MICHAEL MONCADA
&
MICHAEL CLARK

PRESENT

SCALING THE GAME OF PHONES...

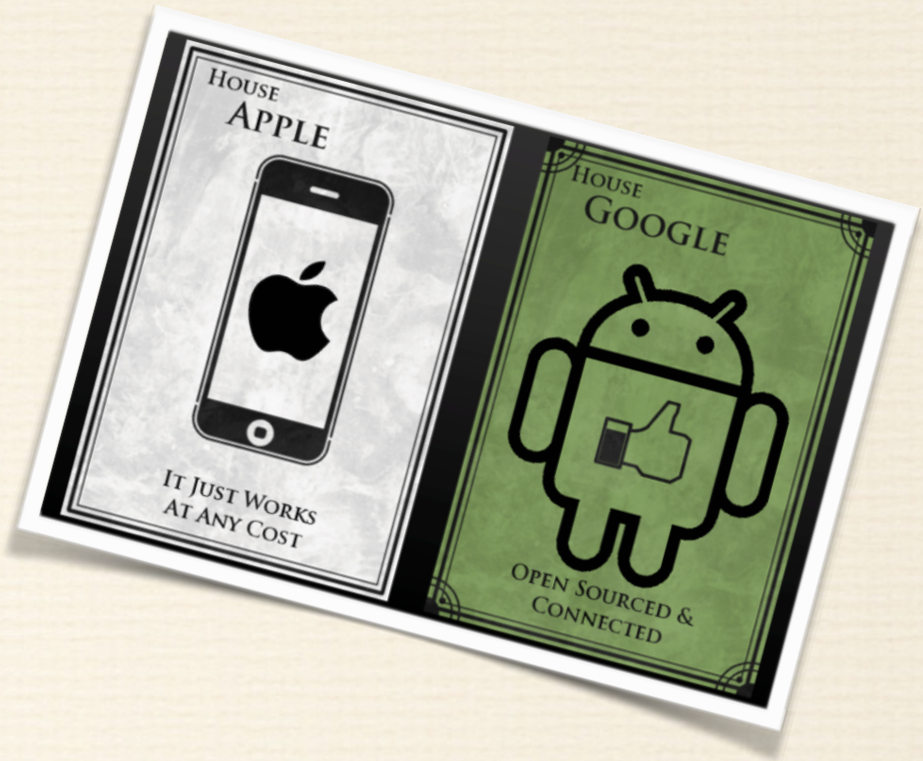
Building Interactive Interfaces in HTML5

"I am accustomed to sleep and in my dreams to imagine the same things that lunatics imagine when awake."

—Rene Descartes

What we're talking about

- **What makes an engaging experience?**
- **HTML Animation Techniques**
- **Why use CSS?**
- **When all else fails…always have a plan B**
- **Detecting browser capabilities using Modernizr**
- **Releasing your inner Hulk with Mathematics**
- **Bringing it all together - Geeky Recipes**
- **Additional Goodies - Animate.css, Tutorials**
- **Questions and hopefully some answers…**



http://www.applift.com/blog/the-rules-of-mobile-attraction-why-apple-and-google-are-drawn-to-become-more-similar.html

What Makes an Engaging Experience?

Engaging User Experiences

- Hierarchy of app expectations: functional, reliable, usable, pleasurable

- Is a pleasurable, engaging user experience the end or a means to something else?

- **Engaging experience = improved emotional response, improved information retention, improved task performance**

The Makings of an Engaging Experience

- What makes for an engaging experience? A few concepts:
  - Feedback Loops
  - Delighters
  - Peak-End Rule
- Applying these concepts to user interfaces and front-end development techniques.

HTML Animation Techniques

- CSS/CSS3D and DOM (most supported by browsers, including mobile - catch all, especially for older phones!)

- HTML Canvas (supported by newer browsers with decent graphics hardware)

- HTML Canvas using WebGL (supported by new browsers but need more powerful hardware)

The CSS Advantage

- Most supported method for animation across most current browsers.
- CSS/CSS3D transforms (many browsers now support transitions as well) are hardware (**GPU**) accelerated on most modern browsers including IE9+, Mobile Chrome, Mobile Safari and most desktop browsers.
- You can promote your containers to use hardware acceleration simply by specifying the "**transform: translateZ(0);**" CSS rule (more about this later). Webkit based browsers can use "**-webkit-transform: translate3d(0, 0, 0)**"
- Use absolute positioning on animated elements to prevent browser reflow!

