CSS, part 2

CS147L Lecture 3

Mike Krieger

Intro

Welcome back!

HTML Recap

- Set of tags that enclose images, video, text, & other content
- <head>er and <body>
- <div> boxes, <span> around short text

HTML Recap, 2

```
<html>
   <head></head>
   <body>
      <div class='title'>Hello</div>
      <div class='bio'><span
class='greeting'>Hi</span>, this is a little
bit about me</div>
   </body>
</html>
```

Week 2 Recap

- **C**ascading **S**tyle **S**heets apply & transform styles to HTML elements
- Specify *which* elements using selectors, and *what* styles with properties / rules

Selectors recap

- **#***name* (selects by id)
- **.***name* (selects by class)
- *tagname* (selects by tag)

Properties recap

- look like: #selector { property: value; }
- can be a...
  - color (hex, rgb(r,g,b), or name)
  - measurement in pixels or em
  - special keywords like font names, etc

Positioning Recap

- position **absolute** removes an element from the DOM's flow; position **relative** moves the element but rest of DOM pretends it's still in its original spot
- elements are positioned based on the **first element with a 'position' applied to it** that the browser finds, walking up from the node to be positioned

Float Recap

- float: left and float: right take an element out of the flow and send it to as far in that direction of its bounding box as possible.
- other content 'flows' around it like a magazine layout
- **clear: left/right/both** will set that element to occur after any floated elements

3 more things about basic CSS

- Selectors separated by a space will affect children in that order:
  ul li — matches list entries that are children of ul
  #biglist li — matches entries that are children of #biglist
  #biglist li a — matches links(<a>) that are children of list entries inside #biglist
  .article #biglist li a — links that are children of list entries inside a #biglist that's inside something with class "article"

3 more things about basic CSS

- You can combine multiple selectors for an element by joining them:
  div.main — divs that have class main
  #header.callout —element with id "header" and class "callout"
- Combining this rule and the one from last slide:
  #header.callout li — list elements that are children of #header when #header has a "callout" class

3 more things about basic CSS

- **Pseudo-selectors** represent a characteristic of an element, rather than an element:
  - **a:visited** (visited links)
  - **div:hover** (a div that's being moused over at the moment) [we'll use :hover today]

HTML/CSS Questions?

By the end of today...

- Know how to make elements look 'rounded' and shadowed
- Add gradients & reflections to your CSS
- Learn how to do transitions & animations using CSS
- Go through several CSS-heavy exercises

Today's topics

- What are CSS extensions?
- Rounded-ness and shadows
- Gradients
- Transitions
- Animations
- Exercises

To follow along...

- If you've got SVN working, from the Terminal or from Explorer do "svn update"
- If not, go to http://mkrieger.org/cs147/week03.zip

Extensions to CSS

Emerging standards

- Browser makers sometimes want to implement ahead of the standard
- Remember: *experimentation in the open*
- Solution?

Custom extensions

- Prefix!
  - "-moz-" (Mozilla) (like -moz-border-radius: 5px)
  - "-webkit" (WebKit/Safari/iPhone)
  - "-ms-" (Microsoft, IE)
  - "-o-" (Opera)

Survival of the fittest

- Extensions / proposals that are accepted into standards drop the prefix
- Ex:
    - -moz-opacity -> opacity

iPhone supports -webkit-...

(-webkit-)animation
(...)border-radius
(etc)perspective
transform
transition
background-clip
background-composite
background-origin
background-size
marquee
text-fill-color
text-security
text-size-adjust
text-stroke
appearance

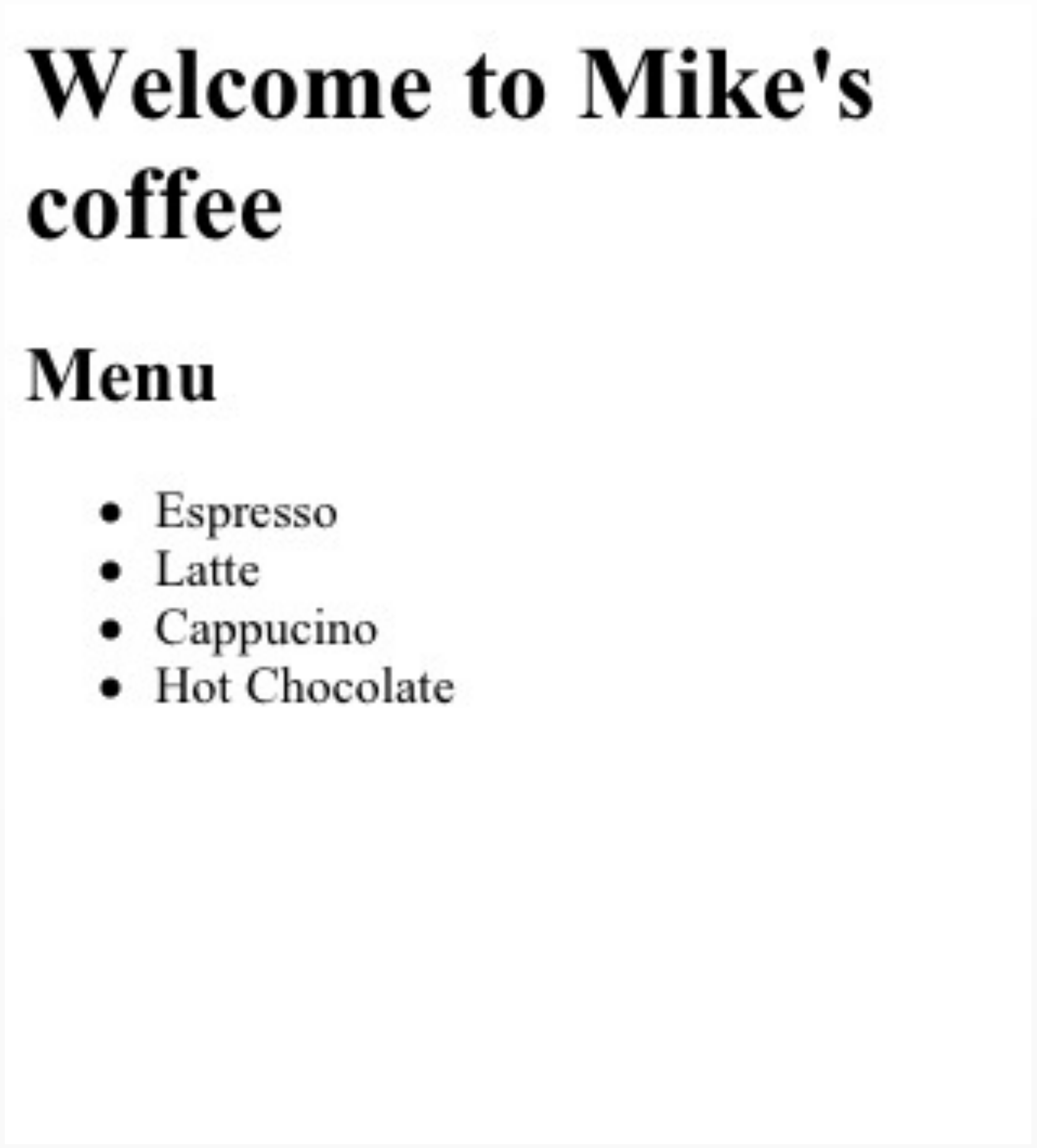
column-count/gap/rule
touch-callout
tap-highlight-color

Roundedness and shadows

Sample HTML: Coffee shop app

```
coffeeshop.html
<html>
<style>...</style>
<body>
<h1>Welcome to Mike's coffee</h1>
<h2>Menu</h2>
<ul>
   <li>Espresso</li>
   <li>Latte</li>
   <li>Cappucino</li>
   <li>Hot Chocolate</li>
</ul>
...
```

Looks like



Welcome to Mike's coffee

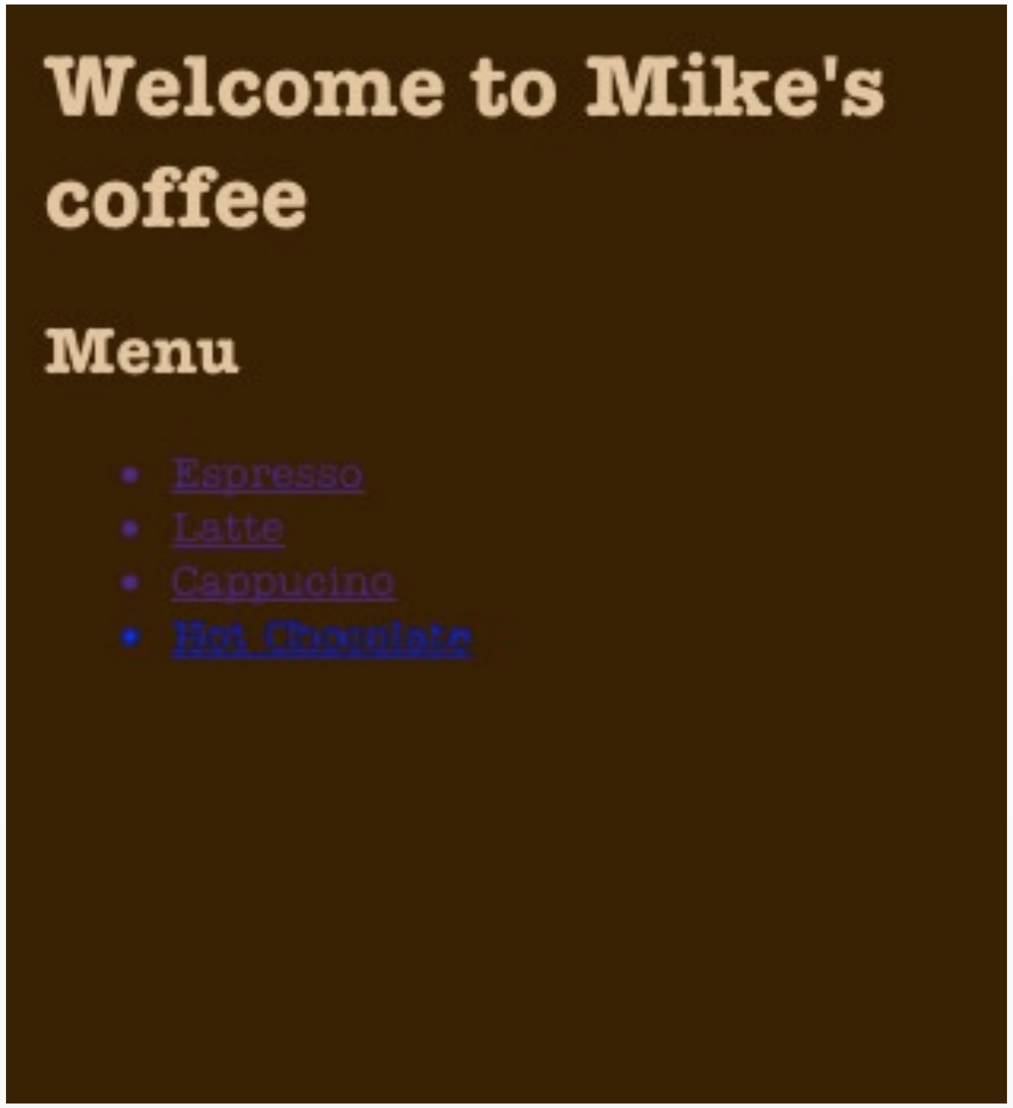
Menu

- Espresso
- Latte
- Cappucino
- Hot Chocolate

Now just the basics

```
coffee-1.css
body {
   background-color: #3d2000;
   color: #ebc7a0;
   font-family: American Typewriter;
   padding: 5px;
   font-size: 14px;
}
```

Looks like...



