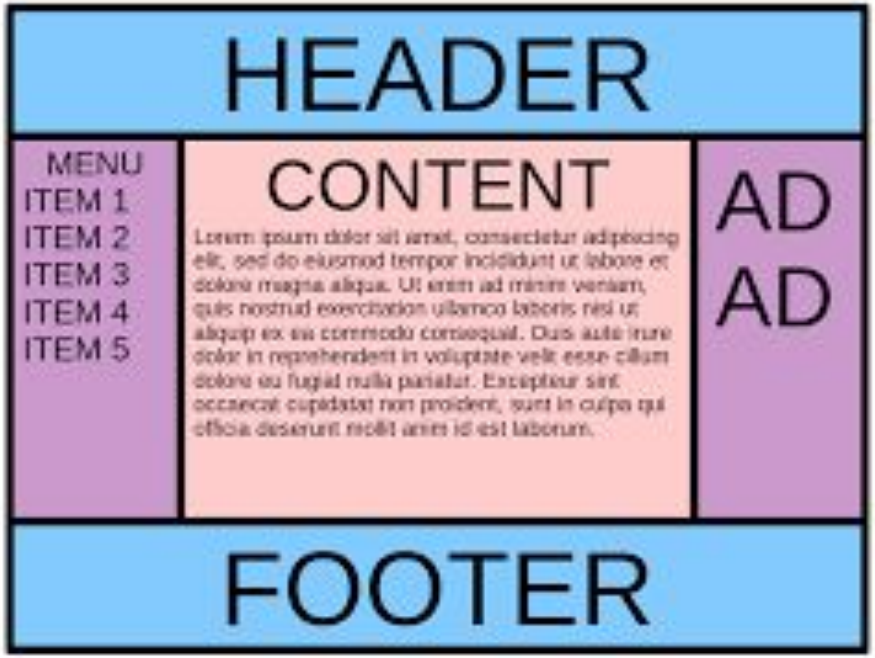
Module 3 - Lecture 3

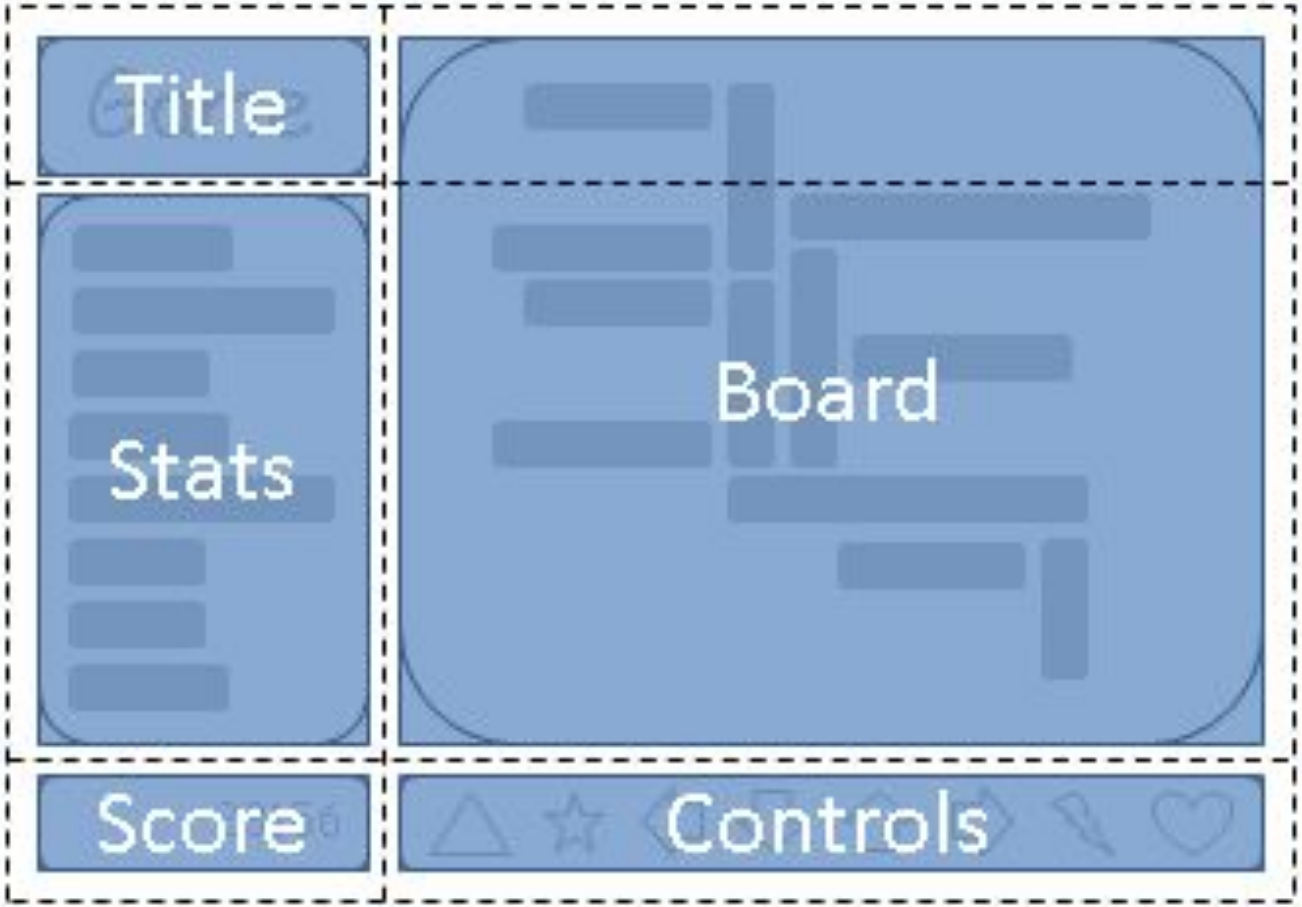
CSS Grid & Intro to Responsive Design

The Holy Grail Layout



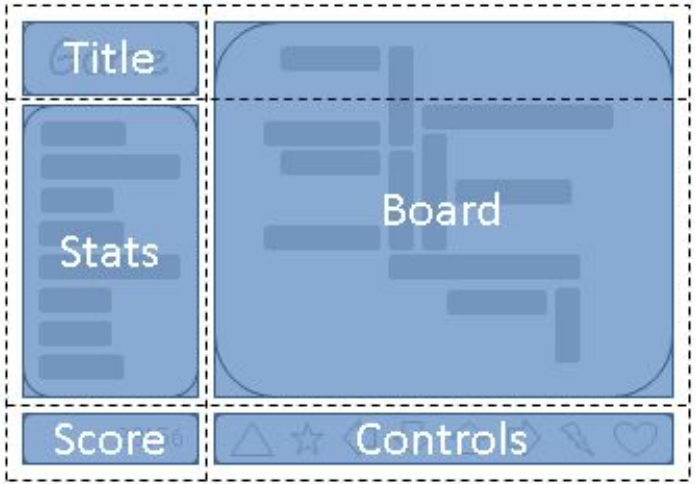
CSS Grid

https://www.w3.org/TR/css-grid-1/


Title
Stats
Board
Score
Controls

Terminology

- **Grid lines** are the vertical and horizontal dividing lines of the grid.
- A **grid item** is one of the items (e.g. Stats) and it may occupy more than one cell.
- A **grid cell** refers to one block within a grid.
- A **grid track** is a term referencing an entire column or row.
- A **grid area** is any rectangular area of one or more cells.
- The **gutters** or **gaps** are the spaces between adjacent grid tracks.