Enabling GPU Hardware Acceleration

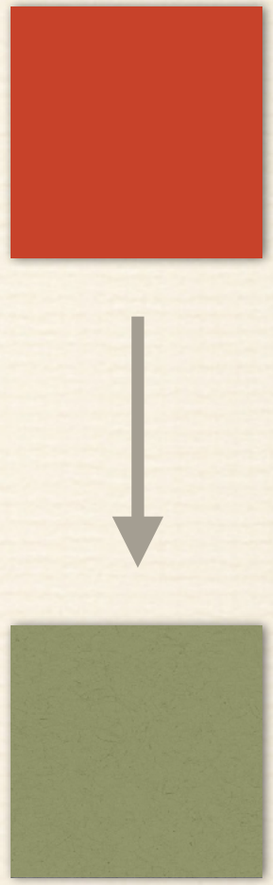
```
.gpu-accelerated-container {
  position: absolute;
  -webkit-transform: translateZ(0);
  -moz-transform: translateZ(0);
  -ms-transform: translateZ(0);
  -o-transform: translateZ(0);
  transform: translateZ(0);
}
```

```
.gpu-accelerated-container-webkit {
  position: absolute;
  -webkit-transform: translate3d(0, 0, 0);
  -moz-transform: translate3d(0, 0, 0);
  -ms-transform: translate3d(0, 0, 0);
  transform: translate3d(0, 0, 0);
}
```

CSS Transitions

```
.box {
  width: 100px;
  height: 100px;
  background: red;
  margin-top: 20px;
  margin-left: auto;
  margin-right: auto;
}

.box:hover {
  background-color: green;
  cursor: pointer;
  -webkit-transition: background-color 2s ease-out;
  -moz-transition: background-color 2s ease-out;
  -o-transition: background-color 2s ease-out;
  transition: background-color 2s ease-out;
}
```



** **

Sourced From: http://css-tricks.com/almanac/properties/t/transition/

Preventing Animation Flicker on Chrome/ Safari Browsers

```
.no-animate-flicker {
  -webkit-backface-visibility: hidden;
  -moz-backface-visibility: hidden;
  -ms-backface-visibility: hidden;
  backface-visibility: hidden;
  -webkit-perspective: 1000;
  -moz-perspective: 1000;
  -ms-perspective: 1000;
  perspective: 1000;
}
```

A disclaimer: When all else fails…always have a Plan B

- It is important to use CSS to enhance not drive UI functionality. **That is unless you are <u>certain</u> of your clients platform**.

- Use JS to compensate either by providing a CPU-based animation or bypassing the animation completely (in the case of older browsers). Not all browsers can render rotating cubes!

- Use tools like **Modernizr** to detect browser capabilities and browser sniff to determine how well those capabilities are supported.

Detecting Browser Capabilities: Modernizr

- A CSS class is added to the HTML entity for every feature the browser supports including "**cssanimations**", "**csstransitions**", "**csstransforms**"
- Use jQuery **hasClass** method to check for capabilities.
- http://modernizr.com/

