CSS Layout Part I

Web Development

CSS Selector Examples

- Choosing and applying Class and ID names requires careful attention
- Strive to use clear meaningful names as far as possible.

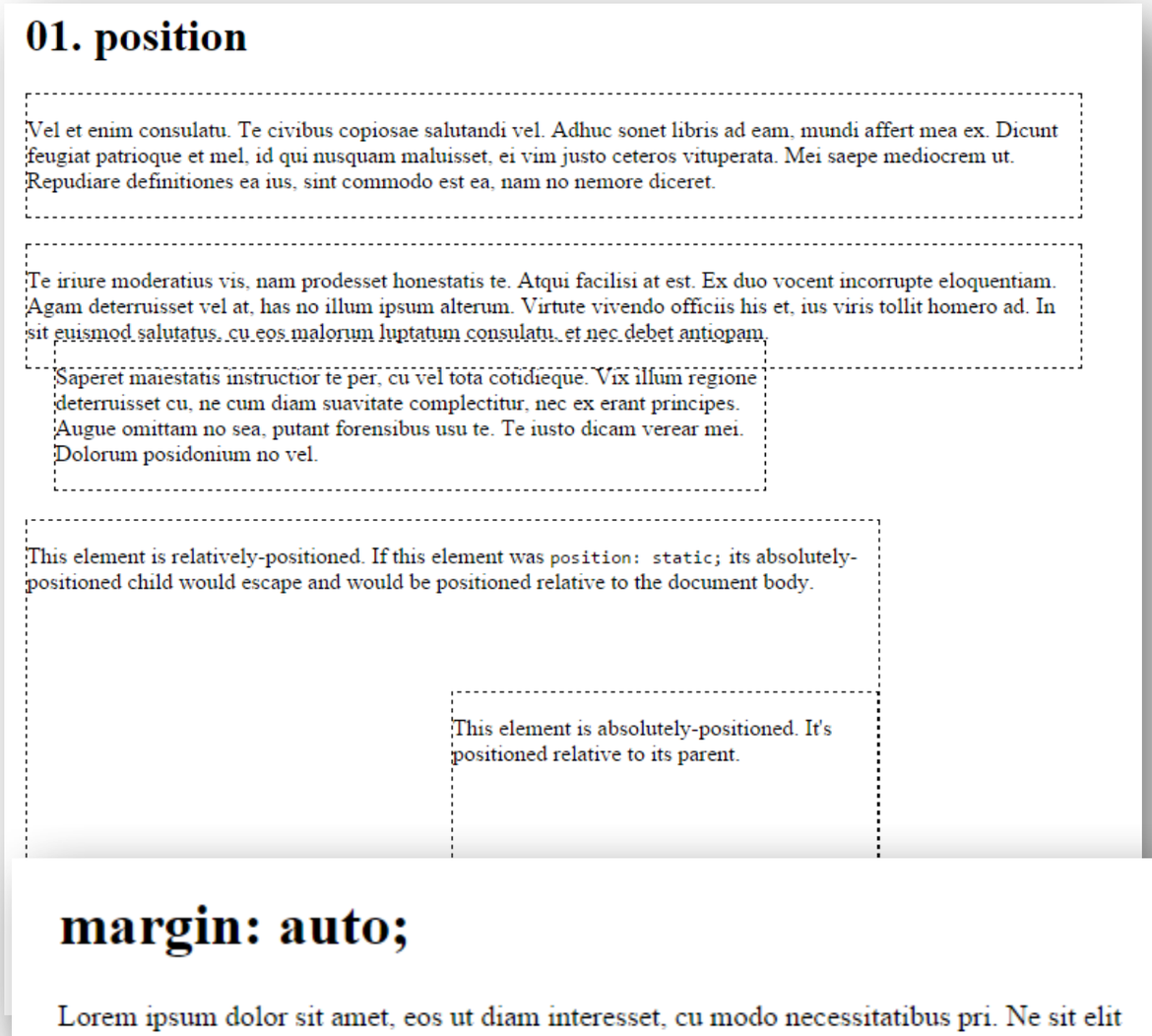
CSS Selectors Summary (simple)

