CSS

STYLE ATTRIBUTE VS ELEMENT VS LINK CSS FILE

Cascading Style Sheets (CSS)

Declarative language configuring look & feel of HTML web pages

CSS can be ***applied to*** an HTML ***element***, ***embedded*** in the HTML document, or ***linked*** to an external file:

```
<link href="/some/external/style/file.css" rel="stylesheet"/>

<style>

  p { text-align: center; }

</style>

<p style="color: green">This paragraph is now green</p>
```

Style attribute

Style attribute configures look & feel of specific HTML elements and is considered bad practice

```
<p style="background-color: yellow">
  This paragraph is yellow
</p>
<p style="background-color: blue;
          color: white">
  This paragraph is blue with white text
</p>
<p style="background-color: red;
          color: white">
  This paragraph is red with white text
</p>
```

This paragraph is yellow

This paragraph is blue with white text

This paragraph is red with white text

Style element

Style element configures style in current web page and is considered bad practice

```
<style>
  .background-yellow { background-color: yellow; }
  .background-blue   { background-color: blue; }
  .background-red    { background-color: red; }
  .color-white       { color: white; }
</style>

<p class="background-yellow">This paragraph is yellow</p>
<p class="background-blue color-white">This paragraph...</p>
<p class="background-red  color-white">This paragraph...</p>
```

Link element

Link element can apply same look and feel to multiple web pages is considered best practice

style.css

```
.background-yellow { background-color: yellow; }
.background-blue   { background-color: blue; }
.background-red    { background-color: red; }
.color-white       { color: white; }
```

```
<link href="style.css" rel="stylesheet"/>
<p class="background-yellow">This paragraph is yellow</p>
<p class="background-blue color-white">This paragraph...</p>
<p class="background-red  color-white">This paragraph...</p>
```

BASIC STYLING

Foreground color

Consider the following HTML snippet

```
<h1>Basic Styling</h1>
<p style="
  color: blue;">
Text can be colored and
styled to be bold, italic
or underlined
</p>
```

Style attribute used only for illustration purposes. Use link instead for best practice

Use the color property to configure the foreground color

Background color

Basic Styling

```
<h1>Basic Styling</h1>
<p style="
  background-color: yellow;
  color: blue;">
Text can be colored and
styled to be bold, italic
or underlined
</p>
```

Text can be colored and styled to be bold, italic or underlined

Use the background-color property to configure the background color

Bold, italic and underlined text

```
<h1>Basic Styling</h1>
<p style="
  background-color: yellow;
  color: blue;">
Text can be colored and
styled to be <b>bold</b>,
<i>italic</i> or
<u>underlined</u></p>
```

Font styles

Basic Styling

```
<h1>Basic Styling</h1>
<p style="...">
<span style="
  font-weight: bold;
  font-family: 'Courier';
  font-size: 20px;">
    Text can be colored and
    styled                  .
</span></p>
```

Text can be colored and styled to be **bold**, *italic* or underlined

CSS styling is preferable to using elements such as b, i, u

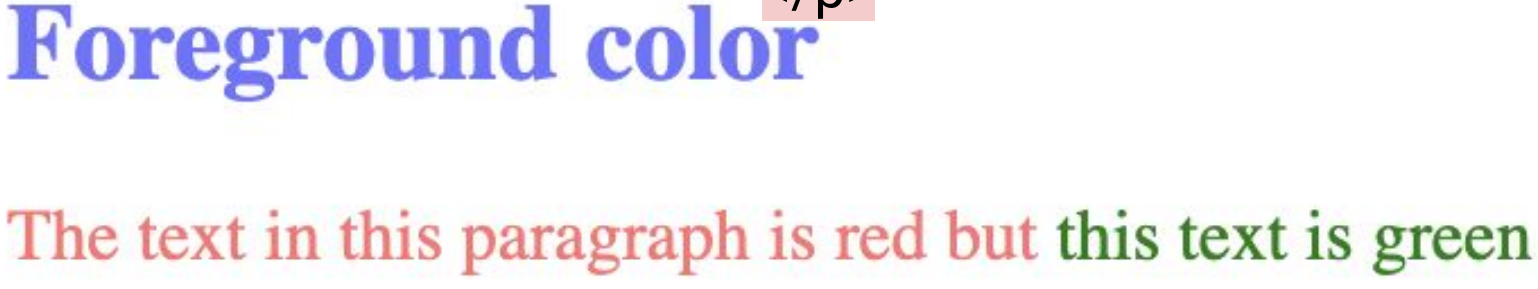
Foreground Color

index.css

```
.fg-color-black { color: black; }
.fg-color-white { color: white; }
.fg-color-blue { color: #7070ff; }
.fg-color-red { color: #ff7070; }
.fg-color-green { color: green; }
```

index.html

```
<h2>Colors</h2>
<h3 class="fg-color-blue">
  Foreground color</h3>
<p class="fg-color-red">
  The text in this paragraph is red but
  <span class="fg-color-green">
    this text is green
  </span>
</p>
```



Background Color

index.css

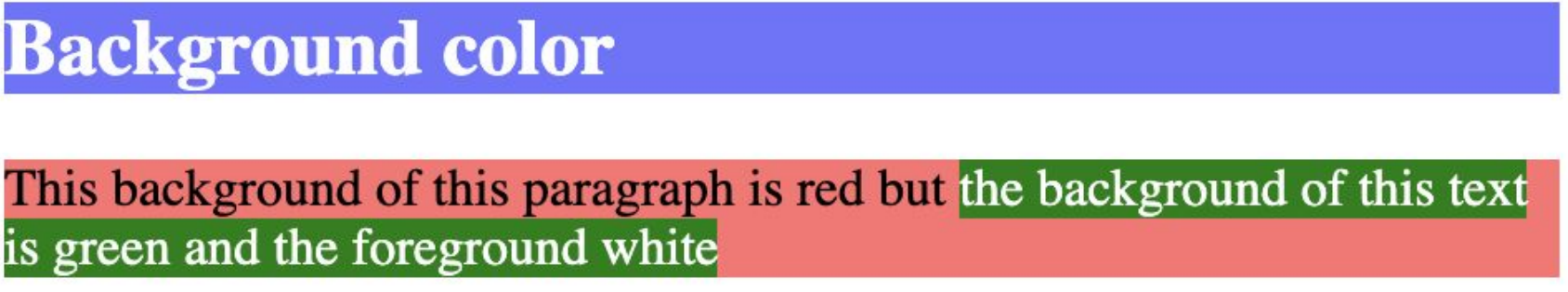
.bg-color-yellow
{ background-color: #ffff07; }
.bg-color-blue
{ background-color: #7070ff; }
.bg-color-red
{ background-color: #ff7070; }
.bg-color-green
{ background-color: green; }

index.html

<h3 class="***bg-color-blue*** fg-color-white">
Background color</h3>

This background of this paragraph is red but

the background of this text is green and the
foreground white



BOX MODEL

Block vs inline elements

- Elements can render inline or as blocks
- Block element
  - widths expand to parent's and height depends on content
  - add vertical break
  - e.g., heading, paragraph, divs
- Inline element
  - width and height depends on content
  - flow with the rest of neighboring content
  - wrap when they don't fit on parent's width
  - e.g., normal text, span, input, button

Box model

All HTML content is rendered using the policies defined by the box model

Defines area content occupies on screen and how it relates to other content

Styling the box model

CSS properties **margin**, **border**, padding, width, and height control the box look and feel

Margin configures the space between boxes

Border configures border thickness, color, and style (dotted, dashed, etc.)

Padding configures the space between the content and the border

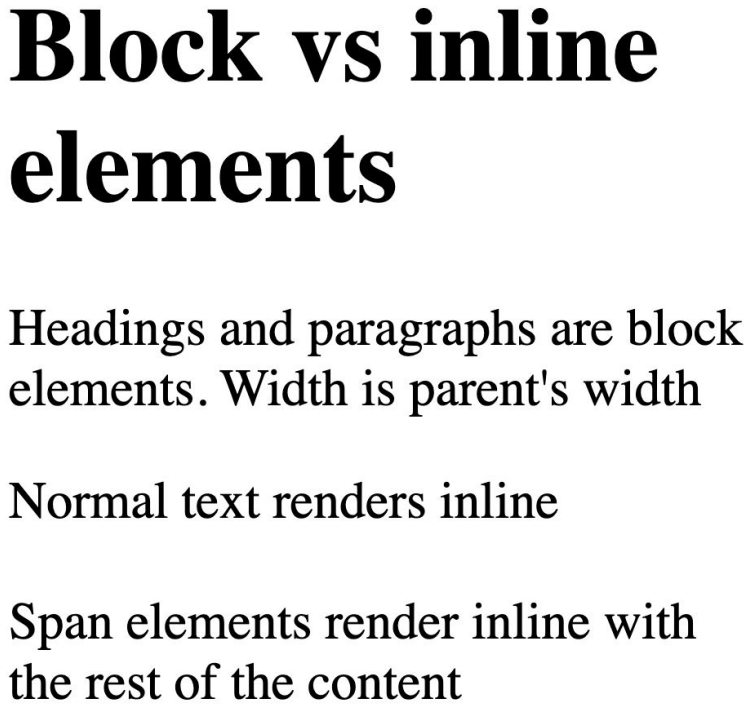
Width and **height** configure the width and height of the content