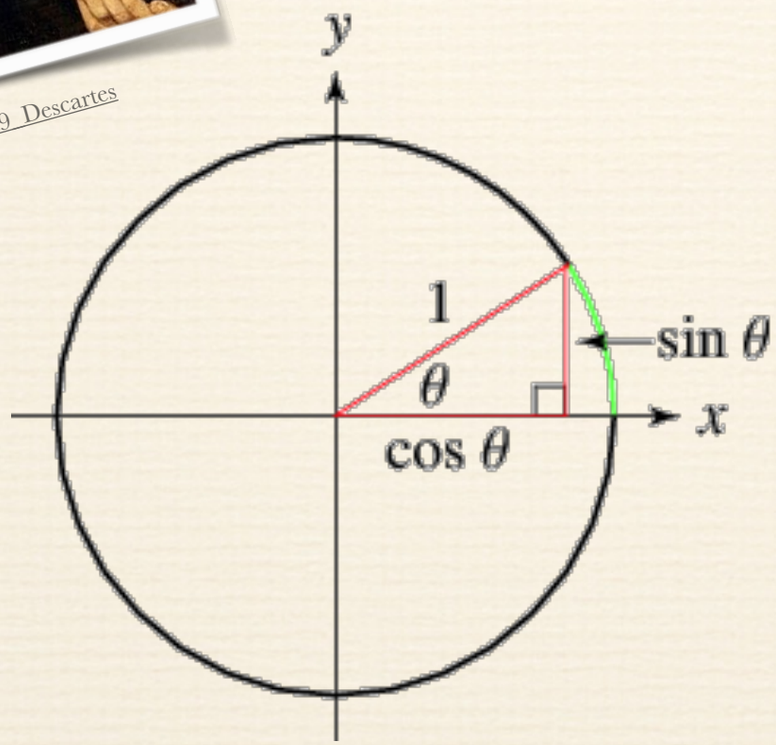
```
<!DOCTYPE html>
▼<html lang="en" dir="ltr" id="modernizrcom" class="js no-touch postmessage history multiplebgs boxshadow
opacity cssanimations csscolumns cssgradients csstransforms csstransitions fontface localstorage
sessionstorage svg inlinesvg no-blobbuilder blob bloburls download formdata wf-proximanova1proximanova2-n4-
active wf-proximanova1proximanova2-i4-active wf-proximanova1proximanova2-n7-active wf-
proximanova1proximanova2-i7-active wf-proximanovacondensed1proximanovacondensed2-n6-active wf-
athelas1athelas2-n4-active wf-active">
  ▶<head>…</head>
  ▼<body id>
    ▼<div class="page">
      ▶<nav role="navigation">…</nav>
      ▼<div class="container clrfx">
        ▶<header id="masthead" class="loading">…</header>
        <!--
                <aside class="quote">
                   <q>Modernizr is an essential part of my toolkit of files.</q>
                   <cite>Andy Clarke</cite>, <a class="cite-source" href="http://www.stuffandnonsense.co.uk/"
        rel="external">Stuff &amp;amp; Nonsense</a>
                </aside>
                -->
        ▼<aside class="quote">
```

Releasing your inner Hulk with Mathematics

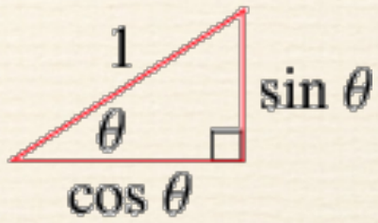
http://en.wikipedia.org/wiki/Ren%C3%A9_Descartes

http://www.fotoswiki.org/filmes/fotos-de-hulk

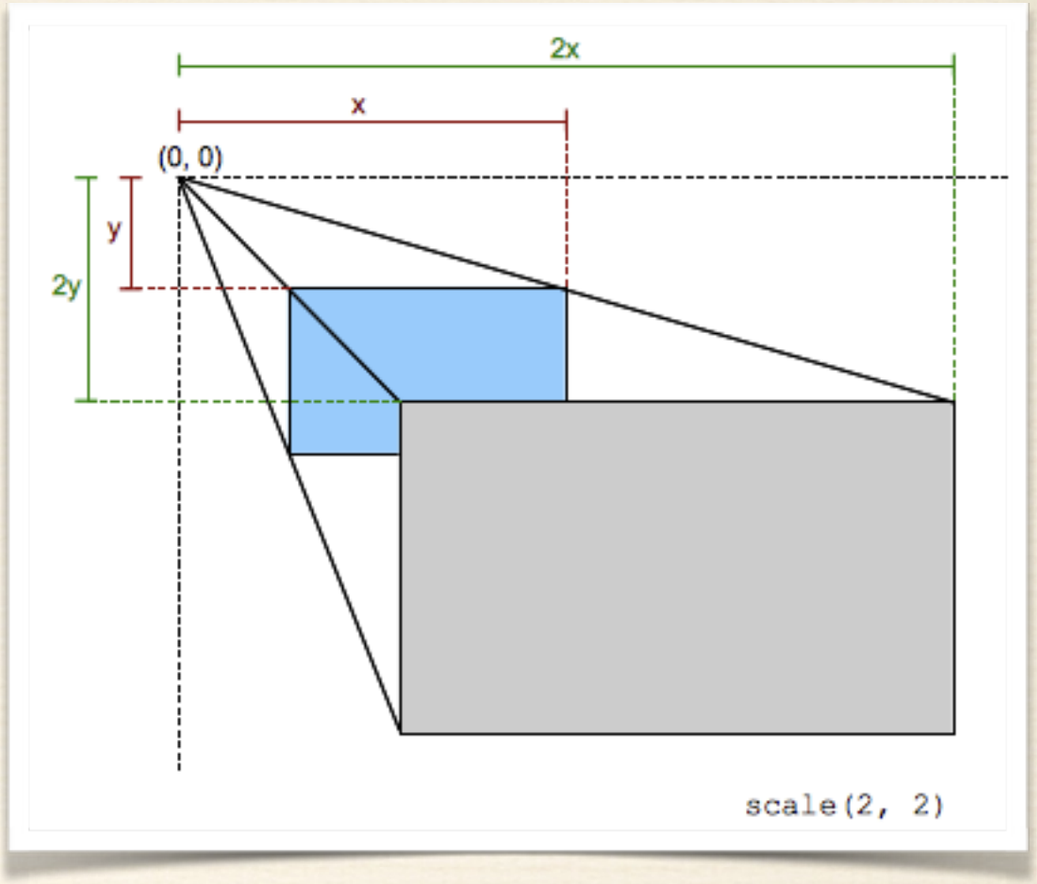
http://cocoadevcentral.com/articles/imgs/matrix001.png