Welcome to Mike's coffee

Menu

- Espresso
- Latte
- Cappucino
- Hot Chocolate

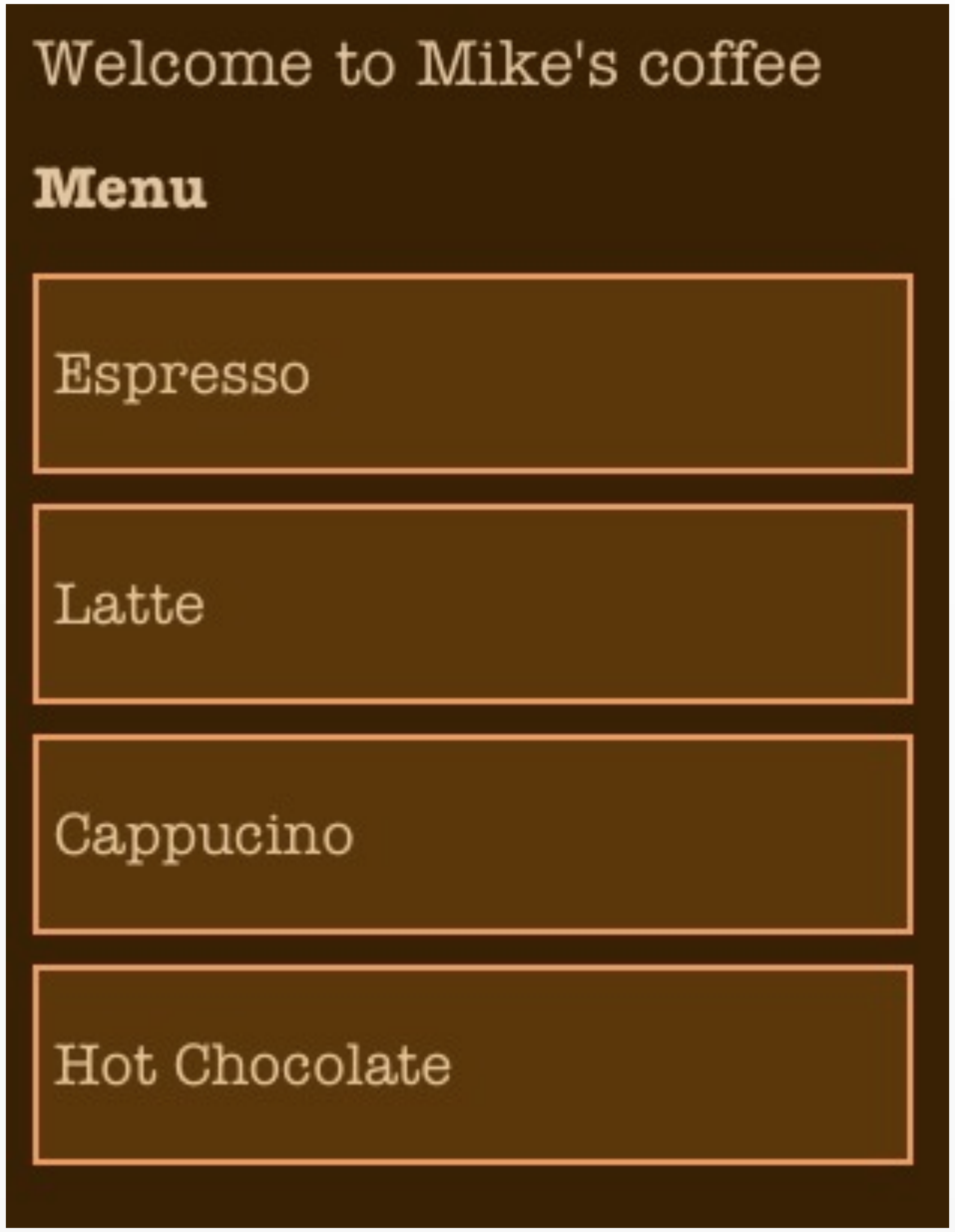
Next step...

coffee-2.css

```
h1.welcome {
     font-size: 1.6em;
     font-weight: normal;
}

h2.menu-header{
     font-size: 1.4em;
}

ul {
     list-style-type: none;
     padding-left: 0px;
     margin-left: 0px;
     width: 100%;
}
```

```
a {
     text-decoration: none;
     color: inherit;

}


li {
     border: 2px solid #ff995e;
     padding: 20px 5px;
     font-size: 1.4em;
     background-color: #623600;
     margin-bottom: 10px;
}
```

Looks like...


Welcome to Mike's coffee

Menu

Espresso

Latte

Cappucino

Hot Chocolate

Border radius

- Specify *how rounded* your border is
- Two ways:
  - -webkit-border-radius: 5px;
  - -webkit-border-top-left-radius: 5px (etc)

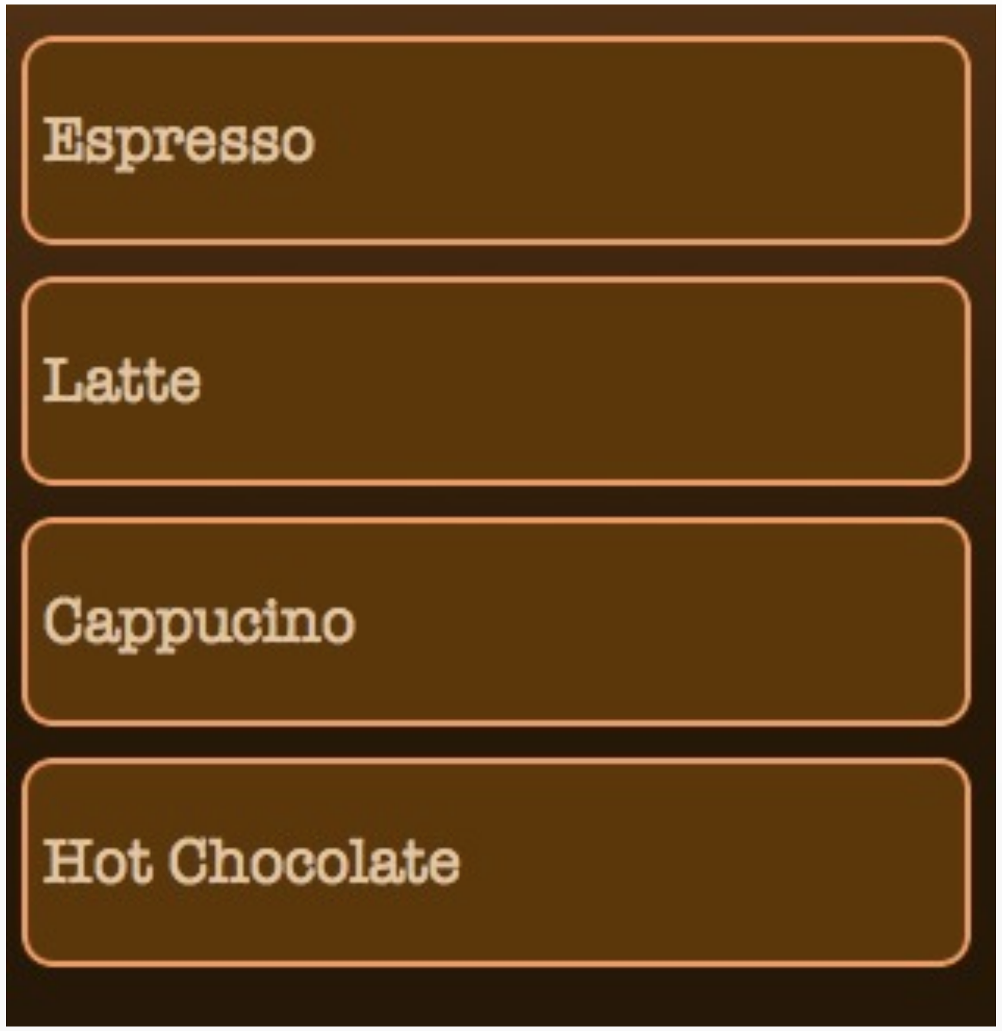
Why rounded corners?

- Makes buttons look more clickable
- Feels less jarring

The CSS is simple

```
li {
   -webkit-border-radius: 10px;
}
```

Looks like:



Box shadow

- Specify a drop shadow for your elements
- Format:
  - -webkit-box-shadow: [color] [x-offset] [y-offset] [softness];

