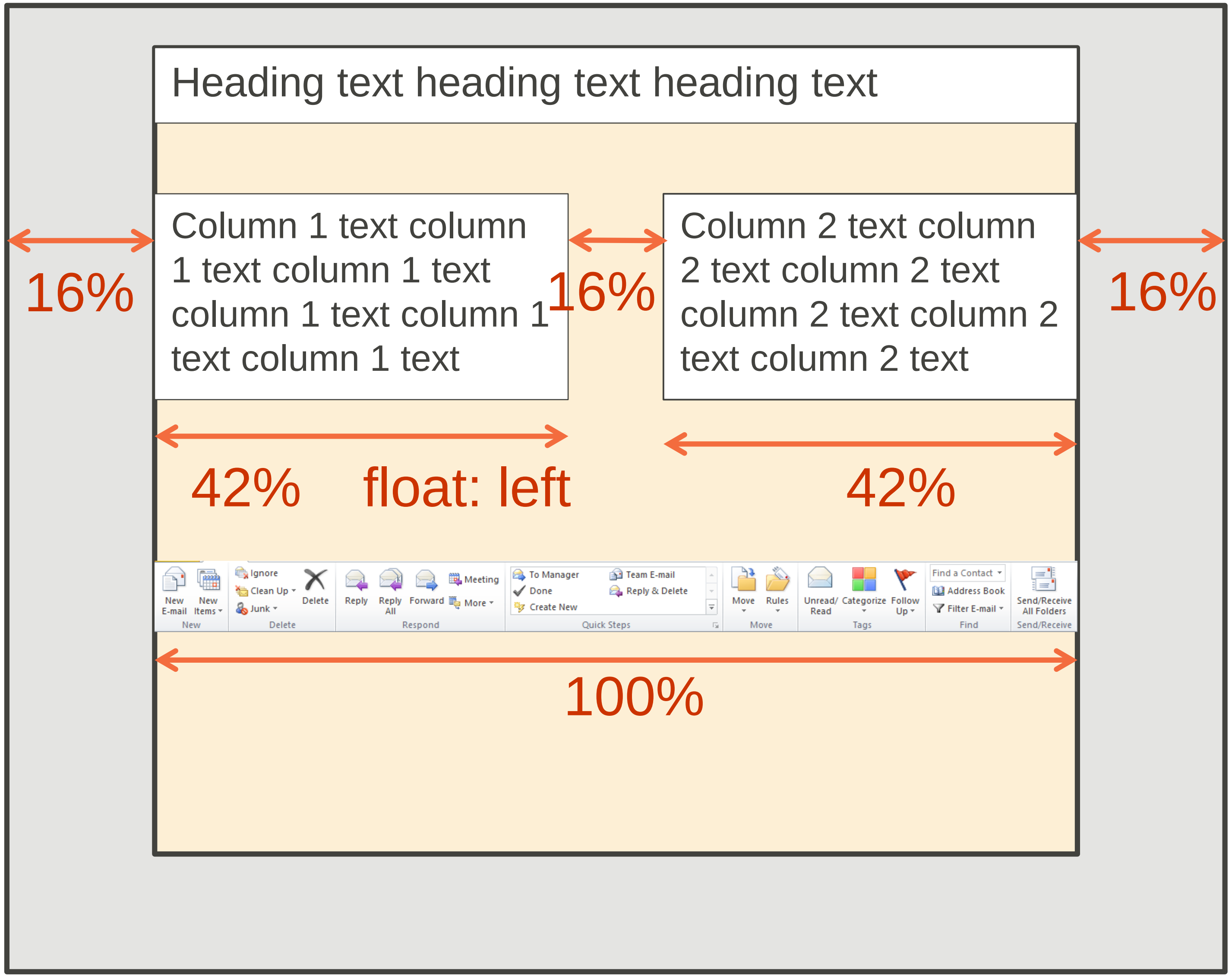
42%

100%

body

viewport

VIEWPORT LESS WIDE



Heading text heading text heading text

Column 1 text column 1 text column 1 text column 1 text column 1 text column 1 text

Column 2 text column 2 text column 2 text column 2 text column 2 text column 2 text

16%

16%

16%

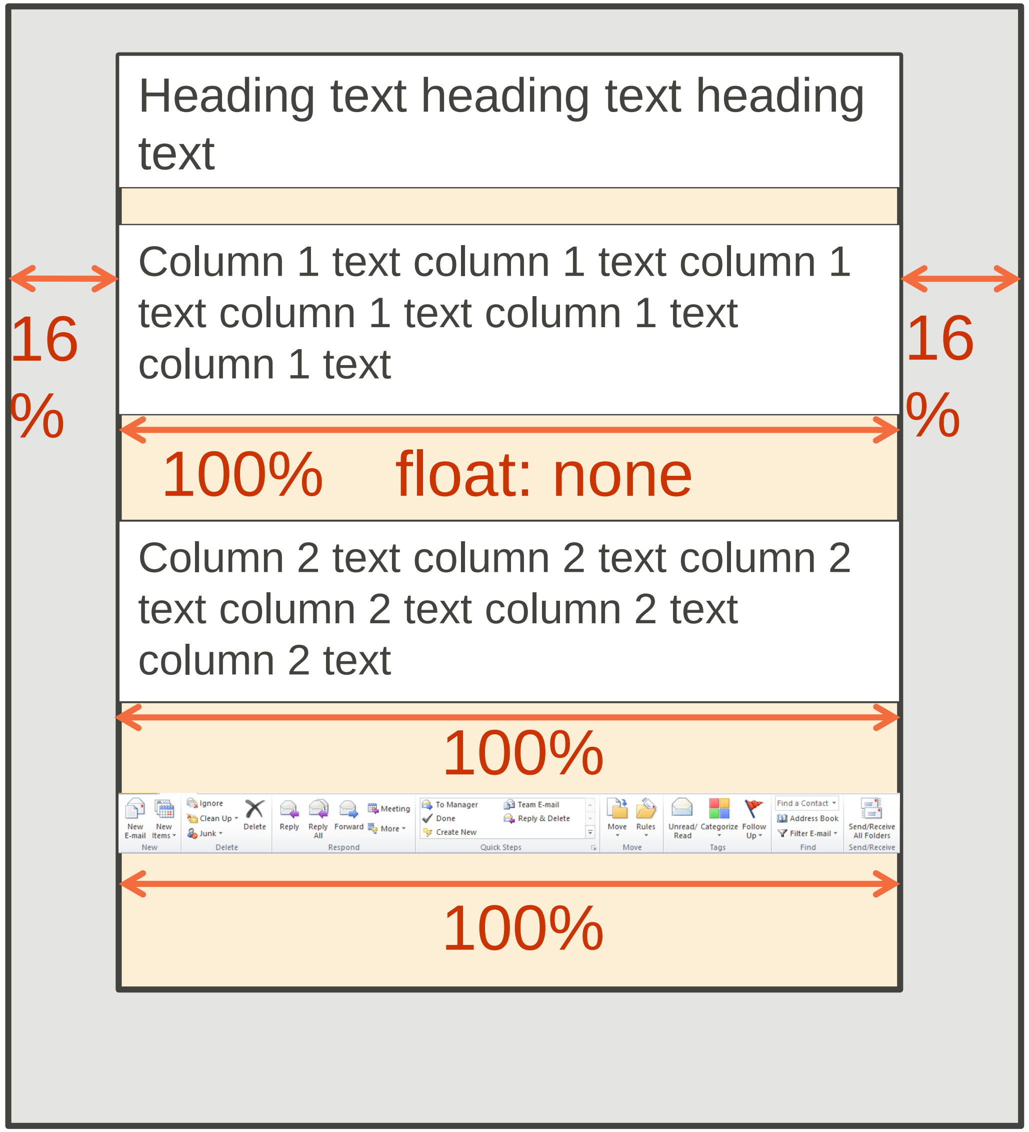
42%

float: left

42%

100%

PAST BREAKPOINT



Heading text heading text heading text

16 %

Column 1 text column 1 text column 1 text column 1 text column 1 text column 1 text

16 %

100% float: none

Column 2 text column 2 text column 2 text column 2 text column 2 text column 2 text

100%

100%

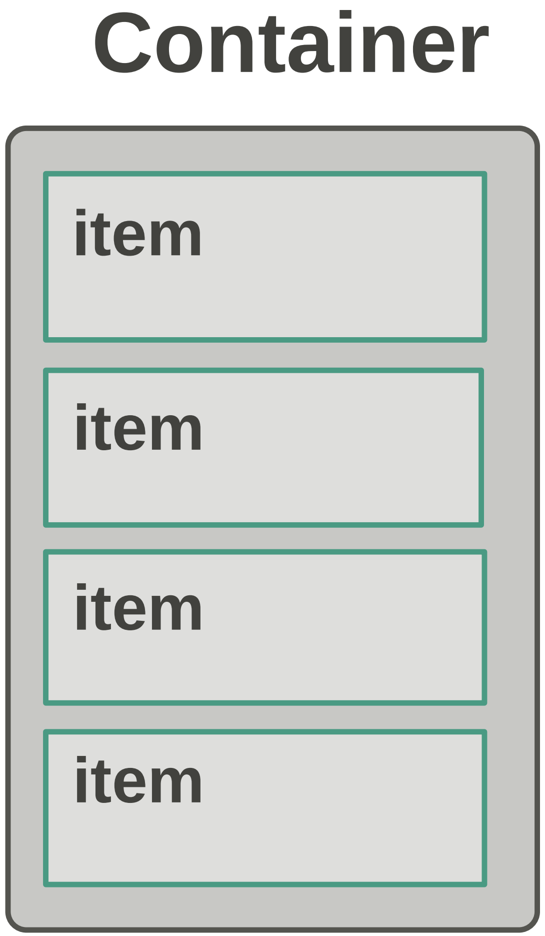
INTRODUCING CSS FLEXBOX

- Creates intelligent boxes that are stretchable, squeezable and capable of changing visual order
- Simple solution for layout paradigms that CSS has always struggled with:
  - vertical centering
  - equal heights

