Week5- Topics

CSS positioning

CSS Boxes

CSS treats each HTML element as if it is in its own box

Control the appearance of each box:

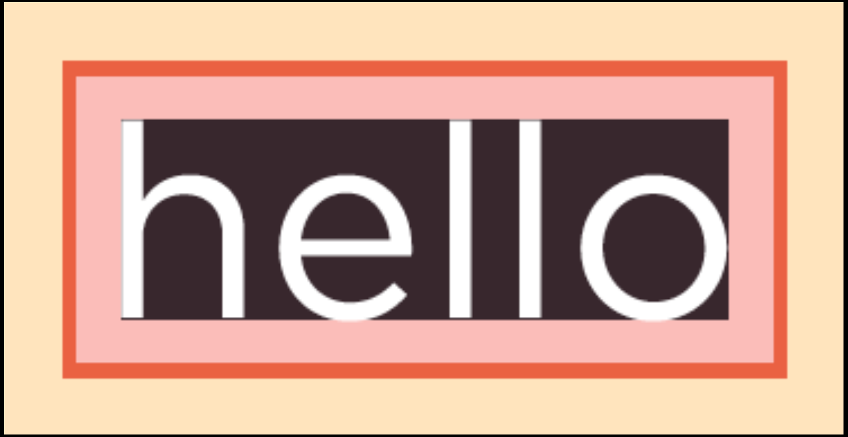
- Dimensions
- Borders
- Margins & padding
- Show & hide

CSS Boxes

Border

Margins – outside the border

Padding – space between the border and the content



CSS Boxes

```
height: 300px;
width: 300px;

min-width: 450px;
max-width: 650px;

min-height: 10px;
max-height: 30px;

overflow: hidden;
overflow: scroll;
```

CSS Boxes

border-width:

thin
medium
thick

border-top-width
border-right-width
border-bottom-width
border-left-width

```
p.one {
        border-width: 2px;}
p.two {
        border-width: thick;}
p.three {
        border-width: 1px 4px 12px 4px;}
```

School of Design

border-style:

```
p.one {border-style: solid;}
p.two {border-style: dotted;}
p.three {border-style: dashed;}
p.four {border-style: double;}
p.five {border-style: groove;}
p.six {border-style: ridge;}
p.seven {border-style: inset;}
p.eight {border-style: outset;}
```

ex.css_border_style.html

border-color:

border-top-color
border-right-color
border-bottom-color
border-left-color

```
p.one {
        border-color: #0088dd;}
p.two {
        border-color: #bbbbaa #111111 #ee3e80 #0088dd;}
```

border:

width, style and color in that specific order

```
p {
        width: 250px;
        border: 3px dotted #0088dd;}
```

padding:

Space between the content of an element and its border

Most often - px

padding-top

padding-right

padding-bottom

padding-left

```
p.example {
        padding: 10px;}
```

margin:

Space between the boxes

Most often - px

margin-top

margin-right

margin-bottom

margin-left

```
p.example {
        margin: 20px;}
```

margin:

Space between the boxes (most often – px)

margin-top

margin-right

margin-bottom

margin-left

clockwise order: top, right, bottom, left

```
p.example {
        margin: 20px;}
p.example2 {
        margin: 1px 2px 3px 4px; }
```

Centering boxes

- set a width for the box (otherwise it will take up the full width of the page)
- setting the left and right margins to auto will make the browser put an equal gap on each side of the box
- for older browsers the element that the box sits inside should have a text-align property with its value set to center

display:

converts inline elements into a block-level elements or vice versa

inline

block

inline-block

causes a block-level element to flow like an inline element, while retaining other features of a block-level element

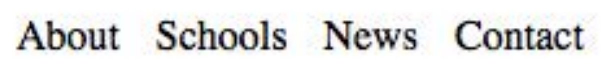
none

hides element form the page

display:

```html
<ul>
<li>Home</li>
<li>Products</li>
<li class="coming-soon">Services</li>
<li>About</li>
<li>Contact</li>
</ul>
```

```css
li {
        display: inline;
        margin-right: 10px;}
li.coming-soon {
        display: none;}
```