http://mathworld.wolfram.com/images/eps-gif/Trigonometry_1001.gif



Scaling



- Increases the size of an object's **x** (width) or **y** (height)

- **scale**(sX), **scale**(sX, sY)

- **scaleX**(sX), **scaleY**(sY), **scaleZ**(sZ) - use convention **scale**<X,Y, Z>(s)

- **scale3d**(sX, sY, sZ)

http://www.sitepoint.com/advanced-css3-2d-and-3d-transform-techniques/

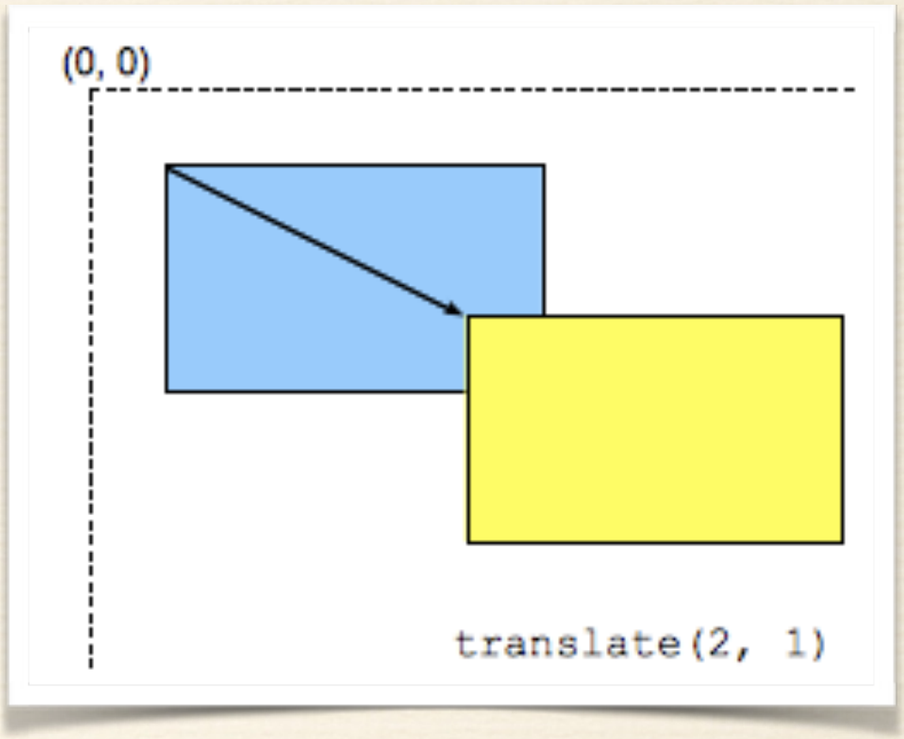
Translations

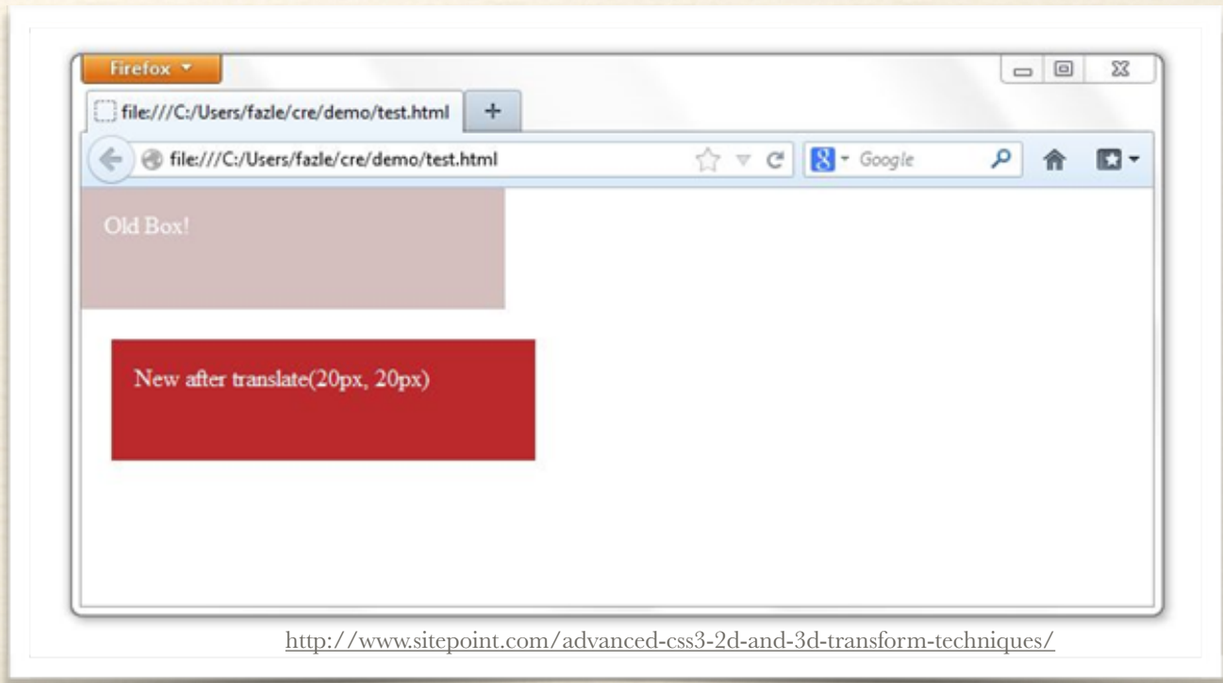
