HTML Layouts using Flexbox + CSS Grids

author: thomasburleson@gmail.com
date: Thursday January 4th 2018

This document aims to explain in detail the enhancements to the Angular Flex-Layout library to support layouts with either (or both) FlexBox CSS and CSS Grid.

Flexbox CSS APIs

The current @angular/flex-layout library supports directive-based API to layout HTML UI elements using Flexbox CSS. The API uses HTML directive syntax to support both static and responsive layouts.

The Angular Flex-Layout API includes directives for HTML containers:

- fxLayout
- fxLayoutAlign
- fxLayoutGap

And directives for HTML elements *inside* flexbox-styled HTML containers:

- fxFlex
- fxFlexOrder
- fxFlexOffset
- fxFlexAlign

And directives for DOM element manipulation:

- fxHide
- fxShow
- ngClass
- ngStyle

And directives for responsive img sources

- src
- srcset [pending]

These directives black-box many of the complexities of using Flexbox CSS within a HTML document. Unfortunately browser support has some known issues, especially regarding **column** layouts. Additional, many developers would like to use CSS Grids for layouts instead of Flexbox.

CSS Grids

While FlexBox css allows developers to create 1-dimensional layouts (with support for nested 1-dimensional layouts), **CSS Grid** supports 2-dimensional layouts analogous to tables.

The standard raw CSS APIs include: [CSS Grid Specifications]

- display
- grid-template-rows
- grid-template-columns
- grid-template-areas
- grid-area

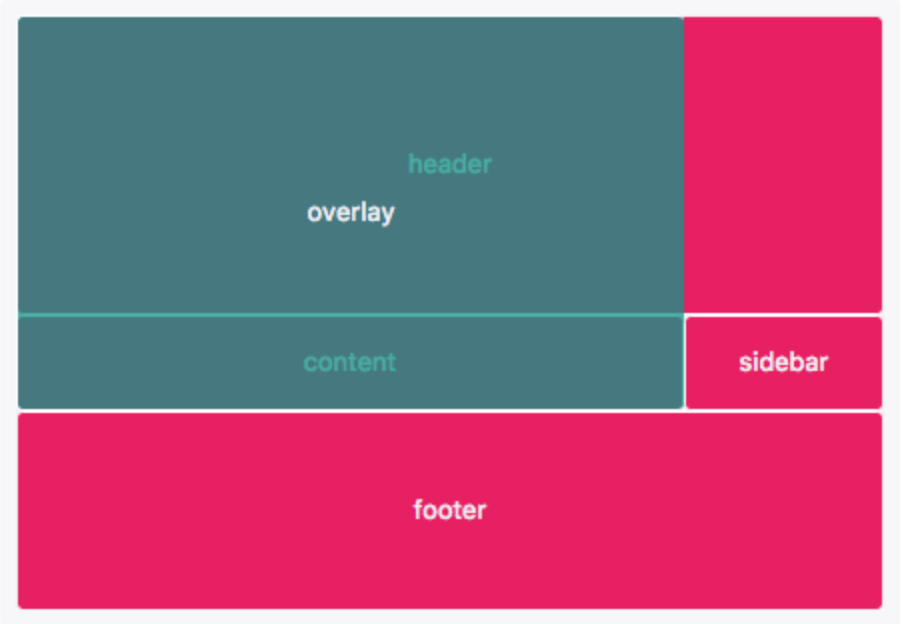
- grid-gap
- grid-row-gap
- grid-column-gap

- grid-auto-flow
- grid-auto-rows
- grid-auto-columns

- grid-row
- grid-column
- grid-area

- grid-row-start
- grid-row-end
- grid-column-start
- grid-column-end

- justify-items (align items along row axis)
- align-items (align items along column axis)
-
- justify-content (align tracks along column axis)
- align-content (align tracks along row axis)



Proposed Enhancements

Leveraging the architectures and patterns implemented for the Flexbox CSS API, the CSS Grid API would add static and responsive support for CSS Grids.