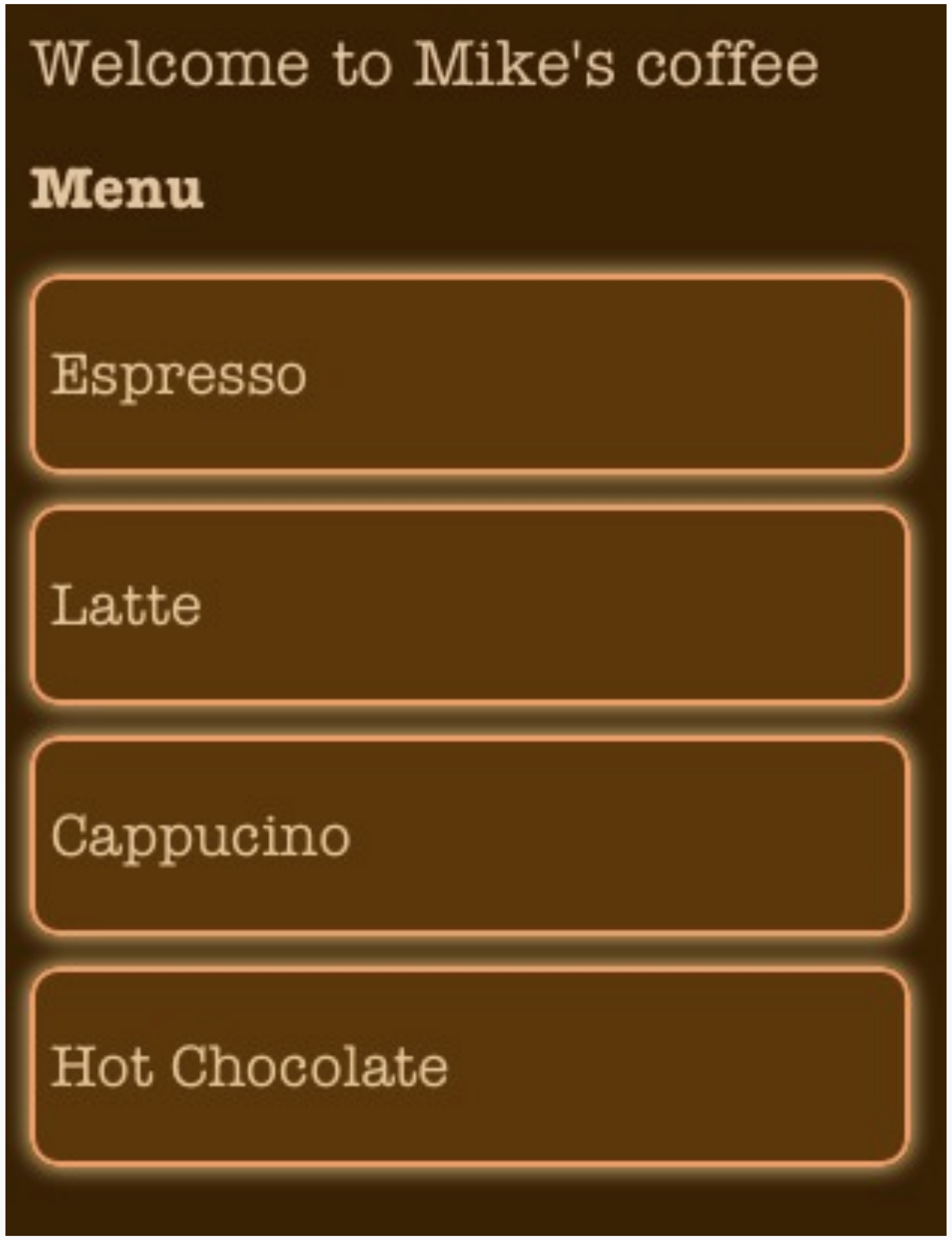
Why drop shadow?

- Gives elements depth relative to page
- Can also create more of a "click" affordance

Next step...

```
coffee-3.css:
li {
      -webkit-border-radius: 10px;
      -webkit-box-shasdow: #ffe2a0 0px 0px 8px;
}
```

Looks like...



Welcome to Mike's coffee

Menu

Espresso

Latte

Cappucino

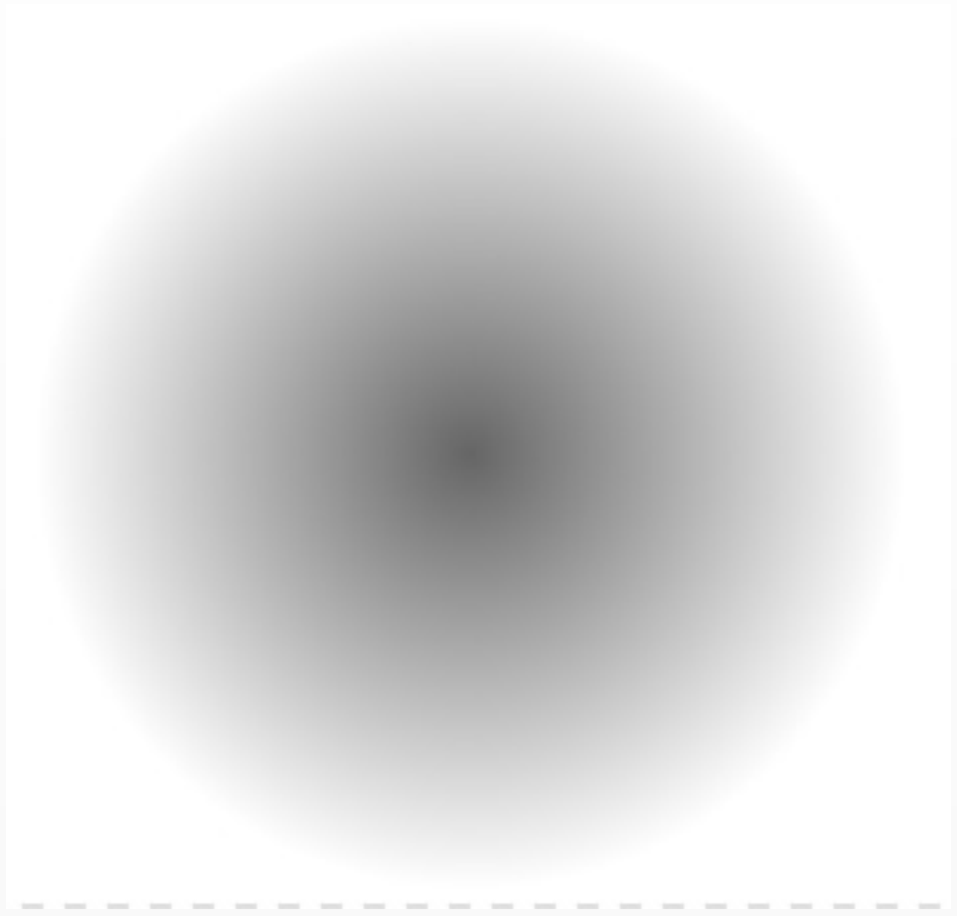
Hot Chocolate

Gradients

Gradients, briefly

- **Linear:** *n* color points in line from x to y, interpolate between
- **Radial:** center color and focus color, interpolate between
- (we'll focus on linear, more applicable in UI design)

Some gradients



Why Gradients in UI?

- Creates depth on screen
- Emphasize/de-emphasize, or guide users' eyes
- Create a more "organic" design

Gradient best practice

- From one shade of a color to another, *not* to another color
- From gray to black
- From white to transparent

CSS Gradients

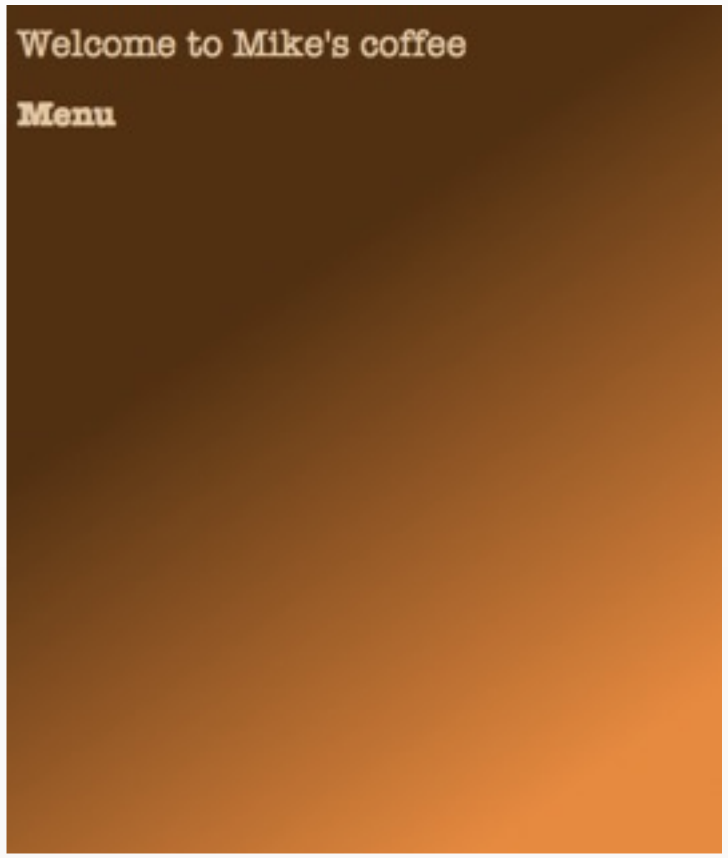
- -webkit-gradient (takes the role of an **image**)
- body { background-image: -webkit-gradient([*linear*|*radial]*, [origin point], [end point], [stops] }

Origin and End Points

background-image: -webkit-gradient(*linear*|*radial*, **[start-point], [end-point]**, [stops]

- keywords (**left top** | **left bottom**)
- or x, y points, no units, separated by space

  -webkit-gradient(linear, 125 200, 320 480...), shown at right
- or percentages separated by space



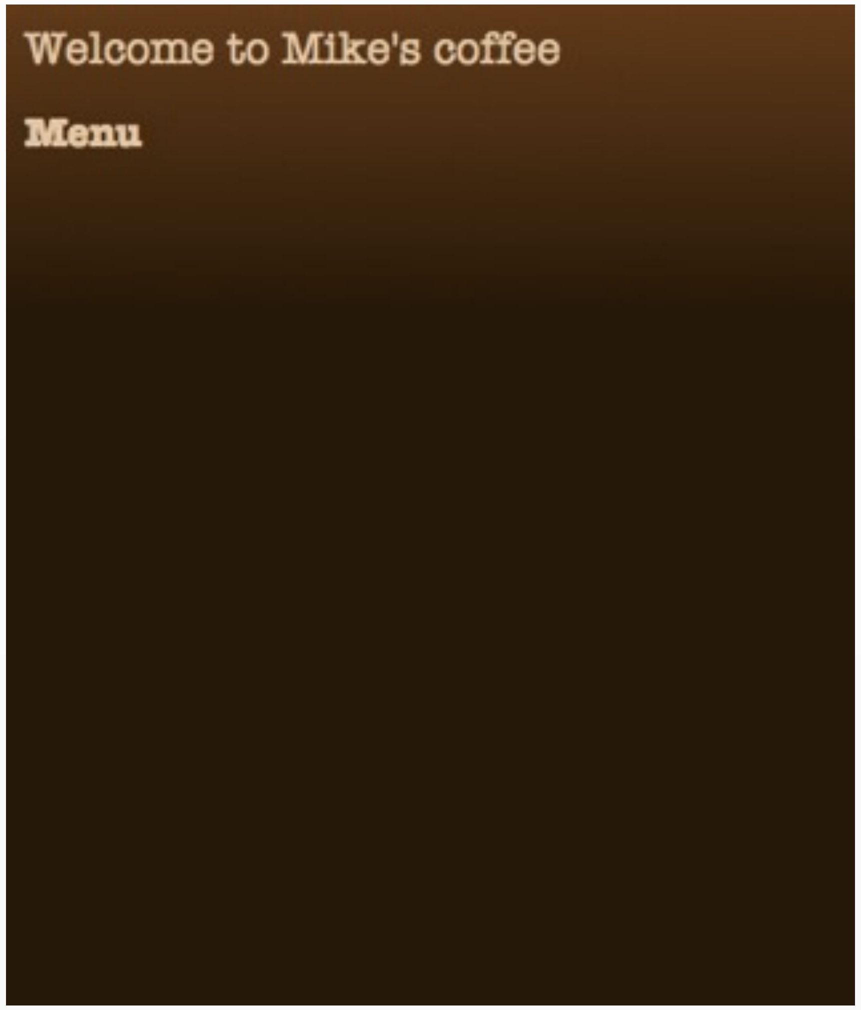
Gradient color stops

background-image: -webkit-gradient(*linear*|*radial*, [start-point], [end-point], **[stops]**