Actual width and **height** of content is **combination** of all above

Block vs Inline Elements Example

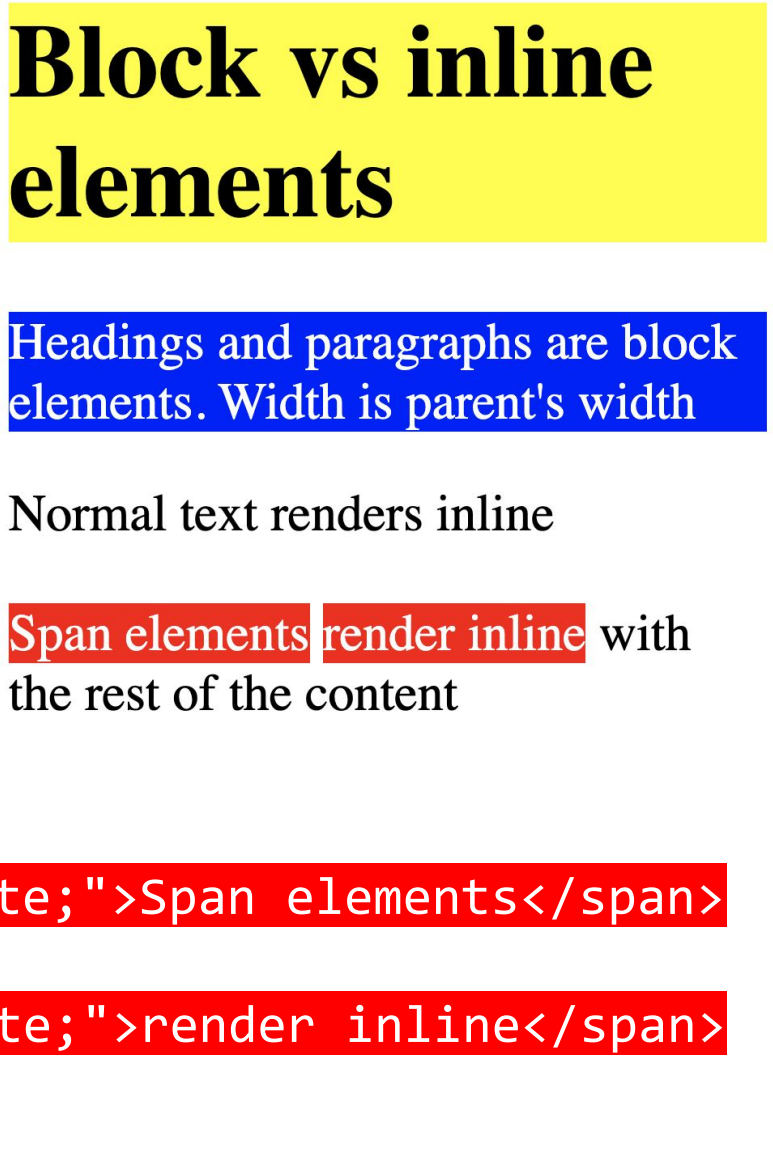
Consider the following HTML snippet

```
<h1>Block vs inline elements</h1>

<p>Headings and paragraphs are
block elements. Width is parent's
width</p>

Normal text renders inline

<br/><br/>

<span>Span elements</span>

<span>render inline</span> with the rest of the content
```



Let's add some colors

```
<h1 style="
  background-color: yellow">
  Block vs inline elements
</h1>
<p style="
  background-color: blue;
  color:white;">
  Headings ... width</p>
Normal text renders inline
<br/><br/>
<span style="
  background-color: red;color:white;">Span elements</span>
<span style="
  background-color: red;color:white;">render inline</span>
with the rest of the content
```



Styling borders

```
<span style="
   background-color: red;
   color: white;
   border-style: solid;
   border-width: 10px;
   border-color: green;">
      Span elements</span>
<span style="
   background-color: red;
   color: white;
   border: solid 10px green;">
      render inline</span>
```

Block vs inline elements

Headings and paragraphs are block elements. Width is parent's width

Normal text renders inline

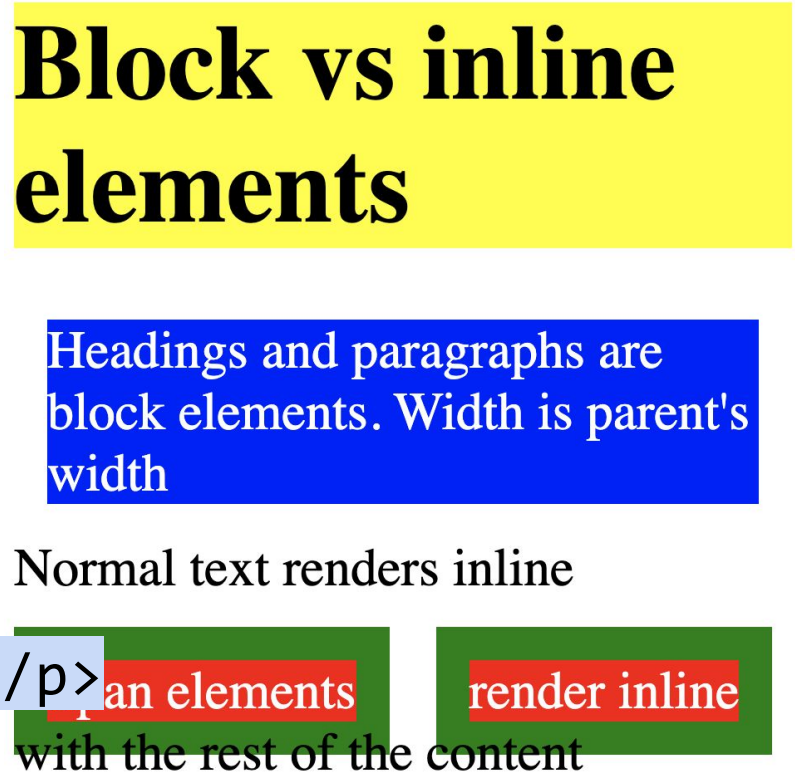
Span elements render inline with the rest of the content

Styling margins

Margins add spacing between elements
Note space all around paragraph and between span elements

```
<p style="
  margin: 10px;
  background-color: blue;
  color:white;">Headings...</p>
```

```
<span style=" ...
  margin-right: 10px;">
    Span elements</span>
<span ... >render inline</span>
```



Styling padding

Padding adds space around the content, but within the border

Note new space in paragraph and spans

```
<p style="
  padding: 10px
  background-color: blue;
  color:white;">Headings...</p>
```

```
<span style=" ...
  padding-left: 10px;
  padding-top: 10px;">
    Span elements</span>
<span ... >render inline</span>
```

Block vs inline elements

Headings and paragraphs are block elements. Width is parent's width

Normal text renders inline

Span elements render inline with the rest of the content

Border Colors, Widths and Styles

```
<p class="border-fat border-red border-solid"> Solid fat red border</p>
<p class="border-thin border-blue border-dashed">Dashed thin blue border
</p>
```

```
.border-fat { border-width: 20px 30px 20px 30px; }
.border-thin { border-width: 4px; }
.border-dashed { border-style: dashed; }
.border-red { border-color: #ff7070; }
.border-solid { border-style: solid; }
.border-yellow { border-color: #ffff07; }
.border-blue { border-color: #7070ff; }
```

Solid fat red border

Dashed thin blue border

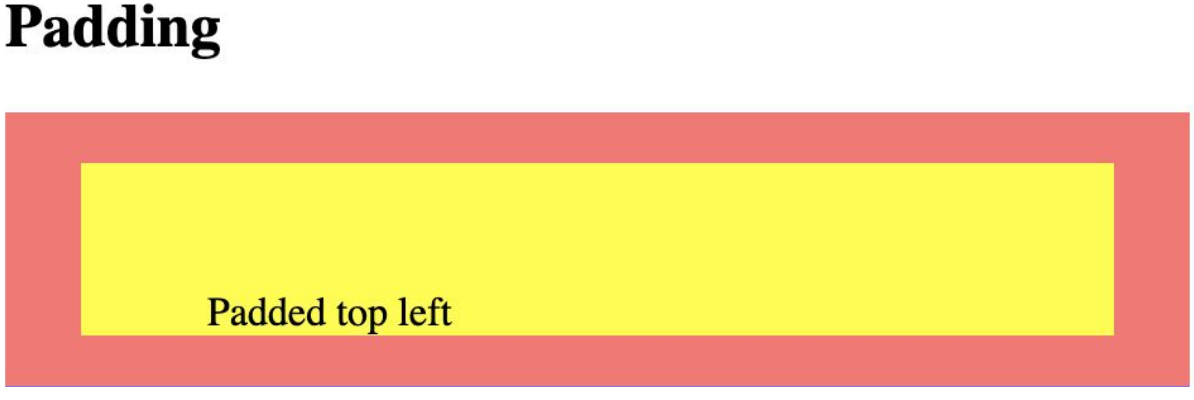
Padding Top Left

index.html

```
<h2>Padding</h2>
<div class="padded-top-left border-fat border-red border-solid bg-color-yellow">
    Padded top left
</div>
```

index.css

```
.padded-top-left {
    padding-top: 50px;
    padding-left: 50px;
}
```



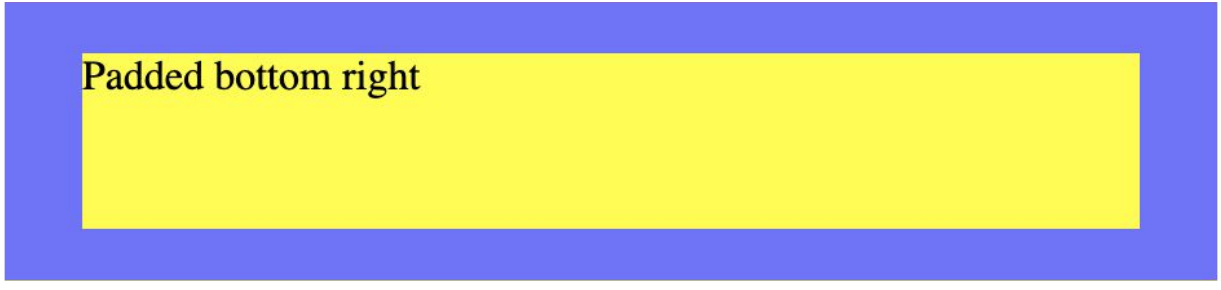
Padding Bottom Right

index.html

```
<div class="padded-bottom-right border-fat border-blue
            border-solid bg-color-yellow">
    Padded bottom right</div>
```

index.css

```
.padded-bottom-right {
    padding-bottom: 50px;
    padding-right: 50px;
}
```



