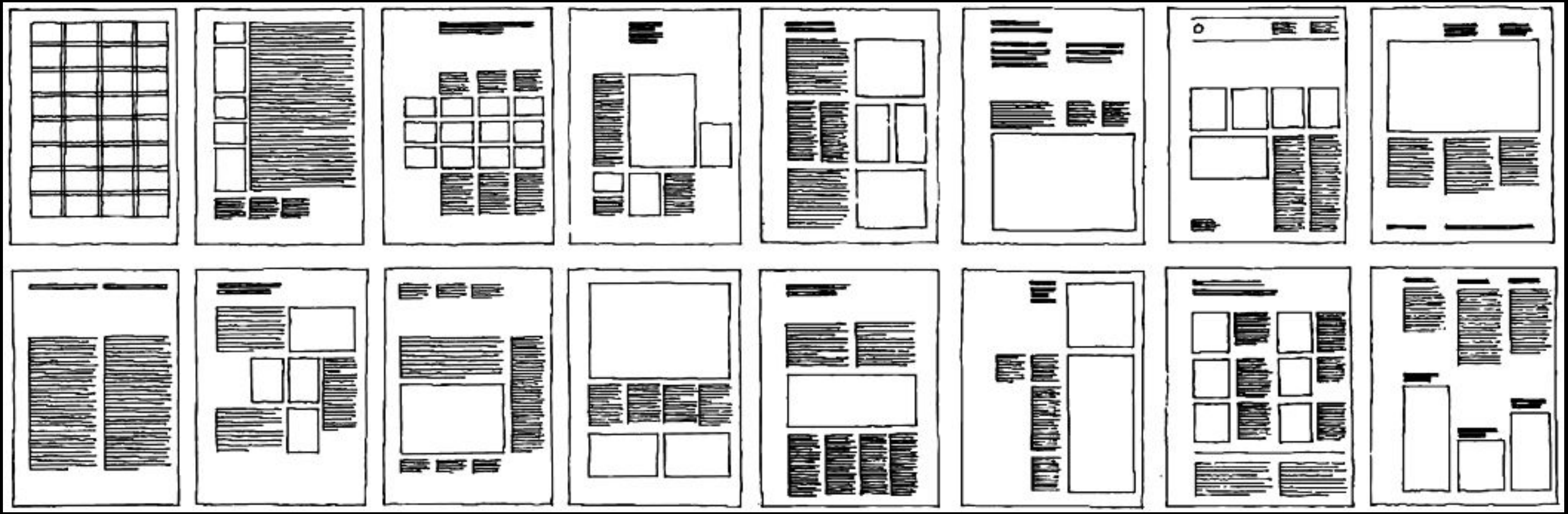
ADVANCED LAYOUT



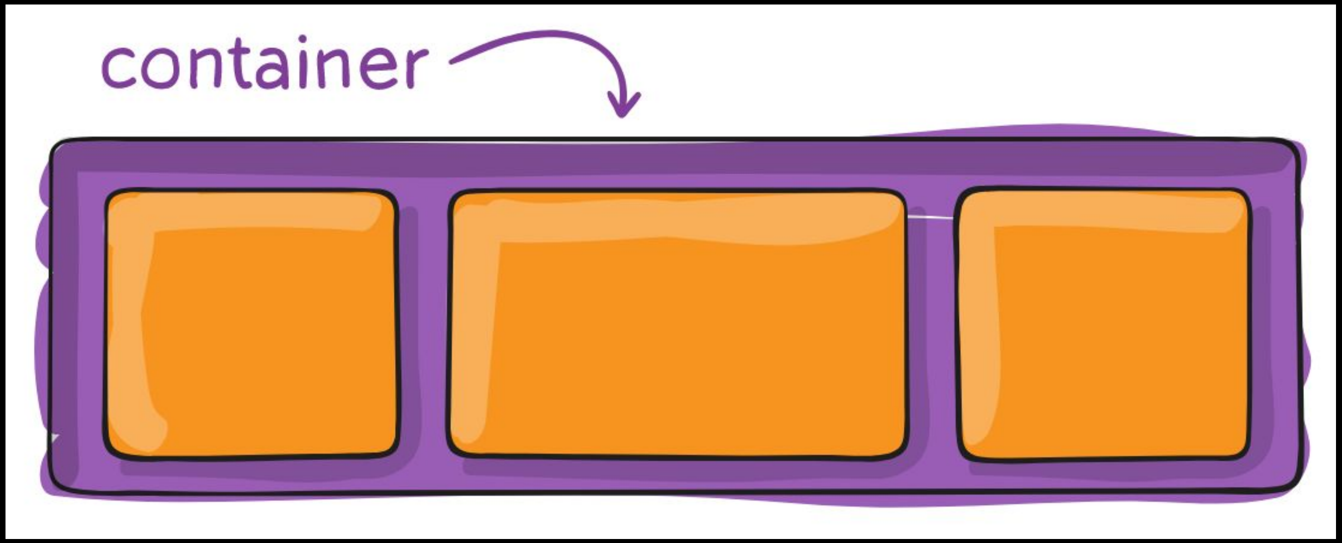
ADVANCED LAYOUT

WE'VE COVERED PRIMITIVE METHODS FOR LAYOUT:

- `display (inline, block, etc.)`
- `<div> <span>`
- `position (absolute, relative, etc.)`
- `float`

BUT THERE ARE MODERN AND MORE LOGICAL METHODS

ADVANCED LAYOUT



FLEXIBLE BOX (FLEXBOX)

