Introduction to Web Programming

Lecture 5: Floating and Positioning

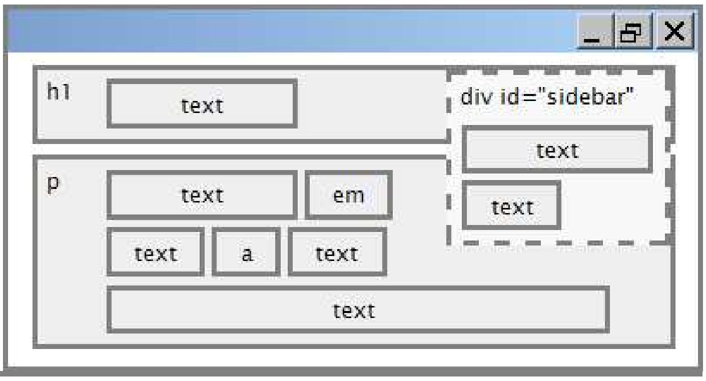
5.1: Floating Elements

- **5.1: Floating Elements**
- 5.2: Sizing and Positioning

The CSS `float` property (reference)

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/borat.jpg" alt="Borat" class="headericon" />
Borat Sagdiyev (born July 30, 1972) is a ...
```

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

Borat Sagdiyev (born July 30, 1972) is a fictional Kazakhstani journalist played by British-Jewish comedian Sacha Baron Cohen. He is the main character portrayed in the controversial and successful film Borat: Cultural Learnings of America for Make Benefit Glorious Nation of Kazakhstan ...

Float vs. alignment

none 1 before

none 2 before

left #1 left #2 none 1 after right #2 right #1

none 2 after

- using Firebug, toggle the above `div`s from being aligned to floated...

Floating content and width

I am not floating, no width set

I am floating right, no width set

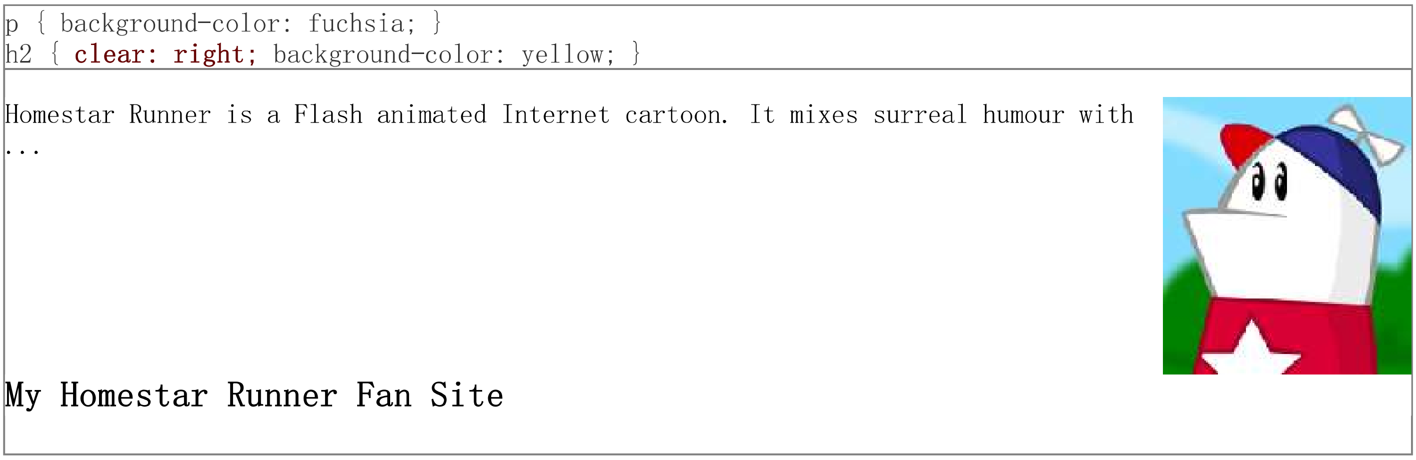
I am floating right, no width set, but my text is very long so this paragraph doesn't really seem like it's floating at all, darn

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: yellow; }
```



