

# Page Layout with CSS

고급 웹 개발응용 실습  
제주대학교 컴퓨터공학부

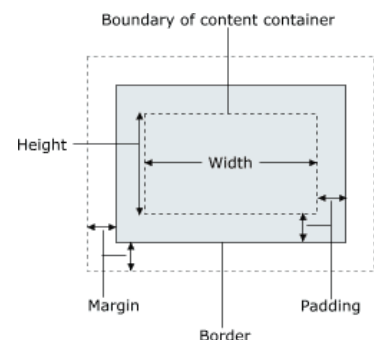
본 내용은 University of Washington, CSE 190 M (Web Programming), Spring 2007 수업에서 사용된 자료입니다.

Except where otherwise noted, the contents of this presentation are © Copyright 2007 Marty Stepp and are licensed under the Creative Commons Attribution 2.5 License.



## CSS Box Model

- for layout purposes, every element is composed of:
  - the actual element's content
  - a border around the element
  - padding between the content and the border (*inside*)
  - a margin between the border and other content (*outside*)
- width = content width + L/R padding + L/R border + L/R margin  
height = content height + T/B padding + T/B border + T/B margin
  - IE6 doesn't do this right, and sucks



## CSS properties for borders

```
h2 { border: 5px solid red; }
```

**This is a heading.**

- border: all properties of border on all 4 sides
- a border is specified as three items:
  - its thickness (specified in px, pt, em, %, or one of the following general widths: thin, medium, thick)
  - its style (one of none, hidden, dotted, dashed, double, groove, inset, outset, ridge, solid)
  - its color (specified as seen previously for text and background colors)

## More border properties

- border-color, border-width, border-style: specific properties of border on all 4 sides
- border-bottom, border-left, border-right, border-top: all properties of border on a particular side
- border-bottom-color, border-bottom-style, border-bottom-width, border-left-color, border-left-style, border-left-width, border-right-color, border-right-style, border-right-width, border-top-color, border-top-style, border-top-width: specific properties of border on a particular side
- Complete list of border properties

## Border example 2

```
h2 {
  border-left: thick dotted #CC0088;
  border-bottom-color: rgb(0, 128, 128);
  border-bottom-style: double;
}
```

**This is a heading.**

- each side's border properties can be set individually
- if you omit some properties, they receive default values (e.g. border-bottom-width above)

---

## CSS properties for padding

---

- padding: padding on all 4 sides
- padding-bottom: padding on bottom side only
- padding-left: padding on left side only
- padding-right: padding on right side only
- padding-top: padding on top side only
- Complete list of padding properties

---

### Padding example 1

---

```
p { padding: 20px; border: 3px solid black; }
h2 { padding: 0px; background-color: yellow; }
```

This is the first paragraph

This is the second paragraph

**This is a heading**

Here I am!

- notice that padding shares the background color of the element

---

### Padding example 2

---

```
p { padding-left: 200px; padding-top: 30px;
  background-color: fuchsia; }
```

This is the first paragraph

This is the second paragraph

- each side's padding can be set individually

---

## CSS properties for margins

---

- margin: margin on all 4 sides
- margin-bottom: margin on bottom side only
- margin-left: margin on left side only
- margin-right: margin on right side only
- margin-top: margin on top side only
- Complete list of margin properties

---

### Margin example 1

---

```
p {
  margin: 70px;
  background-color: fuchsia;
}
```

This is the first paragraph

This is the second paragraph

- notice that margins are always transparent  
(they don't contain the element's background color, etc.)

---

## Margin example 2

---

```
p {  
  margin-left: 200px;  
  background-color: fuchsia;  
}
```

This is the first paragraph

This is the second paragraph

- 
- each side's margin can be set individually
- 

## Recall: properties for dimensions

---

```
p { width: 400px; background-color: yellow; }  
h2 { width: 50%; background-color: aqua; }
```

Here I am!

This paragraph uses the first style above.

**This heading uses the second style above.**

- 
- width, height: how wide or tall to make this element
  - max-width, max-height, min-width, min-height: the maximum or minimum size of this element in the given dimension
  - all of these apply only to block elements; ignored for inline elements
- 

## Centering a block element: auto margins

---

```
p { width: 500px; margin-left: auto; margin-right: auto; }
```

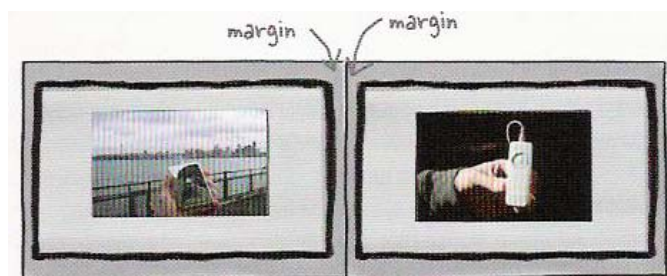
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do  
eiusmod tempor incididunt ut labore et dolore magna aliqua.