- Three points (start, middle, end)

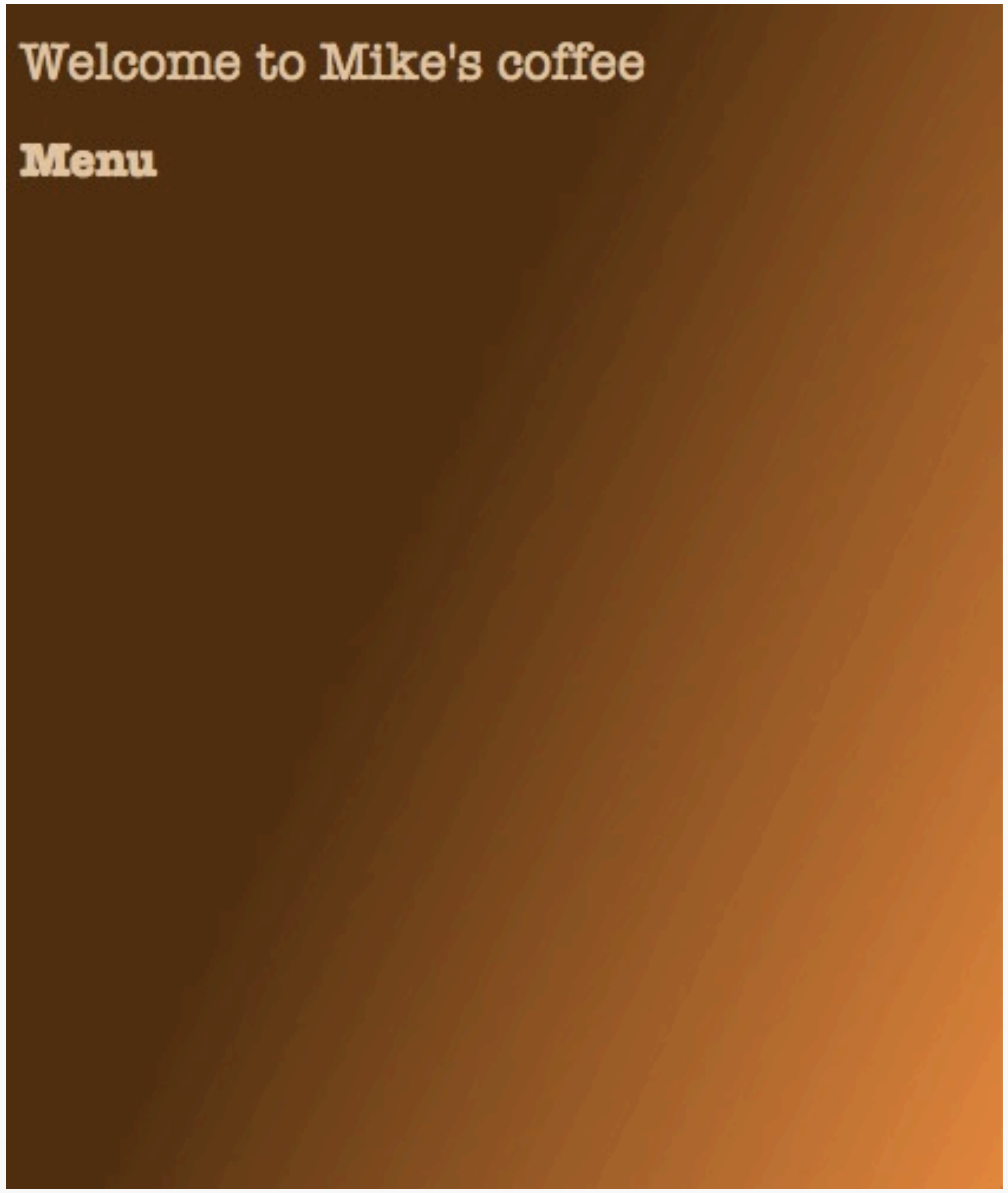
  color-stop([value between 0 and 1], [color])

  background-image: -webkit-gradient(linear, left top, left bottom, color-stop(0, #693a15), color-stop(0.3,#2a1806), color-stop(0.9, #2a1806));



Gradient Color Stops

- Also use **from**( ) **to**( ) shortcuts for 0% and 100%:
- background-image: -webkit-gradient(linear, 25% 70%, 100% 100%, from(#572f0c), to(#f78632));

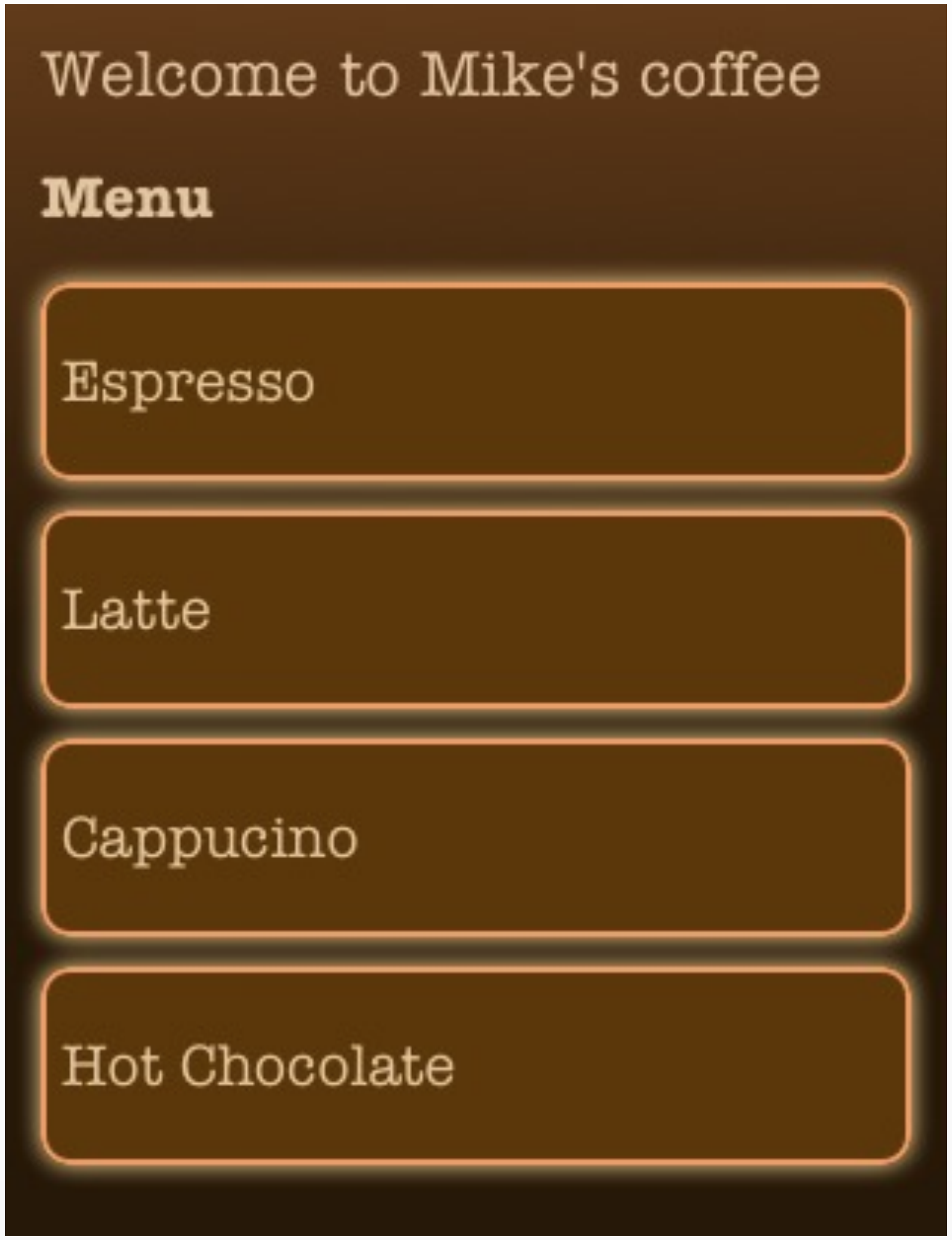

Welcome to Mike's coffee

Menu

Back to our app

```
coffee-4.css
body {
   background-image: -webkit-gradient(linear, left
top, left bottom, color-stop(0, #693a15), color-stop
(0.5,#2a1806), color-stop(0.9, #2a1806));
}
```

Looks like...


Welcome to Mike's coffee

Menu

Reflections

Reflections

- Easy to over-use
- Mind the perspective!

Syntax

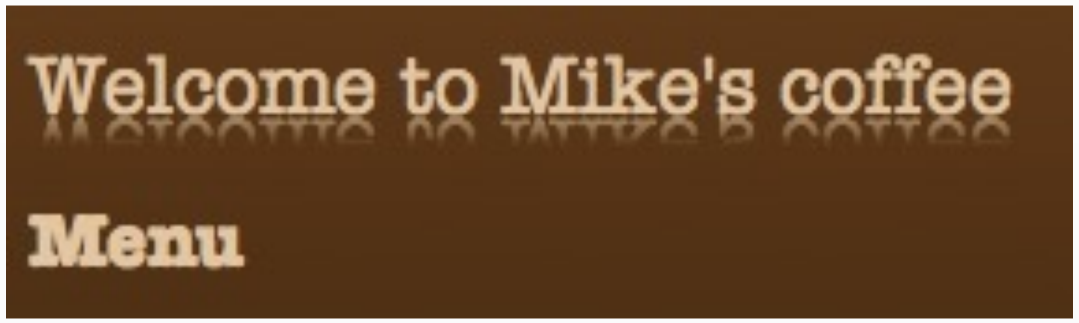
- -webkit-box-reflect: [direction] [offset] [image-mask]
- tip: you can use the gradients we just learned about as image masks

Example

```
.welcome{
   -webkit-box-reflect: below -10px  -webkit-
gradient(linear, left top, left bottom, from
(transparent), color-stop(0.5, transparent),
to(white));
}
```

this says: give me a reflection on the welcome header that's below the header, 10 pixels up from where it would normally be, and mask it with a gradient that goes from transparent to white

Looks like...


Welcome to Mike's coffee

Menu

Notes

- This works on any box element, like a div or a span
- Doesn't receive clicks & touches
- Updates in real time, including for videos!