The API will include directives for HTML containers:

- gdLayout (alias for grid-auto-flow) **DONE**

- gdLayoutRows **DONE**
- gdLayoutColumns **DONE**
- gdLayoutAreas **DONE**

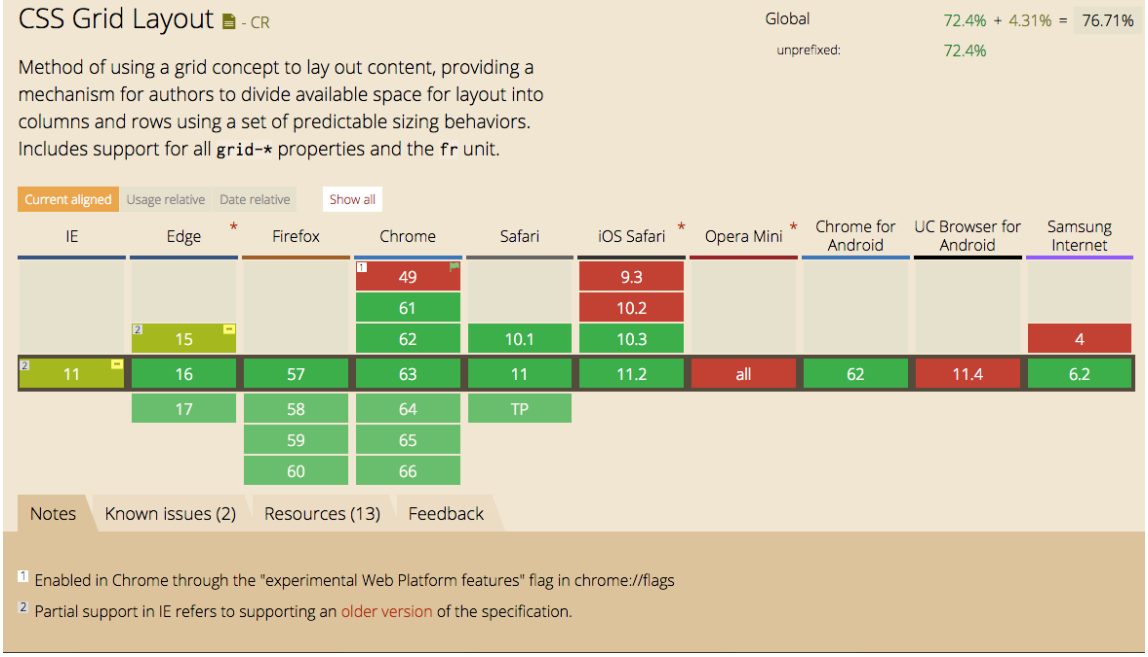
- gdAlignRows (align items) **DONE**
- gdAlignColumns (align tracks) **DONE**
- gdGap **DONE**

And directives for HTML elements *inside* CSS Grid HTML containers:

- gdRow **DONE**
- gdColumn **DONE**
- gdArea **DONE**
- gdAlign **DONE**

API	**Options [or sample values]**	
gdLayout	= "column|row|inline-row|inline-column dense"	(default == **row**)
gdLayoutRows	= "repeat(5, 1fr) [auto]"	(<val> == **<val>fr**)
gdLayoutColumns	= "1fr 20vw [auto]"	
gdLayoutAlign	= "center stretch" (align items in row or column)	
gdTrackAlign	= (align tracks relative to grid container)	(default == **normal**)
gdGap	= "row [column]" , "row" , "_ column", where row/column are values	
gdRow	= "start [/ end]" , "start", "_ / end"	
gdColumn	= "start [/ end]" , "start", "_ / end"	
gdArea	= <name>	
gdAlign	= "stretch" (align child within parent)	

Browser Support



With the exception of IE11, **CanIUse.com** shows evergreen browser support for CSS Grids is excellent.

Known Issues

Of course, every feature set has known issues. Similar to the well-known FlexBugs listing, **GridBugs** maintains a listing of known issues and [when possible] workarounds.