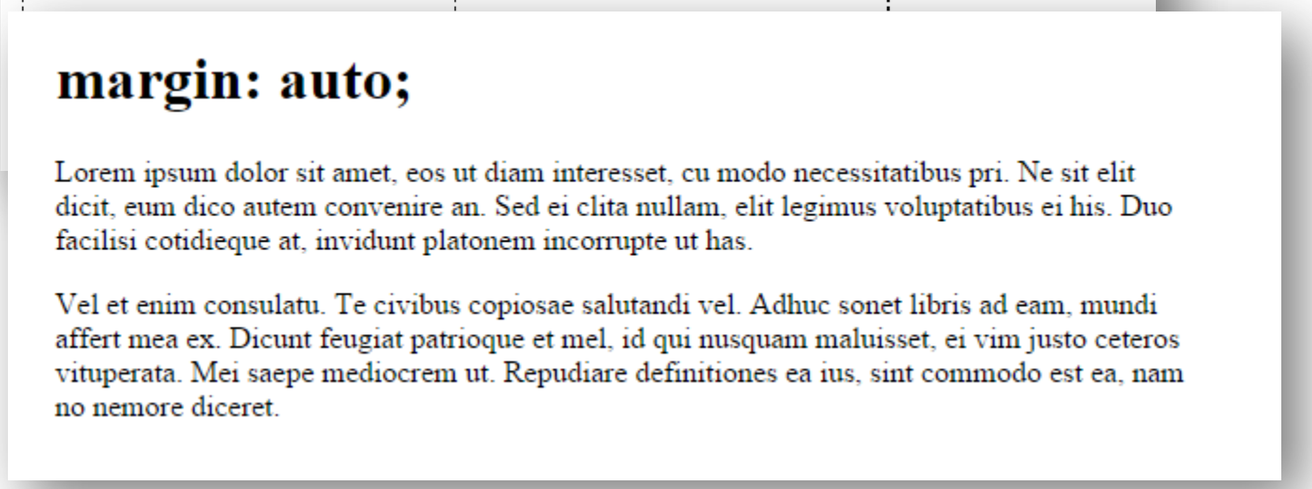
Selector	Applies to
p	All paragraphs in the document
.about	All elements in the document with a class value of about
#corporatehistory	The element in the document with an id value of corporate history (if present)
h1,h2,h3	All first-, second-, and third-level headings in the document
.privacy, .copyright	All elements with a class of privacy or copyright
#header,#footer	The element assigned an id of header, and the element assigned an id of footer
p.footnote	All paragraphs assigned a class of footnote

CSS Selectors Summary (advanced)

Selector	Applies to
#bodycopy.usergenerated	An element that has been assigned both an id of bodycopy and a class of usergenerated
.navigation a	All links with an ancestor parent assigned a class of navigation
#primarynavigation li.current	All list items with a class of current and an ancestor parent with an id of primarynavigation
.about #bodycopy	Any element on the site with an id of bodycopy and an ancestor parent assigned a class of about
body#personalproducts, body#proproducts, body#enterpriseproducts	The body elements within the site assigned the ids personalproducts, proproducts, and enterpriseproducts
body#personalproducts #bodycopy, body#proproducts #bodycopy, body#enterpriseproducts #bodycopy	The elements assigned an id of bodycopy, within the documents suggested by the previous example
ol li ol li ol li	A list item in the third level of a nested ordered list

Lab 06: Learn CSS Layout Part 1





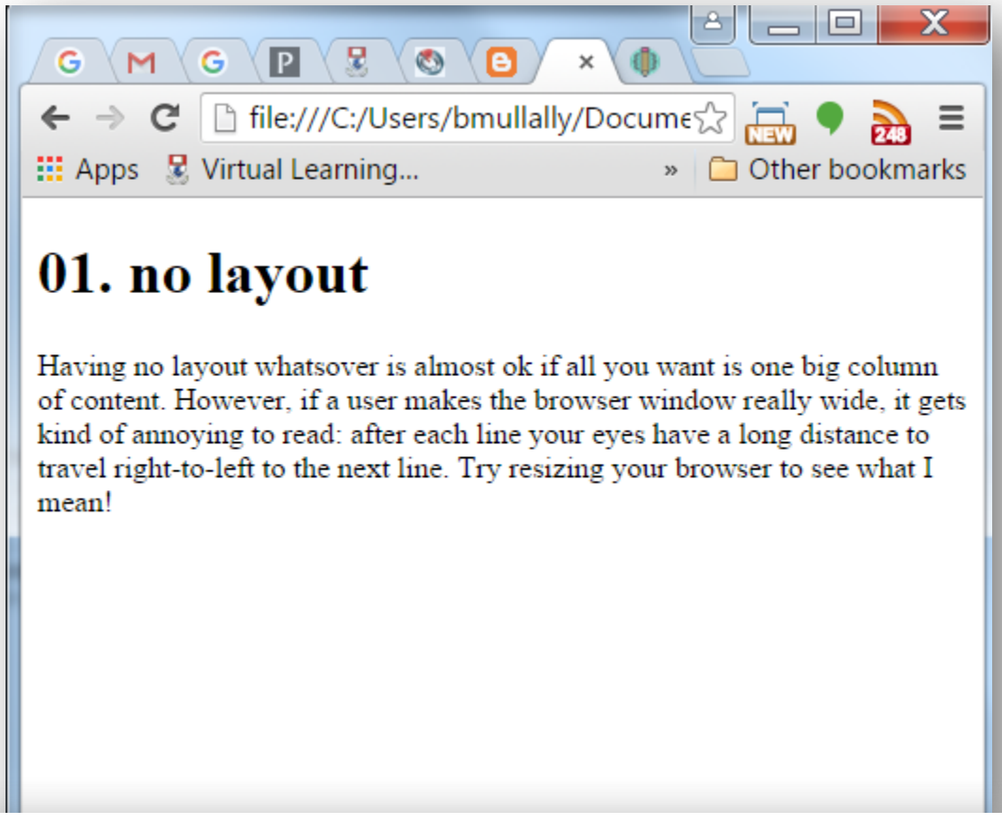
CSS Layout I

- No layout
- The “display” property
- Margin: auto;
- Max-width
- The box model
- Box-sizing
- Position
- Float
- Clear

No Layout

- No layout is ok if you want one big column of content.