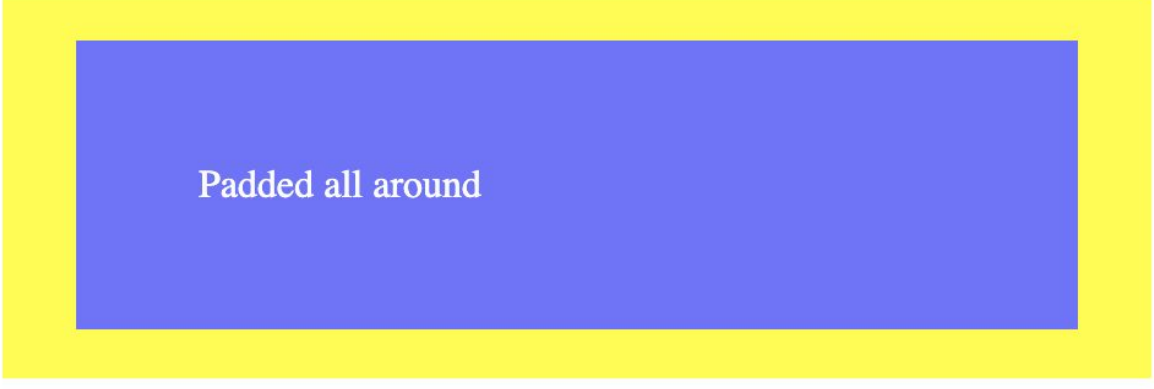
Padding All Around

index.html

```
<div class="padding-fat border-fat border-yellow border-solid
        bg-color-blue fg-color-white">Padded all around</div>
```

index.css

```
.padding-fat {
    padding: 50px;
}
```



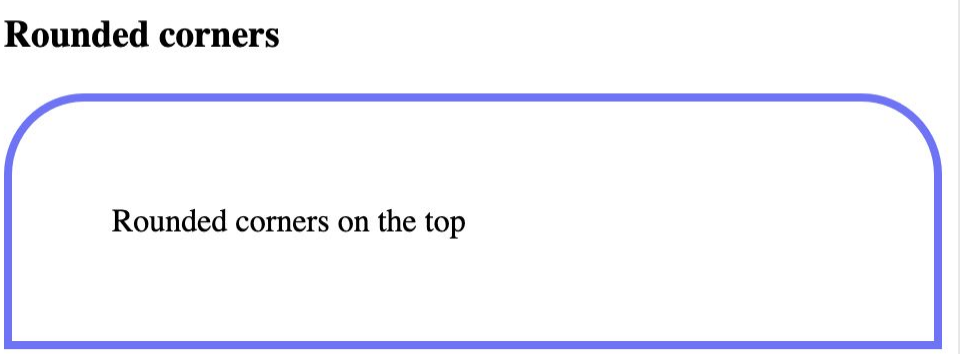
Rounded Corners

index.html

```
<h3>Rounded corners</h3>
<p class="rounded-corners-top border-thin
                border-blue border-solid padding-fat">
   Rounded corners on the top
</p>
```

index.css

```
.rounded-corners-top {
   border-top-left-radius: 40px;
   border-top-right-radius: 40px;
}
```



Different Rounded Corners

index.html

```
<p class="rounded-corners-inline border-thin border-blue
                border-solid padding-fat">
   Different rounded corners
</p>
```

index.css

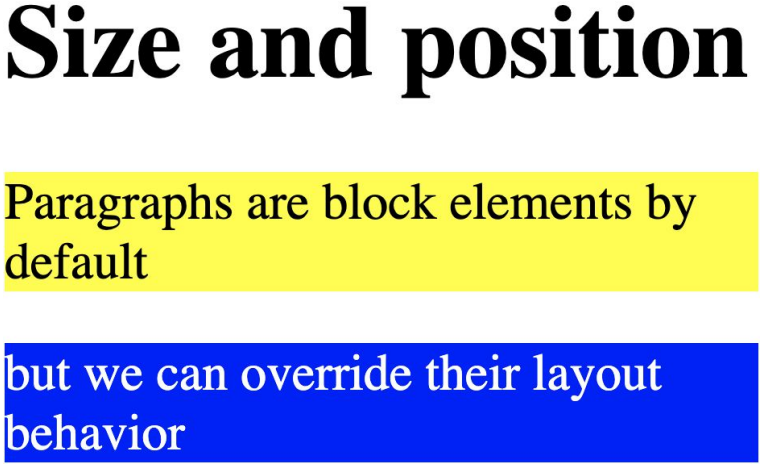
```
.rounded-corners-inline {
   border-radius: 30px 0px 20px 50px;
}
```

SIZE & POSITION

Styling size

Consider the following HTML snippet

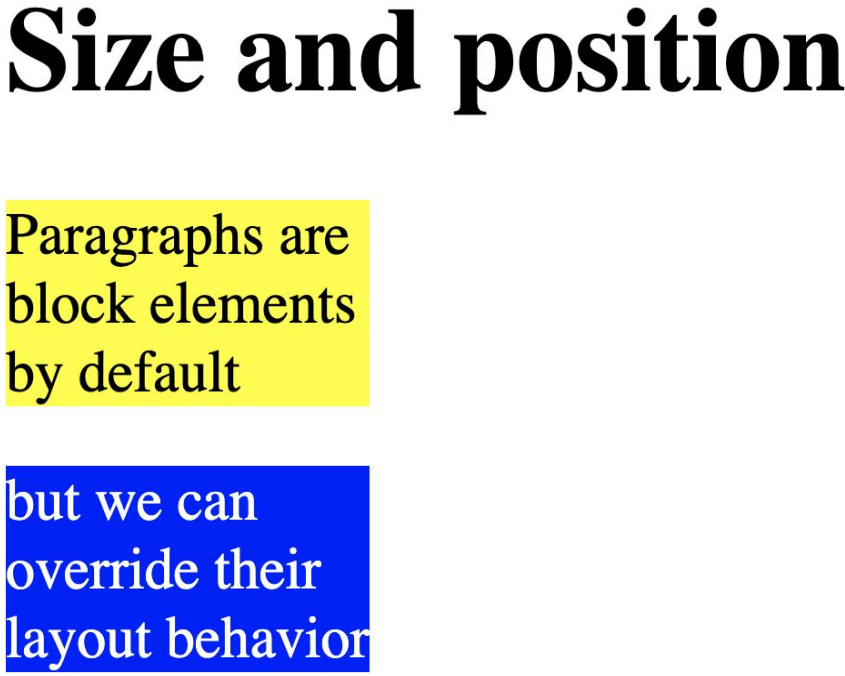
```
<h1>Size and position</h1>
<p style="
  background-color: yellow;">
Paragraphs are block
elements by default
</p>
<p style="
  background-color: blue;
  color: white;">
but we can override
their layout behavior
</p>
```



Styling width

Width configures width of element.
Note block elements still add vertical break

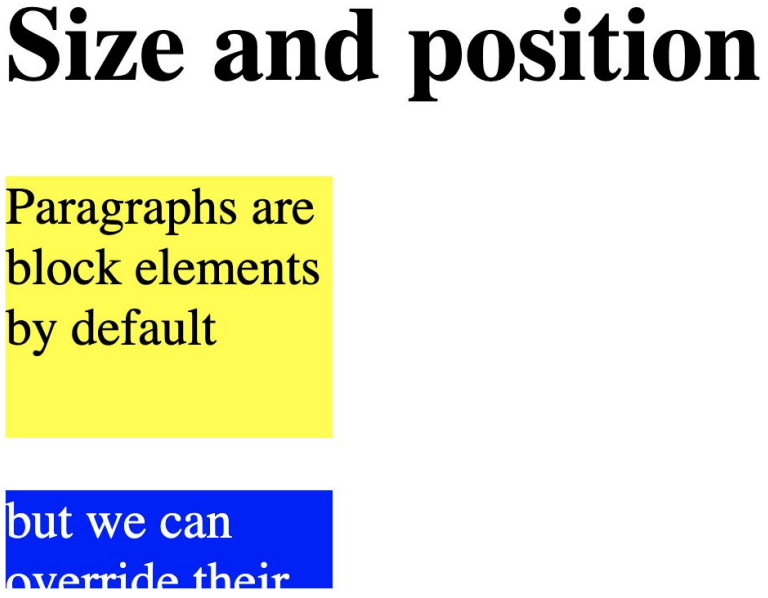
```
<h1>Size and position</h1>
<p style="
  width: 100px;
  background-color: yellow;
"> ...
</p>
<p style="
  width: 100px;
  background-color: blue;
  color: white;
"> ...
</p>
```