CONTAINER AND ITEMS

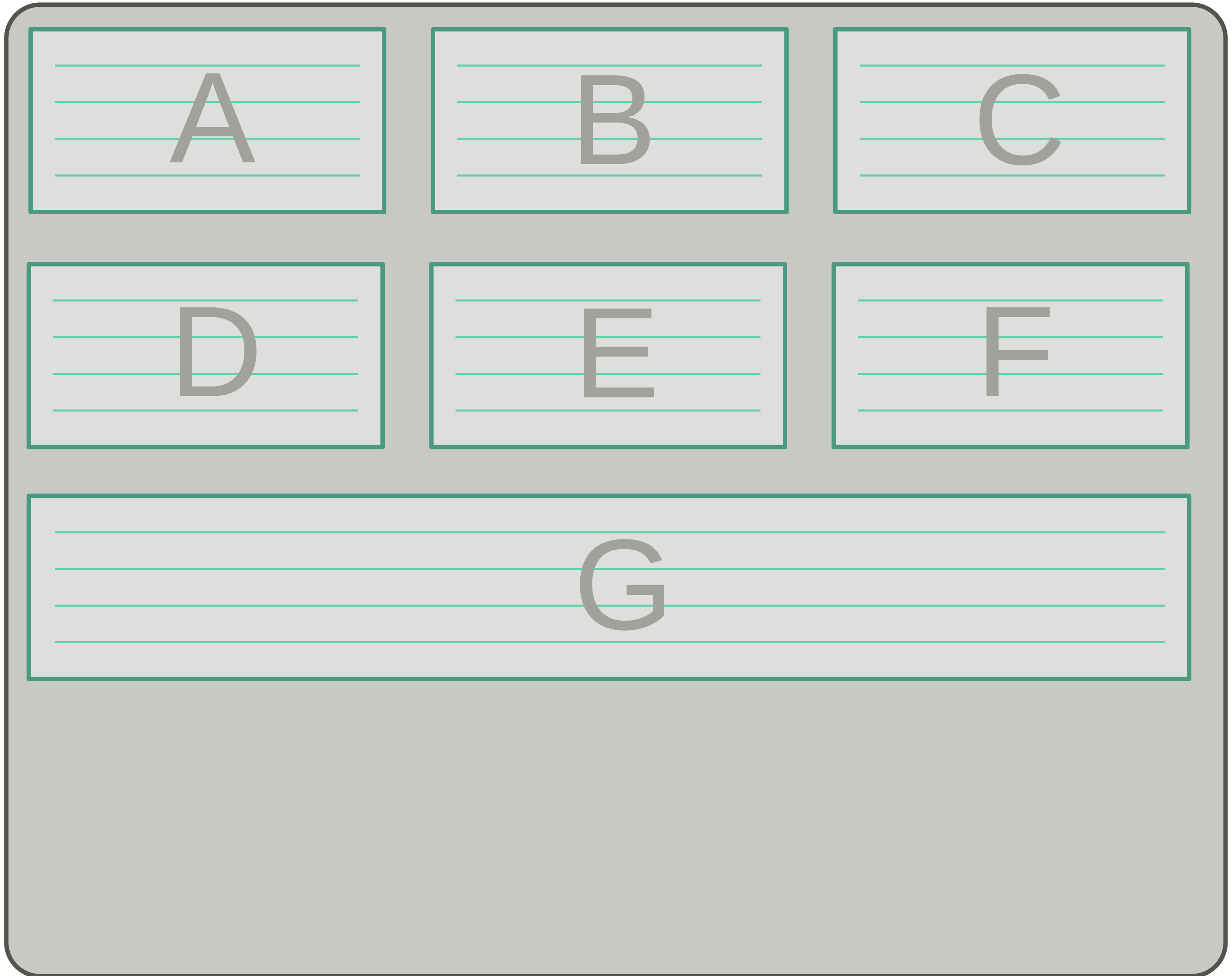
- Within a flex container, items typically:
  - Arrange themselves in rows or columns
  - expand to fill available free space or
  - shrink to prevent overflow

- Flex container's margins do NOT collapse with the margins of its content

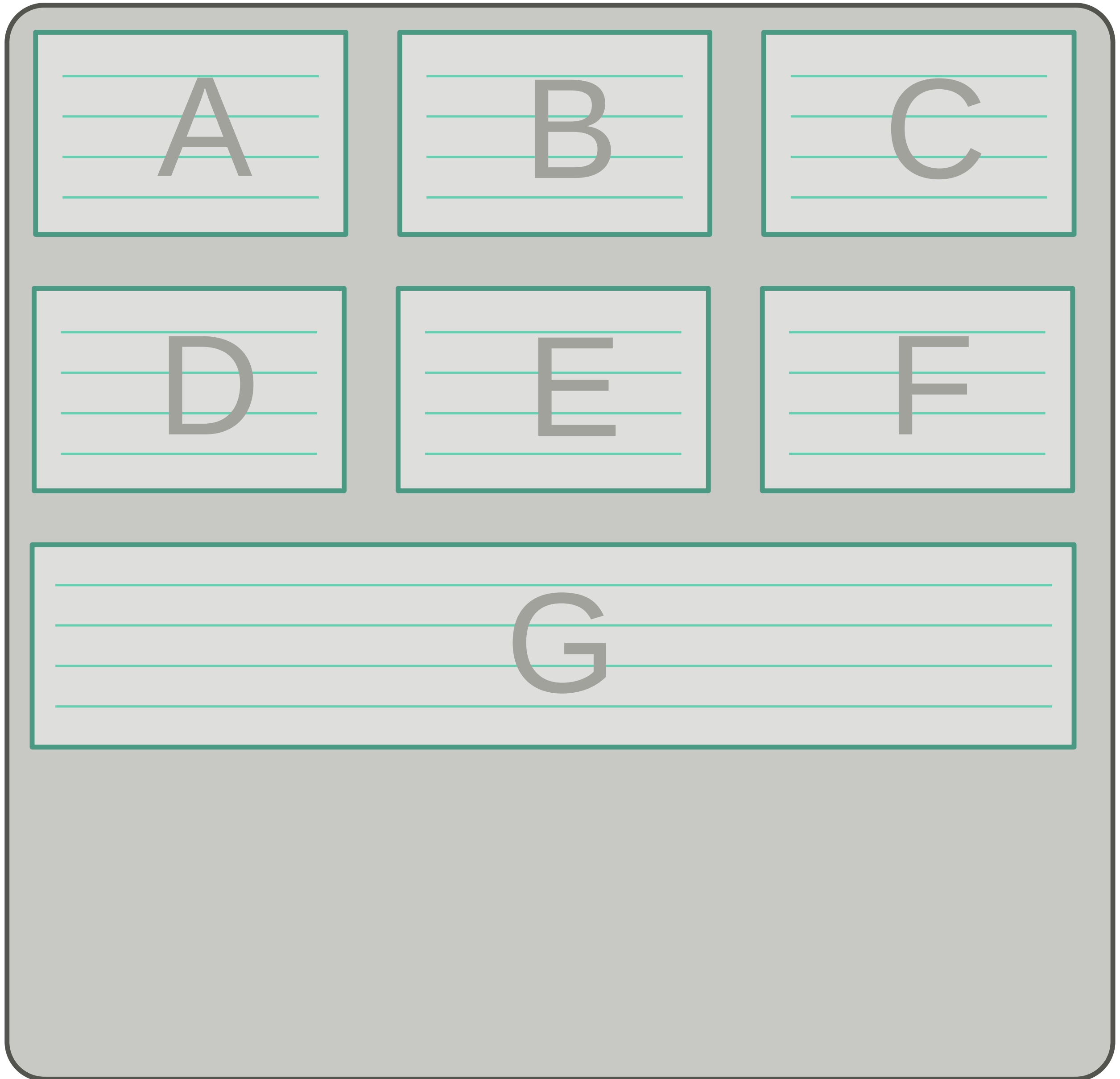


A Visual Demonstration of Flexbox principles…

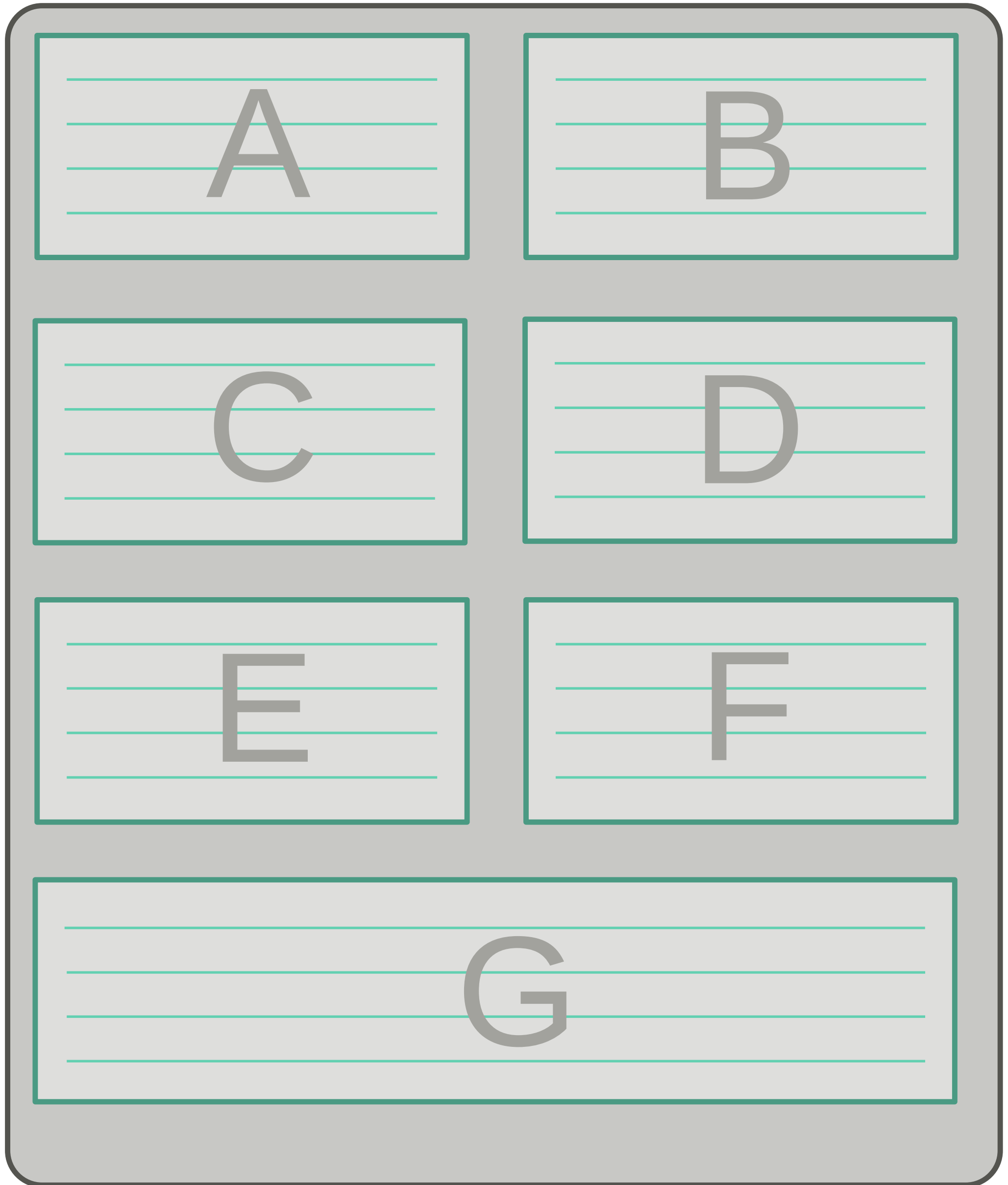
WIDE SCREEN (DESKTOP)