- What happens if you make the browser really wide?



Your eyes have to travel a long distance right to left to the next line.

01. no layout

Having no layout whatsover is almost ok if all you want is one big column of content. However, if a user makes the browser window really wide, it gets kind of annoying to read: after each line your eyes have a long distance to travel right-to-left to the next line. Try resizing your browser to what I mean!

`display` is CSS's most important property for controlling layout.

Every element has a default display value depending on what type of element it is. The default for most elements is usually `block` or `inline`.

block

`<div>` is the standard block level element. It starts on a new line and stretches to the left and right as far as it can. Others include `<p>` and new in HTML5 are `<header>` `<footer>` `<section>` and more.

inline

`<span>` is the standard inline element. The `<a>` element is the most common inline element.

none

Setting `display` to `none` will render the page as though the element does not exist.

`visibility: hidden;` will hide the element, but the element will take up the space it would usually.

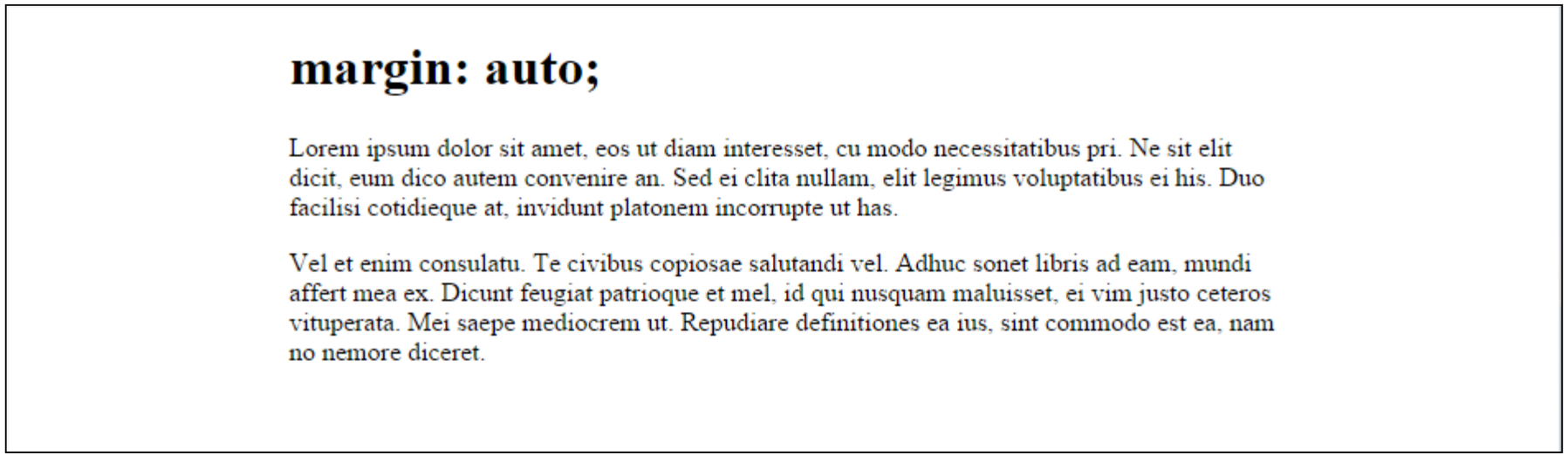
"display" property

```
.test-one {
  display:block;
}

test-two {
  display:inline;
}

test-three {
  display:none;
}

.test-four {
  visibility:hidden;
}
```

margin: auto;

```css
#main {
  width: 600px;
  margin: 0 auto;
}
```

- Setting the **width** of a block-level element will stop it stretching out to the edges left and right. Then set the margin to auto left and right. This horizontally centres that element within its container.
- The only problem is when the browser window is narrower than the width you set. What will the browser do?