Styling height

Height controls height of element

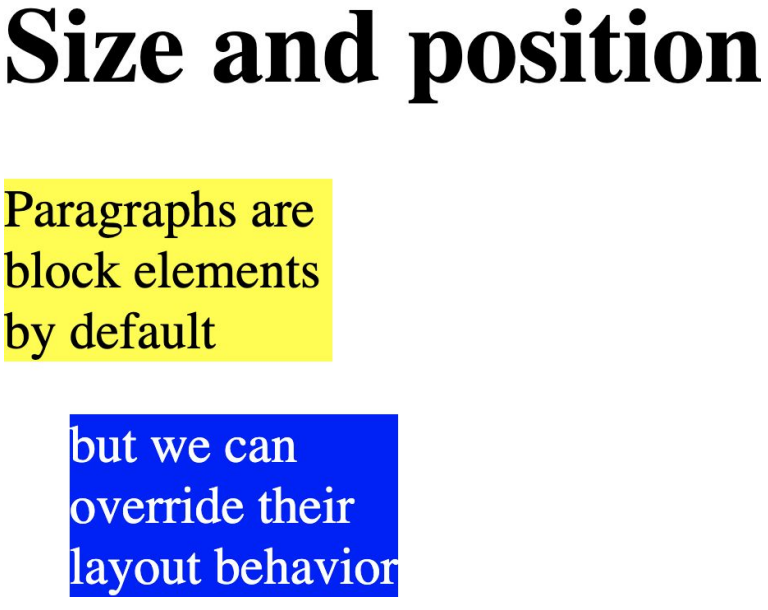
```
<h1>Size and position</h1>
<p style="
  width: 100px;
  height: 80px;
  background-color: yellow;
"> ... </p>
<p style="
  width: 100px;
  height: 30px;
  background-color: blue;
  color: white;
"> ...</p>
```



Styling relative position

Left and right move horizontally, can be -

```
<h1>Size and position</h1>
<p style="
  width: 100px;
  height: 80px;
  background-color: yellow;
"> ... </p>
<p style="
  position: relative;
  left: 20px;
  background-color: blue;
  color: white;
"> ...</p>
```



relative position is measured from default position

Styling relative position

Top and bottom moves vertically, can be -

```
<h1>Size and position</h1>
<p style="
  width: 100px;
  height: 80px;
  background-color: yellow;
"> ... </p>
<p style="
  position: relative;
  left: 20px;
  top: -30px;
  background-color: blue;
  color: white;
"> ... </p>
```

Size and position

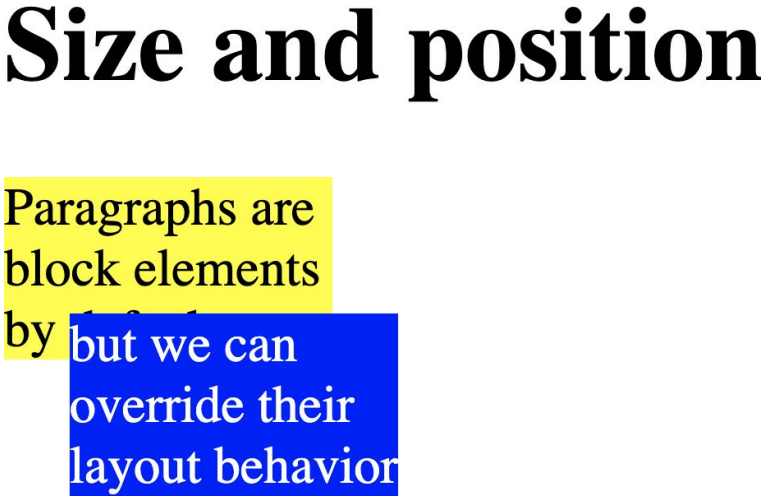
Paragraphs are block elements by [illegible] but we can override their layout behavior

vertical and horizontal can be combined to position anywhere

Styling relative position

Top and bottom moves vertically, can be -

```
<h1>Size and position</h1>
<p style="
  width: 100px;
  height: 80px;
  background-color: yellow;
"> ... </p>
<p style="
  position: relative;
  left: 20px;
  top: -30px;
  background-color: blue;
  color: white;
"> ... </p>
```



vertical and horizontal can be combined to position anywhere

Stretching horizontally

Use both left and right write to stretch

```
<p style="
  position: relative;
  left: 0px;
  right: 0px;
  width: 100px;
  top: -30px;
  background-color: blue;
  color: white;
"> ...</p>
```

Size and position

Paragraphs are block elements

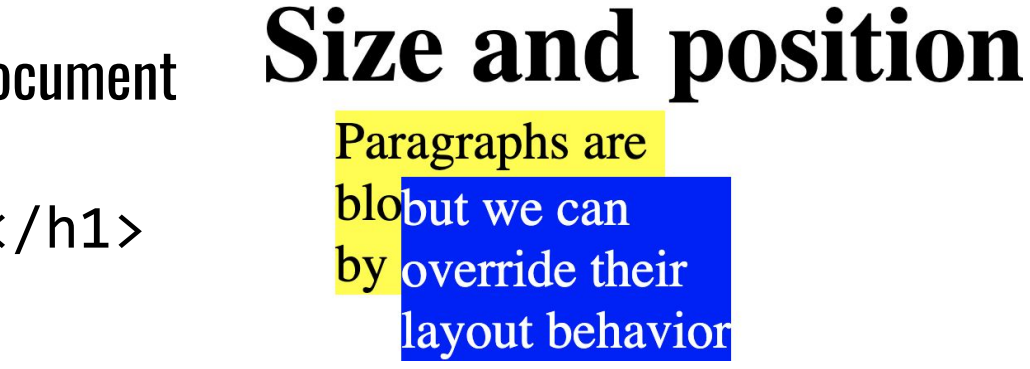
but we can override their layout behavior

remove **width** for stretch to work horizontally

Absolute position

Absolute position is relative to document

```
<h1>Size and position</h1>
<p style="
  position: absolute;
  left: 30px; top: 30px;
  width: 100px;">...</p>
<p style="
  position: absolute;
  left: 50px; top: 50px;
  width: 100px;">...</p>
```



Sticky header

```
<body style="
  position: relative;
  top: 60px;">
<h1 style="
  position: fixed;
  top: 0px;
  margin-top: 0px;
  background: white;">
   Size and Position
</h1>
<p>Lorem ipsum dolor sit amet.
<p>Lorem ipsum dolor sit amet.
```

Size and Position

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Portrait and Landscape Dimensions

index.css

```
.dimension-portrait {
    width: 75px;
    height: 100px;
}
.dimension-landscape {
    width: 100px;
    height: 75px;
}

.dimension-square {
    width: 75px;
    height: 75px;
}
```

index.html

```
<h2>Dimension</h2>
<div>
    <div class="dimension-portrait bg-color-yellow">
        Portrait
    </div>
    <div class="dimension-landscape bg-color-blue
                fg-color-white">
        Landscape
    </div>

    <div class="dimension-square bg-color-red">
        Square
    </div>
</div>
```

Browser

Relative Position

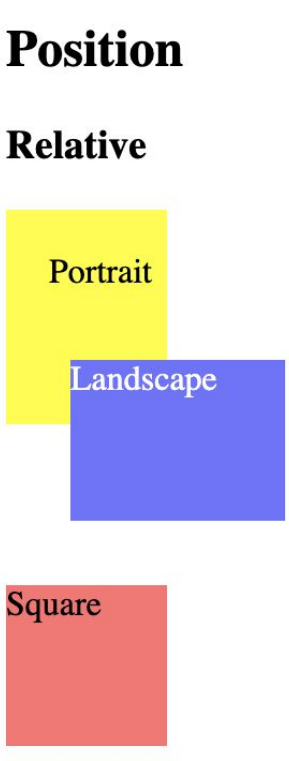
index.css

```
.pos-relative-nudge-up-right {
    position: relative;
    bottom: 30px;
    left: 30px;
}
.pos-relative-nudge-down-right {
    position: relative;
    top: 20px;
    left: 20px;
}
.pos-relative {
    position: relative;
}
```

index.html

```
<h2>Position</h2>
<h3>Relative</h3>
<div>
    <div class="bg-color-yellow ...">
        <div class="pos-relative-nudge-down-right">
            Portrait</div>
    </div>
    <div class="pos-relative-nudge-up-right ...">
        Landscape</div>
    <div class="bg-color-red ...">
        Square</div>
</div>
```

Browser



Absolute Position

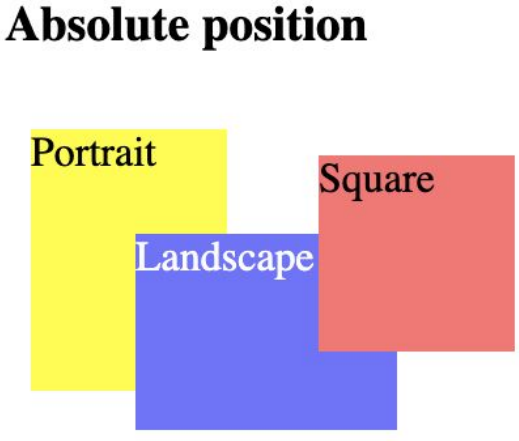
index.css

```
.pos-absolute-10-10 {
    position: absolute;
    top: 10px;
    left: 10px;
}
.pos-absolute-50-50 {
    position: absolute;
    top: 50px;
    left: 50px;
}
.pos-absolute-120-20 {
    position: absolute;
    top: 20px;
    left: 120px;
}
```

index.html

```
<h3>Absolute position</h3>
<div class="pos-relative">
    <div class="pos-absolute-10-10">
        Portrait
    </div>
    <div class="pos-absolute-50-50">
        Landscape
    </div>

    <div class="pos-absolute-120-20">
        Square
    </div>
</div>
```