ARRIVED WITH CSS3

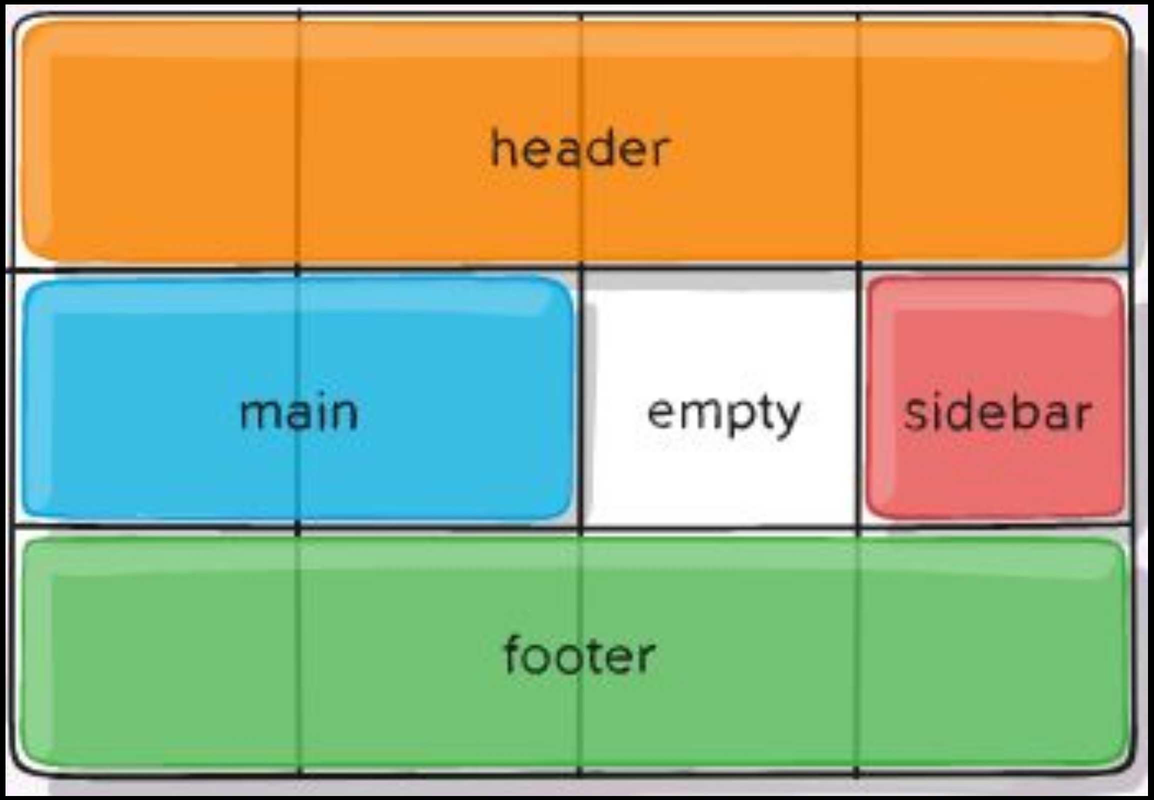
CONSISTS OF FLEXIBLE CONTAINERS AND FLEXIBLE ITEMS WITHIN

EXPANDS ITEMS TO TAKE UP AVAILABLE SPACE OR SHRINKS THEM TO PREVENT OVERFLOW (ITEMS FLOWING OUT OF NORMAL FLOW)

MORE CONTROL OVER SIZING, DEVICES, ETC. THAN FLOATING.

1 DIMENSIONAL: FLEX CAN BE APPLIED TO EITHER A COLUMNAL OR ROW DISPLAY (CONTENTS