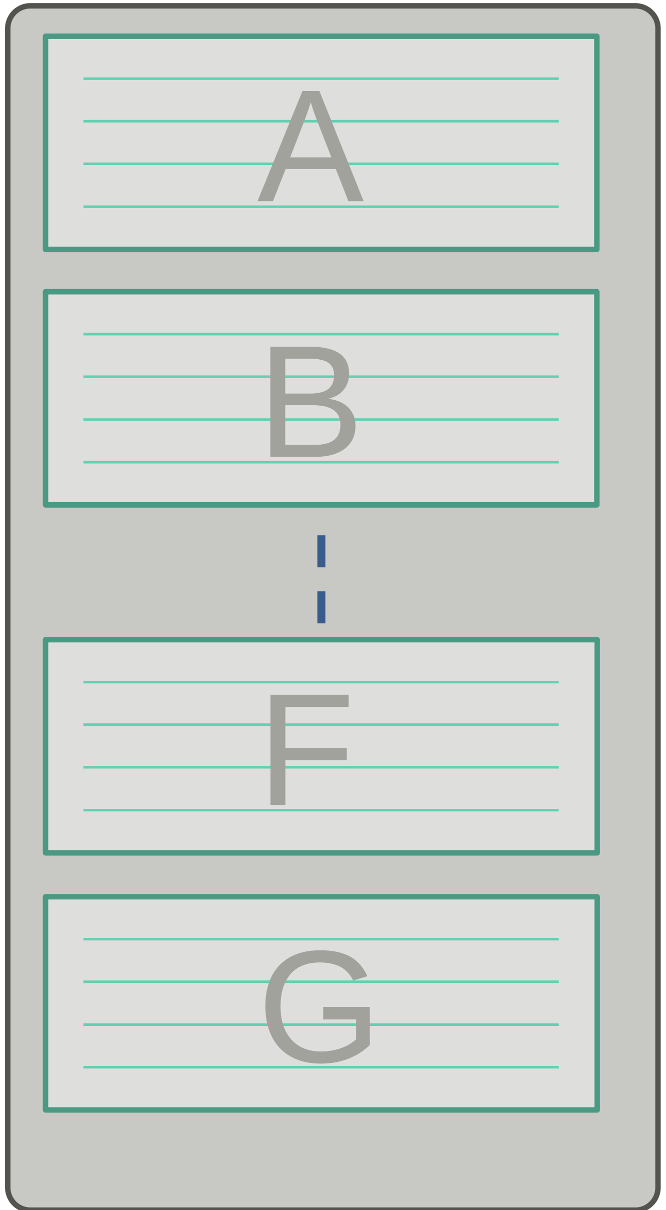
LESS WIDE



NARROWER



NARROW SCREEN (PHONE)

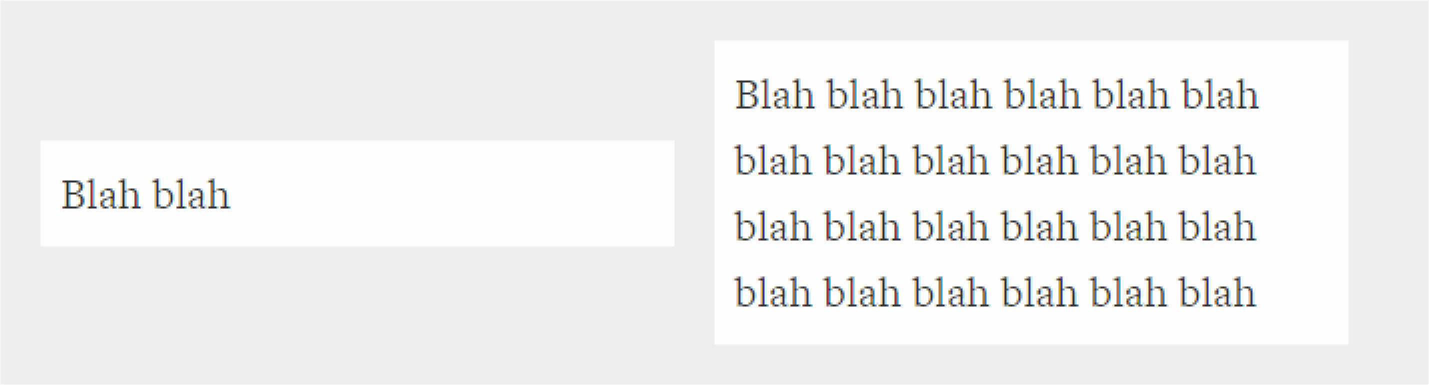


FLEXBOX SCENARIOS

- Vertical centering
- Unknown number of items within a container (photos, landing page, toolbar/menu items)
- Flexible layout for website components (header, sidebar, main, footer)