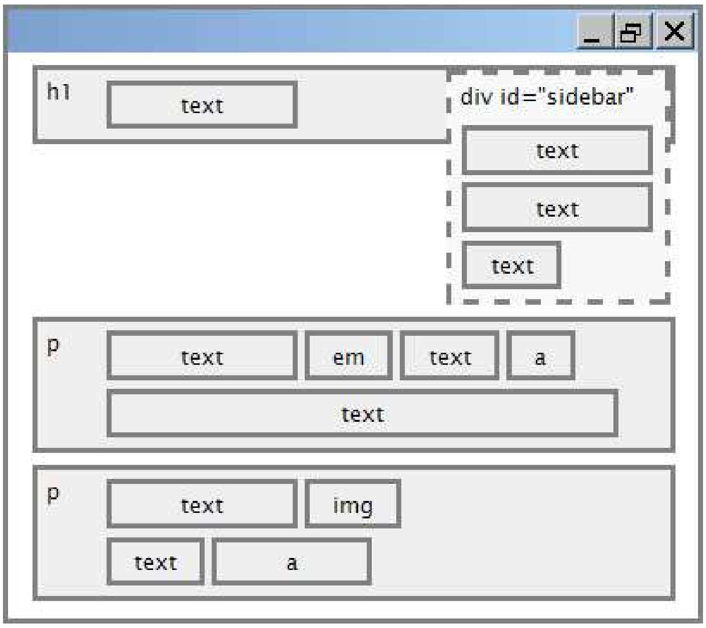
Homestar Runner is a Flash animated Internet cartoon. It mixes surreal humour with ...

My Homestar Runner Fan Site

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

Clear diagram

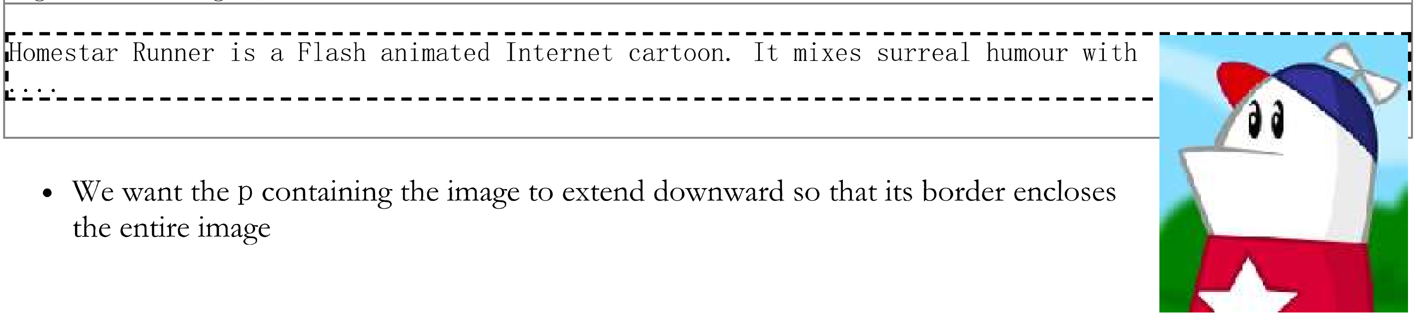
```
div#sidebar { float: right; }
p { clear: right; }
```



Common error: container too short

```
<p><img src="images/homestar_runner.png" alt="homestar runner" />
  Homestar Runner is a Flash animated Internet cartoon.
  It mixes surreal humour with ....</p>
```