CAN WRAP HOWEVER, FORCING CONTENT TO ANOTHER ROW OR COLUMN)

ADVANCED LAYOUT



GRID

ARRIVED WITH CSS3

FOR GRIDDED LAYOUT/SYSTEMS

EXCELLENT CONTROL OVER WHERE IN THE GRID ITEMS/CONTENT SHOULD GO

2 DIMENSIONAL: CONTROL IS EXERTED OVER ROWS AND COLUMNS

ADVANCED LAYOUT

FLEXBOX VS GRID - WHAT'S THE DEAL?

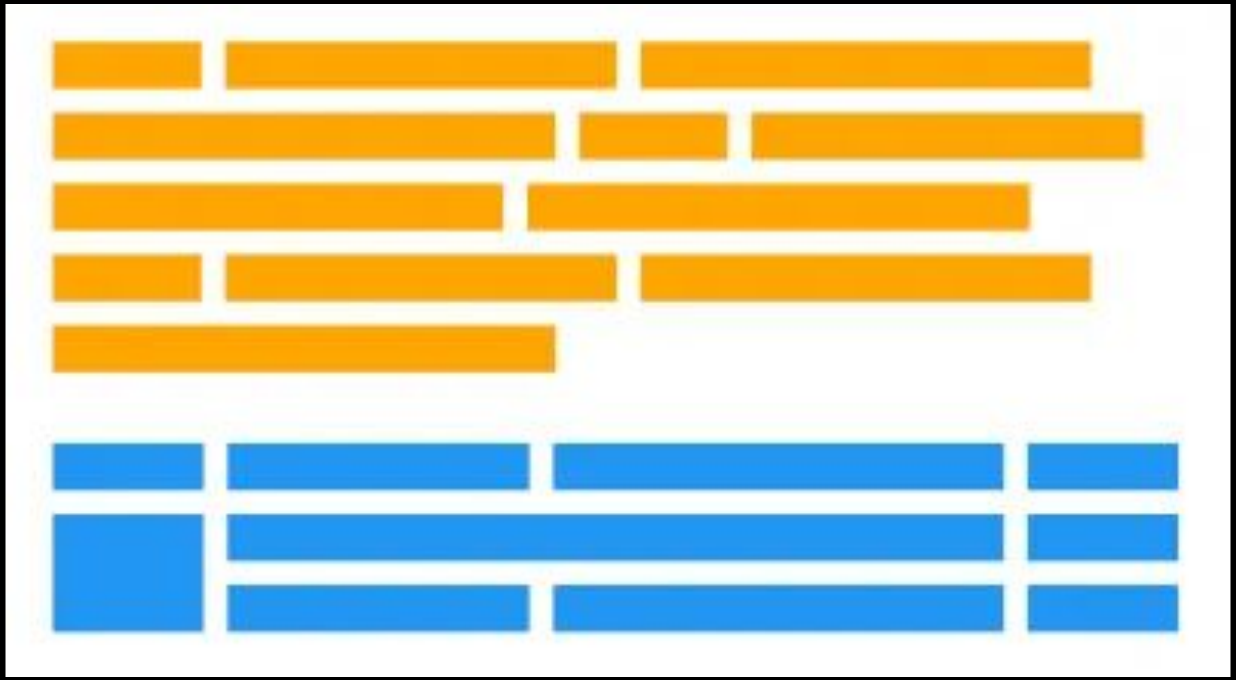
GENERALLY:

