HTML/CSS: INTERMEDIATE

STEPS TO ACHIEVE A MULTI-COLUMN LAYOUT USING FLEXBOX:

- In your HTML: Each column has a wrapper around it in your HTML
- In your HTML: Wrap the columns directly with a container (the flex container)
- In your CSS: Add `display:flex` to flex container
- In your CSS: Give each column a width in percentages
- In your CSS:: Use justify-content on flex container to determine how columns are distributed.

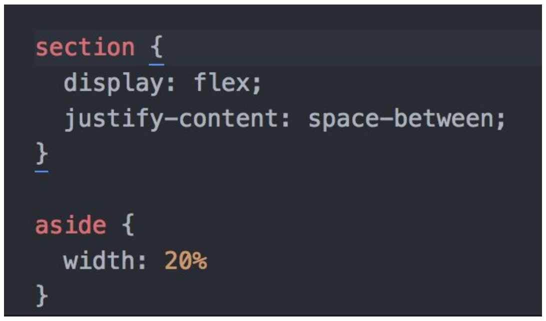
DISPLAY:

`inline`	Make a block-level element act inline element.
`inline-block`	Make a block-level element flow like an inline element, while retaining width, height, padding, and margin.
`block`	Make an inline element act like a block-level element.
`none`	Hide an element from the page.

FLEX CONTAINER PROPERTIES:

`display`	flex
`flex-direction`	row, column
`flex-wrap`	wrap, nowrap
`justify-content`	flex-start, flex-end, center, space-between, space-around

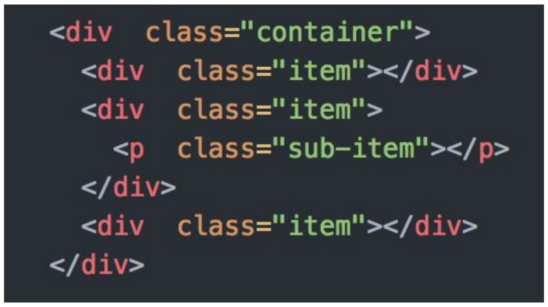
Example:



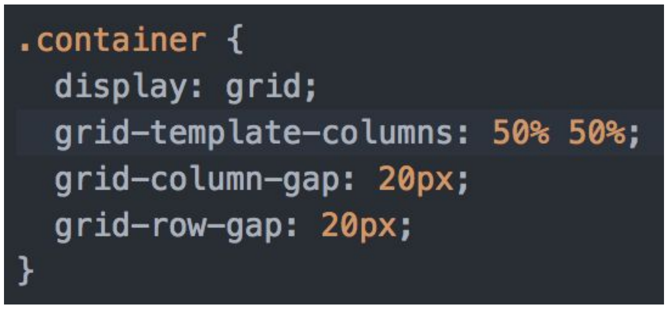
```
section {
  display: flex;
  justify-content: space-between;
}

aside {
  width: 20%
}
```

CSS GRID PROPERTIES:



```
<div  class="container">
  <div  class="item"></div>
  <div  class="item">
    <p  class="sub-item"></p>
  </div>
  <div  class="item"></div>
</div>
```



```
.container {
  display: grid;
  grid-template-columns: 50% 50%;
  grid-column-gap: 20px;
  grid-row-gap: 20px;
}
```

grid-wrapper	the selector assigned to be your grid container.	`.grid-container{}`
display	to set up CSS grid.	`display: grid | inline-grid | subgrid;`
grid-template-columns:	to set columns, if there are 2 items in the list, it will be a 2 column layout.	`grid-template-columns: 50% 50%;`
grid-column-gap:	space between columns.	`grid-column-gap:10px;`
grid-row-gap:	space between rows.	`grid-row-gap:10px;`
grid-gap:	shorthand for the space between tracks (columns and rows) if you have an even space between.	`grid-gap: 10px;`
align-content:	aligns grid content along the columns axis.	`.grid { align-content: stretch| center | start | end | space-between | space-around | space-evenly};`
align-items	aligns content inside the grid along the column axis.	`.grid { align-items: stretch | center | start | end};`

MEDIA QUERIES:

```
@media screen and (max-width: 600px) {
        .hero {
              display: block;
               }
        section {
              flex-basis: 100%
        }
}
```

POSITIONING:

position	static | relative | fixed | absolute	`position: fixed;`
top	% | length (px or em) |	`top: 30px;`
right	% | length(px or em)	`right: 30px;`
bottom	% | length(px or em)	`bottom: 15px;`

left	% | length(px or em)	`left: 5px;`
z-index	number	`z-index: 100;`