Starting a Grid Layout

```
section {
	display: grid;
}

<section>
	<div>
		<ul>
			<li>Item one</li>
			<li>Item two</li>
		</ul>
	<div>

	<p>Text goes here</p>
</section>
```

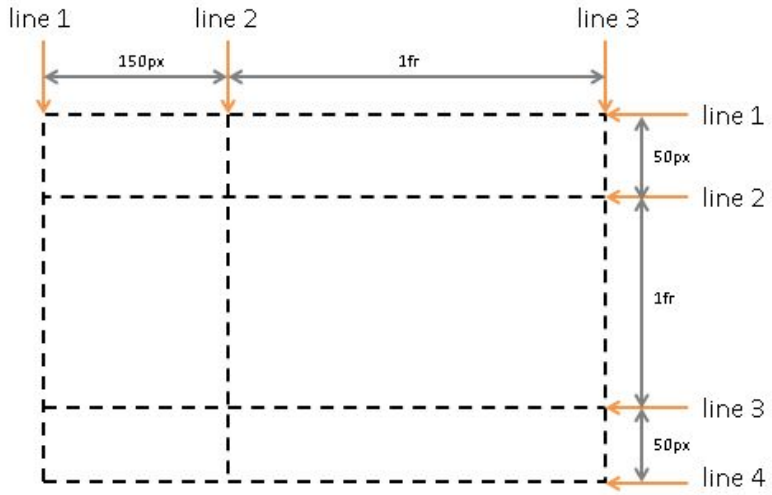
- Grid items are defined by the direct descendants of the element that is displayed as a grid.
  - In the example above, and <p> become the grid items.

- The grid will be 1 column by default include as many rows as there are grid items.
  - In the example above, this would result in a 2 row, 1 column grid.

Defining a Grid Layout

```
{
      display: grid;
      grid-template-columns: 150px 1fr;
      grid-template-rows: 50px 1fr 50px;
}
```

* The fr unit is a flexible length representing a fraction of the remaining space.