Browser



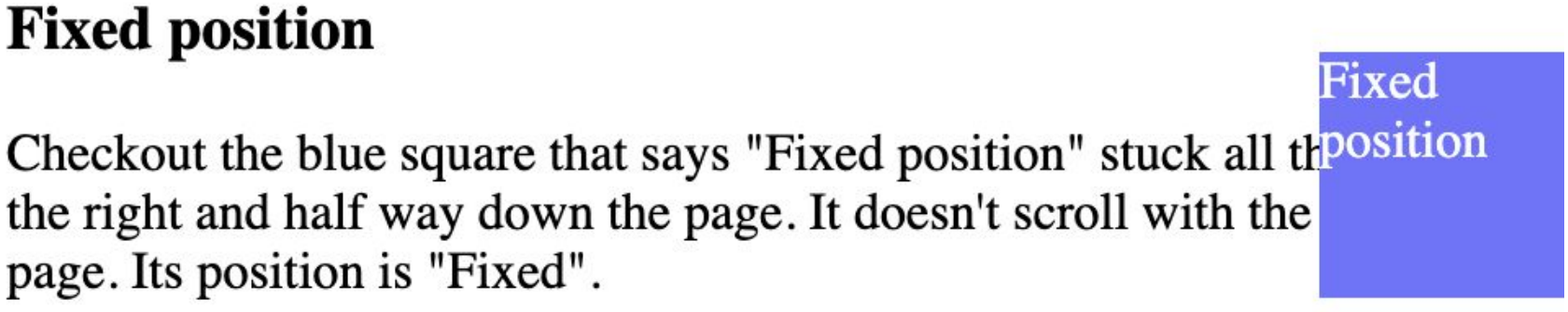
Fixed Position

index.css

```
.pos-fixed {
    position: fixed;
    right: 0px;
    bottom: 50%;
}
```

index.html

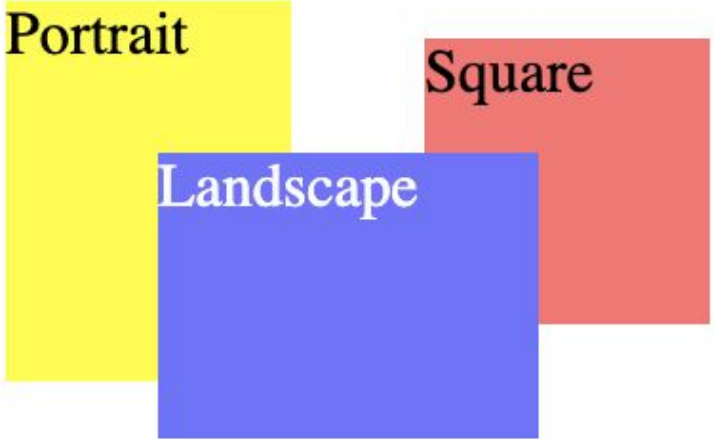
```
<h3>Fixed position</h3>
Checkout the blue square that says "Fixed position" stuck all the way on the right and half way down the page. It doesn't scroll with the rest of the page. Its position is "Fixed".

<div class="pos-fixed dimension-square bg-color-blue fg-color-white">Fixed position</div>
```



Z-Index

index.css

```
.zindex-bring-to-front {
    z-index: 10;
}
```



index.html

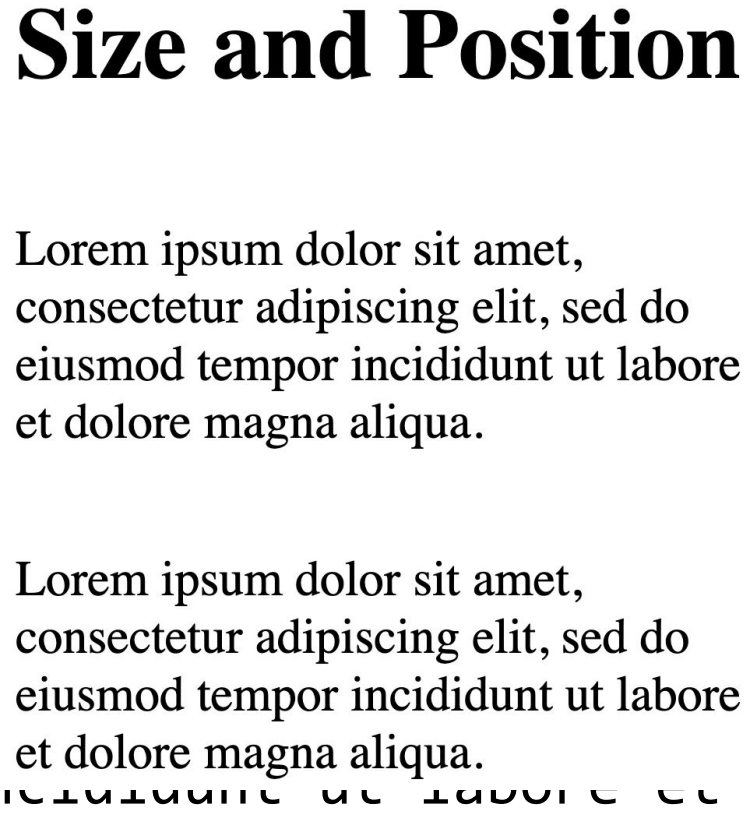
```
<h2>Z index</h2>
<div class="pos-relative">
    <div class="pos-absolute-10-10
                dimension-portrait">
                Portrait</div>
    <div class="zindex-bring-to-front
                pos-absolute-50-50
                dimension-landscape">
                Landscape</div>
    <div class="pos-absolute-120-20
                dimension-square">
               Square</div>
</div>
```

FLOAT

Use float to layout columns

Consider the following HTML snippet

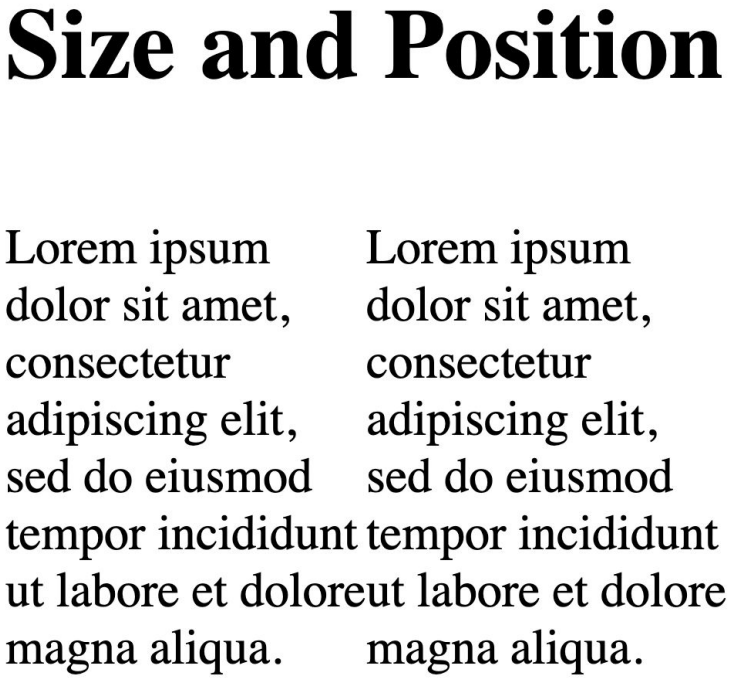
```
<h1>Size and Position</h1>
<p>
Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua.</p>
<p>
Lorem ipsum dolor sit amet, co
elit, sed do eiusmod tempor in
dolore magna aliqua.</p>
```



Laying out two columns

Consider the following HTML snippet

```
<h1>Size and Position</h1>
<p style="
  float: left;
  width: 50%">
Lorem ipsum dolor sit amet,
...</p>
<p style="
  float: left;
  width: 50%">
Lorem ipsum dolor sit amet,
...</p>
```



Laying out three columns

Consider the following HTML snippet

```
<h1>Size and Position</h1>
<p style="
  float: left;
  width: 33%">
Lorem ipsum dolor sit amet,
...</p>
<p style="
  float: left;
  width: 33%">
Lorem ipsum dolor sit amet,
...</p>
```

Size and Position

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

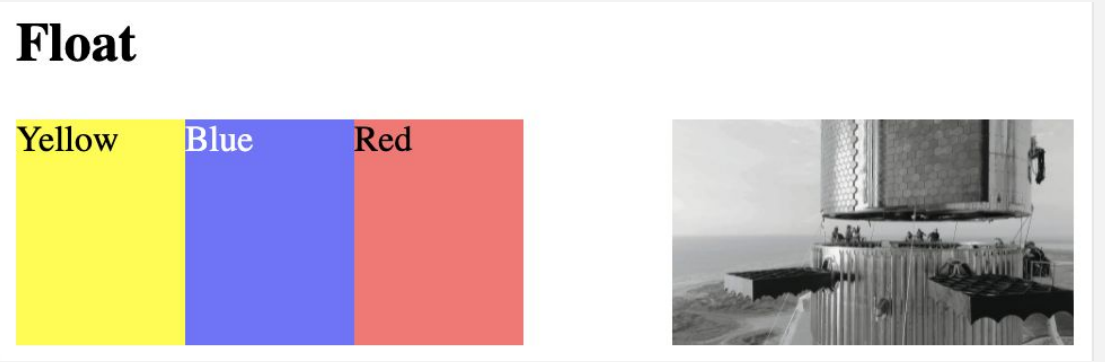
Float

index.css

```
.float-left {
    float: left;
}

.float-right {
    float: right;
    height: 100px;
}

.float-done {
    clear: both;
}
```

index.html

```
<h2>Float</h2>
<div>
    <div class="float-left dimension-portrait">Yellow</div>
    <div class="float-left dimension-portrait">Blue</div>
    <div class="float-left dimension-portrait">Red</div>
    <img class="float-right" src="starship.jpg"/>
    <div class="float-done"></div>
</div>
```



