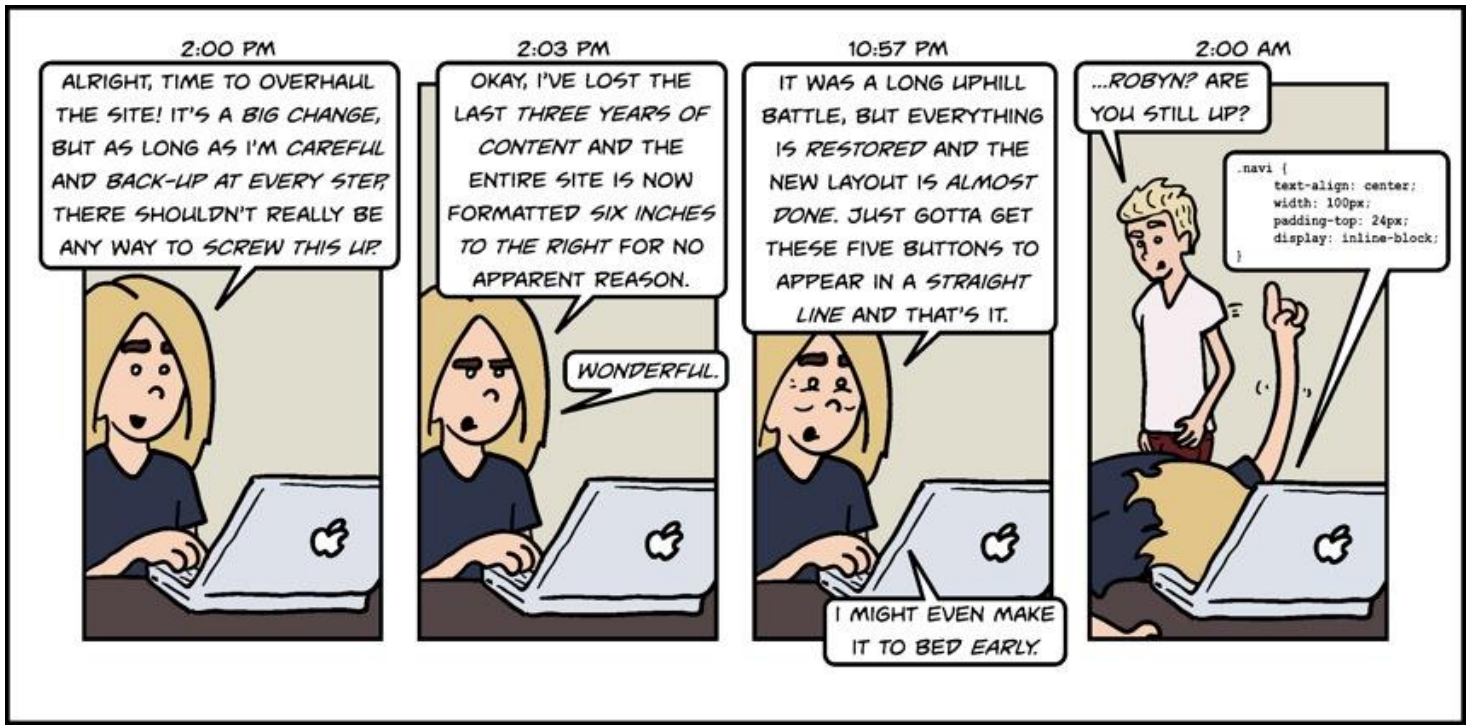
CSE 154



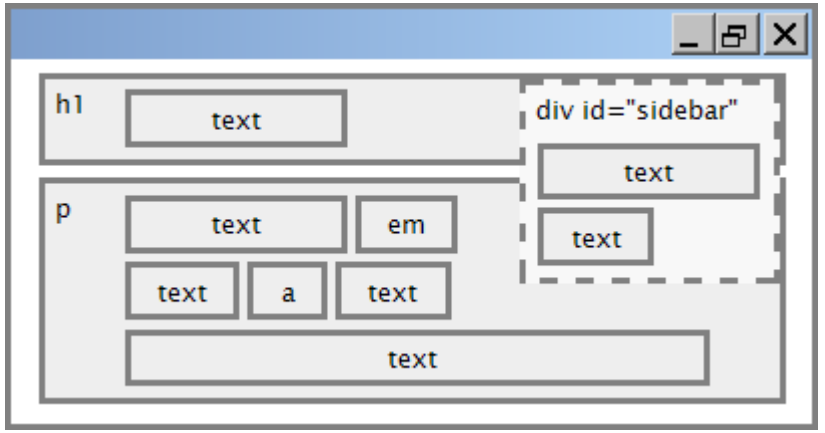
LECTURE 4: FLOATING AND POSITIONING

The CSS float property

property	description
float	side to hover on; can be left, right, or none (default)

- a ***floating*** element is removed from normal document flow

- underlying text wraps around it as necessary



Float example

```
<img src="images/koala.jpg" alt="Koala" class="headericon" />

Lorem ipsum dolor sit amet, consectetur adipiscing elit....
```
HTML

```
img.headericon {
    float: left;
}
```
CSS