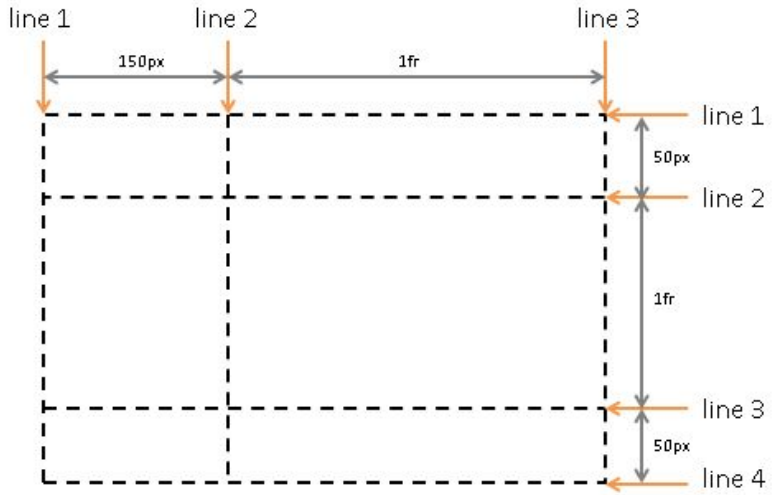


Placing Items

```
{
      display: grid;
      grid-template-areas: “. a”
                           “b .”
                           “. c”;
      grid-template-columns: 150px 1fr;
      grid-template-rows: 50px 1fr 50px;
}

#item1 { grid-area: a }
#item2 { grid-area: b }

#item3 { grid-area: c }
```

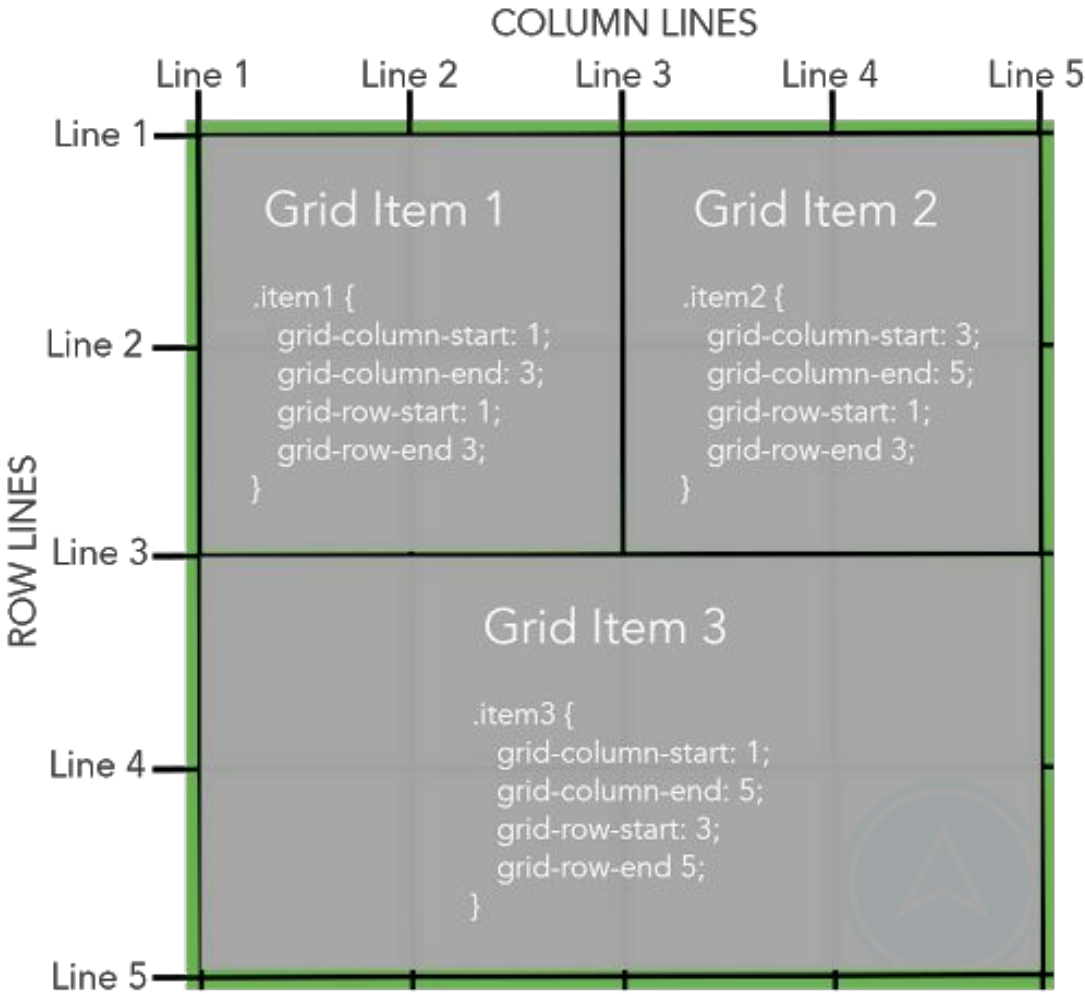


* A . is used within grid-template-areas to leave a grid cell empty.

grid-column-start,
grid-column-end,

grid-row-start,
grid-row-end

control the starting and ending location within the grid where a grid item appears.