Grid Layout

index.css

```
.grid-row {
    clear: both;
}

.grid-col-half-page {
    width: 50%;
    float: left;
}
```

index.html

```
<h2>Grid layout</h2>
<div class="grid-row">
    <div class="grid-col-half-page ...">
        <h3>Left half</h3>
    </div>
    <div class="grid-col-half-page ...">
        <h3>Right half</h3>
    </div>
</div>
```

Grid layout

Grid Layout

index.css

```
.grid-row {
    clear: both;
}

.grid-col-third-page {
    width: 33%;
    float: left;
}
.grid-col-two-thirds-page {
    width: 67%;
    float: left;
}
```

index.html

```
<div class="grid-row">
    <div class="grid-col-third-page ...">
        <h3>Left third</h3>
    </div>
    <div class="grid-col-two-thirds-page ...">
        <h3>Right two thirds</h3>
    </div>
</div>
```

Grid Layout

index.css

```
.grid-row {
    clear: both;
}
.grid-col-left-sidebar {
    width: 20%;
    float: left;
}
.grid-col-main-content {
    width: 60%;
    float: left;
}
.grid-col-right-sidebar {
    width: 20%;
    float: left;
}
```

index.html

```
<div class="grid-row">
    <div class="grid-col-left-sidebar ...">
        <h3>Side bar</h3>
        <p>This is the left sidebar</p>
    </div>
    <div class="grid-col-main-content ...">
        <h3>Main content</h3>
        <p>This is the main content... </p>
    </div>
    <div class="grid-col-right-sidebar ...">
        <h3>Side bar</h3>
        <p>This is the right sidebar</p>
    </div>
</div>
```

ROTATING CONTENT

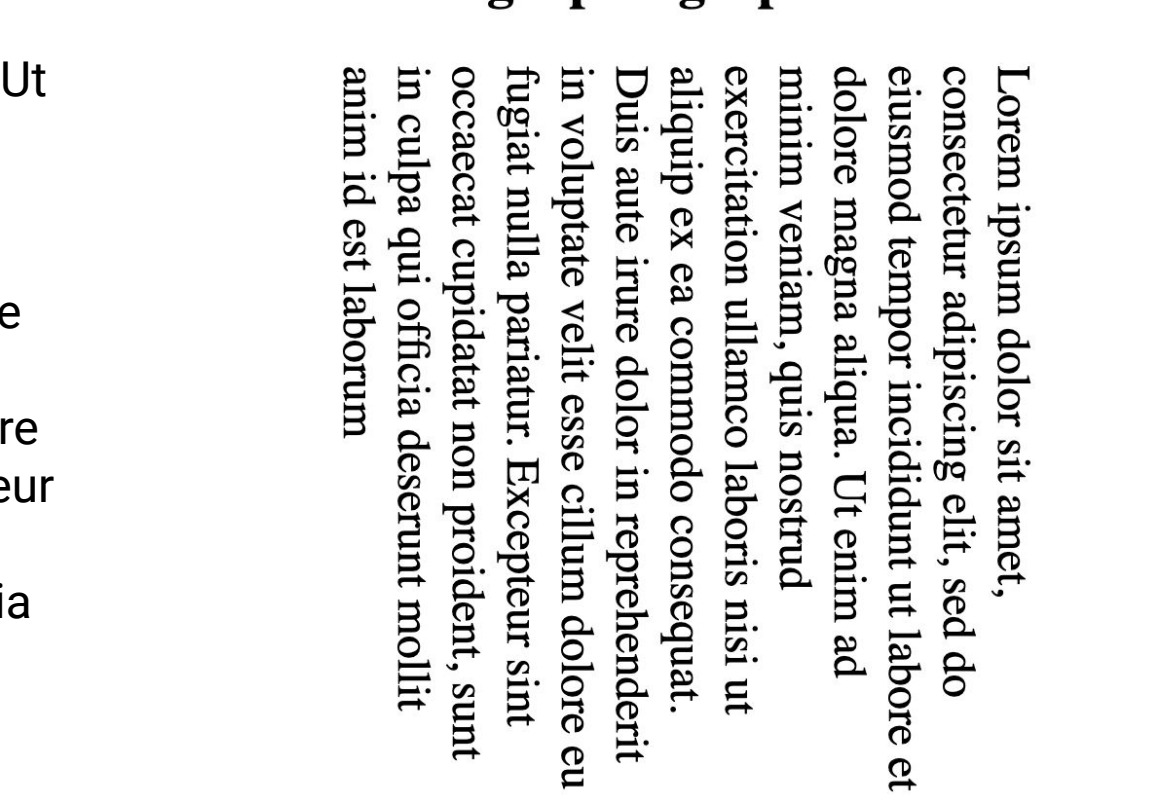
Rotate 90

index.html

```
<h2>Rotations</h2>
<h3>Rotating a paragraph</h3>
<p class="rotate-paragraph">
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p>
```

index.css

```
.rotate-paragraph {
    width: 250px;
    height: 250px;
    transform: rotate(90deg);
}
```



Rotating a List

index.html

```
<h3>Rotating a lists</h3>
<ul class="rotate-list">
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
```

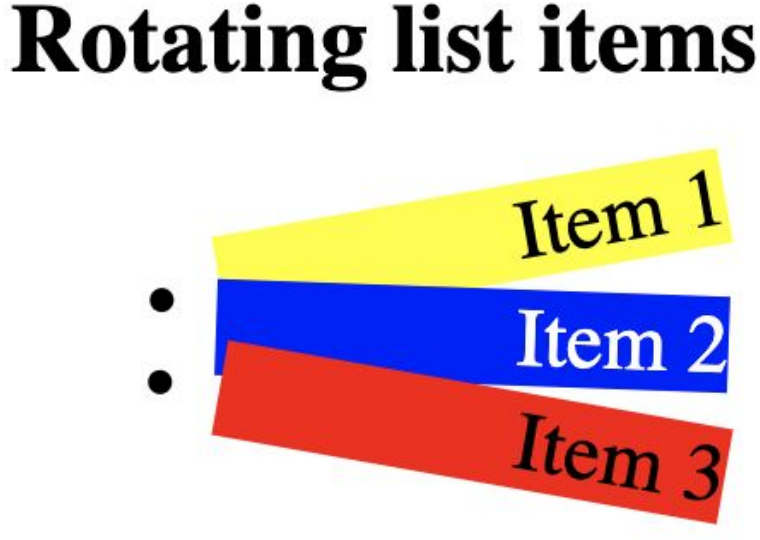
index.css

```
.rotate-list {
    transform: rotate(12deg);
    width: 100px;
    background-color: red;
    color: white;
}
```

Rotating List Items

index.html

```
<h3>Rotating list items</h3>
<ul class="rotate-list-items">
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
```



index.css

```
.rotate-list-items li {
    width: 100px;
    text-align: right;
}
.rotate-list-items li:nth-child(1) {
    transform: rotate(-10deg);
    background-color: yellow;
}
.rotate-list-items li:nth-child(2) {
    transform: rotate(2deg);
    background-color: blue;
    color: white;
}
.rotate-list-items li:nth-child(3) {
    transform: rotate(10deg);
    background-color: red;
}
```

Rotating Images

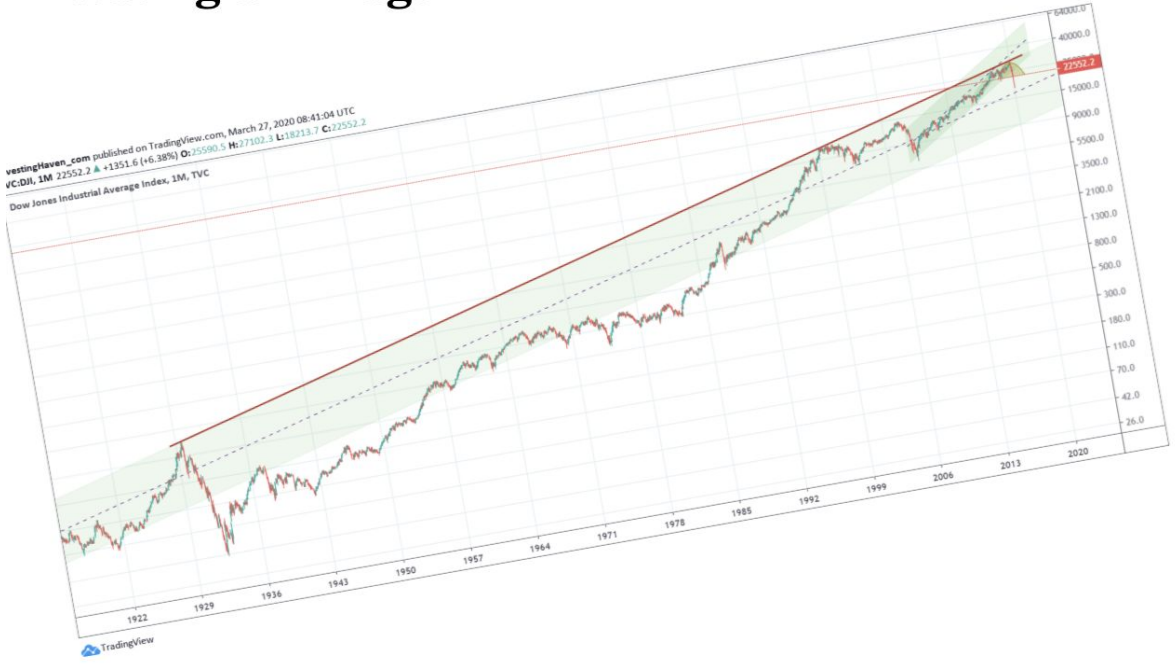
index.html

```
<h3>Rotating an image</h3>
<img class="rotate-image" width="400px"
    src="dowjones_100_years.png"/>
```