- 
- only works if width is set  
(otherwise, element occupies entire width of page)
  - to center inline elements within a block element,  
use `text-align: center;` instead
- 

## Top/bottom margin collapse

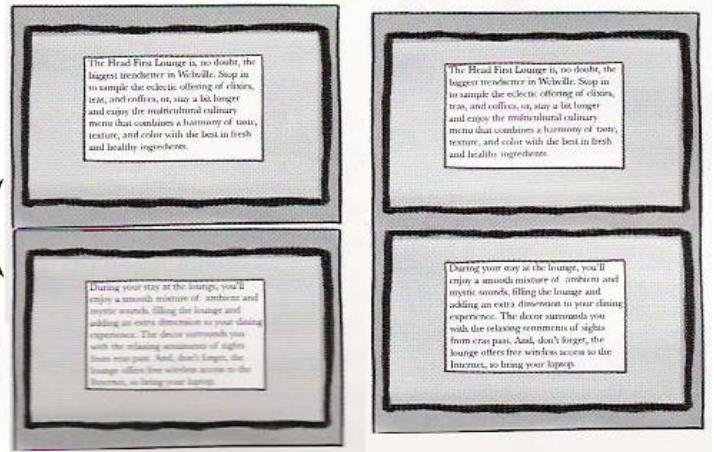
---

- when two block elements appear  
on top of each other, their  
margins are collapsed
- their shared margin is the larger  
of the two individual margins



When the browser places two block elements on top of each other, it collapses their margins.

Their shared margin is the size of the larger of the two margins.



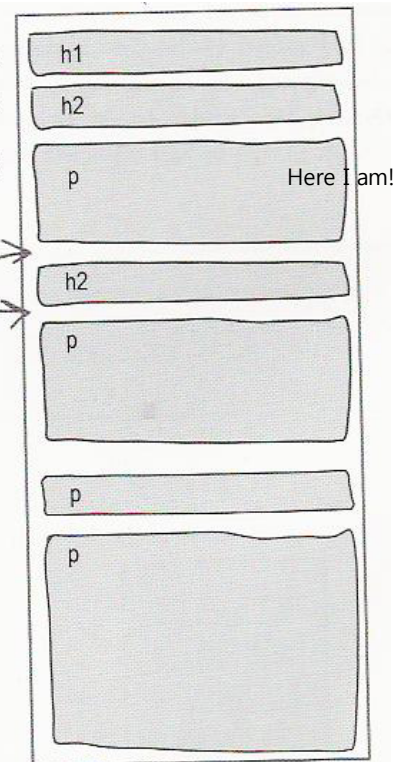
## Document flow - block elements

```
<html>
  <head>...</head>
  <body>
    <h1>...</h1>
    <h2>...</h2>
    <p>...</p>
    <h2>...</h2>
    <p>...</p>
    <p>...</p>
    <p>...</p>
  </body>
</html>
```

Each block element is taken in the order it appears in the markup, and placed on the page.

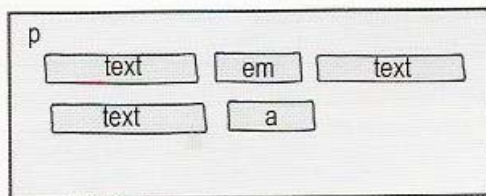
Each new block element causes a linebreak.

Notice that elements take up the full width of the page.

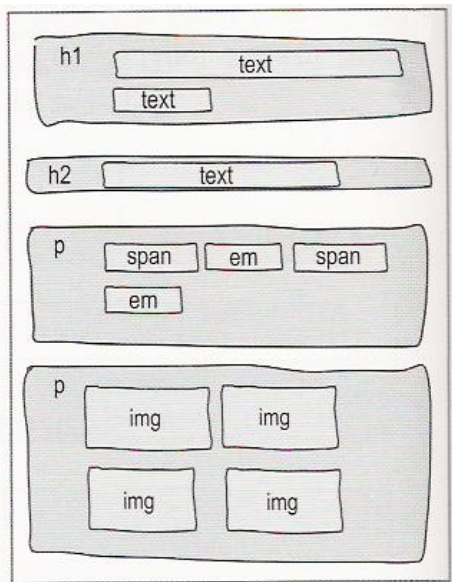


## Document flow - inline elements

```
<p>
Join us <em>any evening</em> for
these and all our other wonderful <a
href="beverages/elixir.html" title="Head
First Lounge Elixirs">elixirs</a>.
</p>
```



## Document flow - a larger example



Here I am!