VERTICAL CENTERING

Vertical centering without flexbox is hopeful at best

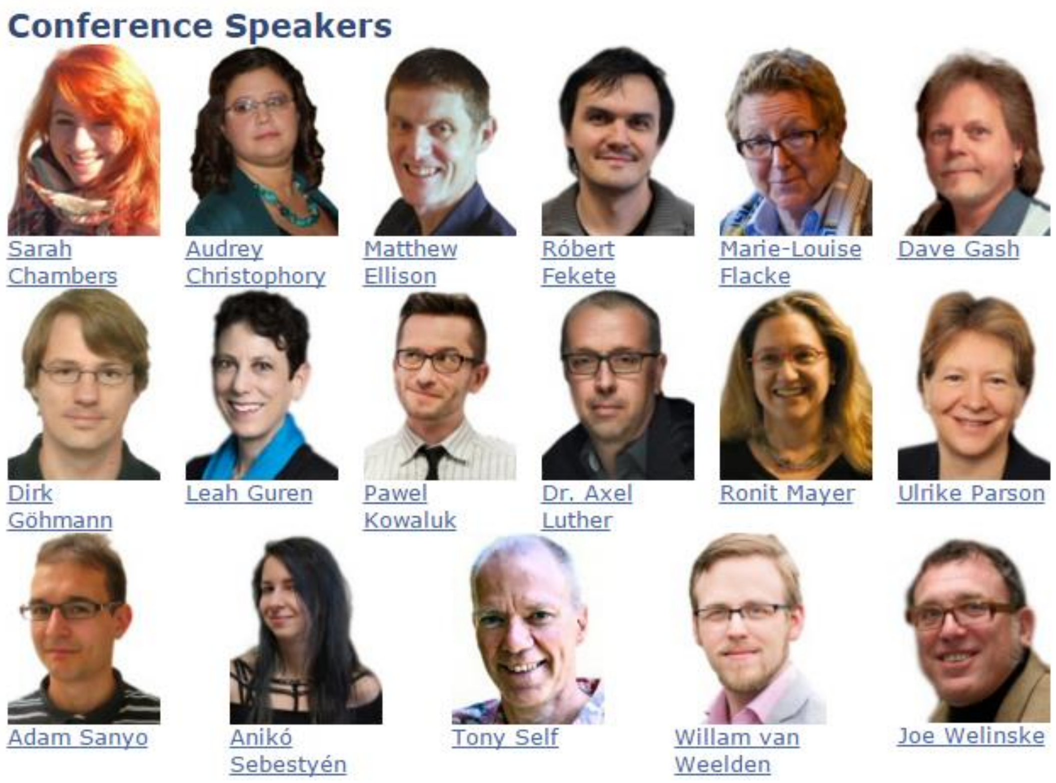


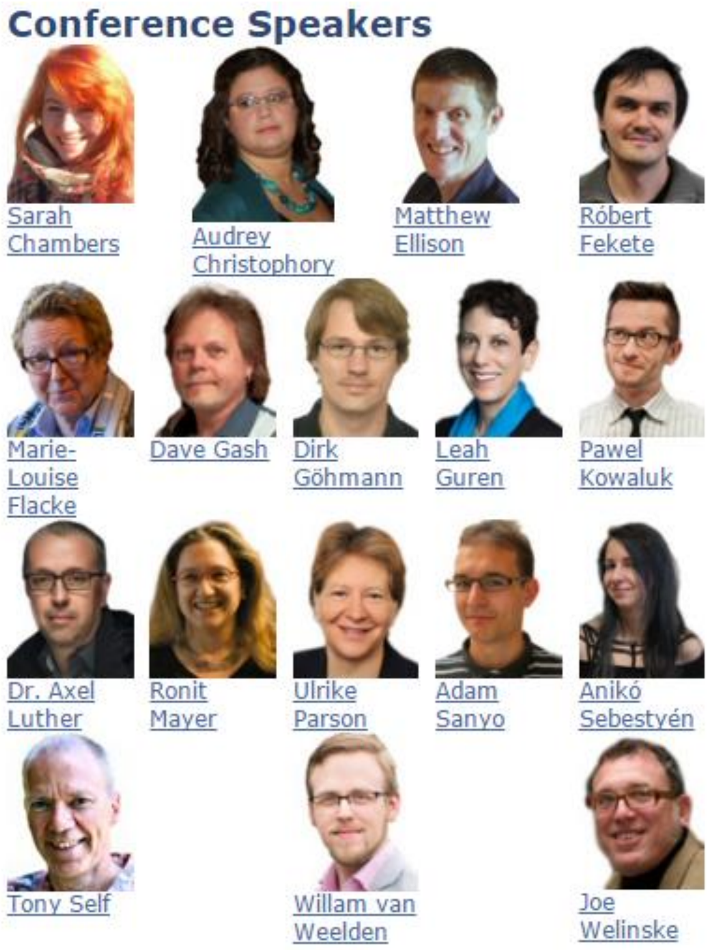
Blah blah

Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah

Demo by David Walsh at
davidwalsh.name/css-vertical-center-flexbox

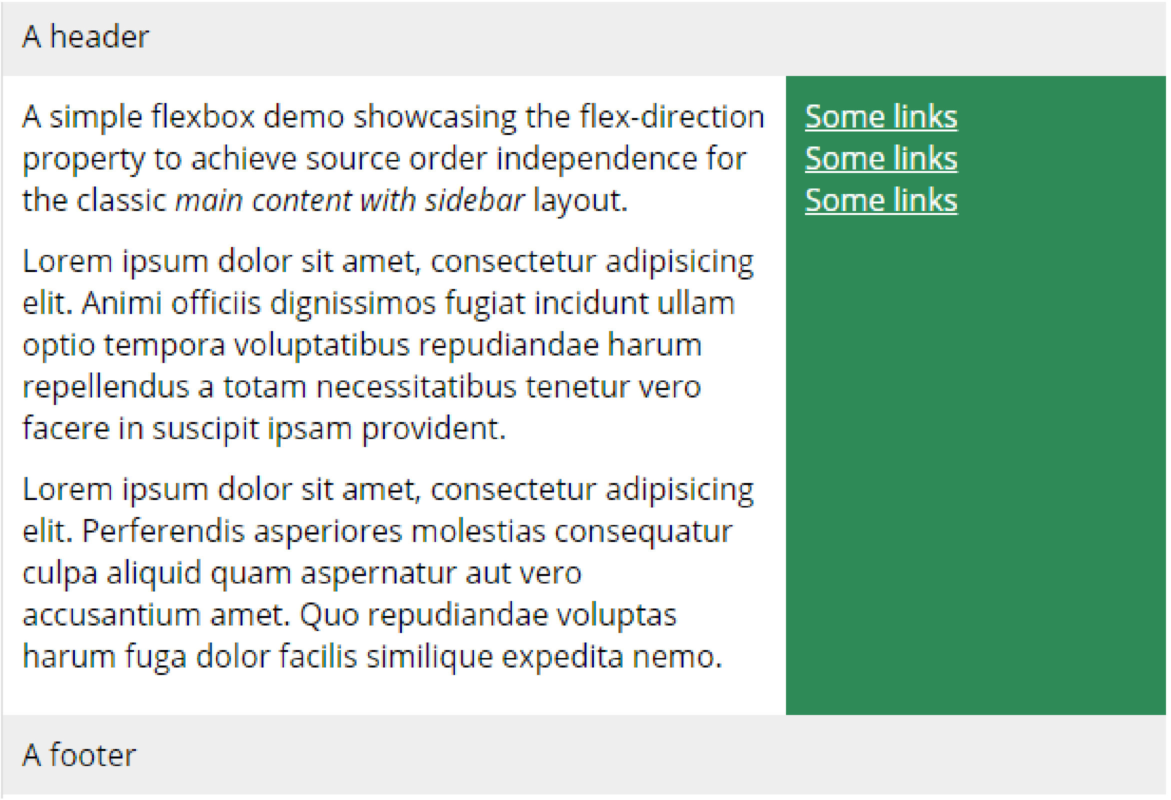
CONTINUOUS REARRANGEMENT OF ITEMS WITHIN A CONTAINER AS SCREEN SIZE CHANGES





UA Europe speakers at
www.uaconference.eu/speakers.html

FLEXIBLE LAYOUT FOR WEBSITE COMPONENTS



A header

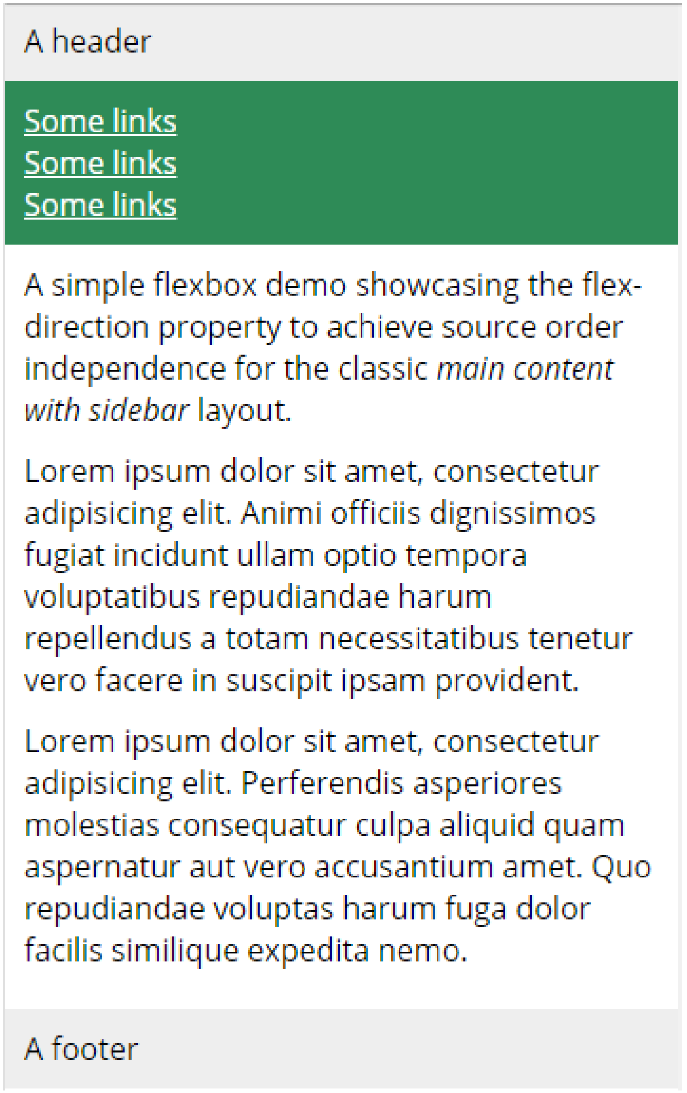
A simple flexbox demo showcasing the flex-direction property to achieve source order independence for the classic *main content with sidebar* layout.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Animi officiis dignissimos fugiat incidunt ullam optio tempora voluptatibus repudiandae harum repellendus a totam necessitatibus tenetur vero facere in suscipit ipsam provident.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Perferendis asperiores molestias consequatur culpa aliquid quam aspernatur aut vero accusantium amet. Quo repudiandae voluptas harum fuga dolor facilis similique expedita nemo.

Some links
Some links
Some links

A footer



A header

Some links
Some links
Some links

A simple flexbox demo showcasing the flex-direction property to achieve source order independence for the classic *main content with sidebar* layout.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Animi officiis dignissimos fugiat incidunt ullam optio tempora voluptatibus repudiandae harum repellendus a totam necessitatibus tenetur vero facere in suscipit ipsam provident.

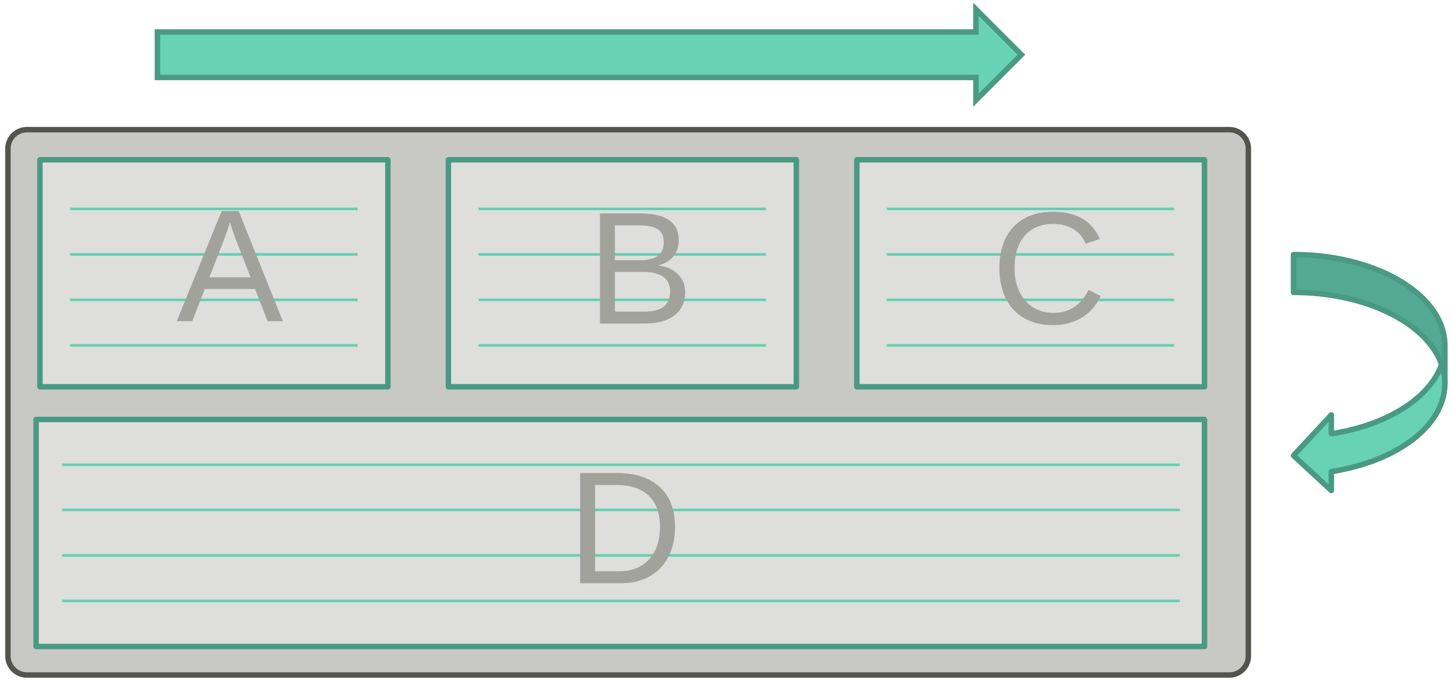
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Perferendis asperiores molestias consequatur culpa aliquid quam aspernatur aut vero accusantium amet. Quo repudiandae voluptas harum fuga dolor facilis similique expedita nemo.