index.css

```
.rotate-image {
    transform: rotate(-10deg);
}
```

Rotating an image



Rotating a Table

index.html

```
<h3>Rotating a table</h3>
<br/><br/>
<table border="1" class="rotate-table">
   <tr><td>Return rate</td><td>Monthly contributions</td>
      <td>Balance after 40 years</td></tr>
   <tr><td>5</td><td>250</td><td>372,141.15</td></tr>
   <tr><td>5</td><td>500</td><td>744,282.29</td></tr>
   <tr><td>5</td><td>1000</td><td>1,488,564.58</td></tr>
   <tr><td>7</td><td>250</td><td>621,379.12</td></tr>
   <tr><td>7</td><td>500</td><td>1,242,758.23</td></tr>
   <tr><td>7</td><td>1000</td><td>2,485,516.46</td></tr>
   <tr><td>10</td><td>250</td><td>1,398,651.85</td></tr>
   <tr><td>10</td><td>500</td><td>2,797,303.70</td></tr>
   <tr><td>10</td><td>1000</td><td>5,594,607.40</td></tr>
   <tr><td>13</td><td>250</td><td>3,251,260.21</td></tr>
   <tr><td>13</td><td>500</td><td>6,502,520.42</td></tr>
   <tr><td>13</td><td>1000</td><td>13,005,040.83</td></tr>
</table>
```

index.css

```
.rotate-table {
   position: relative;
   left: 50px;
   transform: rotate(15deg);}
.rotate-table tr:nth-child(1) {
   background-color: black;
   color: white;}
.rotate-table tr:nth-child(n+2) {
   background-color: yellow;}
.rotate-table tr:nth-child(n+5) {
   background-color: blue;
   color: white;}
.rotate-table tr:nth-child(n+8) {
   background-color: red;}
.rotate-table tr:nth-child(n+11) {
   background-color: green;
   color: white;
}
```

Rotating a table

Return rate	Monthly contributions	Balance after 40 years
5	250	372,141.15
5	500	744,282.29
5	1000	1,488,564.58
7	250	621,379.12
7	500	1,242,758.23
7	1000	2,485,516.46
10	250	1,398,651.85
10	500	2,797,303.70
10	1000	5,594,607.40
13	250	3,251,260.21
13	500	6,502,520.42
13	1000	13,005,040.83

GRADIENT BACKGROUNDS

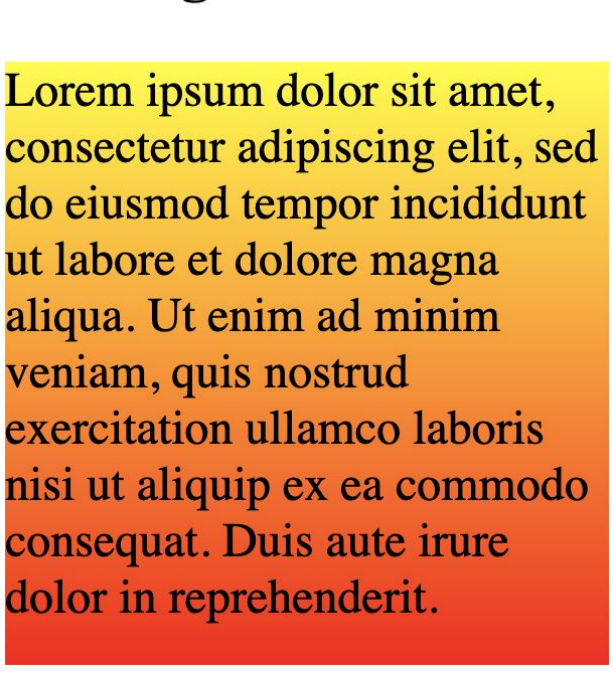
Linear Gradients

index.html

```
<h3>Linear gradients</h3>
<p class="gradients-linear">
Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore
et dolore magna aliqua. Ut enim ad
minim veniam, quis nostrud
exercitation ullamco laboris nisi ut
aliquip ex ea commodo consequat.
Duis aute irure dolor in
reprehenderit.</p>
```

index.css

```
.gradients-linear {
    background: linear-gradient(yellow, red);
    width: 200px;
    height: 200px;
}
```



Radial Gradient

index.html

```
<h3>Radial gradients</h3>
<p class="gradients-radial">
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit.</p>
```

index.css

```
.gradients-radial {
    background: radial-gradient(yellow, green);
    width: 200px;
    height: 200px;
}`
```

Radial gradients

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit.

FLEX

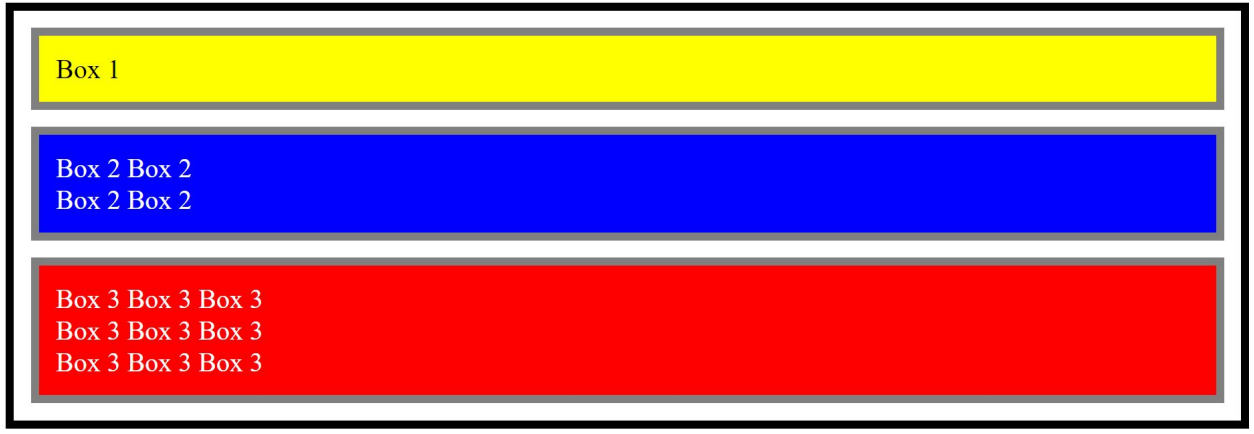
Consider

index.css

```
.flex-container {
    border-width: 5px;
    border-color: black;
    border-style: solid;
}
.flex-box {
    border: 5px gray solid;
    margin: 10px;
    padding: 10px;
}
.flex-box-1 {
    background-color: yellow;
}
.flex-box-2 {
    background-color: blue;
    color: white;
}
.flex-box-3 {
    background-color: red;
    color: white;
}
```

index.html

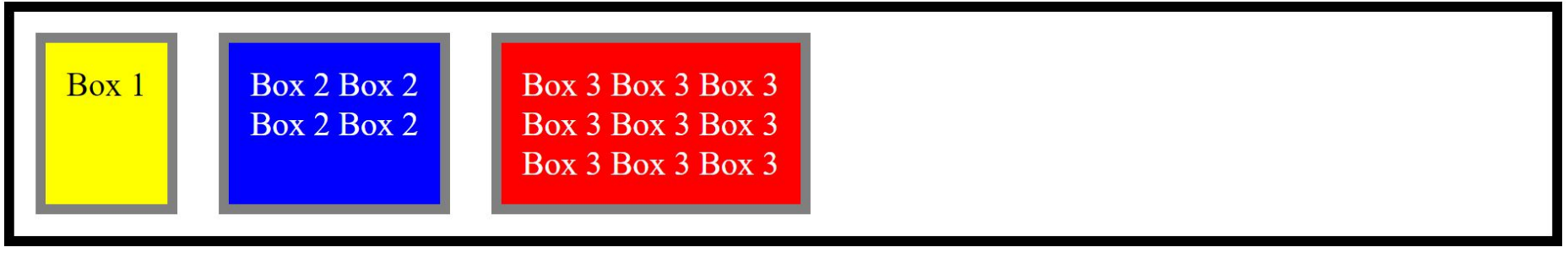
```
<div class="flex-container">
    <div class="flex-box flex-box-1">Box 1</div>
    <div class="flex-box flex-box-2">
        Box 2 Box 2<br/>Box 2 Box 2
    </div>
    <div class="flex-box flex-box-3">
        Box 3 Box 3 Box 3<br/>Box 3 Box 3 Box 3<br/>Box 3 Box 3
    </div>
</div>
```



Horizontal Align

index.css

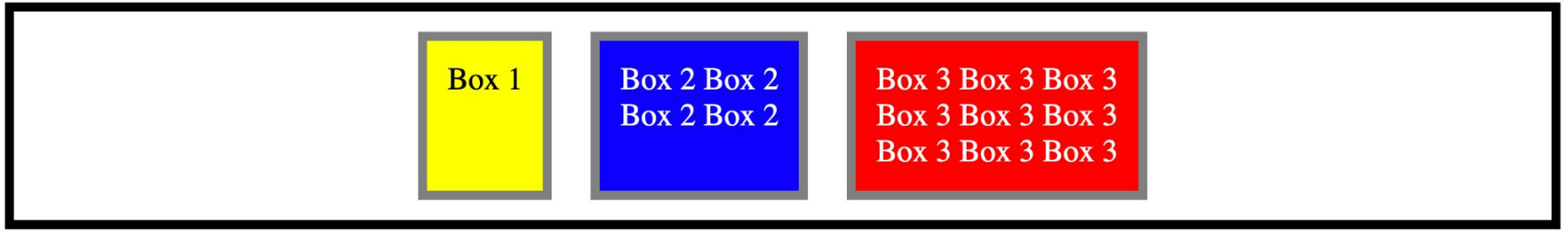
```
.flex-container {
    border-width: 5px;
    border-color: black;
    border-style: solid;
    display: flex;
}
```