## The CSS float property (reference)

```
img.floatright { float: right; width: 130px; }
```

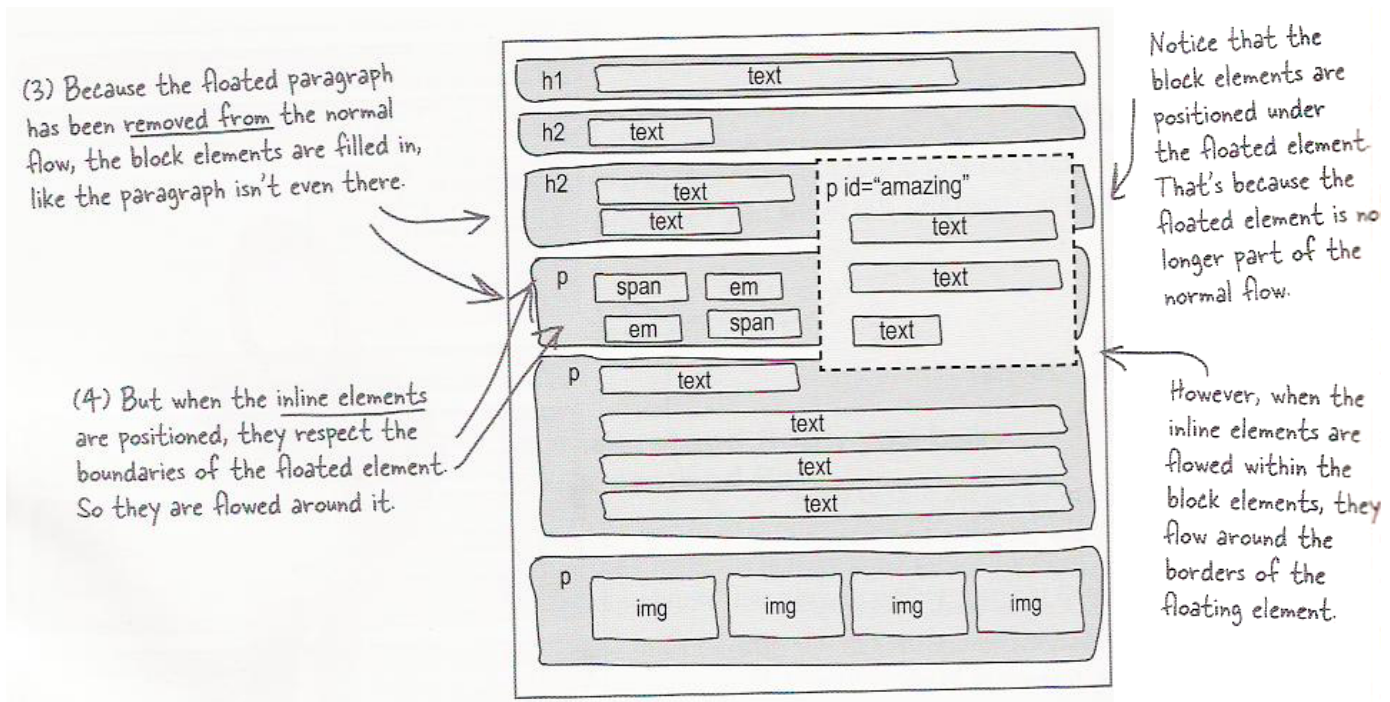
Borat Sagdiyev (born July 30, 1972) is a fictional Kazakhstani journalist played by British-Jewish comedian Sacha Baron Cohen. He is the main character portrayed in the controversial and successful film *Borat: Cultural Learnings of America for Make Benefit Glorious Nation of Kazakhstan*. Borat ...



- float can be left, right, or none (default)
- floating elements are removed from normal document flow
- underlying text wraps around floating element as necessary

## Floating elements diagram





## Common float bug: missing width

Here I am!

I am not floating, no width

I am floating right, no width

I am not floating, 45% width

I am floating right, 45% width

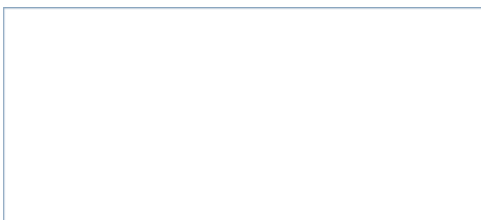
- often floating block elements must have a width property value
  - if no width is specified, the floating element may occupy 100% of the page width, so no content can wrap around it

## Practice Problem

**I**t was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

Result:

Interaction Pane:



Show Answer

Evaluate

## 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 references to 1980s and 1990s pop culture, notably video games, classic television and popular music.