FLEXBOX WHEN YOU JUST WANT TO MAKE A BUNCH OF CONTENT FIT ON THE PAGE

GRID WHEN YOU HAVE A SPECIFIC, STRICT DESIGN OR LAYOUT TO A PAGE

FLEXBOX CAN WRAP CONTENTS BUT SPACING BETWEEN ITEMS OF VARIED WIDTHS IS MISALIGNED

GRID CAN WRAP CONTENTS AS WELL BUT SPACING IS STRICT TO THE GRID



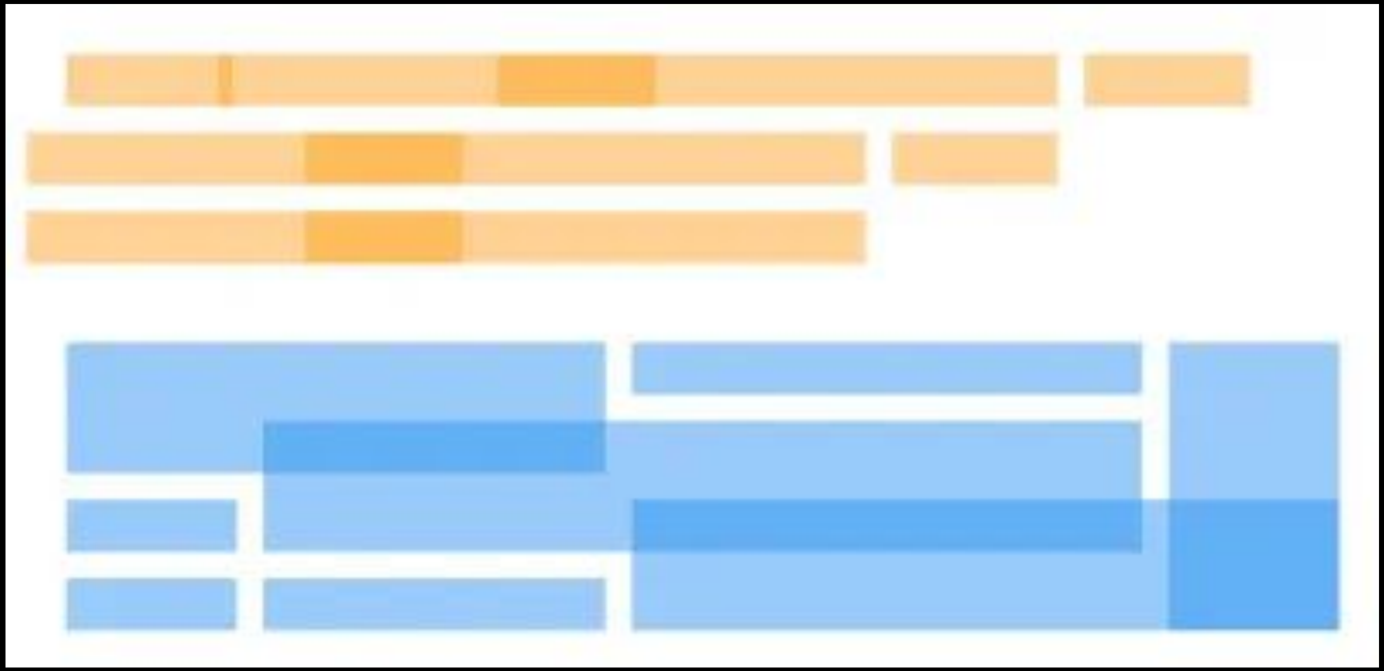
ADVANCED LAYOUT

FLEXBOX VS GRID - WHAT'S THE DEAL?