A footer

Demo by Karen Menezes at
http://codepen.io/imohkay/details/JoYoRE

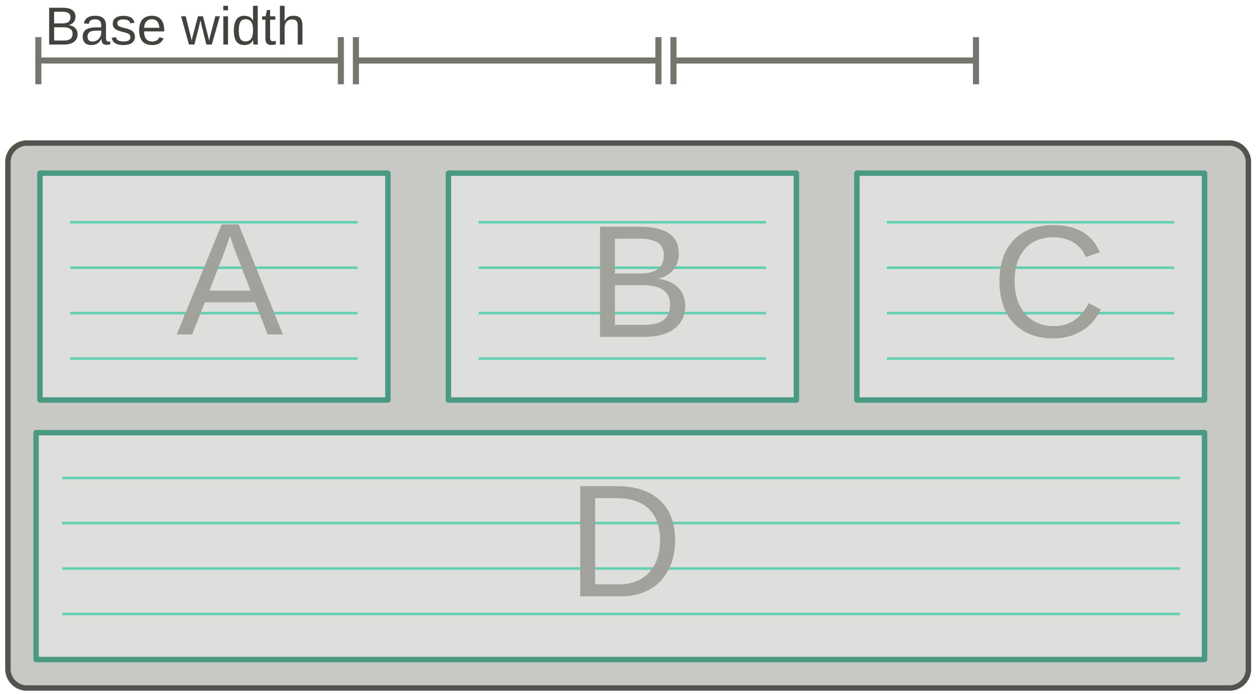
BASIC SPECIFICATIONS THAT WE CAN SPECIFY USING CSS

- Arrangement of items (row or column)
- …and whether they are allowed to wrap



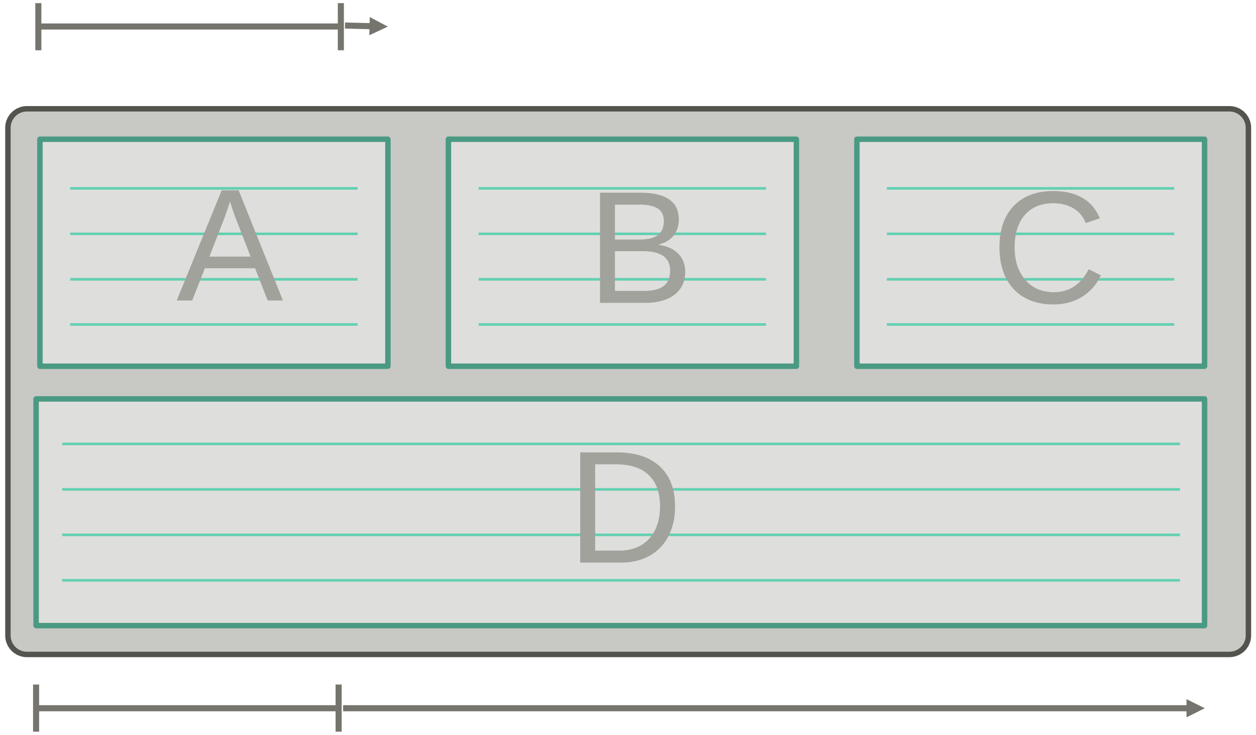
BASIC SPECIFICATIONS THAT WE CAN SPECIFY USING CSS

- Base width of items
  (used to calculate how many items can be fitted within each row)



BASIC SPECIFICATIONS THAT WE CAN SPECIFY USING CSS

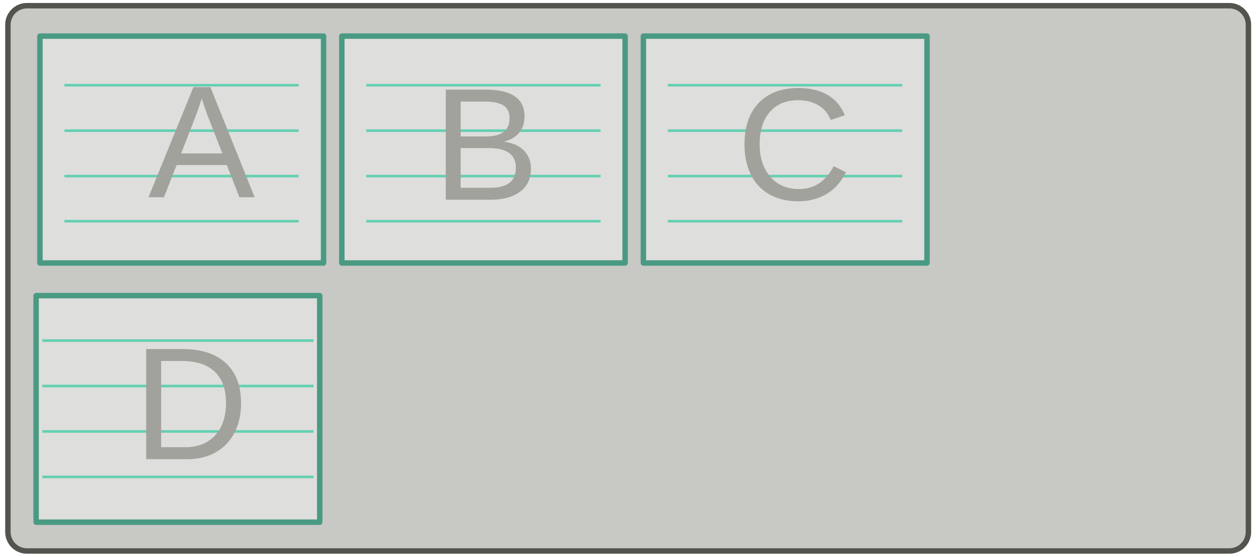
- Whether the width of items is allowed to grow to fill the available space