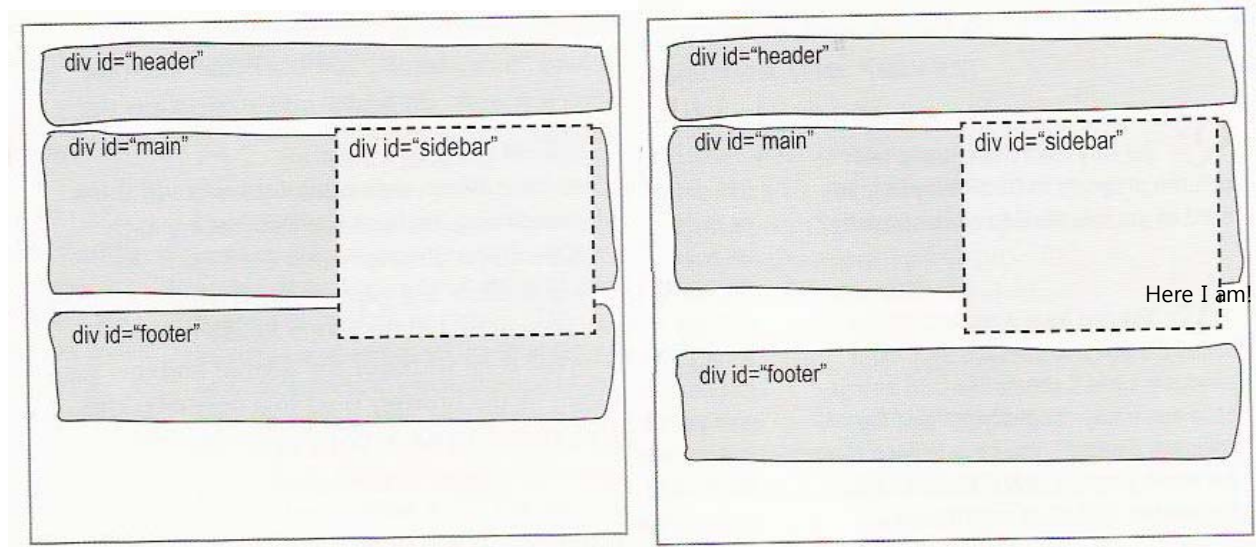


## My Homestar Runner Fan Site

- disallows any floating elements from overlapping this element
- clear can be left, right, both, or none (default)

## Clear diagram

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



## Practice problem (HTML) (CSS)

### W3Schools.com

"Never increase, beyond what is necessary, the number of entities required to explain anything." William of Ockham (1285-1349)

### Free Web Building Tutorials

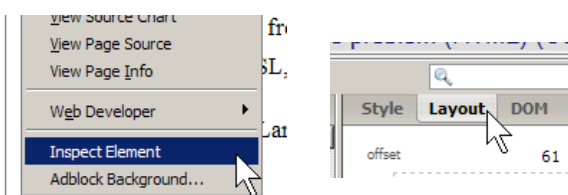
At W3Schools you will find all the Web-building tutorials you need, from basic HTML and XHTML to advanced XML, XSL, Multimedia and WAP.

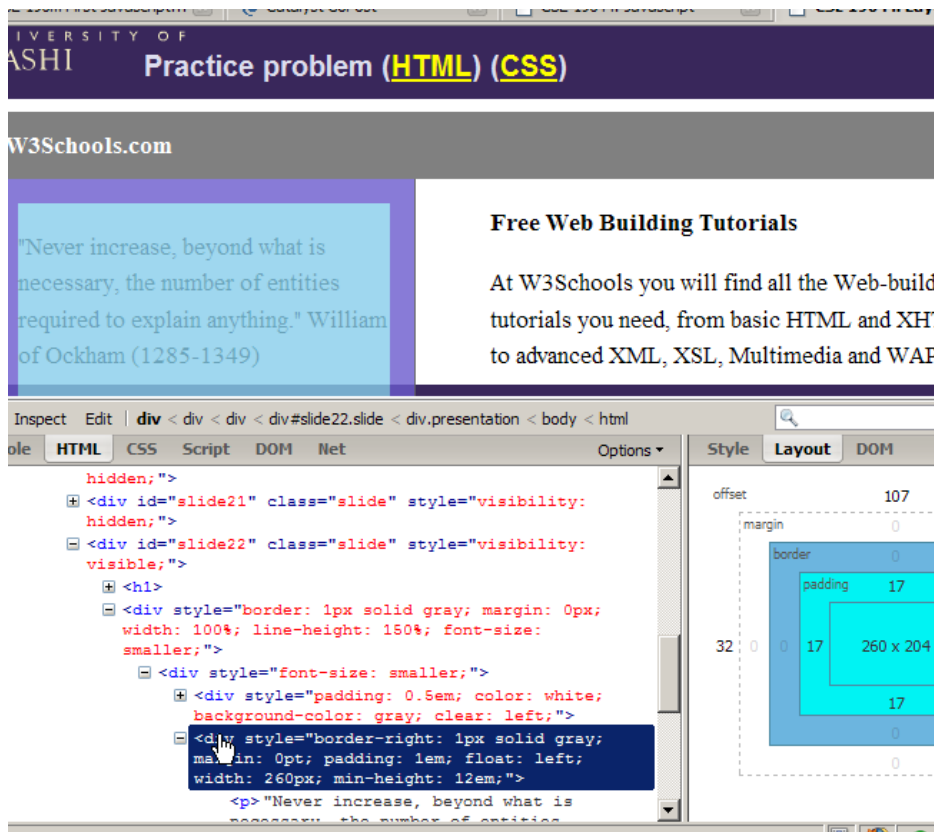
W3Schools - The Largest Web Developers Site On The Net!