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam scelerisque purus ut dui mollis, sed malesuada leo pretium. Morbi bibendum mi at lacus rutrum convallis. Duis id eros dolor. In id eros blandit lectus viverra facilisis at commodo velit. Cras pretium nunc id nisl elementum, at interdum odio blandit. Donec luctus rutrum iaculis. Praesent luctus ante et cursus suscipit. Nullam congue egestas lorem nec luctus. Donec tincidunt tortor mi, nec ultricies orci bibendum a. Aliquam viverra metus nec ligula varius feugiat. In lacinia ligula accumsan tortor porttitor ornare. Donec interdum mattis purus sit amet ultrices.

output

Floating content and width

I am not floating, no width set

I am floating right, no width set

I am floating right, no width set, but my text is very long so this paragraph doesn't really seem like it's floating at all, darn

I am not floating, 45% width

I am floating right, 45% width

- often floating elements should have a `width` property value
  - if no `width` is specified, other content may be unable to wrap around the floating element

The clear property

```
p { background-color: fuchsia; }
h2 { clear: right; background-color: cyan; }
```
CSS

XKCD a webcomic of romance, sarcasm, math, and language...

My XKCD Fan Site

property	description
clear	disallows floating elements from overlapping this element; can be left, right, both, or none (default)

Common error: container too short

```
<p><img src="images/xkcd.png" alt="the man in the hat" />
  XKCD a webcomic of romance, sarcasm,
    math, and language...</p>
```
HTML

```
p { border: 2px dashed black; }
img { float: right; }
```
CSS

XKCD a webcomic of romance, sarcasm, math, and language...

- We want the p containing the image to extend downward so that its border encloses the entire image

The overflow property

```
p { border: 2px dashed black; overflow: hidden; }
```
CSS

XKCD a webcomic of romance, sarcasm, math, and language...

property	description
overflow	specifies what to do if an element's content is too large; can be auto, visible, hidden, or scroll

The position property

```
div#ad {
  position: fixed;
  right: 10%;
  top: 45%;
}
```

Here I am!

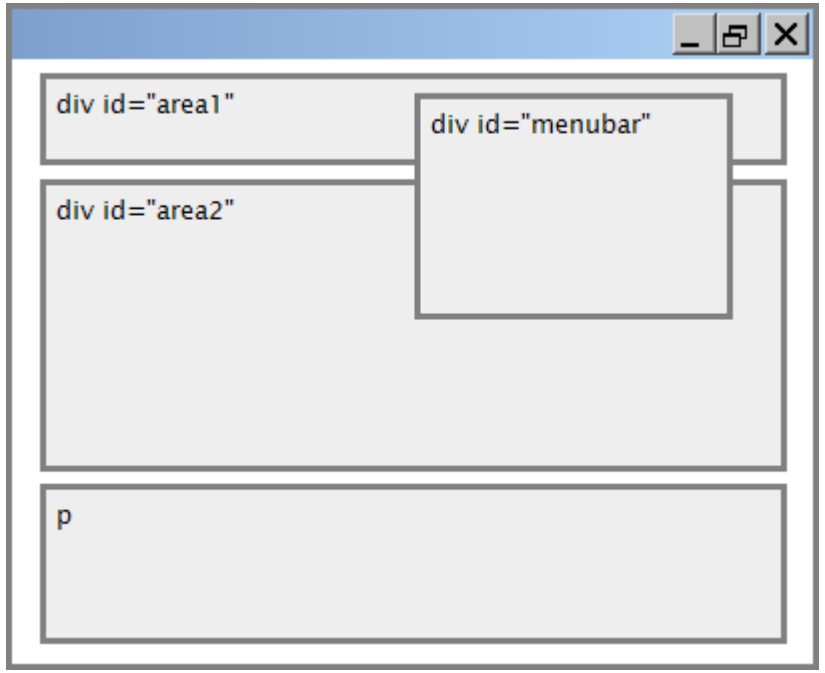
CSS

property	value	description
position	static	default position
	relative	offset from its normal static position
	absolute	a fixed position within its containing element
	fixed	a fixed position within the browser window
top, bottom, left, right	positions of box's corners	

Absolute positioning

```
#menubar {
      position: absolute;
      left: 400px;
      top: 50px;
}
```
CSS

- removed from normal flow (like floating ones)
- positioned relative to the block element containing them (assuming that block also uses `absolute` or `relative` positioning)
- actual position determined by `top`, `bottom`, `left`, `right` values
- should often specify a `width` property as well