GENERALLY:

GRID IS BETTER AT OVERLAPPING OVERLAPPING IN FLEX IS MUCH MORE TIME CONSUMING

FLEX



GRID

FLEXBOX IS BEST WHEN YOU HAVE A BUNCH OF STUFF OF DIFFERENT SIZES AND JUST WANT A REASONABLE LAYOUT FROM THEM

GRIDS FOR FULL PAGE LAYOUTS AND FLEXBOX FOR INDIVIDUAL ELEMENTS/PARTS OF A PAGE.

THEY CAN BE NESTED INSIDE OF EACH OTHER AS WELL. GRIDS IN GRIDS, FLEX IN FLEX, GRIDS IN FLEX, FLEX IN GRIDS

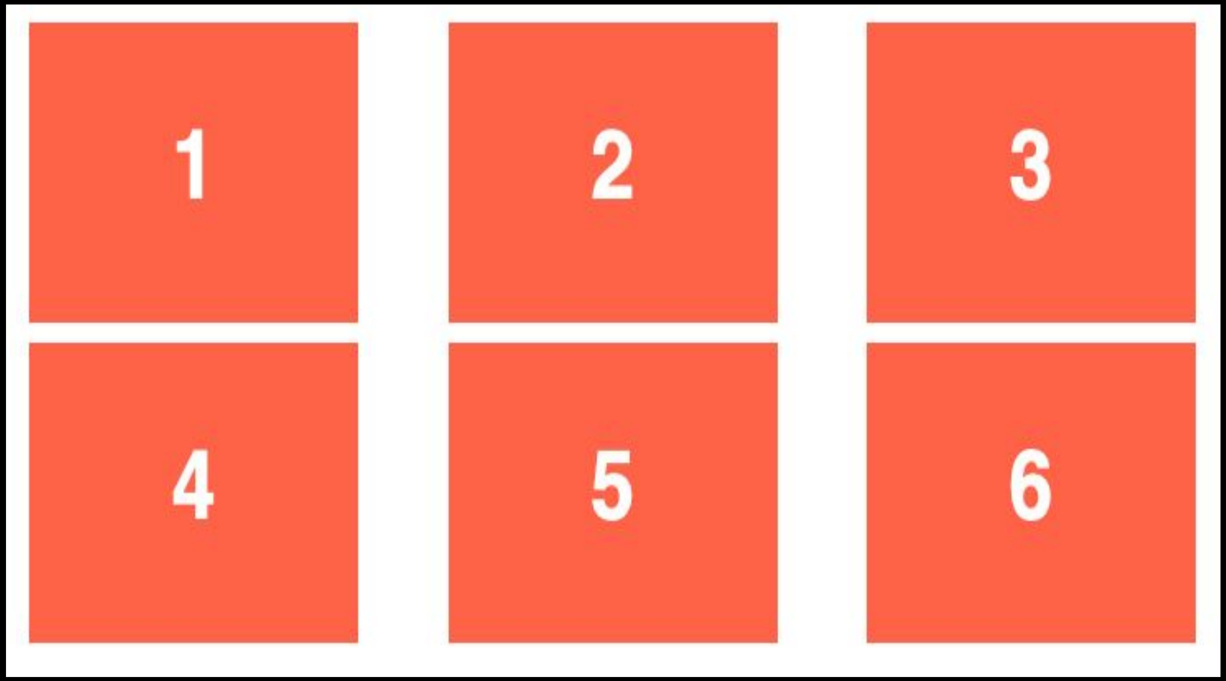
ADVANCED LAYOUT

A FLEXBOX IMPLEMENTATION

```
.parent {
  display: flex;
   /* Then we define the flow direction
     and if we allow the items to wrap
   * This is the same as:
   * flex-direction: row;
   * flex-wrap: wrap;
   */
  flex-flow: row wrap;
  /* Then we define how is distributed
     the remaining space */
  justify-content: space-around;

}

.child {
  width: 200px;
  height: 150px;
}
```