Copyright 1999-2005 by Refsnes Data.

- which div elements should float, and how?
- how would we add a second column on the left?

## Firefox Firebug extension





Here I am!

## ----- Advanced CSS Layout -----

### Advanced CSS Layout

Reading: Sebesta Ch. 6 sections 6.1 - 6.4

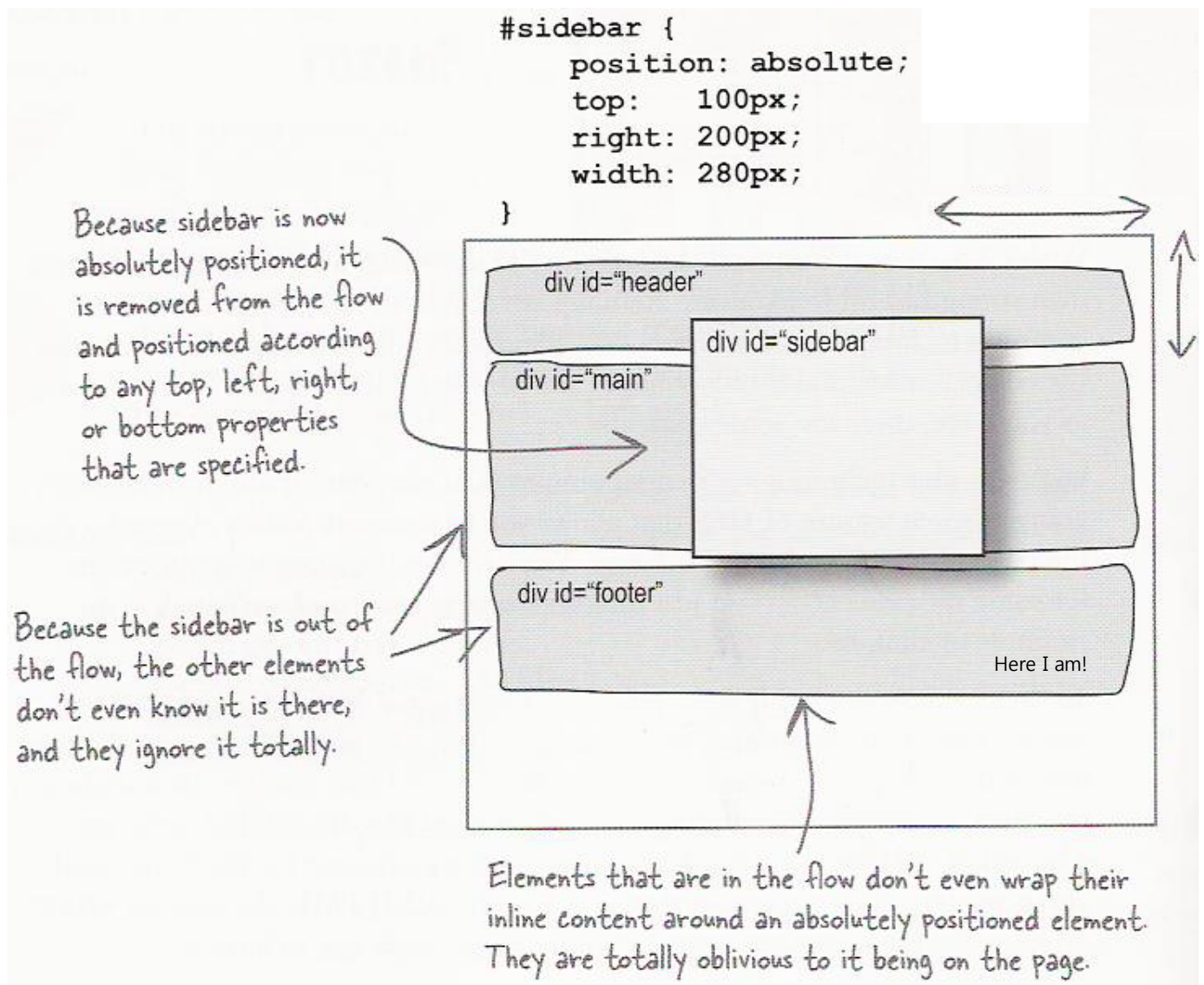
### The position property (examples)

```
div#rightside {
  position: fixed;
  right: 10%;
  top: 36%;
}
```