About Schools News Contact

display:

hide boxes but leaves a space where the elements would have been

a blank space will appear in place of an element

hidden

visible

Ex. css_display.html

visibility:

```
<ul>
<li>Home</li>
<li>Products</li>
<li class="coming-soon">Services</li>
<li>About</li>
<li>Contact</li>
</ul>
```

```
li {
        display: inline;
        margin-right: 10px;}
li.coming-soon {
        visibility: hidden;}
```

CSS layout

Layout

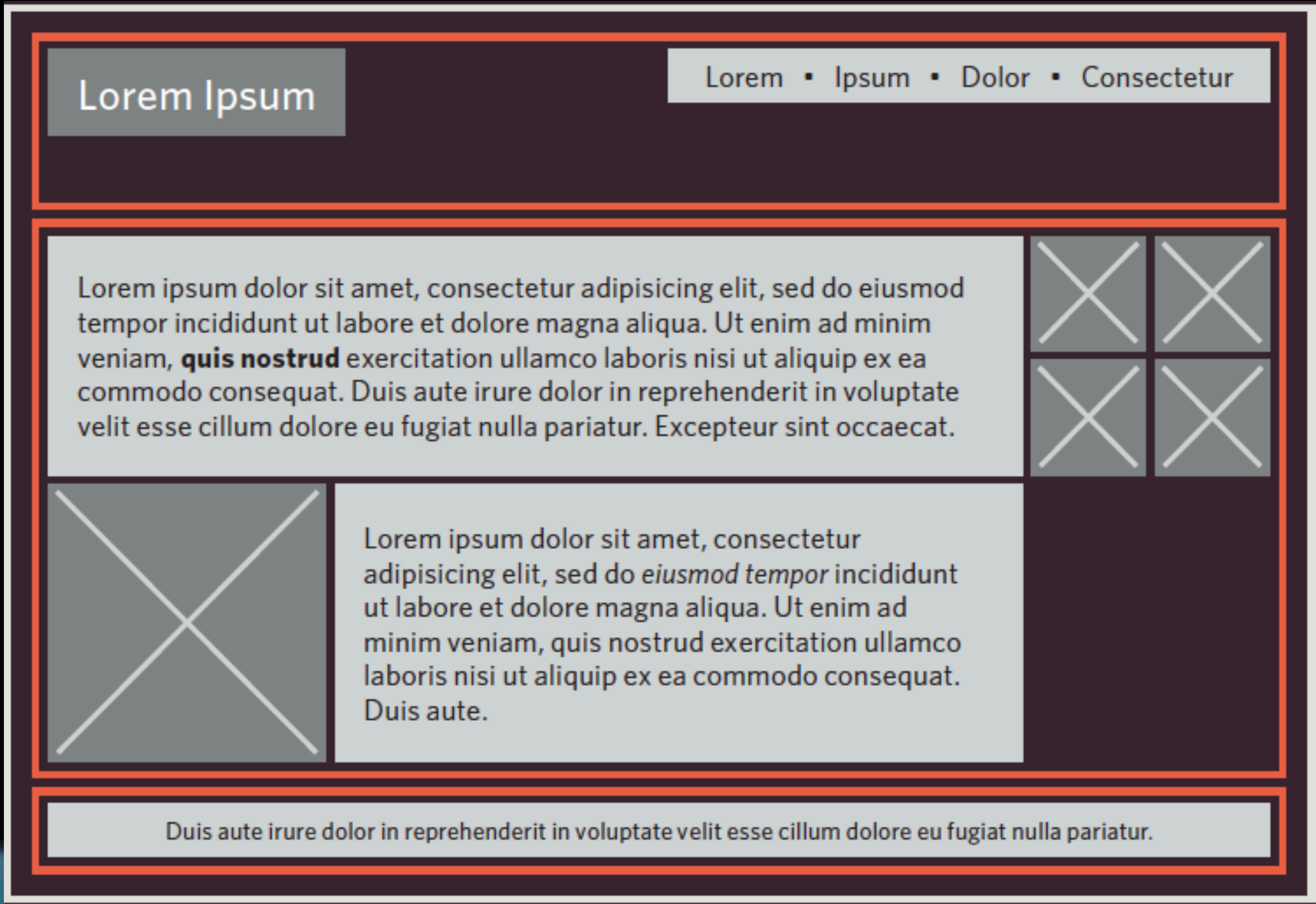
Composition

Positioning

CSS layout

grouping
a number
of elements
together inside
a block - <div>

parent
child



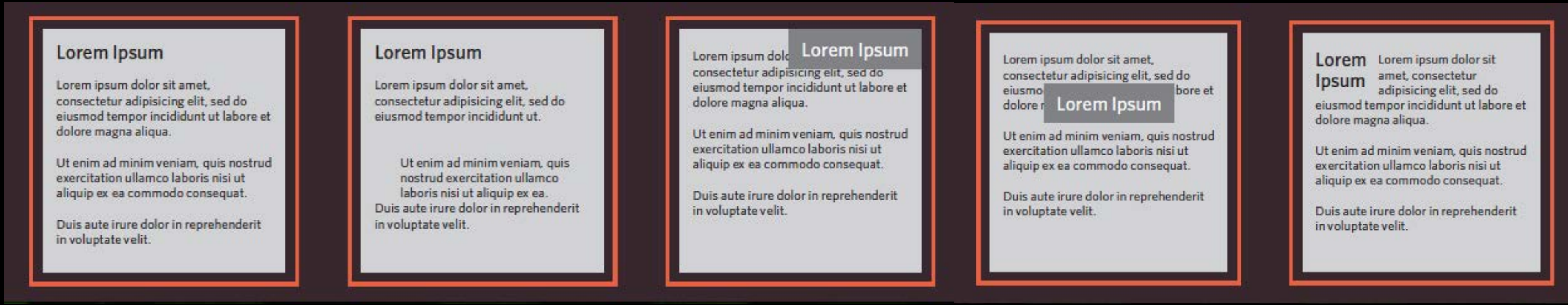
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, **quis nostrud** exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat.

Lorem ipsum dolor sit amet, consectetur adipisicing 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.

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

CSS positioning

Normal | Relative | Absolute | Fixed | Floating


Lorem Ipsum
Lorem ipsum dolor sit amet, consectetur adipisicing 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.
Lorem Ipsum
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut.
Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea.
Duis aute irure dolor in reprehenderit in voluptate velit.
Lorem Ipsum
Lorem ipsum dolor sit amet, consectetur adipisicing 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.
Lorem Ipsum
Lorem ipsum dolor sit amet, consectetur adipisicing 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.
Lorem Ipsum
Lorem ipsum dolor sit amet, consectetur adipisicing 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.

CSS positioning

normal flow (static)

- Every block-level element appears on a new line, causing each item to appear lower down the page than the previous one – default

```
position: static;
```

relative positioning

- A relative positioned element is positioned relative to its normal position
- does not affect the position of surrounding elements; they stay in the position they would be in in normal flow