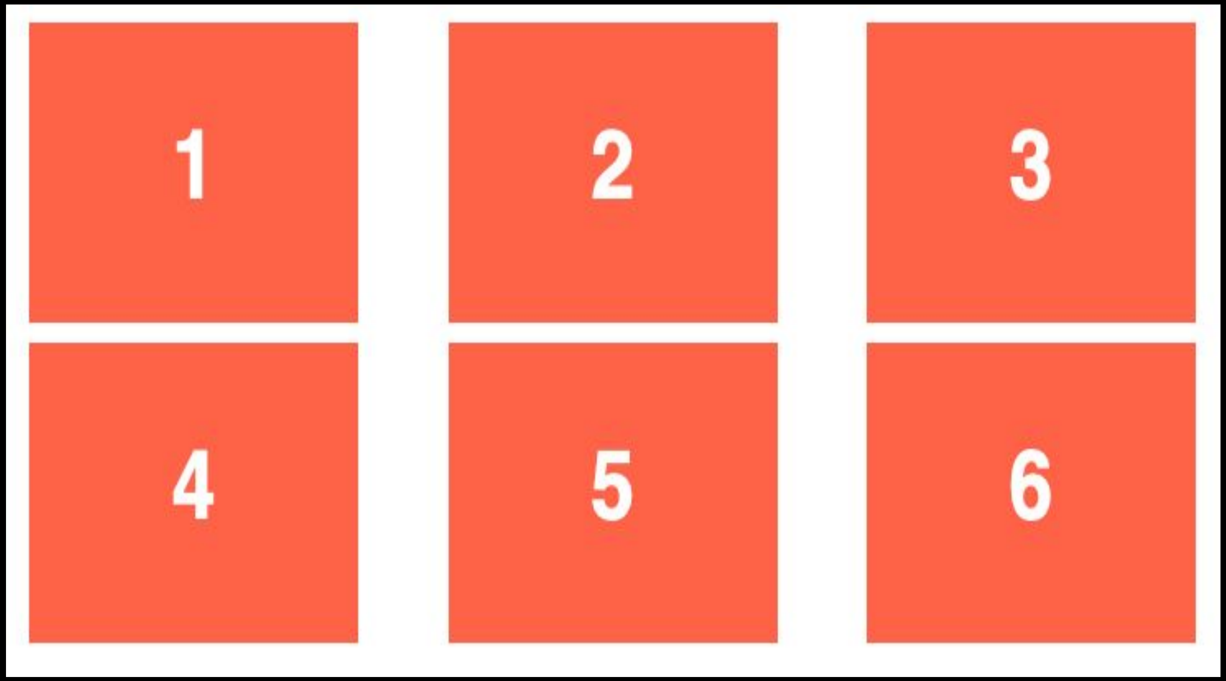
```
<section class="parent">
  <div class="child">1</div>
  <div class="child">2</div>
  <div class="child">3</div>
  <div class="child">4</div>
  <div class="child">5</div>
  <div class="child">6</div>
</section>
```

ADVANCED LAYOUT

A GRID IMPLEMENTATION

```
.parent {
  display: grid;
  grid-gap: 20px;
/* Then we define the columns using
  a measurement: here its 'fr' which
  means fractional unit. 3 columns split
  equally */
  grid-template-columns: 1fr 1fr 1fr;
}

.child {
  width: 200px;
  height: 150px;
}
```



```
<section class="parent">
  <div class="child">1</div>
  <div class="child">2</div>
  <div class="child">3</div>
  <div class="child">4</div>
  <div class="child">5</div>
  <div class="child">6</div>
</section>
```