max-width

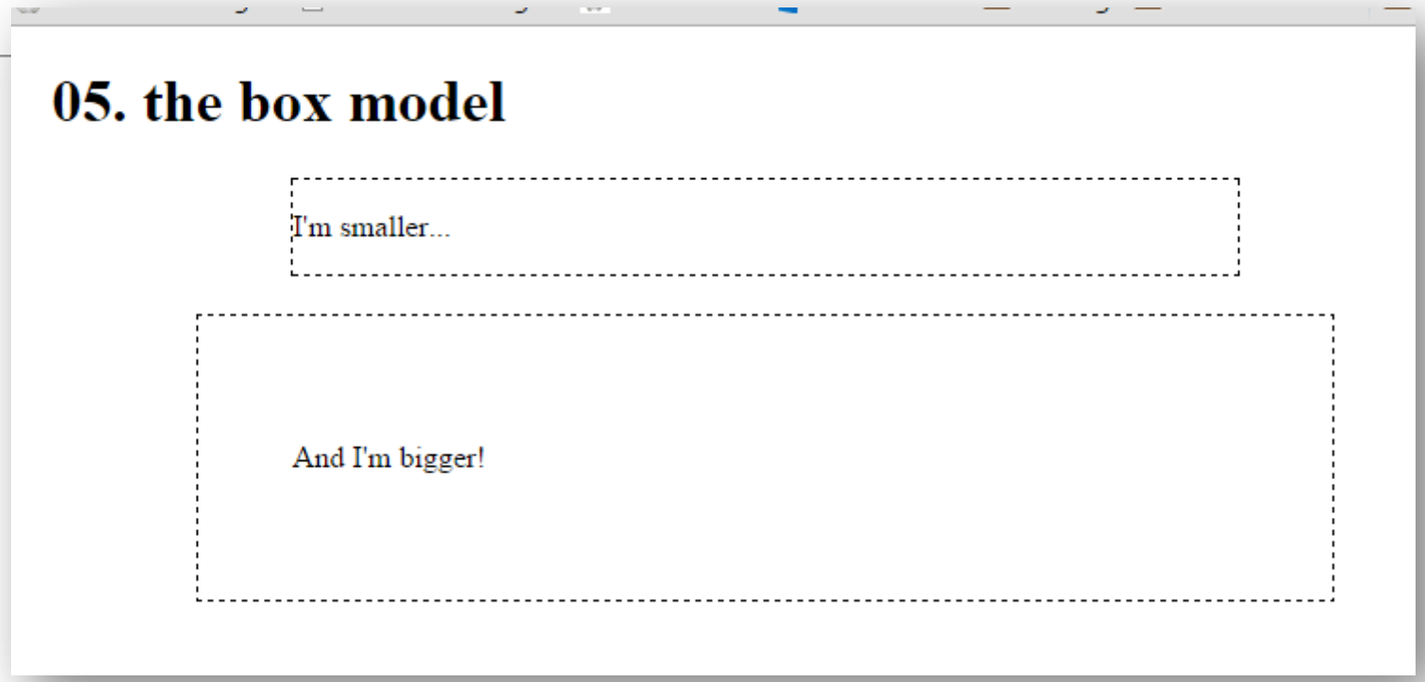
```css
#main {
  max-width: 600px;
  margin: 0 auto;
}
```

- To prevent a horizontal scroll bar appearing we can use the max-width property. For a site that needs to be usable on a mobile it is important to use max-width.

- max-width is supported by all major browsers including IE7

the box model

- When you set the width of an element, it can actually appear bigger because of the border and padding.
- These two elements have their width set the same but they end up rendered as different sizes.
- The solution might be to write smaller width value than we wanted

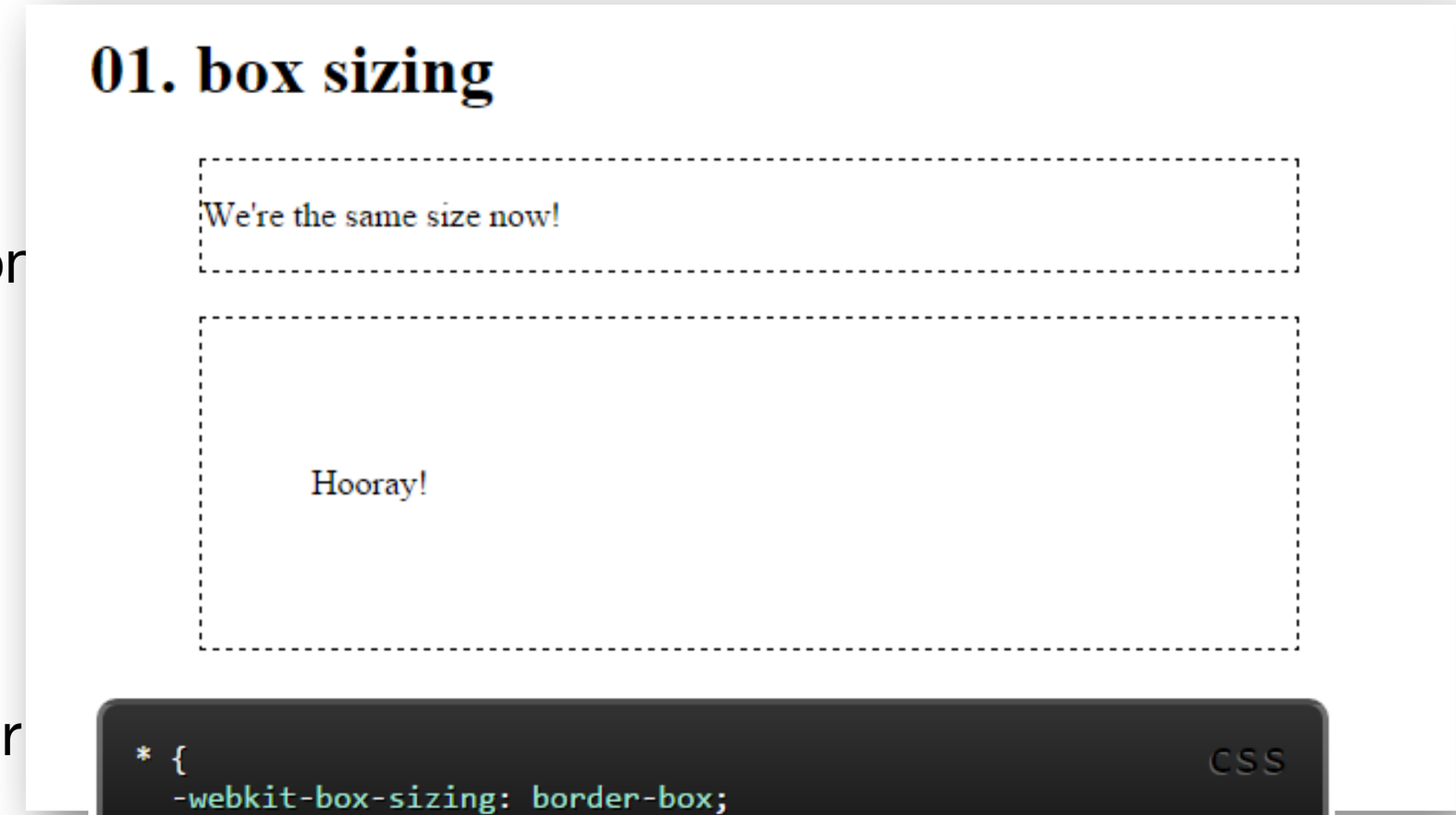



```css
.simple {
  width: 500px;
  margin: 20px auto;
}

.fancy {
  width: 500px;
  margin: 20px auto;
  padding: 50px;
  border-width: 10px;
}
```