Transitions

General idea

- Interpolate between two CSS values
- Specify transition on **starting class**

A brief break from coffee

```
transition-basic.html
<style>
</style>
</head>
<body>
    <div>Hover your mouse over me!</div>
```

WebKit Transition

```
-webkit-transition:
[property] [duration in seconds] [easing
function];
```

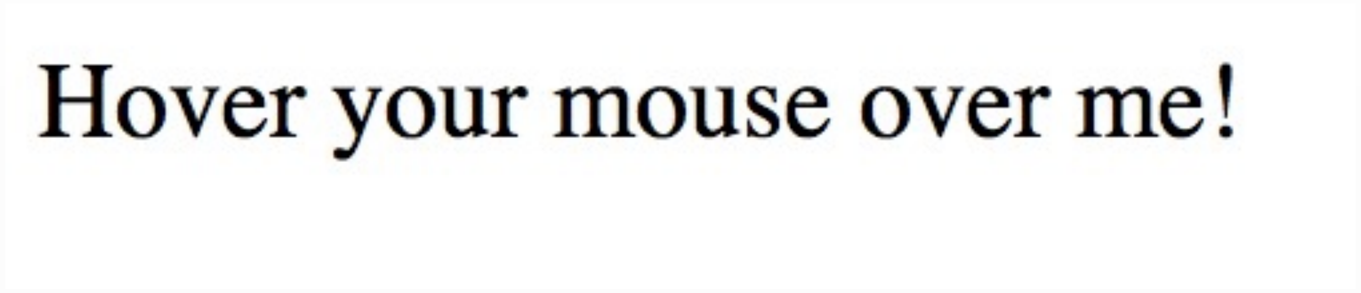
easing functions:
linear
ease
ease-in
ease-out
ease-in-out
cubic-bezier

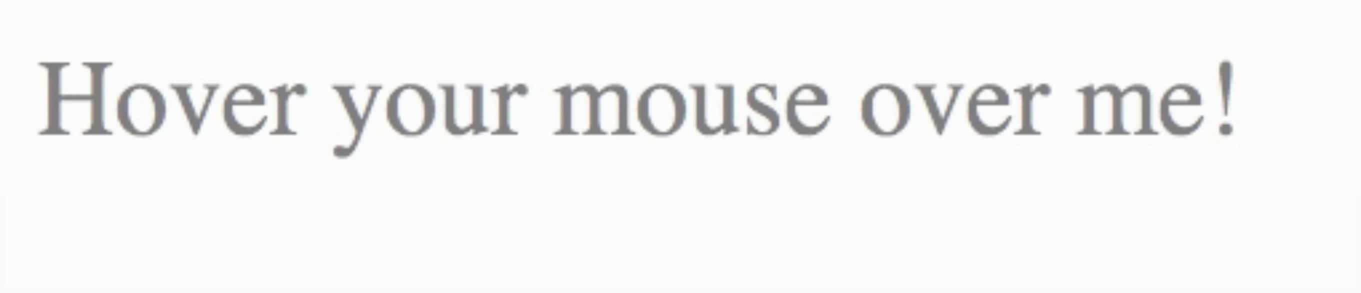
Adding a transition

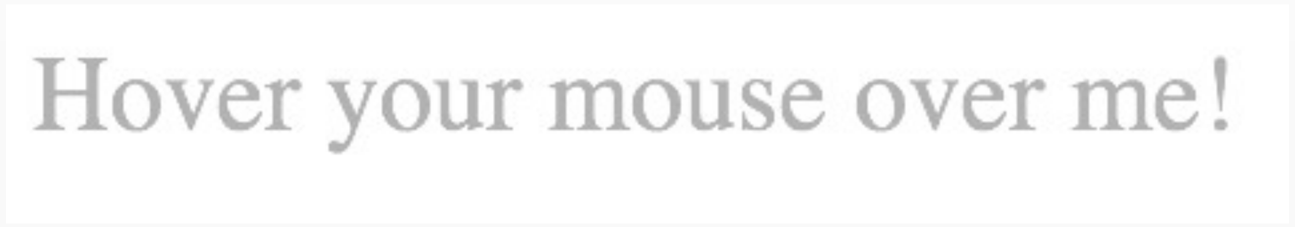
```
div {
    opacity: 1.0;
    -webkit-transition: opacity 1s linear;
}

div:hover {
    opacity: 0.3;
}
```

on **starting class**, we set that we want any changes to opacity to animate in 1 seconds, and using a linear easing function

Does...


Hover your mouse over me!


Hover your mouse over me!


Hover your mouse over me!

What the browser does

- If transition specific in starting class:
  - Watch for changes in value that are marked "transition"
  - For every point between now and *n* seconds in the future, update the value to an interpolation between original value and desired value using an *easing function*

So what can I transition?

- Almost any numeric properties
- Colors (!)
- Opacity
- Transforms

Back to coffee

```
a little bit of JavaScript, we'll cover it next week:

function setupTouchEvents() {
    var lis = document.getElementsByTagName('li');
    for(var i = 0; i < lis.length; i++){
        lis[i].addEventListener("touchstart", function(){
           this.className = "touched";}, false);
        lis[i].addEventListener("touchend", function(){
           this.className = "";}, false);
    }
}
```

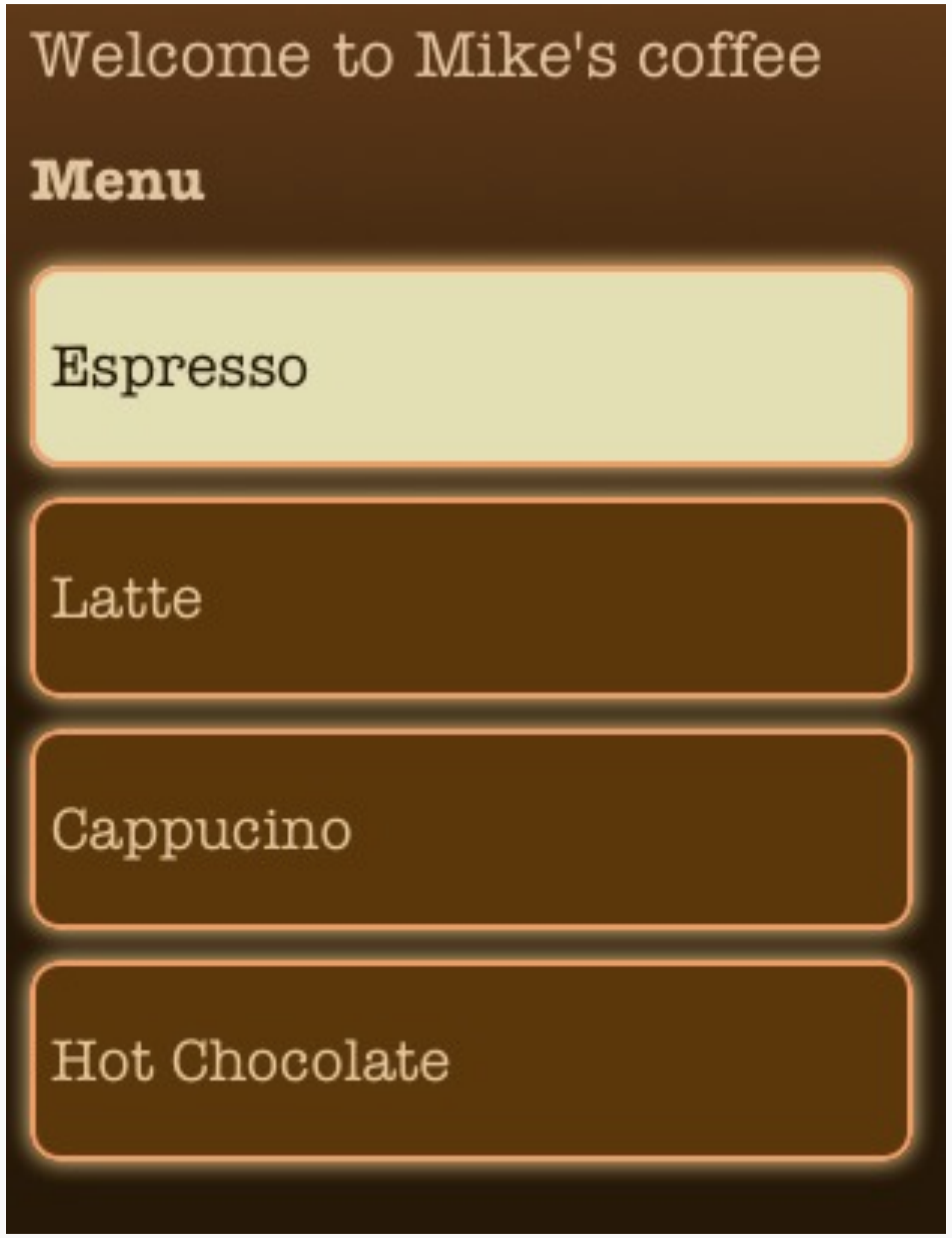
don't worry about details, this is adding a class of "touched" when we touch a list entry, and take it off after we stop touching

The touched class

```
coffee-5.css
li {
   -webkit-transition: background-color 1s
linear;
}

li.touched {
   background-color: #e4e1b3;
   color: #2a1806;
}
```

what this is saying: when background-color changes, animate it for 1 second

Looks like...


Welcome to Mike's coffee

Menu

Espresso

Latte

Cappucino

Hot Chocolate

(demo on iPhone)

http://mkrieger.org/cs147/week03/coffeehome.html

Transforms

Transforms

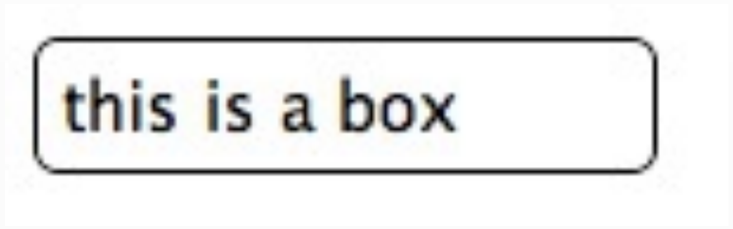
- Scaling, rotating, displacing
- iPhone even supports 3D!