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

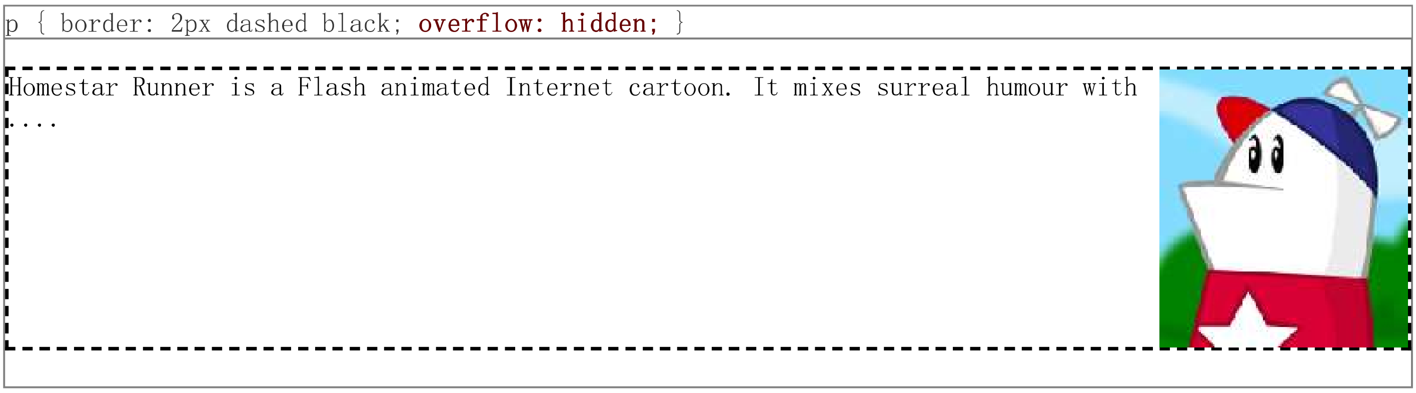


Homestar Runner is a Flash animated Internet cartoon. It mixes surreal humour with ....

- 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; }
```



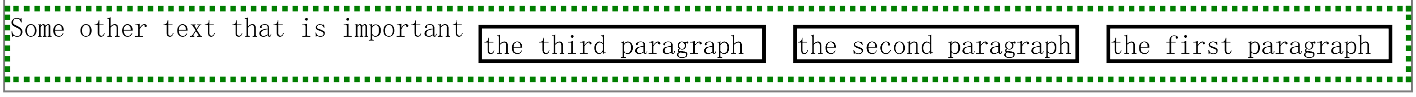
Homestar Runner is a Flash animated Internet cartoon. It mixes surreal humour with ....

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

Multi-column layouts

```
<div>
  <p>the first paragraph</p>
  <p>the second paragraph</p>
  <p>the third paragraph</p>
  Some other text that is important
</div>
```

```
p { float: right; width: 20%; margin: 0.5em;
    border: 2px solid black; }
div { border: 3px dotted green; overflow: hidden; }
```



Some other text that is important

the third paragraph

the second paragraph

the first paragraph

5.2: Sizing and Positioning

- 5.1: Floating Elements
- **5.2: Sizing and Positioning**

The position property (examples)

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

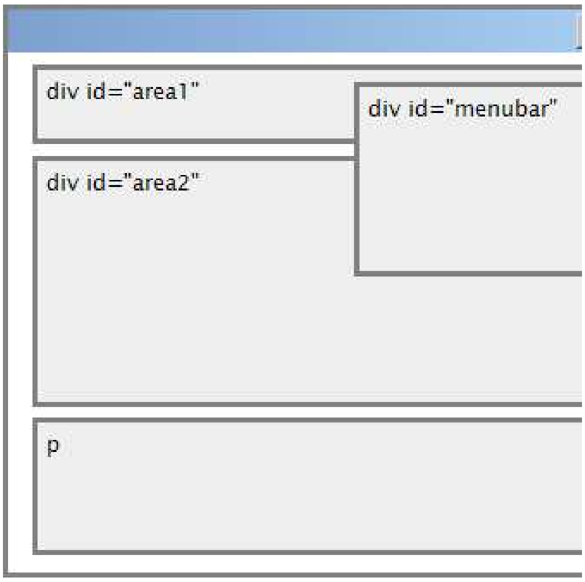
Here I am!

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;
}
```