```
position: relative;
```

CSS positioning

absolute positioning

- An absolute position element is positioned relative to the first parent element that has a position other than static. If no such element is found, the containing block is <html>
- Removed from the normal flow
- The document and other elements behave like the absolutely positioned element does not exist
- can overlap other elements

```
position: absolute;
```

- Due to the way different mobile browsers treat the viewport, fixed positioning can be somewhat unreliable

CSS positioning

fixed positioning

- An element with fixed position is positioned relative to the browser window
- will not move even if the window is scrolled

```
position: fixed;
```

floating elements

- Floated element is taken out of normal flow and positioned
- it to the far left or right of a containing box
- other content can flow around

```
float: right;
```

School of Design

CSS positioning

floating elements

- Floated element is taken out of normal flow and positioned
- it to the far left or right of a containing box
- other content can flow around

```
float: right;
```

Ex. css_float.html

CSS positioning

clear:

- no other element should touch the sides of a box

    left

    right

    both

    none

```
.clear { clear: left;}
…
<p class="clear">
```

CSS positioning

- Parents of floating elements can be treated as no width, no border, etc.
- To avoid it add overflow: auto; & width: properties.

```
div {
        border: 1px solid #665544;
        /*overflow: auto;*/
        /*width: 100%;}*/
        }
```

Ex. css_parent_float.html

Multiple columns with <div> and floats

- <div> element used to group items in a box

width:	sets the width of the columns
float:	positions the columns next to each other
margin:	creates a gap between the columns

```
.column1 { float: left; width: 620px; margin: 10px;}
.column2 { float: left; width: 300px; margin: 10px;}
```

CSS Layout

Dektop	Mobile
960-1000 px width	960 – 1024 px- .etc.
570-600 px height	640 – 768 px – etc.