Syntax

- -webkit-transform: function(values..)
- functions: scale, scaleX/Y/Z, rotateX/Y/Z, translateX/Y/Z

Back to boxes

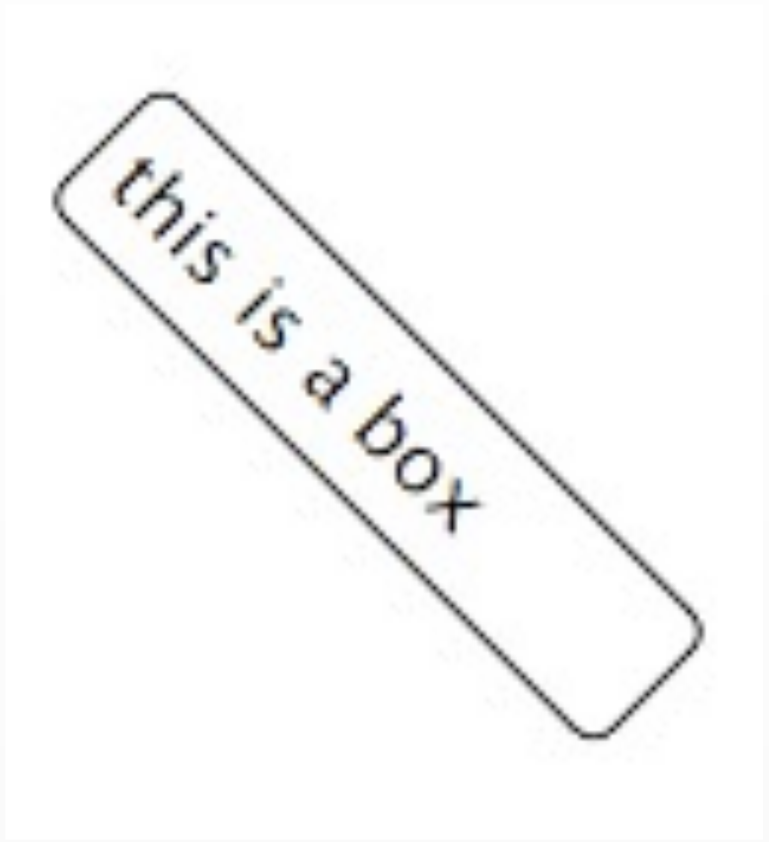
```
boxy.html:
<html>
<head>
<style>
.box {
border: 1px solid black;
padding: 5px;
-webkit-border-radius: 5px;
width: 120px
 }

</style>
<body>
    <div class='box'>this is a box
</div>
```



this is a box

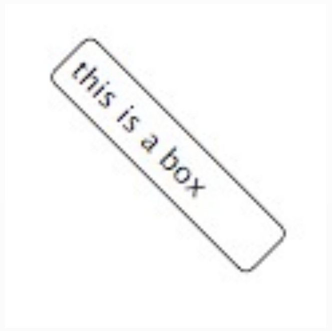
Rotate

```
boxy.html:
<html>
<head>
<style>
.box {
border: 1px solid black;
padding: 5px;
-webkit-border-radius: 5px;
width: 120px;
-webkit-transform: rotate(45deg);
 }
```



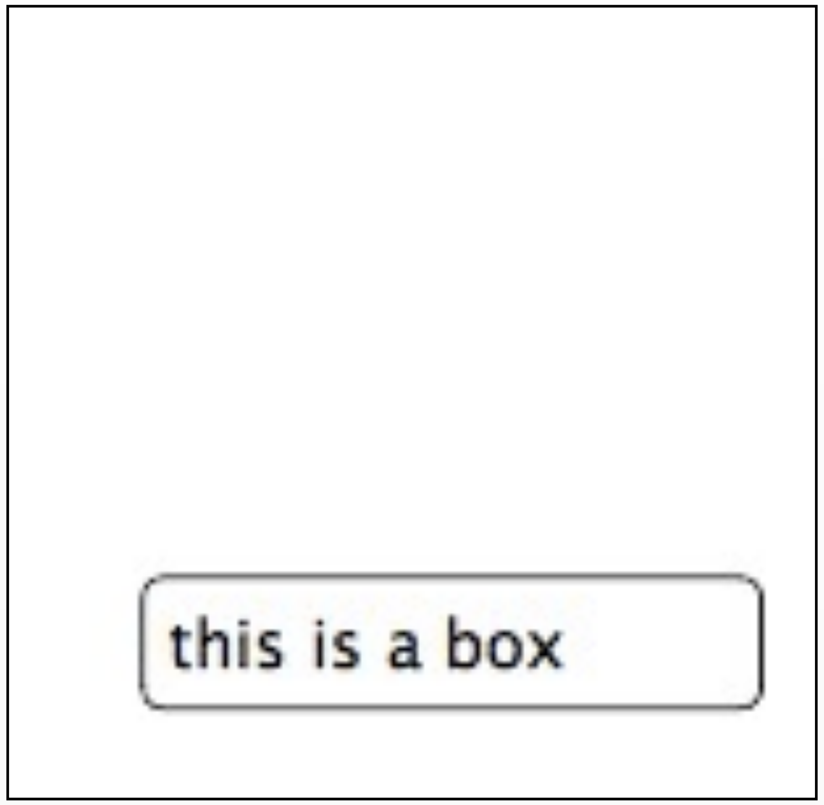
Rotate & Scale

```
boxy.html:
<html>
<head>
<style>
.box {
border: 1px solid black;
padding: 5px;
-webkit-border-radius: 5px;
width: 120px;
-webkit-transform: rotate(45deg),
scale(0.5);
 }
```



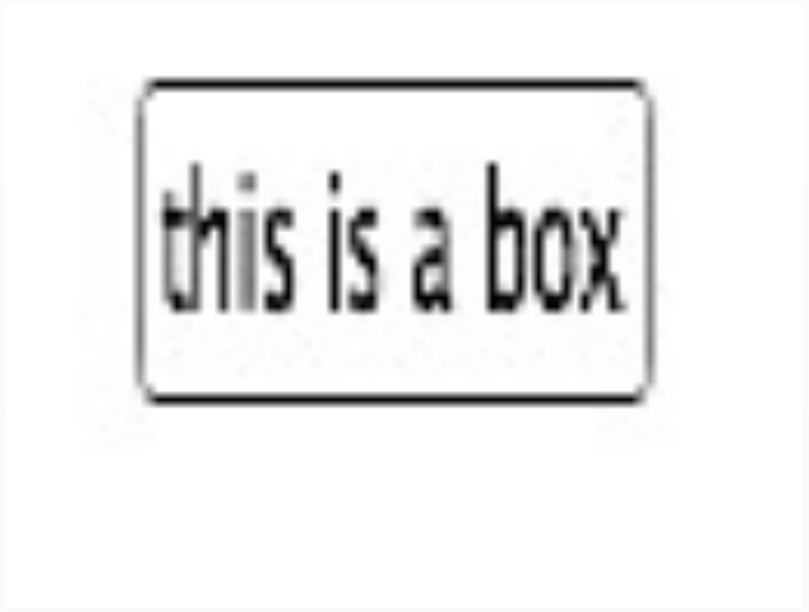
Scale & Translate

```
boxy.html:
<html>
<head>
<style>
.box {
border: 1px solid black;
padding: 5px;
-webkit-border-radius: 5px;
width: 120px;
-webkit-transform: scale(0.9)
translate(1em, 3em);
 }
```



3D rotation

```
boxy.html:
<html>
<head>
<style>
.box {
border: 1px solid black;
padding: 5px;
-webkit-border-radius: 5px;
width: 120px;
-webkit-transform: rotateY(45deg);
 }
```



Transforms + Transitions = Animations

Strategy

- Combine transforms & transitions to create animations, including 3D

Syntax

```
-webkit-animation: [animation-name] [duration] [num
repeats|infinite] [easing function];
```

an animation is like a transition, but doesn't have to be triggered by a class change, and can loop

Keyframes

```
@-webkit-keyframes [animation-name] {
from { [initial transform] }
to { [end transform] }
}

OR

@-webkit-keyframes [animation-name] {
0% { [initial transform] }
33% { [state at 33%] }
90% { [state at 90%] }
...
100% { [final state; back to beginning?] }
}
```

Spinning box

```
.box {
    -webkit-animation: spin 5s infinite linear;

}

@-webkit-keyframes spin {

    from { -webkit-transform:rotateY(0deg); }

    to { -webkit-transform:rotateY(-360deg); }

}
```

Demo

http://mkrieger.org/cs147/week03/boxy-1.html

Cartwheel box

```
.box2 {
    width: 200px;
    color: white;
    padding: 5px;
    background-image: -webkit-gradient(linear, left top, left bottom,
from(black), to(gray));
    -webkit-animation: cartwheel 5s infinite ease;
}

@-webkit-keyframes cartwheel {

0% { -webkit-transform: translateX(0) rotateY(0deg); }

50% { -webkit-transform: translateX(600px) rotateY(-360deg); }

100% { -webkit-transform: translateX(0px) rotateY(0deg); }

}
```