BASIC SPECIFICATIONS THAT WE CAN SPECIFY USING CSS

- How items are justified within the container if they are not allowed to grow

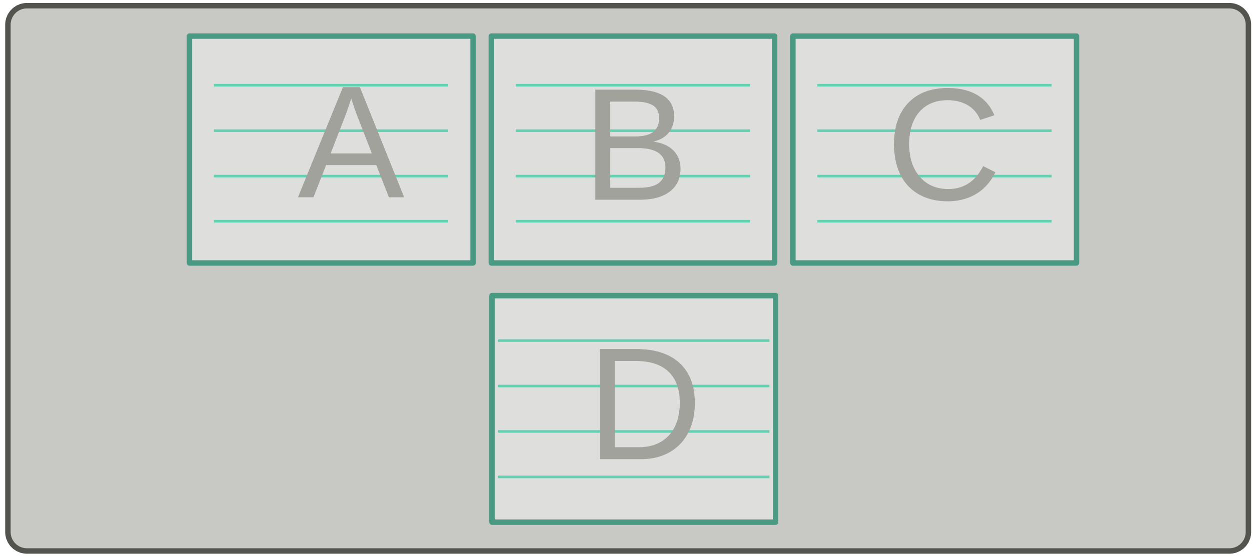
`flex-start`



BASIC SPECIFICATIONS THAT WE CAN SPECIFY USING CSS

- How items are justified within the container if they are not allowed to grow

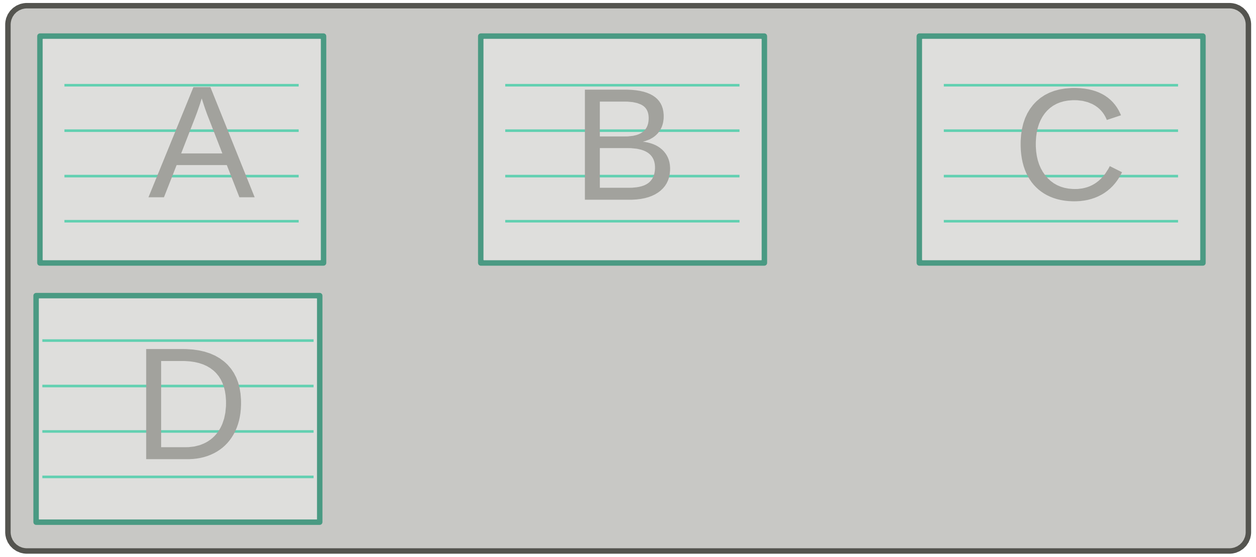
`center`



BASIC SPECIFICATIONS THAT WE CAN SPECIFY USING CSS

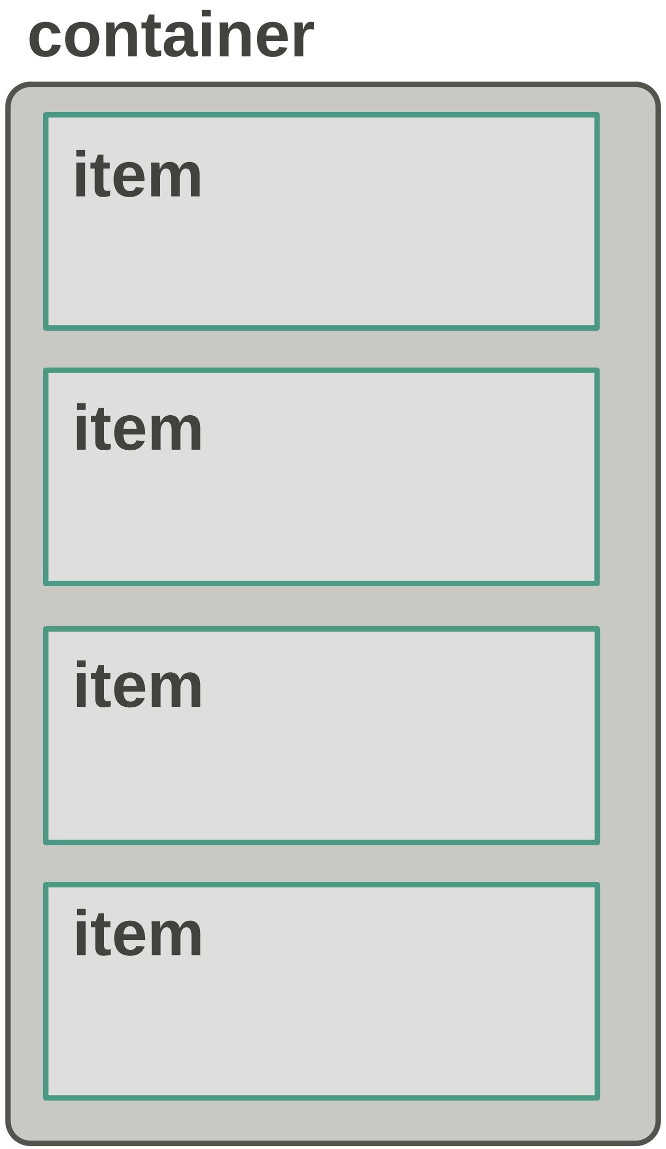
- How items are justified within the container if they are not allowed to grow

`space-between`



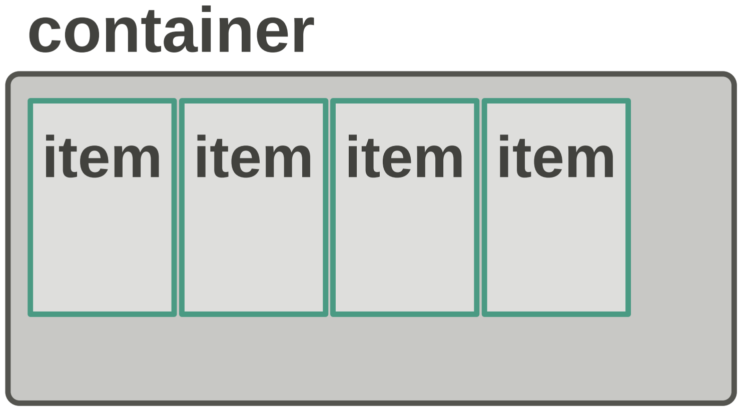
Building the Flexbox CSS Code…

HTML CONTENT



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

CSS - CONTAINER PROPERTY: `FLEX`



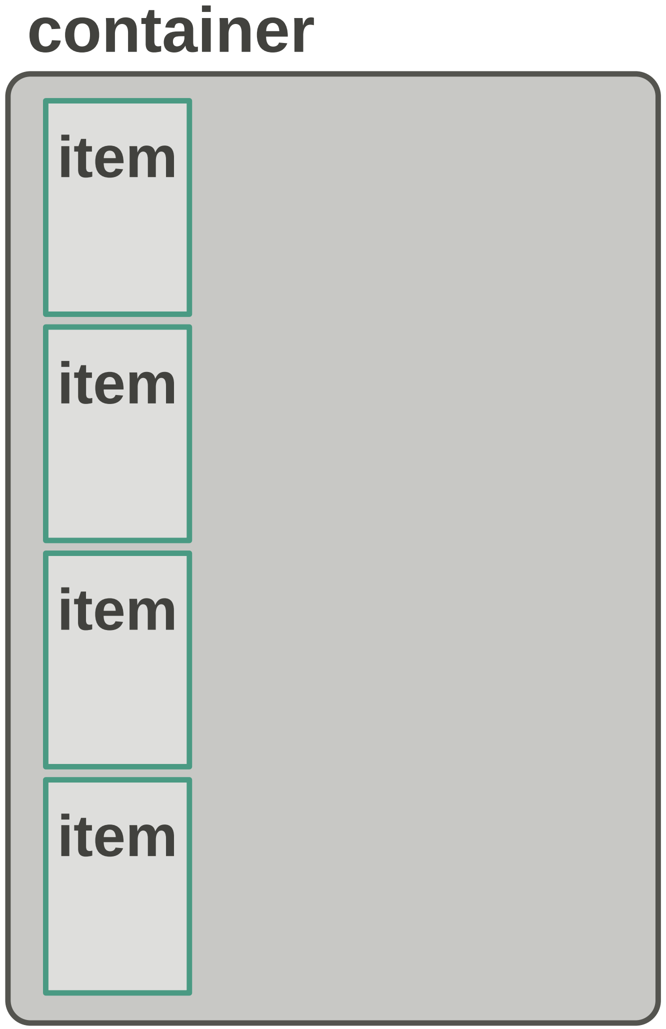
```
div#container
{
        display: flex;
}

div.item
{
}
```

The only property that you **need** to specify

Default behavior:

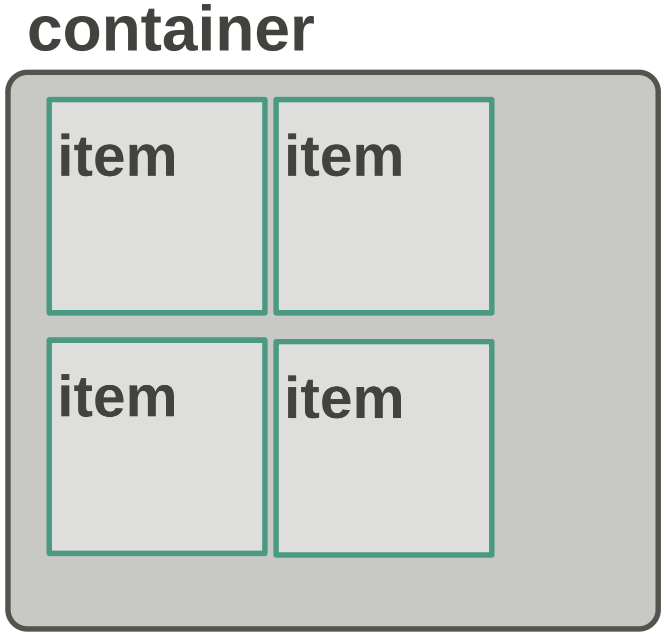
- Arrangement by row
- No wrap
- Justified to start of flex container
- Items shrink to fit within container
- Items do not grow to fill available space

CONTAINER PROPERTY: `FLEX-DIRECTION`



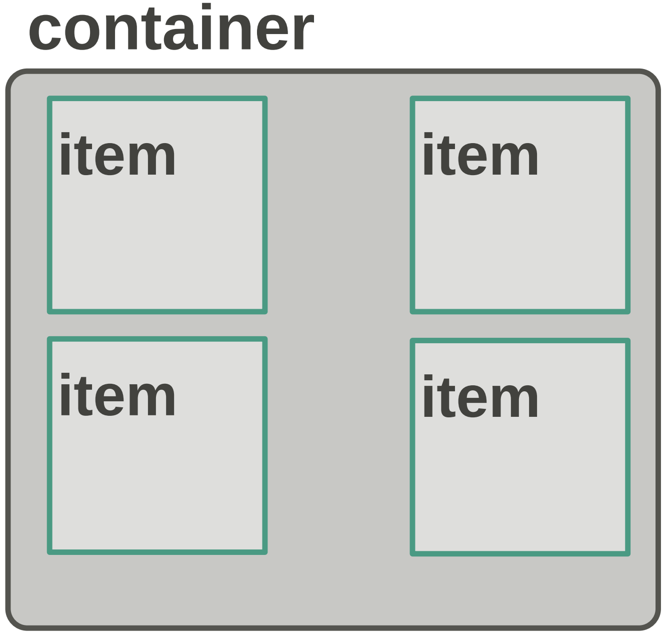
```
div#container
{
        display: flex;
        flex-direction: column;
}

div.item
{
}
```

CONTAINER PROPERTY: `FLEX-WRAP`



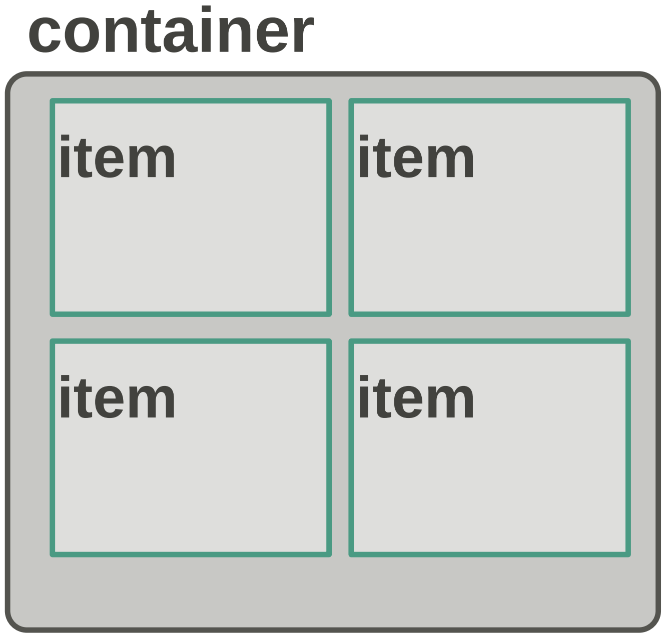
```
div#container
{
        display: flex;
        flex-direction: row;
        flex-wrap: wrap;
}

div.item
{
}
```