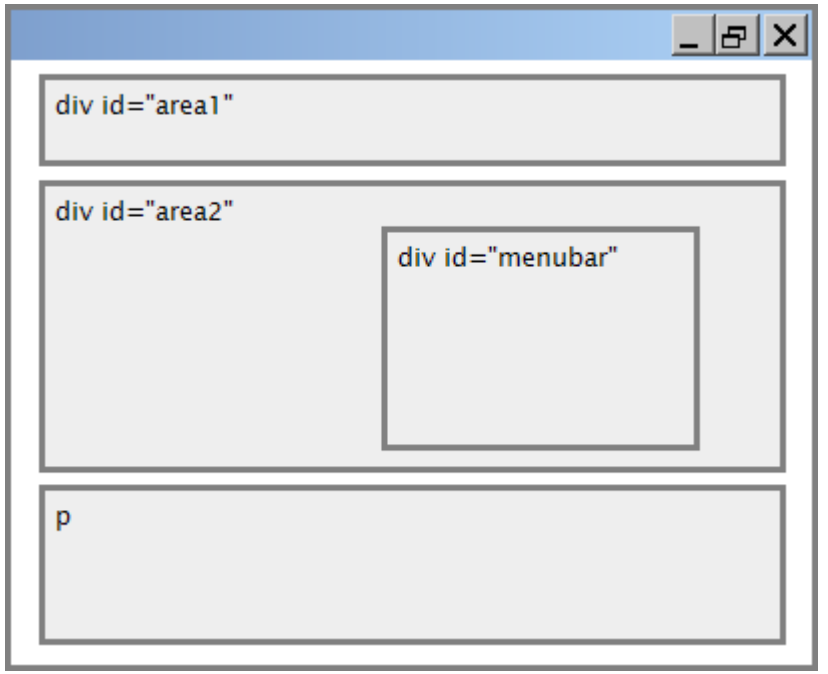


Relative positioning

```
#area2 { position: relative; }                                CSS
```

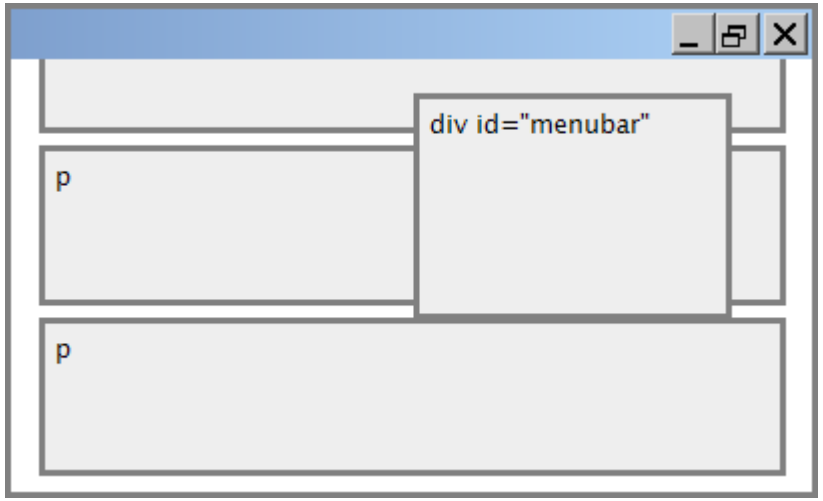
- absolute-positioned elements are normally positioned at an offset from the corner of the overall web page

- to instead cause the absolute element to position itself relative to some other element's corner, wrap the `absolute` element in an element whose `position` is `relative`



Fixed positioning

- removed from normal flow (like floating ones)
- positioned relative to the browser window
  - even when the user scrolls the window, element will remain in the same place



Alignment vs. float vs. position

1. if possible, lay out an element by *aligning* its content
    - horizontal alignment: `text-align`
        - set this on a block element; it aligns the content within it (not the block element itself)
    - vertical alignment: `vertical-align`
        - set this on an inline element, and it aligns it vertically within its containing element
2. if alignment won't work, try *floating* the element
3. if floating won't work, try *positioning* the element
    - absolute/fixed positioning are a last resort and should not be overused

Details about inline boxes

- size properties (`width`, `height`, `min-width`, etc.) are ignored for inline boxes
- `margin-top` and `margin-bottom` are ignored, but `margin-left` and `margin-right` are not
- the containing block box's `text-align` property controls horizontal position of inline boxes within it
    - text-align does not align block boxes within the page
- each inline box's `vertical-align` property aligns it vertically within its block box

The display property

```
h2 { display: inline; background-color: yellow; }
```
CSS

This is a heading **This is another heading**

output

property	description
display	sets the type of CSS box model an element is displayed with

- values: `none`, `inline`, `block`, `run-in`, `compact`, ...
- use sparingly, because it can radically alter the page layout

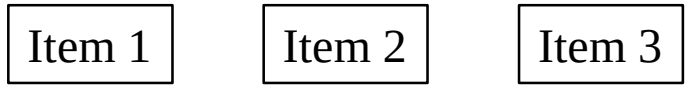
Displaying block elements as inline

```
<ul id="topmenu">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
```

HTML

```
#topmenu li {
  display: inline;
  border: 2px solid gray;
  margin-right: 1em;
}
```

CSS



output

- lists and other block elements can be displayed inline
  - flow left-to-right on same line
  - width is determined by content (block elements are 100% of page width)