Justifying Center

index.css

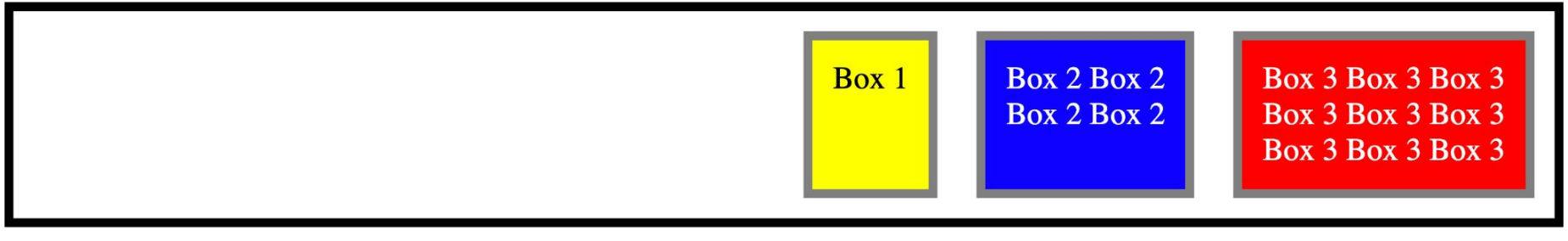
```
.flex-container {
    display: flex;
    justify-content: center;
}
```



Justifying End

index.css

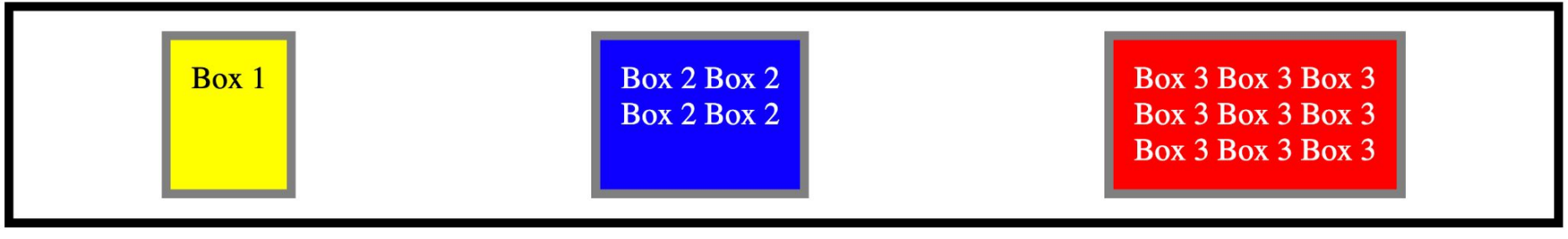
```
.flex-container {
    display: flex;
    justify-content: end;
}
```



Justifying Around

index.css

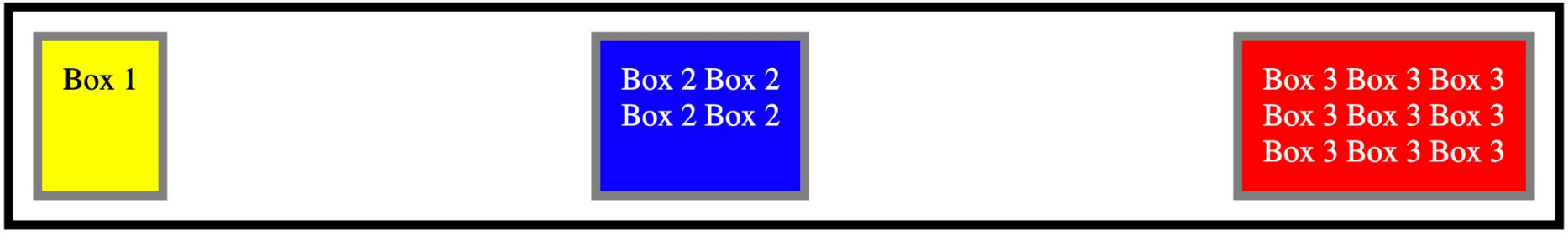
```
.flex-container {
    display: flex;
    justify-content: space-around;
}
```



Justify Between

index.css

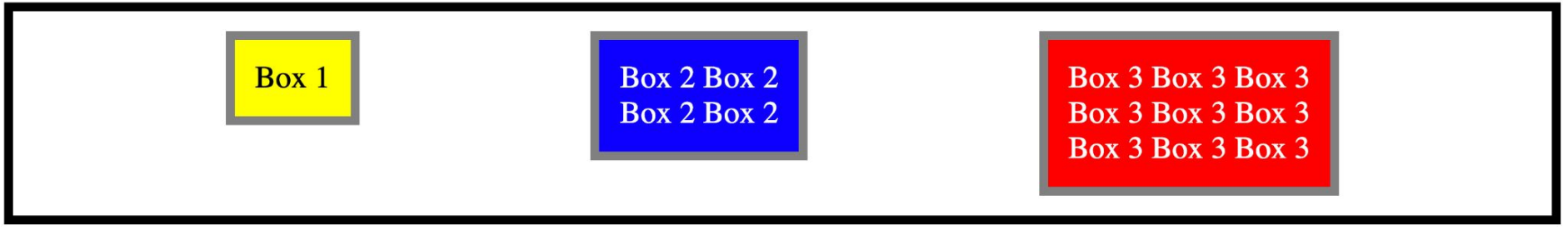
```
.flex-container {
    display: flex;
    justify-content: space-between;
}
```



Align Vertically Start

index.css

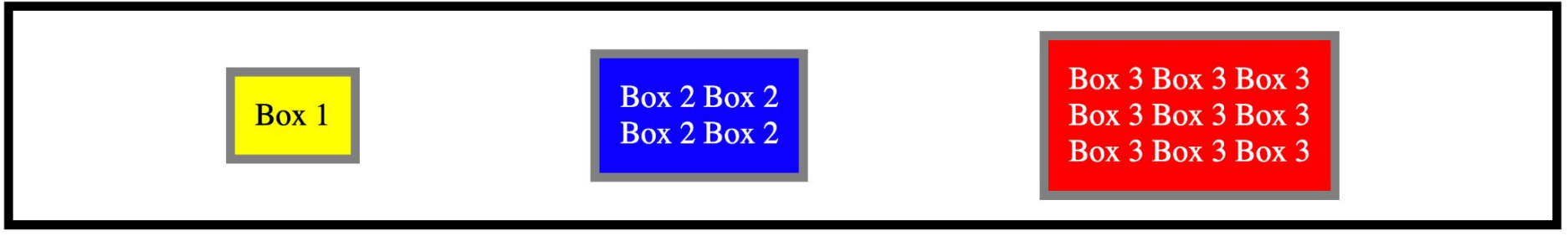
```
.flex-container {
    display: flex;
    justify-content: space-evenly;
    align-items: flex-start;
}
```



Align Vertical Center

index.css

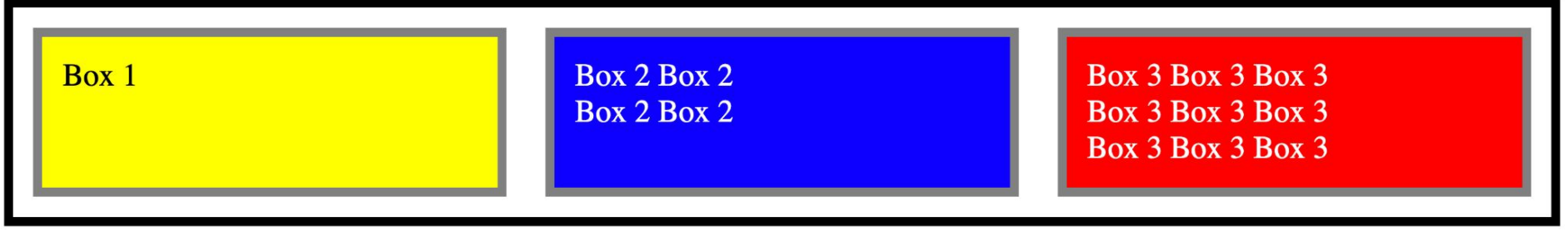
```
.flex-container {
    display: flex;
    justify-content: space-evenly;
    align-items: center;
}
```



index.css

Stretch Evenly

```
.flex-container {
    display: flex;
    justify-content: space-evenly;
    align-items: center;
}
.flex-container > .flex-box {
    flex-basis: 100%;
}
```



FLEX

Consider

index.css

```
.flex-container {
    border-width: 5px;
    border-color: black;
    border-style: solid;
}
.flex-box {
    border: 5px gray solid;
    margin: 10px;
    padding: 10px;
}
.flex-box-1 {
    background-color: yellow;
}
.flex-box-2 {
    background-color: blue;
    color: white;
}
.flex-box-3 {
    background-color: red;
    color: white;
}
```

index.html

```
<div class="flex-container">
    <div class="flex-box flex-box-1">Box 1</div>
    <div class="flex-box flex-box-2">
        Box 2 Box 2<br/>Box 2 Box 2
    </div>
    <div class="flex-box flex-box-3">
        Box 3 Box 3 Box 3<br/>Box 3 Box 3 Box 3<br/>Box 3 Box 3
    </div>
</div>
```