Demo

http://mkrieger.org/cs147/week03/boxy-2.html

Putting it in practice

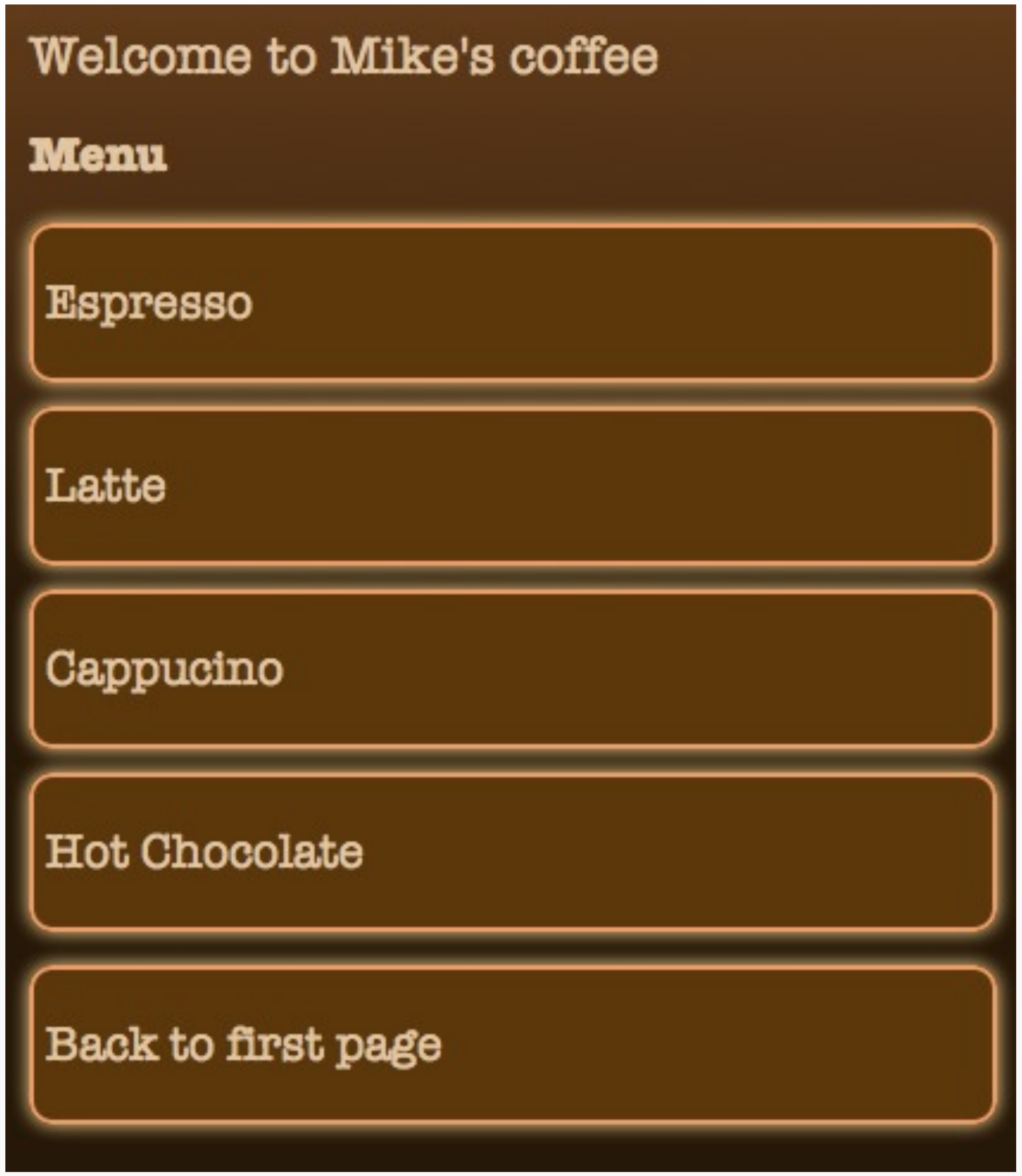
Sliding menu

- Back to our coffee shop...
- coffeeshop-navigation.html

What's new

```
<div id="container">
    <div id="first-page">
        <h1 class="welcome">Welcome to Mike's coffee</h1>
        <h2 class="menu-header">Menu</h2>
        <ul>
           <li><a href="#espresso">Espresso</a></li>
           <li><a href="#latte">Latte</a></li>
           <li><a href="#cappucino">Cappucino</a></li>
           <li><a href="#hotchocolate">Hot Chocolate</a></li>
        </ul>
    </div>
    <div id="second-page" class="hidden">
        <ul>
            <li>Back to first page</li>
        </ul>
    </div>
</div>
```

Looks like...



Welcome to Mike's coffee

Menu

Strategy

- Have two pages **positioned absolutely**, next to each other
- Use CSS transitions and transforms to slide between the pages
- Use as little JavaScript as possible

Event flow

- Page loads; one "page" has no class, other has class "hidden"
- On user tap, we add "hidden" to the first page, remove "hidden" from the second page (and vice-versa when the user is on page 2)

The JavaScript

again, don't worry about details; the important thing is that we add a "hidden" class to the page we want offscreen

```
var listEntries = document.getElementsByTagName('li');
var currentPage = 1;
var onTouchStart = function(){
    this.className = "touched";
};
var onTouchEnd = function(){
    this.className = "";
    if (currentPage == 1) {
        document.getElementById('first-page').className = "hidden";
        document.getElementById('second-page').className = "";
        currentPage = 2;
    } else {
        document.getElementById('first-page').className = "";
        document.getElementById('second-page').className = "hidden";
        currentPage = 1;
    }
}
for(var i = 0; i < listEntries.length; i++){
    listEntries[i].addEventlistEntriestener("touchstart", onTouchStart , true);
    listEntries[i].addEventlistEntriestener("touchend", onTouchEnd, true);
}
```

So, what does the hidden class do?

```
#container {
    min-height: 600px;
    width: 320px;
    overflow: hidden;
}
```

set a transition on the "left" property (which determines how far off we are positioned)

```
#first-page {
    -webkit-transition: left 0.5s
ease;
    position: absolute;
    left: 10px;
    width: 310px;
    height: 500px;
}

li {
    width: 290px;
}
```

```
#second-page {
    position: absolute;
    left: 10px;
    -webkit-transition: left 0.5s
ease;
}
```

.hidden is specific to the page

```
#first-page.hidden {
    left: -330px;
}

#second-page.hidden {
    left: 330px;
    width: 320px;
}
```

we hide the second page by offsetting it 330px from the left

Demo!

http://mkrieger.org/cs147/week03/coffeehome-navigation.html

Flip portfolio

Goal

- Take biography off front page and put it on the "back", matches iPhone UX patterns

Changes from last week

```
<div id="container">
    <div id="home-screen">
        <div class='content'>
            <div id="info">...</div>
            <div class="clear"></div>
               <a href="#" id="bio-link">See bio</a>
            <div id='assignments'>
                <h2>Course work</h2>
                <ul>...</ul>
            </div>
            <div class='clear'></div>
        </div>
    </div>
    <div id='bio'>
        <div class='content'>
        <h2>Bio</h2>
            This is my bio.<br/>
            <a href="#" id="back-home">Back home</a>
        </div>
    </div>
</div>
```

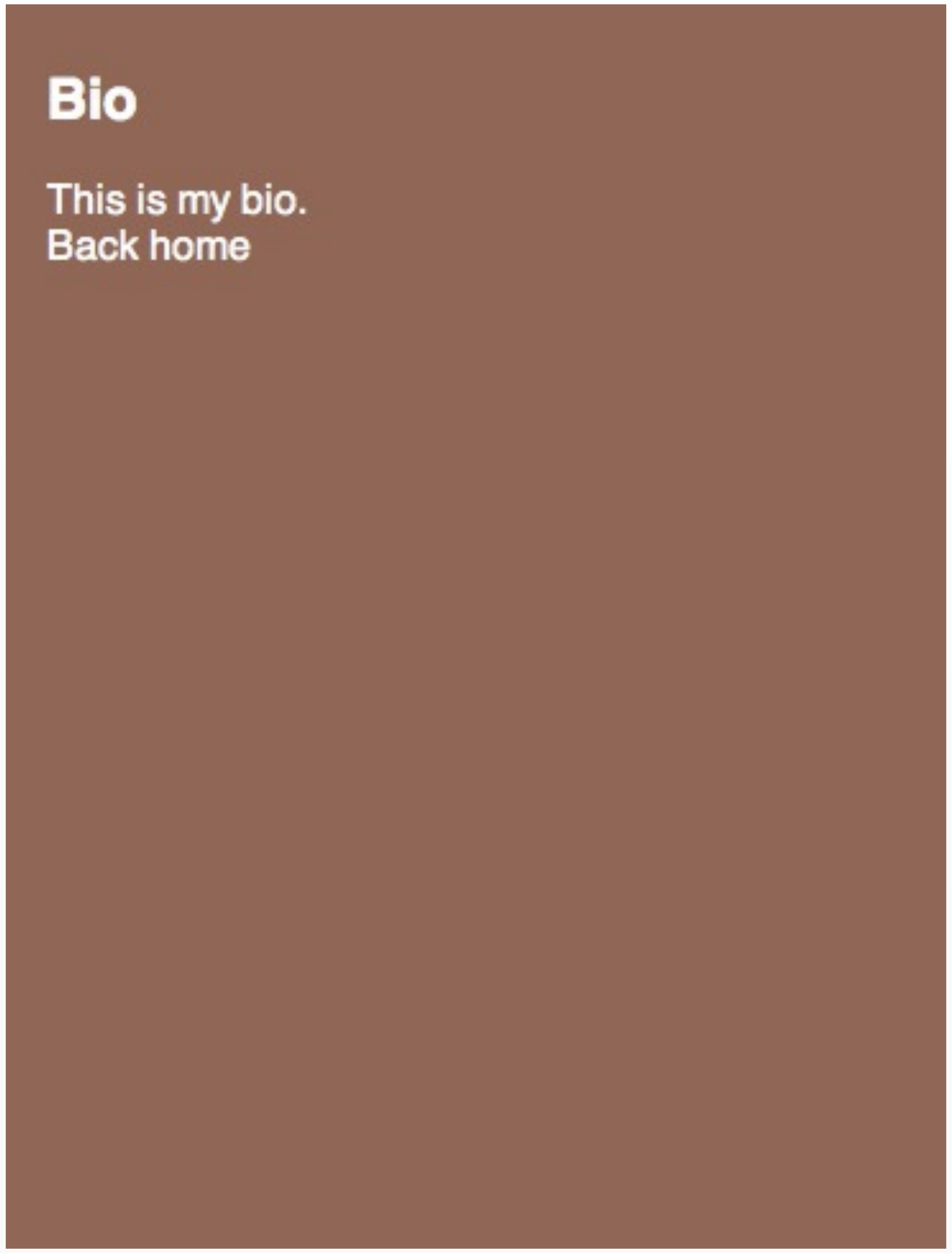
Strategy

- Pretend we have two cards in a box
- One card is front page, other is bio
- Bio page is rotated 180 degrees so the two cards are back to back
- If we rotate the whole box, we can control whether we see the front page or the back page

In other words...

```
#container {
} /* this is the box around the cards */

#home-screen {
    position: absolute;
} /* this is the first card */

#bio {
    position: absolute;
    left: 0;
    width: 320px;
}
```

Looks like...