Aligning Content

*Justify can be thought of as left to right alignment.
*Align can be thought of as vertical alignment.
*Options for alignment include start, center, end.

- **justify-items** is applied to the grid container to define justification of grid items along the row axis, within the individual grid cells

- **justify-self** is applied to any grid item to define row-axis justification within its individual grid cell

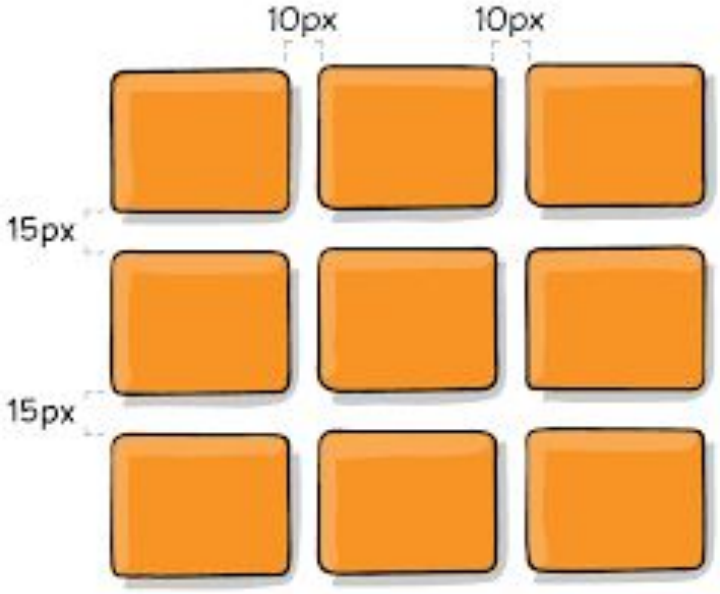
- **align-items** is applied to the grid container to define justification of grid items along the column axis, within the individual grid cells

- **align-self** is applied to any grid item to define column-axis justification within its individual grid cell

Grip Gap

The space between the grid tracks. The gutter.

```
{
      grid-template-columns: 1fr 1fr 1fr;
      grid-column-gap: 10px
      grid-row-gap: 15px;
}
```



Responsive Design

Mobile First

- **Mobile-first design** is a design philosophy that aims to create better experiences for users by starting the design process from the smallest of screens: mobile. Designing and prototyping your websites for mobile devices first helps you ensure that your users' experience is seamless on any device.

- **Mobile First design** can also include considering the performance constraints of a mobile device, such as slower network speeds, monetary costs of data transfer, and even offline capabilities.

- To complement Mobile First design, we sometimes use a technique referred to as **progressive enhancement**, which is to add more features and functionality as they are deemed accessible based on the browser and/or Internet connectivity.

Mobile First

- When mobile devices were first introduced, web pages were zoomed out in order to display them effectively. For example, the Apple iPhone would start with a default viewport of 960px. That default is overridden by the tag:

  <meta name='viewport" content="width=device-width, initial-scale=1.0">

  DEMO: https://www.w3schools.com/html/example_withoutviewport.htm

- Often this was dealt with by serving separate web pages for mobile devices. This is not always a good solution and can result in twice the effort.

Media Queries

```
@media only screen and (min-width: 1024px) {
  /* Target screen sizes 1024px and above */
}


@media only screen and (max-width: 1023px) {
  /* Target screen sizes 1023px and below */
}
```

Relative Sizing

FONTS

- **em** and root em (or **rem**) are sizing measurements relative to the font size.

ELEMENTS

- Sizing of elements can also be done in **percentages**, from **0-100%**
  - This is not the same as viewport sizing. A percentage is based on an element's parent.

- CSS3 introduced a unit of sizing that enables sizing relative to the height and width of the viewport (browser window).
  - The unit is **vh for viewport height** and **vw for viewport width.** Each ranges from 0 - 100, meaning 0 to 100% of the viewport.

Images

- Images make up 60% of a webpage's size, on average.
- Use relative sizing for images to prevent them from overflowing the container.
- Use the **<picture>** element to specify different images based on media queries. This is called **art direction**.
- Use **srcset** attribute in the **<img>** element to render different images based on the device's pixel density.
- JPG vs. PNG
- Vector vs. Raster

Responsive Image Demos

Resources

Grid

- A Complete Guide to Grid
- Grid Garden
- Holy Grail Layout
- Holy Grail Demo

QUESTIONS?