CONTAINER PROPERTY: `JUSTIFY-CONTENT`



```
div#container
{
        display: flex;
        flex-direction: row;
        flex-wrap: wrap;
        justify-content:
                space-between;
}

div.item
{
}
```

ITEM PROPERTY: `FLEX-GROW`



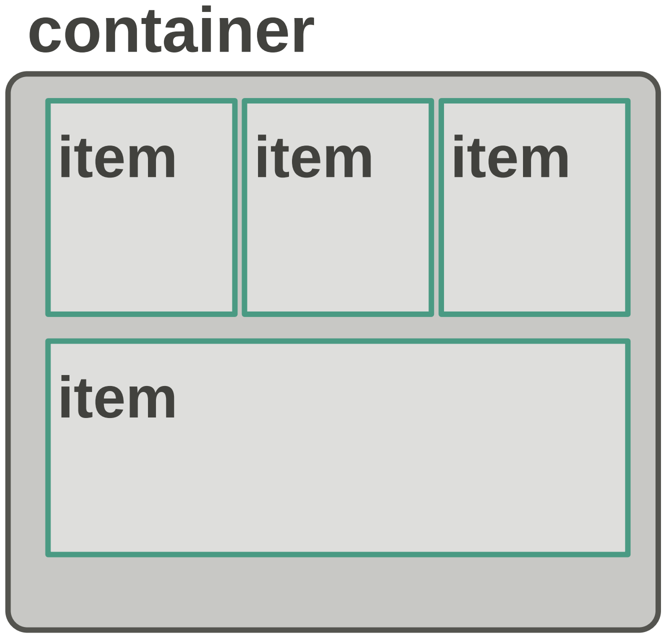
```
div#container
{
        display: flex;
        flex-direction: row;
        flex-wrap: wrap;
        justify-content:
                space-between;
}
```

This property is now redundant

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

All items within a row grow by same amount to fill the available space

ITEM PROPERTY: `FLEX-BASIS`



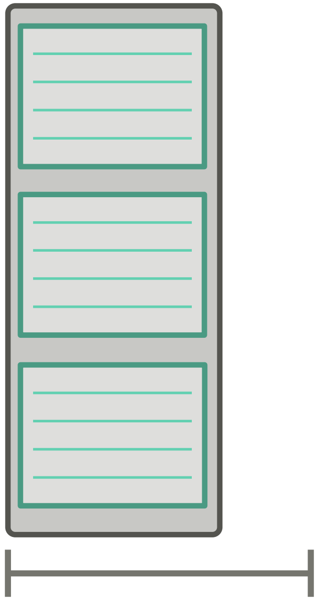
```
div#container
{
        display: flex;
        flex-direction: row;
        flex-wrap: wrap;
        justify-content:
                    space-between;
}

div.item
{
        flex-grow: 1;
        flex-basis: 28%;
}
```

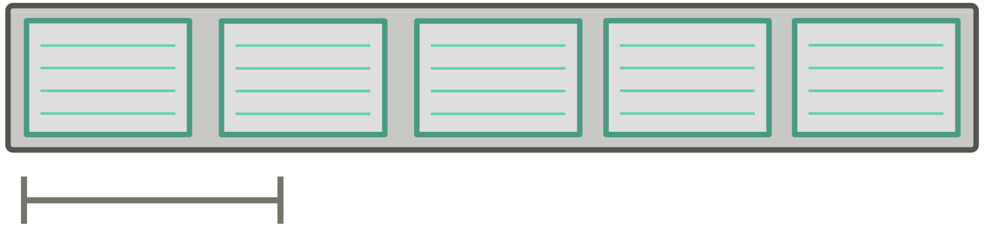
Items grow to fill row

Used to calculate how many items will fit within a row

`FLEX-SHRINK` PROPERTY

- As long as **`flex-shrink`** is not 0, items will shrink if:
  - an item is on its own in a row, and the browser width is less than the **`flex-basis`** property
    or
  - if **`flex-wrap`** has a value of **`nowrap`** and the total of **`flex-basis`** properties for all items is greater than the width of the container





ALL CONTAINER FLEXBOX PROPERTIES

display:	flex;
flex-direction:	**row** | column | row-reverse | column-reverse;
flex-wrap:	wrap | **nowrap** | wrap-reverse;
justify-content:	**flex-start** | flex-end | center | space-between | space-around;
align-items:	**stretch** | flex-start | flex-end | center | baseline;
align-content:	**stretch** | flex-start | flex-end | center | space-between | space-around;

ALL ITEM FLEXBOX PROPERTIES

`order:` `<number>;` default is **0**

`flex-grow:` `<number>;` default is **0** (it will not be allowed to grow)

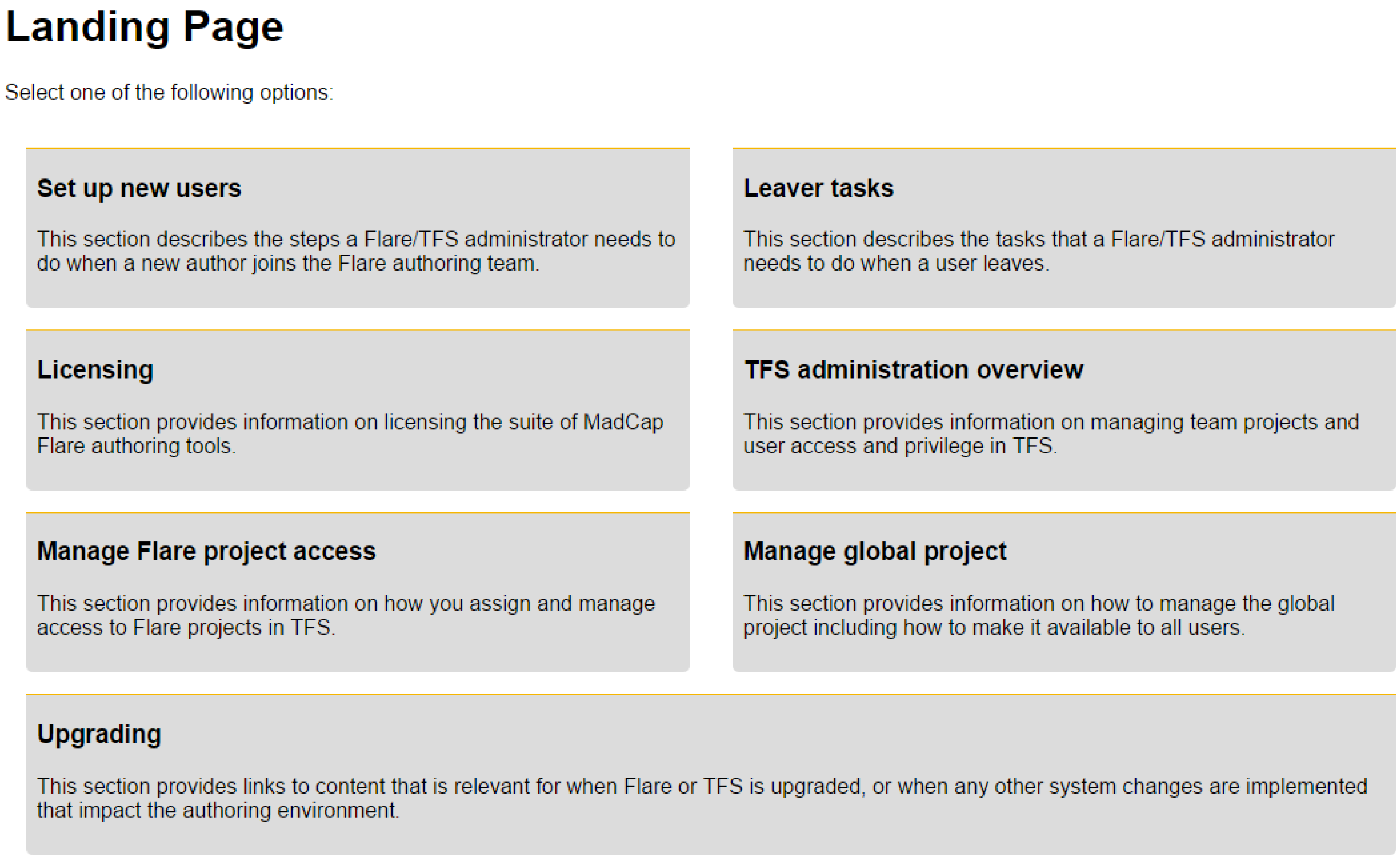
`flex-shrink:` `<number>;` default is **1** (it will be allowed to shrink)

`flex-basis:` **`auto`** `| <width/height in px or %>;`

(**`auto`** means it will be sized automatically
based on its width property)

`align-self:` **`auto`** `| flex-start | flex-end | center |`
`baseline | stretch;`

DEMO OF A MORE COMPLETE SOLUTION



Landing Page

Select one of the following options:

Set up new users

This section describes the steps a Flare/TFS administrator needs to do when a new author joins the Flare authoring team.

Leaver tasks

This section describes the tasks that a Flare/TFS administrator needs to do when a user leaves.

Licensing

This section provides information on licensing the suite of MadCap Flare authoring tools.

TFS administration overview

This section provides information on managing team projects and user access and privilege in TFS.

Manage Flare project access

This section provides information on how you assign and manage access to Flare projects in TFS.

Manage global project

This section provides information on how to manage the global project including how to make it available to all users.