- Variable key message areas
- Above the fold – area that users can see without scrolling
- Less than a second
- Concise content intro
- Hint at more content
- Responsive designs change depending on the screen size

CSS stacking order

Z- index

used to determine the stacking order of positioned elements

```
z-index: 10;
```

used to overlay elements on top of each other to create a specific visual effect

Fixed Width Layouts (px)

Pros

- do not change size
- greater control over the appearance and position of items on the page
- control the lengths of lines of text regardless of the size of the user's window
- image size remain the same relative to the rest of the page

Cons

- change size depending on the user screen / device
- If a user increases font sizes, text might not fit into the allotted spaces
- works best on devices that have a site or resolution similar to target design
- the page will often take up more vertical space than a liquid layout with the same content

Liquid Layouts (%)

Pros

- Expand, Stretch and contract
- the page can contract to fit it without the user having to scroll to the side
- tolerant of users setting font sizes larger than the designer intended (because the page can stretch)

Cons

- no control the width of sections
- the design can look very different
- unexpected gaps around certain
- elements or items squashed together
- lines of text can become illegible
- words may be squashed and you can end up with few words on each line
- fixed width items (images) can overflow over the text

CSS Images

width: height:

consistently sized images across a web site

CSS control the sizes of the

```
img.large { width: 500px; height: 500px;}
img.medium { width: 250px; height: 250px;}
img.small { width: 100px; height: 100px;}
```

CSS Images

```
img.align-left {
        float: left;
        margin-right: 10px;}
img.align-right {
        float: right;
        margin-left: 10px;}
img.medium {
        width: 250px;
        height: 250px;}
```

CSS Images

```
img.align-left {
        float: left;
        margin-right: 10px;}
img.align-right {
        float: right;
        margin-left: 10px;}
img.medium {
        width: 250px;
        height: 250px;}

<img src="images/print1.jpg" alt=”flower" class="align-right medium" />
```

School of Design

CSS Images

```
img.align-center {
	display: block; margin: 0px auto;}
img.medium {
	width: 250px; height: 250px;}



<img src="images/print1.jpg" alt=”flower" class="align-center medium"
" />
```

CSS Images

background-image:

```
body {
background-image: url("images/pattern.gif");}
```

CSS Images

```
background-repeat:
        repeat-x
        repeat-y
        no-repeat
Background-attachment:
        fixed
        scroll
body {
        background-image: url("images/header.gif");
        background-repeat: repeat-x;
        background-attachment: fixed;}
```

CSS Images

background-position:

in the browser window

left top left

center left

bottom center

top center

center center

bottom right

top right

center right

bottom

CSS Images

```
body {
		background-image: url("images/flower.jpg");
		background-repeat: no-repeat;
		background-position: center top;}
```

CSS Images

background:

1: background-color

2: background-image

3: background-repeat

4: background-attachment

5: background-position

```
body {
        background: #ffffff url("images/tulip.gif")
        no-repeat top right;}
```

CSS rollovers

```
a.button {
		height: 36px;
		background-image: "borderimage1.jpg");
		text-indent: -9999px;
		display: inline-block;}
a#add-to-basket {
		width: 174px;
		background-position: 0px 0px;}
a#framing-options {
		width: 210px;
		background-position: -175px 0px;}
a#add-to-basket:hover {
		background-position: 0px -40px;}
a#framing-options:hover {
		background-position: -175px -40px;}
```

```
<a class="button" id="add-to-basket">
Add to basket</a>
<a class="button" id="framing-options">
Framing options</a>
```

CSS3 gradient

```
#gradient {
/* fallback color */
background-color: #66cccc;
/* fallback image */
background-image: url(images/fallback-image.png);
/* Firefox 3.6+ */
background-image: -moz-linear-gradient(#336666,
#66cccc);
/* Safari 4+, Chrome 1+ */
background-image: -webkit-gradient(linear, 0% 0%,
0% 100%, from(#66cccc), to(#336666));
/* Safari 5.1+, Chrome 10+ */
background-image: -webkit-linear-gradient(#336666,
#66cccc);
/* Opera 11.10+ */
background-image: -o-linear-gradient(#336666,
```