- static : default position
- relative : offset from its normal static position
- absolute : at a fixed position *within its containing element*
- fixed : at a fixed position *within the browser window*
  - top, bottom, left, right properties specify positions of box's corners

### Absolute positioning

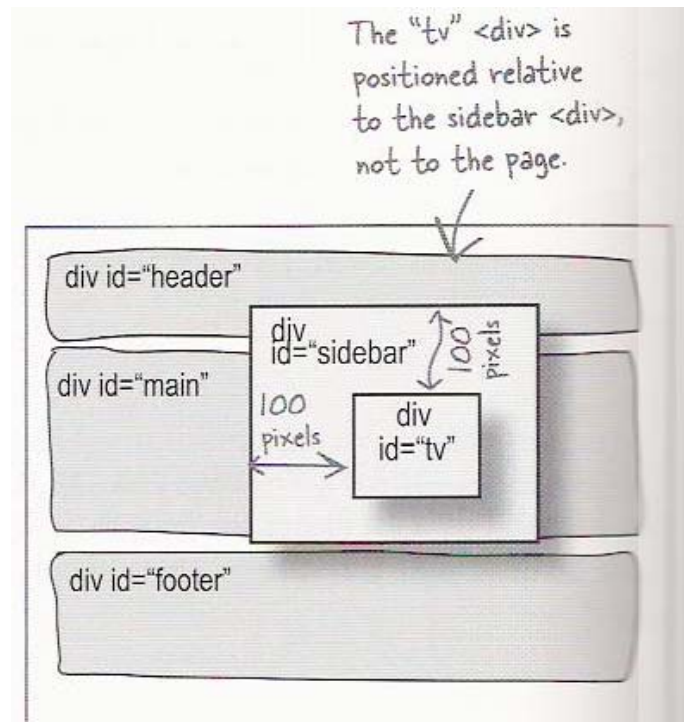




- removed from normal flow (like floating ones)
- positioned relative to the block element containing them (assuming that block also uses absolute or relative positioning)
- actual position determined by top, bottom, left, right values
- should often specify a width property as well

## Absolute positioning details

- positioned relative to the block element containing them
- to position many elements absolutely but close to their normal default position, wrap the absolute elements in a relative element