- Shift an object X, Y and/or Z units in some direction relative to where it is currently.

- **translate**(tx,ty),

- **translate3d**(tx, ty, tz)

- **translate**<X, Y, Z>(t)





http://www.sitepoint.com/advanced-css3-2d-and-3d-transform-techniques/

Rotations

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

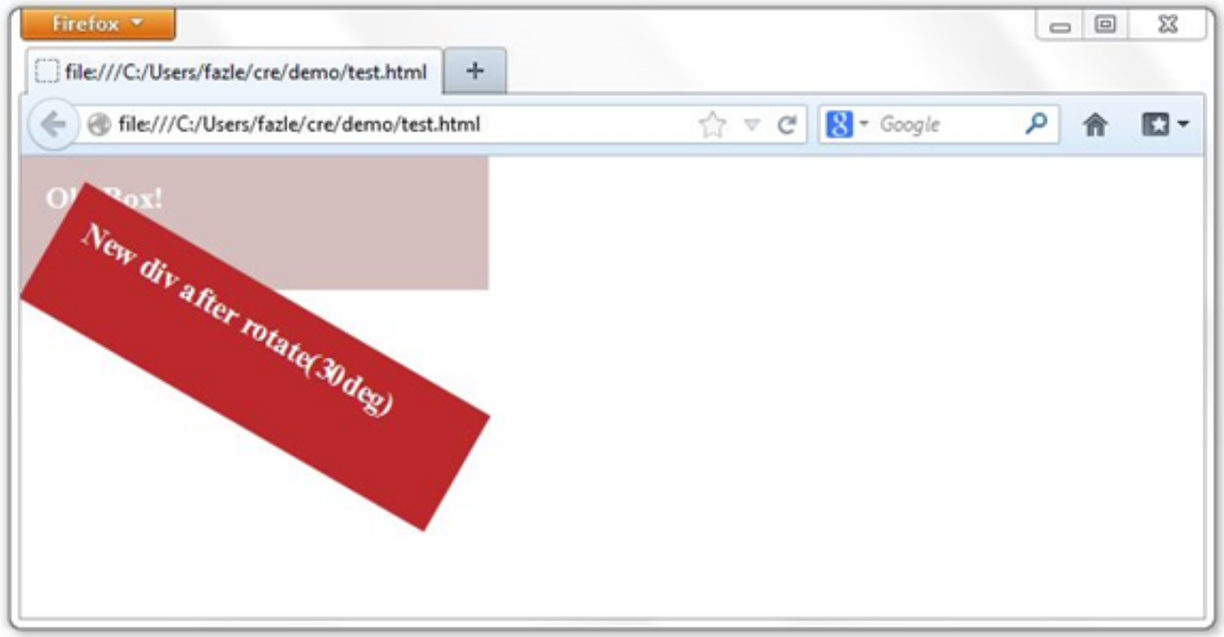
$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- **rotate**(<N>**deg**) rotates clockwise N degrees (right handed coordinate system)