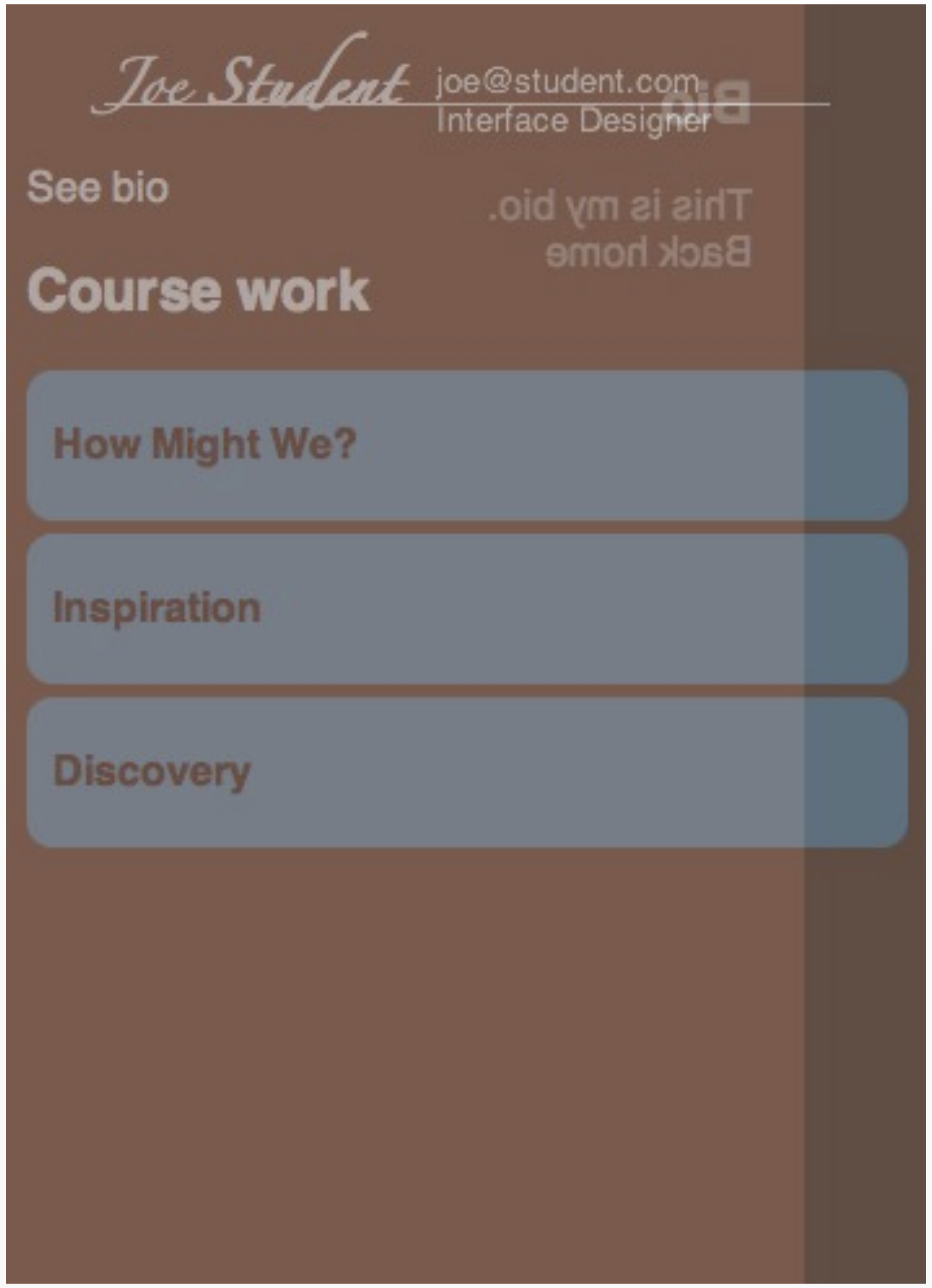
Bio

This is my bio.
Back home

Now flip the back card...

```
#container {
    -webkit-transform-style: preserve-3d;
} /* this is the box around the cards */

#home-screen {
    position: absolute;
} /* this is the first card */

#bio {
    position: absolute;
    left: 0;
    width: 320px;
    -webkit-transform: rotate(180deg);
}
```

Looks like...



(the cards have 0.5 opacity to illustrate what's going on)

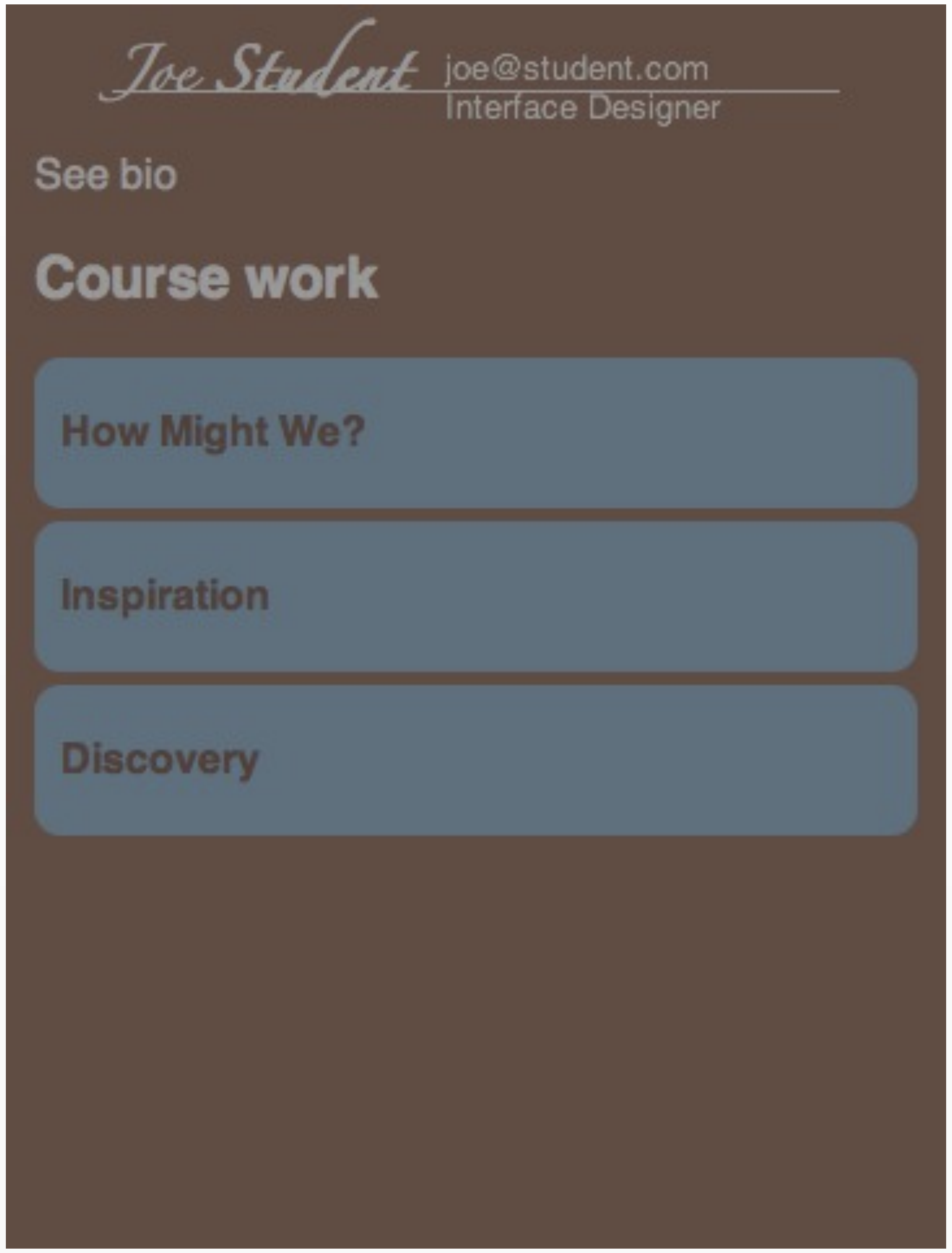
Problem

- We can see the card that's supposed to be behind!
- Solution: -webkit-backface-visibility: hidden;

Backface visibility

```
#container {
    -webkit-transform-style: preserve-3d;
} /* this is the box around the cards */

#home-screen {
    position: absolute;
} /* this is the first card */

#bio {
    position: absolute;
    left: 0;
    width: 320px;
    -webkit-backface-visibility: hidden;
    -webkit-transform: rotate(180deg);
}
```

Now...


Joe Student
joe@student.com
Interface Designer
See bio
Course work
How Might We?
Inspiration
Discovery

Can't see the back card anymore!

Okay, now a teeny bit of JavaScript

```
<script>
function init() {
    var bioLink = document.getElementById("bio-link");
    var container = document.getElementById("container");
    var backHome =  document.getElementById("back-home");
    bioLink.addEventListener("mousedown", function(){
        container.className = "flipped";
        return false;
    }, false);
    backHome.addEventListener("mousedown", function(){
        container.className = "";
        return false;
    }, false);

}
```

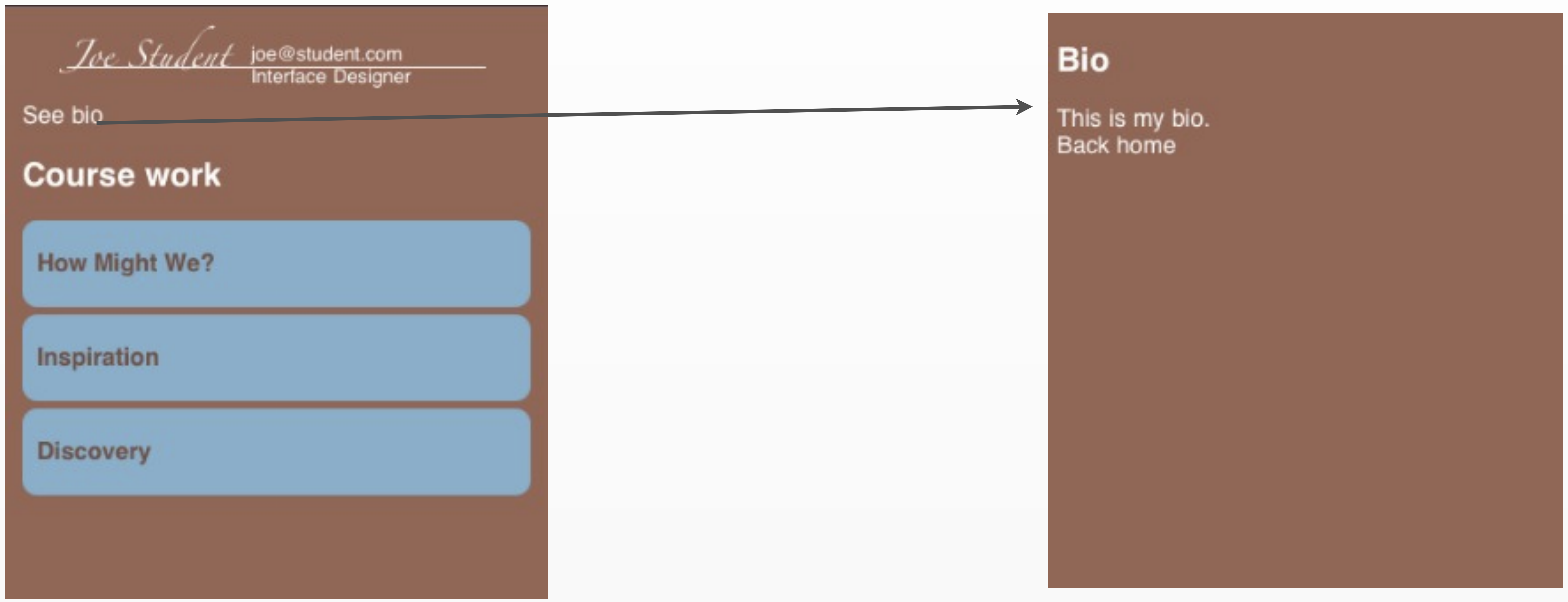
on clicking the link to Bio, add "flipped" to the container's class. on clicking the "back home" link, remove the "flipped" class

the .flipped class

```
#container.flipped{
    -webkit-transform: rotateY(180deg);
}
```

Rotate the whole box (that the two cards are inside of) by 180degrees

Looks like...


Joe Student
joe@student.com
Interface Designer
See bio
Course work
How Might We?
Inspiration
Discovery
Bio
This is my bio.
Back home

Let's get it to animate!

```
#container {
   -webkit-transform-style: preserve-3d;
   -webkit-transition: -webkit-transform 1s
ease;
}
```

In other words: when the -webkit-transform property changes, don't do it all at once, instead do it over 1 seconds with the standard easing function

Demo

http://mkrieger.org/cs147/portfolio/week03.html

Today's summary

- Making elements look 'rounded' and shadowed
- Adding gradients & reflections to your CSS
- Learn how to do transitions & animations using CSS
- Go through a few CSS-heavy exercises

Next week...

- JavaScript!

Q's?