box-sizing

- A new CSS property called `box-sizing` was created for this problem. When you set `box-sizing: border-box;` on an element the padding and border of that element no longer increases its width.
- Here we have set the `box-sizing: border-box;` on both elements.
- Some authors want all elements on their pages to work this way.
- Use –webkit- and –moz- prefixes to ensure all browser engines targeted.

```css
.simple {
  width: 500px;
  margin: 20px auto;
  -webkit-box-sizing: border-box;
     -moz-box-sizing: border-box;
          box-sizing: border-box;
}

.fancy {
  width: 500px;
  margin: 20px auto;
  padding: 50px;
  border: solid blue 10px;
  -webkit-box-sizing: border-box;
     -moz-box-sizing: border-box;
          box-sizing: border-box;
}
```




```css
* {
  -webkit-box-sizing: border-box;
     -moz-box-sizing: border-box;
          box-sizing: border-box;
}
```

position

- position property often used to make more complex layouts.

- Has a range of possible values, and their name can be confusing and difficult to remember.

- **Static**

```
.static {
  position: static;
}
```

- static is the default value. An element with `position: static;` is not positioned in any special way.

- A static element is said to be *not positioned* and an element with its position set to anything else is said to be *positioned*.

position

- **relative**

- relative behaves the same was as static unless you add some extra properties to adjust the element away from its normal position. Other content will not be adjusted to fit into any gap left by that element.

```
<div class="relative1 dashed">
  <p>
    Te iriure moderatius vis, nam prodesset honestatis te. Atqui facilisi
    at est. Ex duo vocent incorrupte eloquentiam. Agam deterruisset vel
    at, has no illum ipsum alterum. Virtute vivendo
    officiis his et, ius viris
    cu eos malorum luptatum co
  </p>
</div>

<div class="relative2 dashed">
  <p>
    Saperet maiestatis instruc
    illum regione deterruisset
    ex erant principes. Augue
    iusto dicam verear mei. Do
  </p>
</div>
```

```
.relative1 {
  position: relative;
}

.relative2 {
  position: relative;
  top: -20px;
  left: 20px;
  width: 500px;
}
```

Te iriure moderatius vis, nam prodesset honestatis te. Atqui facilisi at est. Ex duo vocent incorrupte eloquentiam. Agam deterruisset vel at, has no illum ipsum alterum. Virtute vivendo officiis his et, ius viris tollit homero ad. In sit euismod salutatus, cu eos malorum luptatum consulatu, et nec debet antiopam.