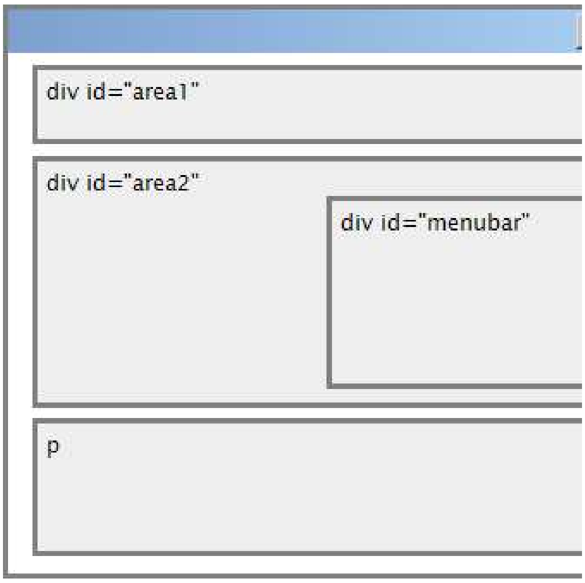
- removed from normal flow (like floating ones)
- positioned relative to the block element containing them (assuming that block also uses absolute or relative positioning)
- actual position determined by top, bottom, left, right values
- should often specify a width property as well



Relative positioning

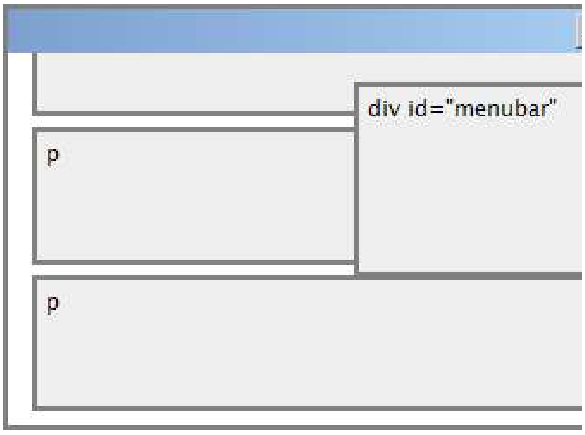
```
#area2 { position: relative; }
```

- 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



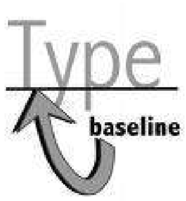
Alignment vs. float vs. position

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

The `vertical-align` property

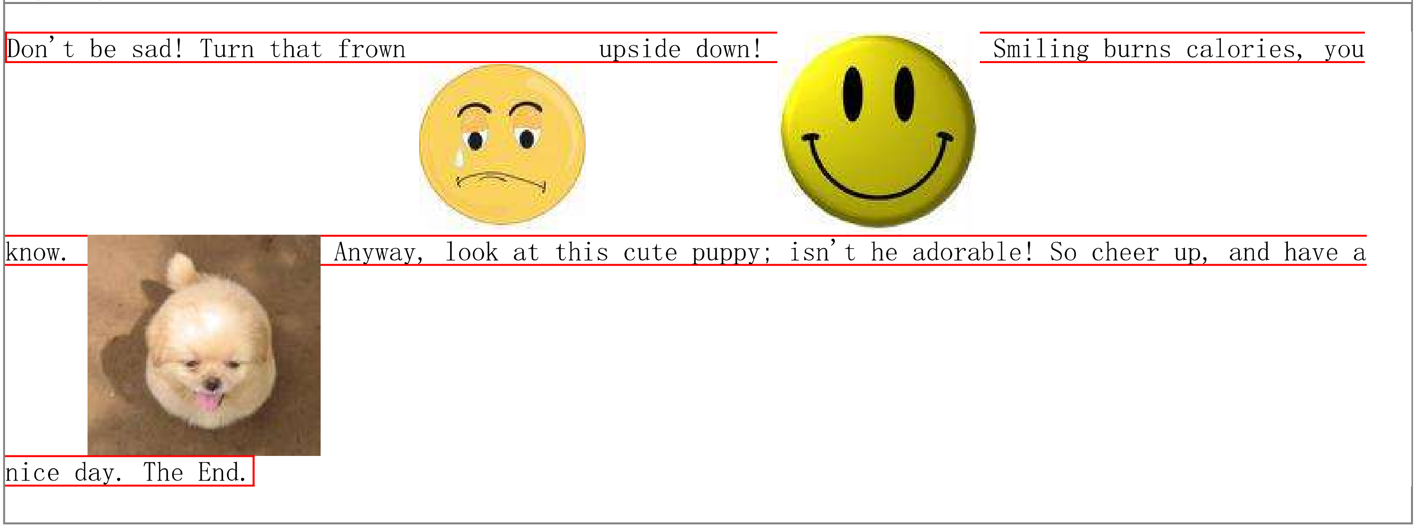
property	description
`vertical-align`	specifies where an inline element should be aligned vertically, with respect to other content on the same line within its block element's box