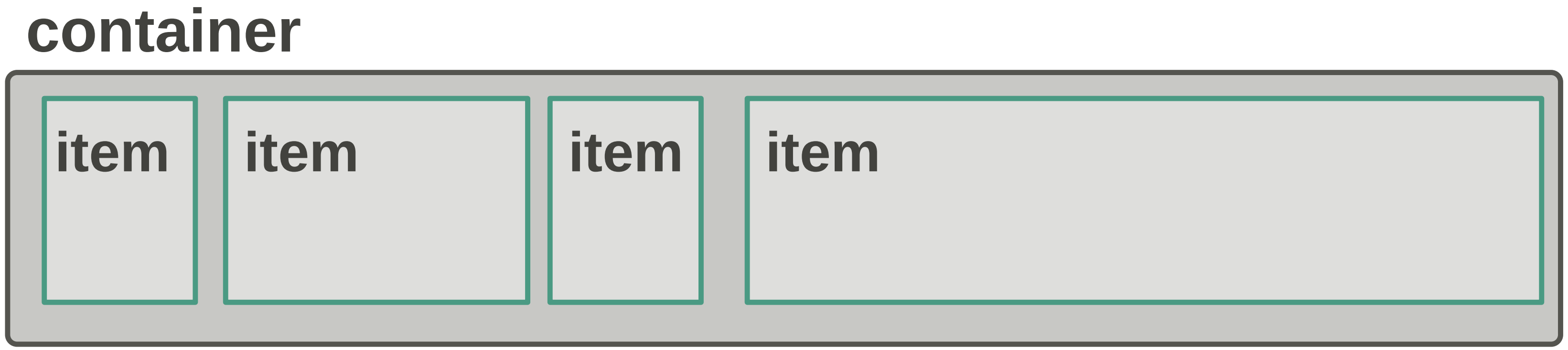
Upgrading

This section provides links to content that is relevant for when Flare or TFS is upgraded, or when any other system changes are implemented that impact the authoring environment.

ABSOLUTE FLEX

- flex-basis of 0:
  - a way of easily controlling relative widths of items
  - **ALL** space within the container is assigned to items according to their relative flex-grow values
  - width property of the items is ignored

ABSOLUTE FLEX



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

div.item                          { flex-basis: 0; }

div.item:nth-of-type(odd)         { flex-grow: 1; }

div.item:nth-of-type(2)           { flex-grow: 2; }

div.item:nth-of-type(4)           { flex-grow: 5; }
```

FLEX SHORTHAND

- W3C encourages authors to use the flex shorthand property for items
- shorthand “correctly resets any unspecified components to accommodate common uses”

```
flex: <flex-grow> <flex-shrink> <flex-basis>
```

- For example:

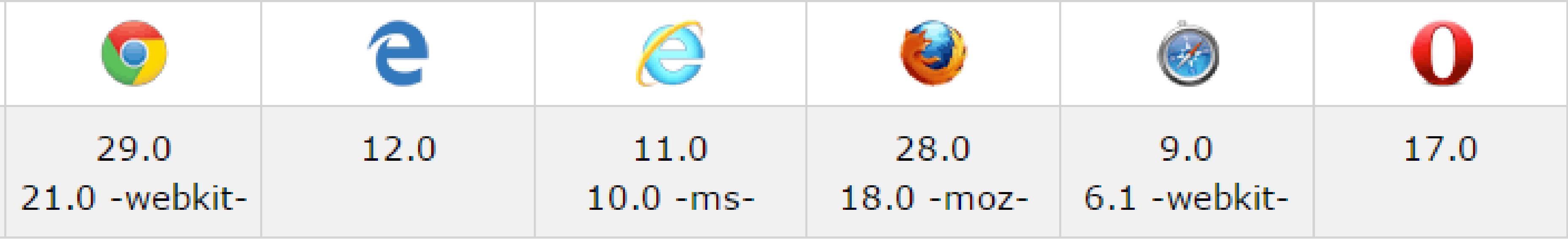
```
flex: 1 1 auto
```

COMMON VALUES

Shorthand	Equivalent to
`flex: initial`	`flex: 0 1 auto`
`flex: auto`	`flex: 1 1 auto`
`flex: none`	`flex: 0 0 auto`
`flex: <+ve number>`	`flex: <+ve number> 1 0`

BROWSER SUPPORT

- Flexbox is supported in all of the latest browsers

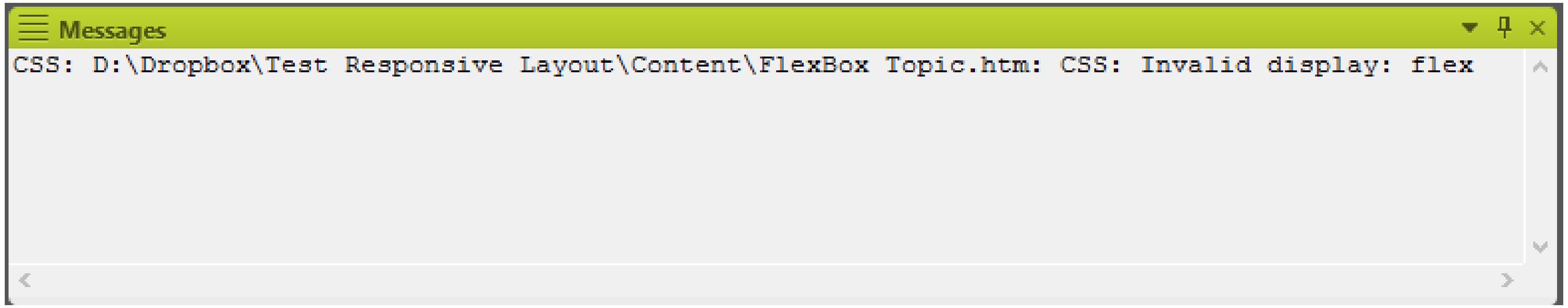


29.0 21.0 -webkit-	12.0	11.0 10.0 -ms-	28.0 18.0 -moz-	9.0 6.1 -webkit-	17.0

- If you need to support older browsers, check out caniuse to see what your options are

ISSUES WORKING WITH FLEXBOX IN FLARE

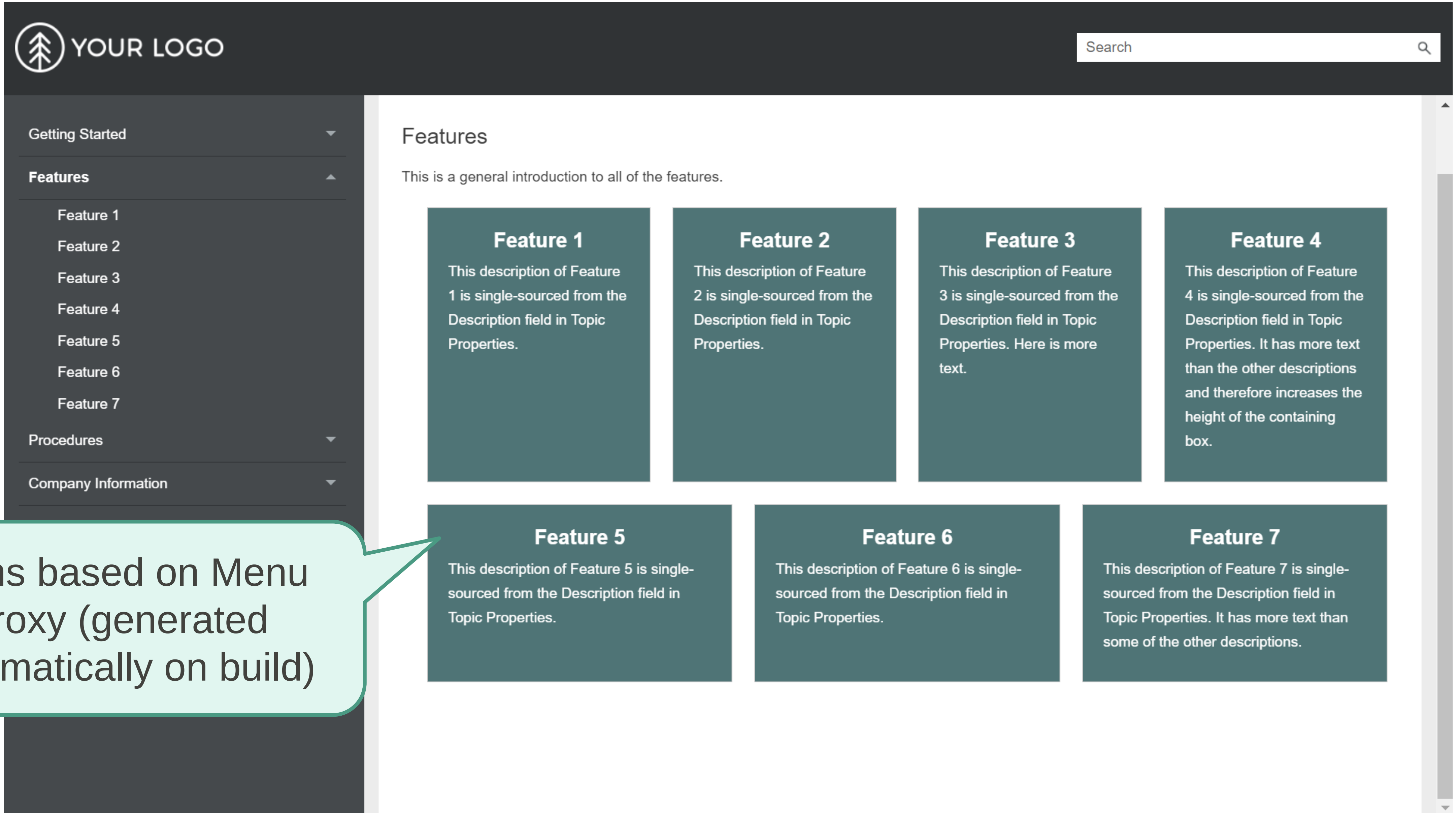
- Flex property not recognized



```
CSS: D:\Dropbox\Test Responsive Layout\Content\FlexBox Topic.htm: CSS: Invalid display: flex
```

- Layout is not shown correctly in XML Editor (but is correct in Preview)

REAL LIFE IMPLEMENTATION OF CSS FLEXBOX IN FLARE



Features

This is a general introduction to all of the features.

Feature 1

This description of Feature 1 is single-sourced from the Description field in Topic Properties.

Feature 2

This description of Feature 2 is single-sourced from the Description field in Topic Properties.

Feature 3

This description of Feature 3 is single-sourced from the Description field in Topic Properties. Here is more text.

Feature 4

This description of Feature 4 is single-sourced from the Description field in Topic Properties. It has more text than the other descriptions and therefore increases the height of the containing box.

Feature 5

This description of Feature 5 is single-sourced from the Description field in Topic Properties.

Feature 6

This description of Feature 6 is single-sourced from the Description field in Topic Properties.

Feature 7

This description of Feature 7 is single-sourced from the Description field in Topic Properties. It has more text than some of the other descriptions.

Items based on Menu proxy (generated automatically on build)

REFERENCES AND FURTHER READING

- https://www.w3.org/TR/css-flexbox/
- https://cvan.io/flexboxin5/ (Great!)
- https://css-tricks.com/snippets/css/a-guide-to-flexbox/
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Using_CSS_flexible_boxes
- http://caniuse.com/#feat=flexbox
- https://www.smashingmagazine.com/2015/03/harnessing-flexbox-for-todays-web-apps/
- https://scotch.io/tutorials/a-visual-guide-to-css3-flexbox-properties

Thanks for attending!
Final questions?

Matthew Ellison
matthew@uaeurope.com

matthew@uaeurope.com