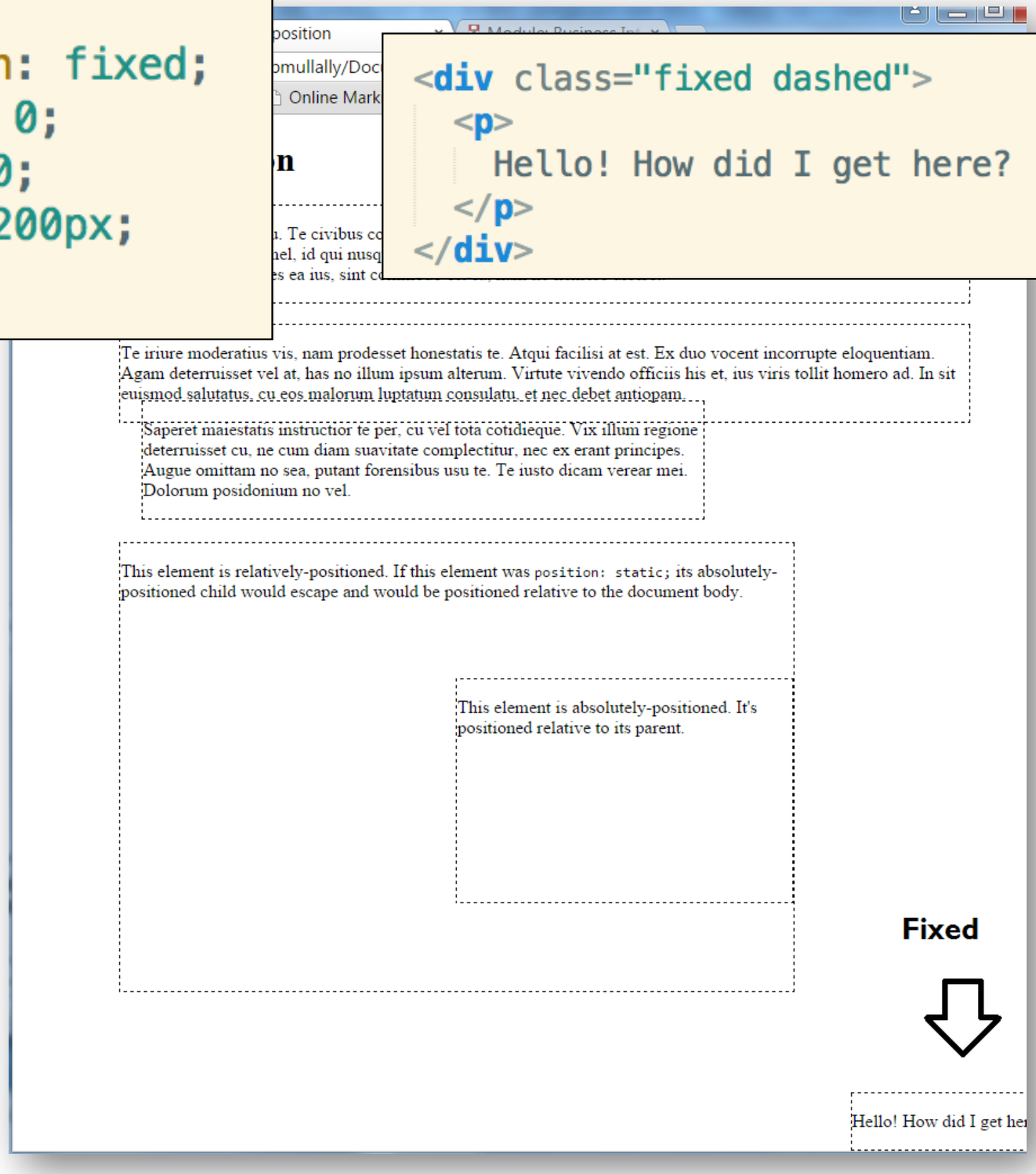
Saperet maiestatis instructior te per, cu vel tota cotidieque. Vix illum regione deterruisset cu, ne cum diam suavitate complectitur, nec ex erant principes. Augue omittam no sea, putant forensibus usu te. Te iusto dicam verear mei. Dolorum posidonium no vel.

position

```
.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 200px;
}
```

```
<div class="fixed dashed">
  <p>
    Hello! How did I get here?
  </p>
</div>
```

- **fixed**
- A fixed element is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled.
- As with relative the top, right, bottom, and left properties are used.
- A fixed element does not leave a gap in the page where it would normally have been located.