- **rotate3d**(rx, ry, rz, Ω) where Ω is the degrees of rotation you want to perform.

- **rotate**<X,Y,Z>(Ω)



http://www.sitepoint.com/advanced-css3-2d-and-3d-transform-techniques/

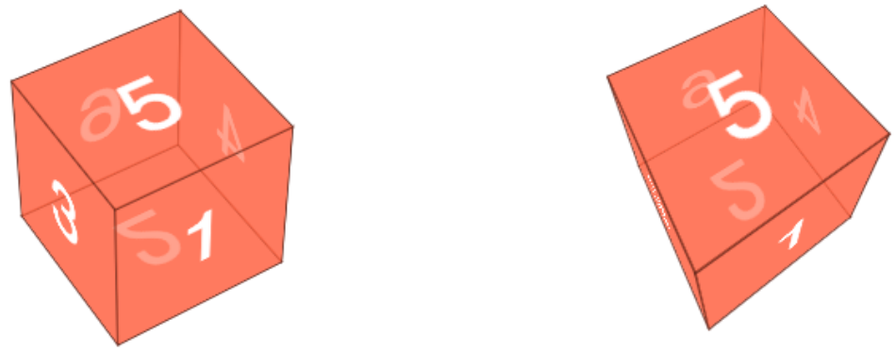
CSS Perspective

- Perspective: In a nutshell elements far from the user (via the z-axis) appear smaller. Elements closer to the user appear larger.

- **transform: perspective(1000px)** gives an element depth

- **perspective: 1000px;** gives the elements children a sense of depth

perspective: 1000px **perspective: 250px**



http://css-tricks.com/almanac/properties/p/perspective/

~ Demo Time ~

Kinetic Scrolling

$$y(t) = \hat{y} - Ae^{-t/\tau}$$

- In a nutshell JS is used to perform all the mathematical work and the final result which is simply a x position is added to the slider's container.
- The builtin **requestAnimationFrame** method is used to drive the render.
- Monitor instantaneous velocity as user is dragging finger across screen.
- Uses Exponential Decay to slow down to a stop
- This simulates an **overdamped** spring/mass system - "The system returns (exponentially decays) to equilibrium without oscillating." - wikipedia

http://ariya.ofilabs.com/2013/11/javascript-kinetic-scrolling-part-2.html

Demo Explanations Cont'd

- Create 5 vertical divs that contain 60 pixel "blinds" (using background-image and background-position) of the original image (**shifted on the X-axis at intervals of 60px**).

- When user click on the image a class is added which performs a transition to rotate those "blinds"

- When the user clicks again the class is removed toggling the transition to rotate the blinds back into their original position.

- AngularJS is used to modularize the code to be able to reuse it to make multiple cards.

Some Other Resources

- Animate.css - http://daneden.github.io/animate.css/
- Cool UI Design Tutorials - http://tympanus.net/codrops/
- CSS-Tricks - http://css-tricks.com/
- Front-end Best Practices - http://isobar-idev.github.io/code-standards/

QUESTIONS

michael.moncada@utoronto.ca

mikej.clark@utoronto.ca

https://github.com/mikecules/TKF2014