**Intro**

CSS Grid Layout (aka "Grid"), is a two-dimensional grid-based layout system that aims to do nothing less than completely change the way we design grid-based user interfaces. CSS has always been used to lay out our web pages, but it's never done a very good job of it. First we used tables, then floats, positioning and inline-block, but all of these methods were essentially hacks and left out a lot of important functionality (vertical centering, for instance). Flexbox helped out, and also Flexbox and css grids share some similarities but the one differences is that the flexbox controls how items flow in one dimention whereas grid controls how items flow in two dimentions. The core difference between Grids and Flex is that Flexbox lets content size itself based on space available or the dimentions of the content itself. Flex is great of handling alignment, distribution and order.

Instead of Flexboxes Grid specifies nothing about how the individual items themselves should be sized. And these items are created in grid container for sizing into.

The css grid items are aligned in rows and columns and exist in or span across cells but they did not really on content structure like tables do, so it allow much more flexibility.

**Basics**

*Grid* is the any element(container) with its display property set to *grid* of *inline-grid*. *Grid* is all about relationship between parent container and child items…The parent container is known as the *Grid Container*, which contain some items are known as the Grid Items. This Grid Container creates *Grid Context* for its children items. The *Grid Container* is also known as the *Grid*.

The *Grid* is made up by 2 sets of lines known as *Grid Lines*.

One set of lines defines the *Columns* for the Grid. This Lines run along where is known as the Column Axis.

The other set of lines defines the *Rows* for the Grid. And this lines run perpendicularly to the *Column Axis* and it names the *Row Axis.*

This *Grid Lines* together make up what are known as the *Grid Tracks*. Grid Track is the space between two grid lines, horizontal or vertical. They give us a size about how wide a row could be and how tall a column could be.

Grid Cell is the space between four grid lines. Where Grid Rows and Grid Columns intersect is where we find what are known as Grid Cells. And that are the smallest units of the Grid where we can place items into.

Grid Area is the area between four grid lines. And finally we have what are known as Grid Areas, which essentially any portion of the Grid that is contained by four Grid Lines.

A quick example if you want to create a

grid container you have this like new

value for the display property called

grid or inline grid depending and if you

notice that in comparison to Flex blocks

flex box by default aligns items and

columns whereas by default if you just

drop a display grid it sets all of the

items in rows

so now we're going to

start defining a grid so here we're

explaining what's called an explicit

grid and we're

telling it to create column tracks and

row tracks so I have two different

properties the grid display grid

template rows and grid template columns

properties

but if you're noticing these are

fixed width values and so we're talking

about the responsive web's

well grid also has this it introduces a

new unit which is really really cool

call the FR or the fraction so it's

very similar to the flexbox unitless

values and FR represents the fraction

of the available space in the across the

grid container

in this example we

have 2 columns defined at one fr each

and third column defined at two fr.

so it's just going to take all of the

available space in the grid container

separate it into 3 sections and then

just evenly distribute it across the

columns

You can also

combine fr with like other length values

There's this also min Max function. It's

this new function you can use inside of

the value when defining grid sizes so

you can tell it to have like a minimum

size and a maximum size and the default

value is auto based on like the content

You can also

combine the repeat function with other

track fixed width values or flexible

values so here you have the first and

last columns at 30 pixels and then in

between instead of defining like 1 fr 1

fr 1 fr. I'm just saying ok let's repeat

3 columns at 1 fr and then it's just

going to distribute it the available

space across the defined columns

So we

have grid gaps I the frustration about

flexbox about evenly spacing out the

items grid gap solves all of these

issues so here we have a defined grid

and there's two different properties the

grid row gap and the grid column gap and

the wonderful thing is it doesn't touch

the outer edge of the grid container it

no it's smart enough to say ok I just

want to evenly spaced out the actual

items and I don't need to add additional

spacing around the container which is

really cool

I got so excited grid gap

is just a shorthand function if you just

put one value in it'll evenly set the

spacing for rows and columns

another

cool thing is you have you're able to

also position grid items on a grip

The same thing with columns

even crazier you can take these lines

that are numbered and then you can

actually like give it names to give it a

little more context and then you can use

those names to position grid items so a

basic example here is you're taking

templates and rows you're defining them

with sizes but you're also have these

grid names and so line names so the line

name should be surrounded by square

brackets and like relative to where they

would be positioned based on where

you're defining these columns and rows

and it's good practice to you can name

them whatever you want

but it's good

practice to use the hyphen start and -

end and append it at the end because

there's some like additional implicit

added benefits

another really cool thing like taking

this even further is now you're naming

grid areas you can take like an entire

block or a section of a grid and like

say I want this to be the header or I

want this to be the footer and then you

can like take items and like position

them into the appropriate places

so here

this is just an example you are we have

the grid template areas property and

then you're using like the syntax to say

I have two columns on the first row and

that entire block I want to name it

header and then on the second row I have

content on the Left sidebar on the right

and on the last row I just want the

entire block to be named footer

and then while we define these names you can use

these names to then start positioning

and creating the grid so here you'll

notice I have these elements I have had

our contents sidebar footer and I have

I'm using a grid row and column properties.

so what does grid

support look like good support is

actually surprisingly pretty good so

this is just a screenshot from can I use

com that I took a couple days ago and

you'll notice the partial support for

RIE and edge surprisingly the spec for

grid came from Microsoft so it's been in

development for over five years and

because the spec has kind of like

evolved it's kind of falling behind but

I believe my understanding is edge will

catch up and I'm not 100% sure about ie