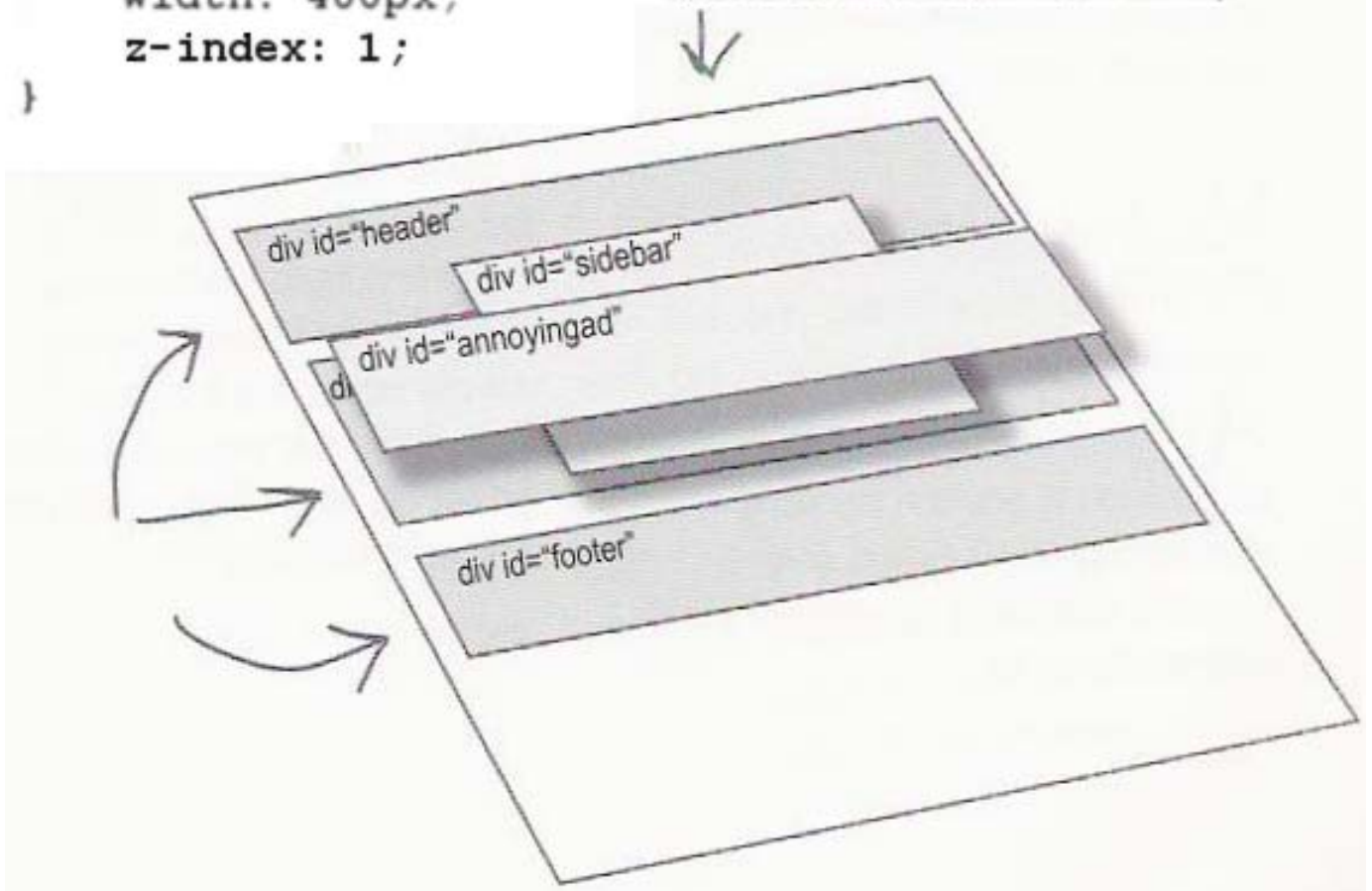


## The z-index property

Here I am!

```
#annoyingad {
  position: absolute;
  top: 150px;
  left: 100px;
  width: 400px;
  z-index: 1;
}
```

The sidebar and annoyingad <div>s are layered on the page, with the annoyingad having a greater z-index than the sidebar, so it's on top.



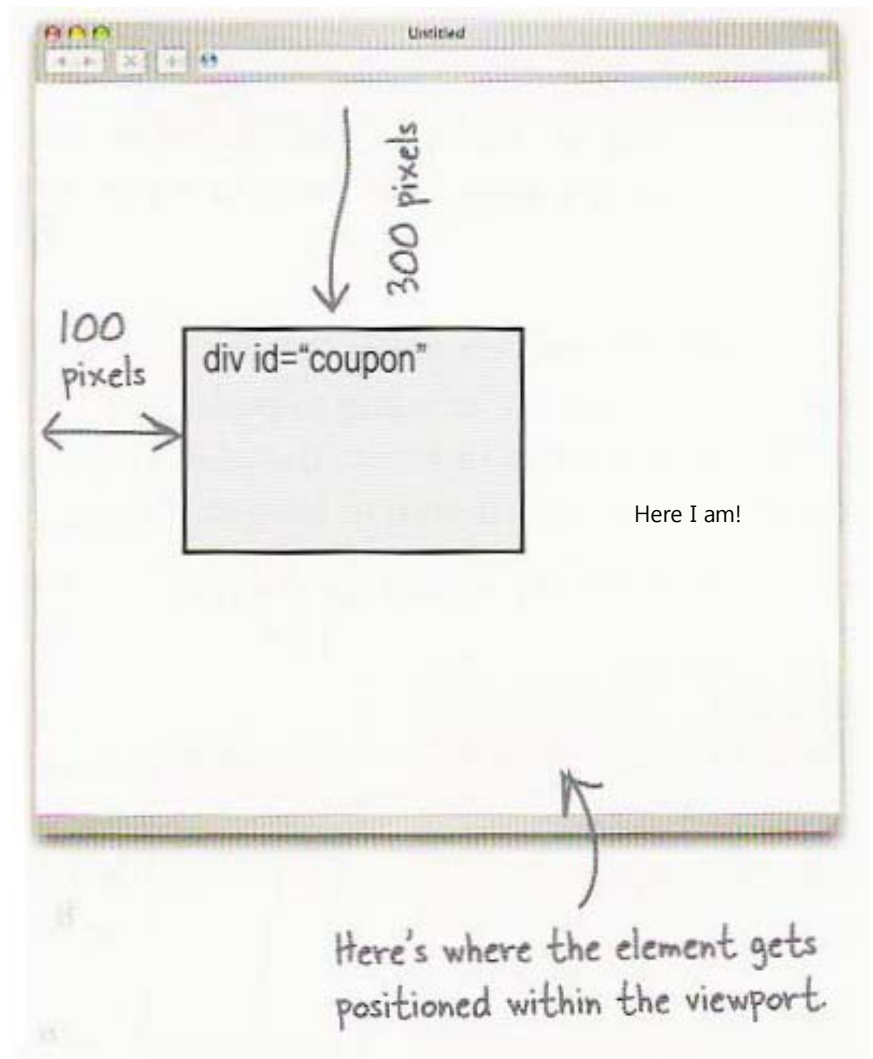
- sets which absolute positioned element will appear on top of another that occupies the same space