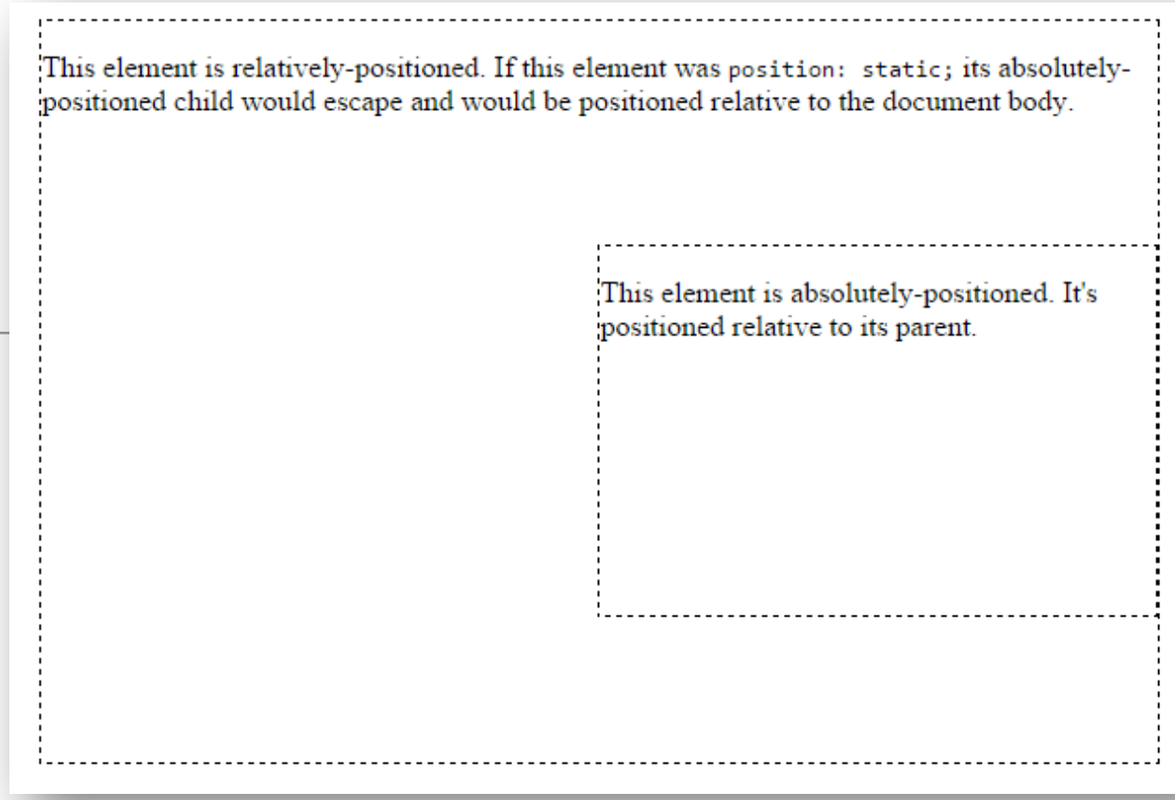


position

- **absolute**

- absolute is the trickiest position value. absolute behaves like fixed except relative to the nearest positioned ancestor instead of relative to the viewport. If an absolutely positioned element has no positioned ancestors, it uses the document body, and still moves along the page scrolling.

- This is tricky, but can be important for creating sophisticated CSS layouts.




```css
.relative3 {
  position: relative;
  width: 600px;
  height: 400px;
}

.absolute {
  position: absolute;
  top: 120px;
  right: 0;
  width: 300px;
  height: 200px;
}
```

```html
<div class="relative3 dashed">
  <p>
    This element is relatively-positioned. If this element was <code>
    position: static;</code> its absolutely-positioned child would escape
    and would be positioned relative to the document body.
  </p>
  <div class="absolute dashed">
    <p>
      This element is absolutely-positioned. It's positioned relative to
      its parent.
    </p>
  </div>
</div>
```

```css
* {
  -webkit-box-sizing: border-box;
     -moz-box-sizing: border-box;
          box-sizing: border-box;
}

body {
  max-width: 80%;
  margin: 0 auto;
  margin-bottom: 100px;
}

.dashed {
  border: dashed 1px;
}

.container {
  position: relative;
}

nav {
  position: absolute;
  left: 0px;
  width: 200px;
}

section {
  margin-left: 200px;
}

footer {
  position: fixed;
  bottom: 0;
  left: 0;
  height: 50px;
  width: 100%;
}
```

```html
<h1> 01. position example </h1>

<div class="container dashed">
  <nav>
    <ul>
      <li> <a href="#">Home</a>       </li>
      <li> <a href="#">Taco Menu</a>  </li>
      <li> <a href="#">Draft List</a> </li>
      <li> <a href="#">Hours</a>      </li>
      <li> <a href="#">Directions</a> </li>
      <li><a href="#">Contact</a>     </li>
    </ul>
  </nav>
  <section class="dashed">
    <p>
      The <code>margin-left</code> style for <code>section</code>s makes sure there is room for
      the <code>nav</code>. Otherwise the absolute and static elements would overlap
    </p>
  </section>
  <section class="dashed">
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et
      dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor.
      Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum
      augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent
      convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed
      ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis.
      Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum.
      Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.
    </p>
  </section>
  <section class="dashed">
    <p>
      Notice what happens when you resize your browser. It works nicely!
    </p>
  </section>
  <footer class="da
    <p>
      If you use a
      bottom</code>
    </p>
  </footer>
</div>
```

position example

01. position example

- Home
- Taco Menu
- Draft List
- Hours
- Directions
- Contact

The `margin-left` style for `sections` makes sure there is room for the `nav`. Otherwise the absolute and static elements would overlap

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.

Notice what happens when you resize your browser. It works nicely!

If you use a fixed header or footer, make sure there is room for it! I put a `margin-bottom` on the `body`.