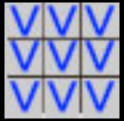
CSS Layout

exercise

CSS3 border-image:

applies an image to the border of any box

background image is sliced it into 9 pieces



1: The URL of the image

2: Where to slice the image

3: What to do with the straight edges;

the possible values are:

stretch stretches the image repeat repeats the image round like repeat but if the tiles do not fit exactly, scales the tile image

Ex.css_border_image.html

CSS3 box-shadow:

Drops shadow around a box

inset – crates inner shadow

horizontal offset - Negative values position the shadow to the left

vertical offset - Negative values position the shadow to the top

blur distance - If omitted, the shadow is a solid line like a border

spread of shadow -positive value will cause the shadow to expand in all directions, and a negative value will make it contract

Ex. css_box_shadow.html

CSS3 border-radius:

Creates rounded corners on any box

The value indicates the size of the radius in pixels

border-top-right-radius

border-bottom-right-radius

border-bottom-left-radius

border-top-left-radius

```
p {
    border: 5px solid #cccccc;
    padding: 20px;
    width: 275px;
    text-align: center;
    border-radius: 10px;
    -moz-border-radius: 10px;
    -webkit-border-radius: 10px;}
```

Ex. css_border_radius.html

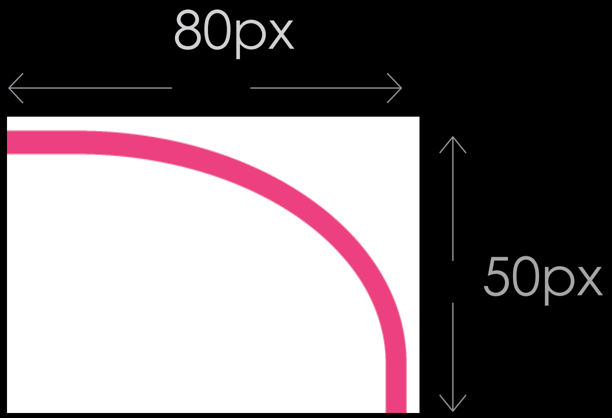
CSS3 elliptical shapes

more complex elliptical shapes

specify different distances for the horizontal and the vertical parts of the rounded corners

border-radius: 80px 50px;

ex.css_elliptical_shapes.html



CSS list properties

list-style-type: - controls the shape/style of a bullet <ol> <ul> <li>

<ul>

none

disc

circle

square

Ex.css_list_tables.html

CSS list properties

<ol>

decimal

1 2 3

decimal-leading-zero

01 02 03

lower-alpha

a b c

upper-alpha

A B C

lower-roman

i. ii. iii.

upper-roman

I II III

CSS list properties

list-style-image: - specifies an image to act as a bullet point

```
ul {
        list-style-image: url("images/star.png");}
li {
        margin: 10px 0px 0px 0px;}
```

CSS list properties

list-style-position: - specifies if the marker appears on the inside or the outside of the box containing the main points

ouside inside

```
ul {
        width: 150px;}
li {
        margin: 10px;}
ul.illuminations {
        list-style-position: outside;}
ul.season {
        list-style-position: inside;}
```

CSS list properties

list-style: - addresses all the above (shorthand)

```
ul {
        list-style: inside circle;
        width: 300px;}
li {
        margin: 10px 0px 0px 0px;}
```

CSS table properties

width:

padding:

text-transform: converts the content of the table headers to uppercase

letter-spacing, font-size: adds additional styling to the content of the table headers

border-top, border-bottom: sets borders above and below the table headers

text-align:

background-color:

:hover - highlights a table row when a user's mouse goes over it

CSS table styling tips

padding to add space between cells

distinguish headings (<th> or bold, etc.)

shade alternate rows for legibility

align numerals

CSS table styling tips

empty-cells: specifies if empty table cell borders are shown

show

hide

inherit - the nested table cells will obey the rules of the container

```
td {
		border: 1px solid #0088dd;
		padding: 15px;}
table.one {
		empty-cells: show;}
table.two {
		empty-cells: hide;}
```

CSS table styling tips

border-spacing: controls the distance between adjacent cells

border-collapse: collapses adjacent borders of inside cells to prevent the width of lines twice that of the outside edges

collapse

separate

Ex.css_list_tables.html