- higher z-index wins
- can be auto (default) or a number
- can be adjusted in DOM:  
object.style.zIndex = "value";

---

## Fixed positioning

```
#coupon {  
  position: fixed;  
  top: 300px;  
  left: 100px;  
}
```



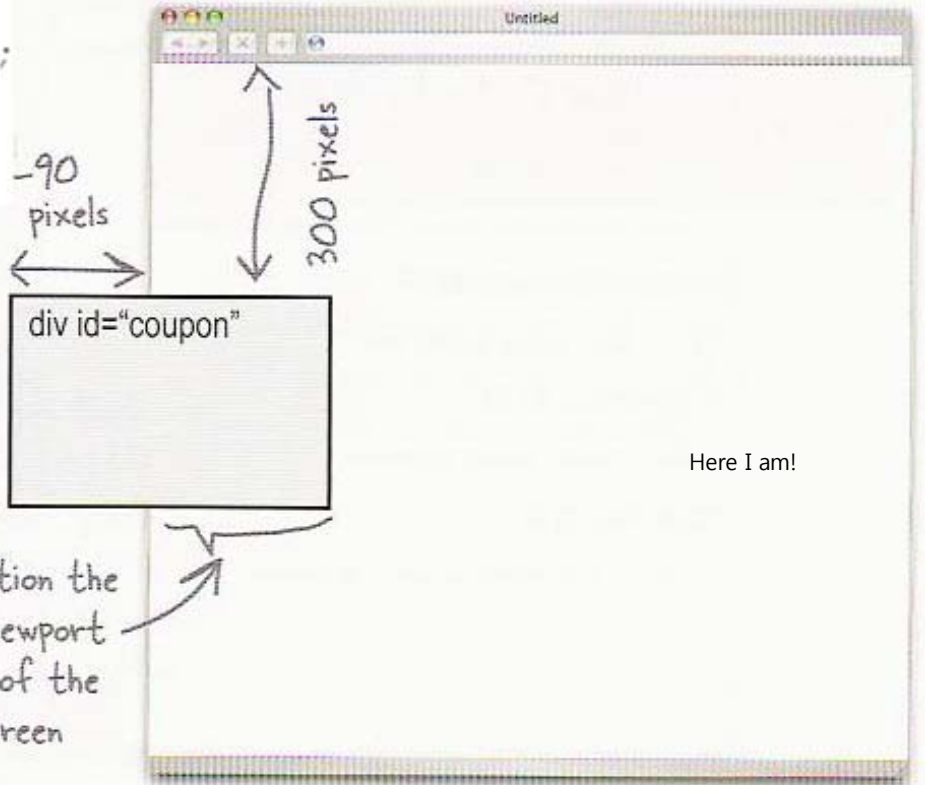
- removed from normal flow (like floating ones)
- positioned relative to the browser window

---

## Negative corners

By specifying `-90 pixels`, we're telling the browser to position the image 90 pixels to the left of the edge of the viewport.

```
#coupon {
  position: fixed;
  top: 300px;
  left: -90px;
}
```



The browser will gladly position the image to the left of the viewport for you, and only the part of the image that is still on the screen will be viewable.

- left, right, top, or bottom value can be negative to create an element that sits outside the visible browser window

## 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 vertical-align property

- 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
- in DOM:
 

```
object.style.verticalAlign = "value";
```



## vertical-align example

```
<p style="background-color: yellow;">
<span style="vertical-align: top; border: 1px solid red;">
Don't be sad! Turn that frown
 upside down!

Smiling burns calories, you know.
```



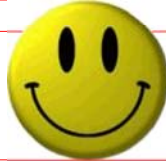
```

Anyway, look at this cute puppy; isn't he adorable! So cheer up,
and have a nice day. The End.
</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">

</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) Here I am!
- setting vertical-align to bottom fixes the problem (so does setting line-height to 0px)

## The display property

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

This is a heading This is another heading

- sets the type of CSS box model an element is displayed with
- can be none, inline, block, run-in, compact, ...
- use sparingly, because it can radically alter the page layout

## Practice problem: Smiley color

- Modify this example page ([HTML](#), [CSS](#)) that displays this [smiley face image](#) to have the following layout and behavior:
  - The smiley face is centered within the overall browser window and is half as tall and wide as the page area.
  - Underneath the smiley face are three checkboxes: Red, Green, and Blue. These controls are centered horizontally within the page and placed vertically at the very top of the page. Each checkbox appears on its own line.
  - When checked, each box adds that color to the background color behind the smiley face. When unchecked, that color is removed from the smiley's background color.

## The visibility property

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

- sets whether an element should be shown onscreen
  - the element 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 visible (default) or hidden
- can be used to show/hide dynamic HTML content on the page in response to events