- can be `top`, `middle`, `bottom`, `baseline` (default), `sub`, `super`, `text-top`, `text-bottom`, or a length value or %
  - `baseline` means aligned with bottom of non-hanging letters



`vertical-align` example

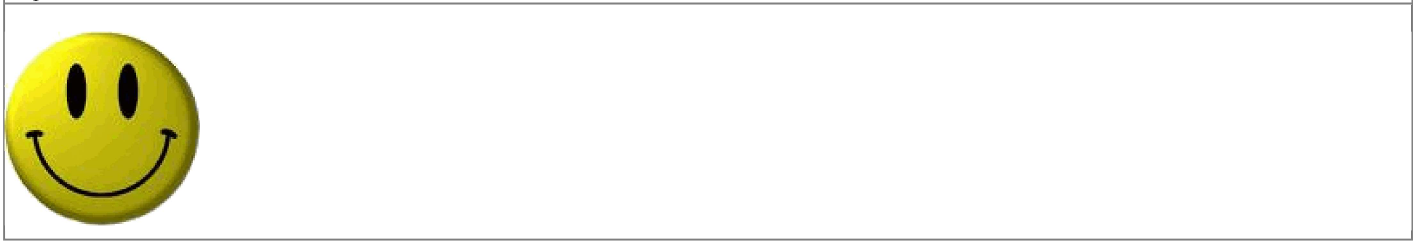
```
<p style="background-color: yellow;">
<span style="vertical-align: top; border: 1px solid red;">
Don't be sad!  Turn that frown
<img src="images/sad.jpg" alt="sad" /> upside down!
<img style="vertical-align: bottom" src="images/smiley.jpg" alt="smile" />
Smiling burns calories, you know.
<img style="vertical-align: middle" src="images/puppy.jpg" alt="puppy" />
Anyway, look at this cute puppy; isn't he adorable!  So cheer up,
and have a nice day.  The End.
</span></p>
```



Don't be sad! Turn that frown upside down! Smiling burns calories, you know. Anyway, look at this cute puppy; isn't he adorable! So cheer up, and have a nice day. The End.

Common bug: space under image

```
<p style="background-color: red; padding: 0px; margin: 0px">
<img src="images/smiley.png" alt="smile" />
</p>
```



- red space under the image, despite `padding` and `margin` of `0`
- this is because the image is vertically aligned to the baseline of the paragraph (not the same as the bottom)
- setting `vertical-align` to `bottom` fixes the problem (so does setting `line-height` to `0px`)

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; }
```

```
This is a heading This is another heading
```

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>
```

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

Item 1 Item 2 Item 3

- 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)

The `visibility` property

```
p.secret {
  visibility: hidden;
}
```

property	description
`visibility`	sets whether an element should be shown onscreen; can be `visible` (default) or `hidden`

- `hidden` elements will still take up space onscreen, but will not be shown
  - to make it not take up any space, set `display` to none instead
- can be used to show/hide dynamic HTML content on the page in response to events

The opacity property

```
body     { background-image: url("images/marty-mcfly.jpg"); background-repeat: repeat; }
p        { background-color: yellow; margin: 0; padding: 0.25em; }
p.mcfly1 { opacity: 0.75; }
p.mcfly2 { opacity: 0.50; }
p.mcfly3 { opacity: 0.25; }
```

```
Marty McFly in 1985
Marty McFly in 1955 fading away, stage 1
Marty McFly in 1955 fading away, stage 2
Marty McFly in 1955 fading away, stage 3
```

property	description
opacity	how not-transparent the element is; value ranges from 1.0 (opaque) to 0.0 (transparent)