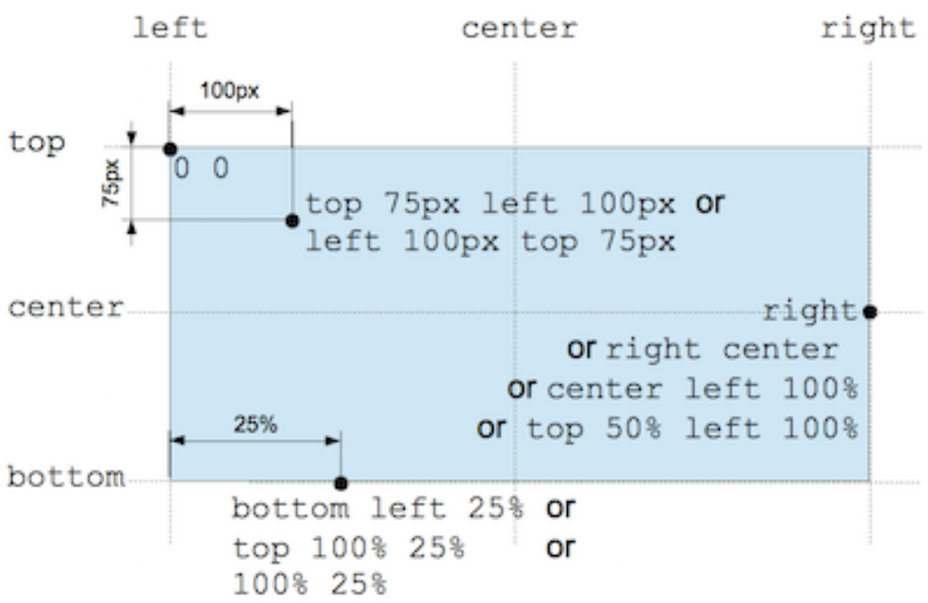
float

- Float can be used for wrapping text around images:

```css
img {
  float: right;
  margin: 0 0 1em 1em;
}
```

```html
<h1> 01. float </h1>
<p>
  <img src="css-positioning.png" alt="An Image">
  the position CSS data type denotes a coordinate in a 2D space used to set a
  location relative to a box. A specific  coordinate can be given by a two keywords,
  with specific offsets. A keyword represent one edge of the element's box or the
  medium line between two edges: left, right, top, bottom or center (which represents
  either the center between the left and right edges, or the center between the top
  or bottom edges, depending on the context). An offset can be either a relative
  value, expressed as a percentage, or an absolute length value. Positive values are
  offset towards the right or towards the bottom, whichever is suitable. Negative
  values are offset in the other.
</p>
```

the position CSS data type denotes a coordinate in a 2D space used to set a location relative to a box. A specific coordinate can be given by a two keywords, with specific offsets. A keyword represent one edge of the element's box or the medium line between two edges: left, right, top, bottom or center (which represents either the center between the left and right edges, or the center between the top or bottom edges, depending on the context). An offset can be either a relative value, expressed as a percentage, or an absolute length value. Positive values are offset towards the right or towards the bottom, whichever is suitable. Negative values are offset in the other.



float without clear

```css
.box {
  float: left;
  width: 200px;
  height: 100px;
  margin: 1em;
}
```

```html
<h1> 09. clear </h1>
<section class="box dashed">
  <p>
    Lorem ipsum dolor sit amet, eos ut diam interesset, cu modo
    necessitatibus pri.
  </p>
</section>
<section class="after-box dashed">
  <p>
    Vel et enim consulatu. Te civibus copiosae salutandi vel. Adhuc sonet
    libris ad eam, mundi affert mea ex. Dicunt feugiat patrioque et mel,
    id qui nusquam maluisset, ei vim justo ceteros vituperata. Mei saepe
    mediocrem ut. Repudiare definitiones ea ius, sint commodo est ea, nam
    no nemore diceret.
  </p>
</section>
```

09. clear

Lorem ipsum dolor sit amet, eos ut diam interesset, cu modo necessitatibus pri.

Vel et enim consulatu. Te civibus copiosae salutandi vel. Ac ad eam, mundi affert mea ex. Dicunt feugiat patrioque et me nusquam maluisset, ei vim justo ceteros vituperata. Mei saep ut. Repudiare definitiones ea ius, sint commodo est ea, nam

diceret.

float with clear

- the **clear** property is important for controlling the behaviour of floats.

You use the value **left** to clear elements floated to the left. You can also clear **right** and **both**.

```css
.box {
  float: left;
  width: 200px;
  height: 100px;
  margin: 1em;
}
.after-box {
  clear: left;
}
```

```html
<h1> 09. clear </h1>
<section class="box dashed">
  <p>
    Lorem ipsum dolor sit amet, eos ut diam interesset, cu modo
    necessitatibus pri.
  </p>
</section>
<section class="after-box dashed">
  <p>
    Vel et enim consulatu. Te civibus copiosae salutandi vel. Adhuc sonet
    libris ad eam, mundi affert mea ex. Dicunt feugiat patrioque et mel,
    id qui nusquam maluisset, ei vim justo ceteros vituperata. Mei saepe
    mediocrem ut. Repudiare definitiones ea ius, sint commodo est ea, nam
    no nemore diceret.
  </p>
</section>
```