Page Layout - Flexbox vs Grid

September 1

ATLAS Front-End Web Development

Today's Goal: fix any and all misconceptions about CSS flexbox and show how very different it is from CSS grid.

CSS Flexbox

```
display: flex;
```

A property/value combo that can be applied to the parent container of some children that you want to be “automatically laid out in a flexible box.”

Very similar to an auto layout frame in Figma. Kinda similar to constraints in Android app development, or auto layout in iOS app development.

Many additional CSS properties specific to flexbox, both on the parent container and the children, impact the way the flexbox behaves.

CSS Flexbox - Flex Direction

`flex-direction: row | row-reverse | column | column-reverse;`

The main and cross axes of the flexbox are determined by the direction that content is layed out within the box

`row` direction will lay out content in the page’s writing direction

`column` lays out content in the page’s flow direction

CSS Flexbox - Flex Wrap

```
flex-wrap: nowrap | wrap;
```

Determines whether or not children in the flex container are wrapped onto a new line or not.

Set to `nowrap` by default.

If set to `nowrap`, existing children will shrink to fit new children, until they can shrink no longer, then content will start overflowing the flex container.

CSS Flexbox - Flex Flow Shorthand

Both the `flex-direction` and `flex-wrap` properties can be combined into one property, `flex-flow`, which controls them both.

Examples:

```
flex-flow: row wrap;

flex-flow: column;

flex-flow: column-reverse wrap;

flex-flow: row-reverse nowrap;
```

CSS Flexbox - Properties Applied to Flex Items

The children inside a flex container are often called “flex items” and can have multiple different properties applied to them which affect the way the flexbox behaves overall.

`flex-basis` is a measurement (`rem`, `em`, `%`, or `px`) that determines the initial/base size of the flex item.

`flex-grow` is a unitless number (just `0`,`1`,`2`,`3`…) that determines how quickly the flex item will grow, relative to the other flex items in the container, to take up available space. For example, an item with `flex-grow: 2;` will grow twice as much as an item with `flex-grow: 1;`

`flex-shrink` is the same as `flex-grow`, but for shrinking past the base size when more space needs to be made for more flex items in the container.

CSS Flexbox - Flex Shorthand

`flex-grow`, `flex-shrink`, and `flex-basis` can all be combined into one property, `flex`, that controls all three of them at once.

Examples:

```
flex: 0 1 200px; ⇒ flex-grow: 0; flex-shrink: 1; flex-basis:200px;

flex: 1 1; ⇒ flex-grow: 1; flex-shrink: 1;

flex: 2 100px; ⇒ flex-grow: 2; flex-basis: 100px;
```

CSS Flexbox - Positioning the Flex Items (Flex Container Properties)

`align-items` controls the position of the items along the cross axis.
Values: `normal | flex-start | flex-end | center | start | end | self-start | self-end | baseline | first baseline | last baseline | stretch | safe <another value> | unsafe <another value>`

`justify-content` controls the position of the items along the main axis.
Values: `normal | flex-start | flex-end | center | start | end | left | right | baseline | first baseline | last baseline | stretch | space-between | space-around | space-evenly | safe <another value> | unsafe <another value>`

CSS Flexbox - Positioning the Flex Items (Flex Item Properties)

`align-self` controls the position of that specific item along the cross axis.

`justify-self` controls the position of that specific item along the main axis.

CSS Grid - Not Flexboxes

A container with the CSS property `display: grid;` will arrange its children in a grid-like fashion.

The size of the grid is set with the CSS properties `grid-template-columns` and `grid-template-rows` applied to the container.

The position of each grid item is set with the CSS properties `grid-row` and `grid-column` applied to the item.