IN4MATX 133: User Interface Software

Lecture 24: Advanced CSS and Visual Design

Professor Daniel A. Epstein
TA Jamshir Goorabian
TA Simion Padurean

Today's goals

By the end of today, you should be able to…

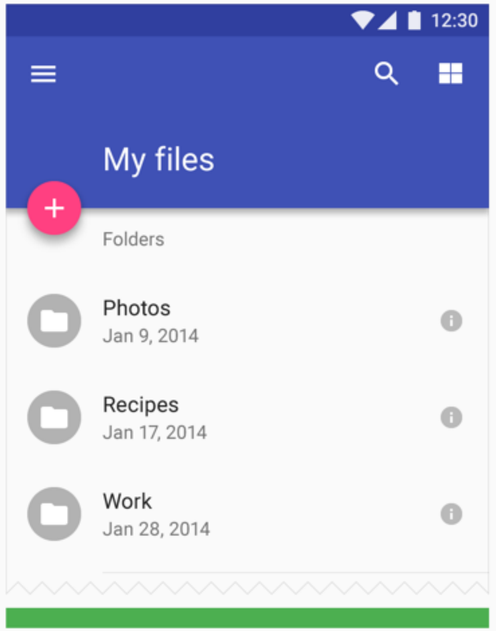
- Follow a few high-level principles for good visual design
- Implement transitions, transforms, and animations in CSS
- Describe situations where these advanced features both add and detract from the user experience

General visual design considerations

Colors

- Don’t be annoying
- Use color to create focal points
- Use bold colors carefully
- Use appropriate colors for the content and audience
- Colors can give meaning to a design
- Accessibility

Although the meaning of color can differ from person to person, some emotive qualities have been commonly linked to certain colors. For example, blues and greens bring to mind the coolness of water; fiery reds and oranges add warmth to a design.



Do.

Use the accent color for your primary action button and components like switches or sliders.

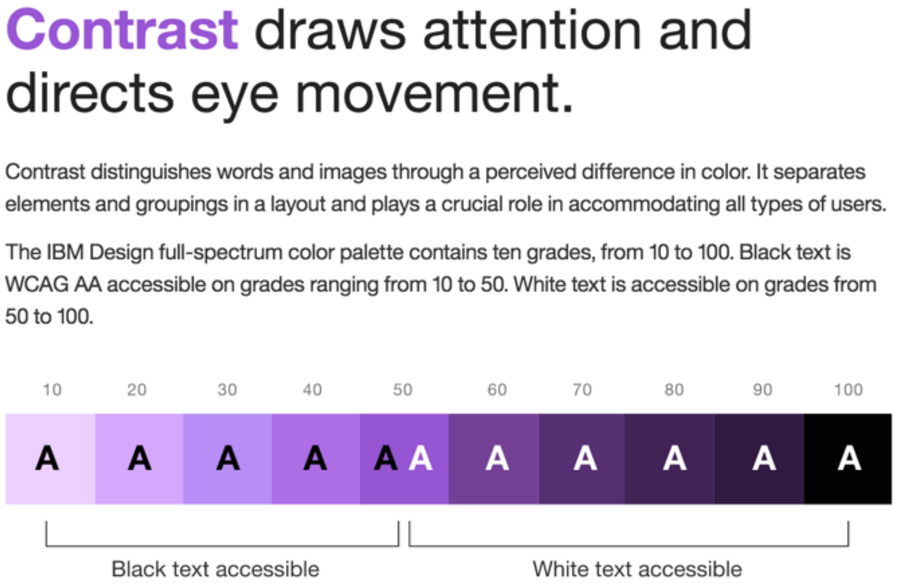
Don't.

Don't use the accent color for app bars or larger areas of color. Avoid using the same color for the floating action button and the background.

https://www.google.com/design/spec/style/color.html#color-ui-color-application

Contrast

- Must differentiate foreground from background
- Think about usability and accessibility
- Consider the context (e.g., at night)
- Light text on dark backgrounds, dark test on light
- You *can* use shadows



https://www.ibm.com/design/language/framework/visual/color#contrast

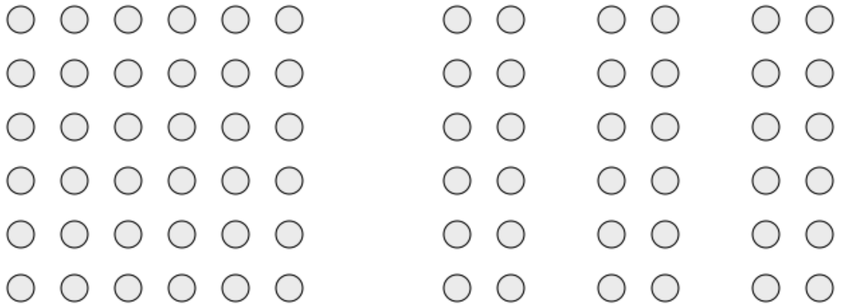
Typography

- Use no more than 3 fonts, 2 is better
- Pair a serif with a sans-serif
  - Use one for headings, the other for body
- Use size and weight to create hierarchy
  - Make your text “scannable”

https://medium.com/gravitdesigner/typography-elements-everyone-needs-to-understand-5fdea82f470d

Spacing and placement

- Gestalt principles: nearby items are thought of as grouped together
- Whitespace is a useful tool for layout
- Align items as much as possible



Lorem

Lorem ipsum dolor sit amet, consectetur adipiscing elit,sed do eiusmod tempor incididunt ut labore et dolore magnaaliqua. Ut enim ad minim veniam, quis nostrud exercitationullamco laboris nisi ut aliquip ex ea commodo consequat.Duis aute irure dolor in reprehenderit in voluptate velitesse cillum dolore eu fugiat nulla pariatur. Excepteur sintoccaecat cupidatat non proident, sunt in culpa qui officiadeserunt mollit anim id est laborum.

Lorem

Lorem ipsum dolor sit amet, consectetur adipiscing elit,sed do eiusmod tempor incididunt ut labore et dolore magnaaliqua. Ut enim ad minim veniam, quis nostrud exercitationullamco laboris nisi ut aliquip ex ea commodo consequat.Duis aute irure dolor in reprehenderit in voluptate velitesse cillum dolore eu fugiat nulla pariatur. Excepteur sintoccaecat cupidatat non proident, sunt in culpa qui officiadeserunt mollit anim id est laborum.

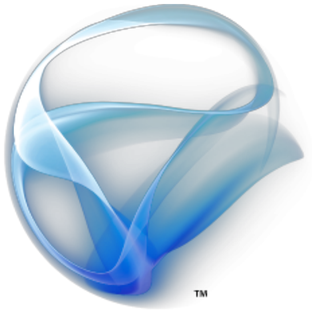
https://medium.com/gravitdesigner/typography-elements-everyone-needs-to-understand-5fdea82f470d

Don’t hesitate to steal design ideas from online examples…
You have to start somewhere

Animations in CSS

Flash, Silverlight, and HTML Canvas

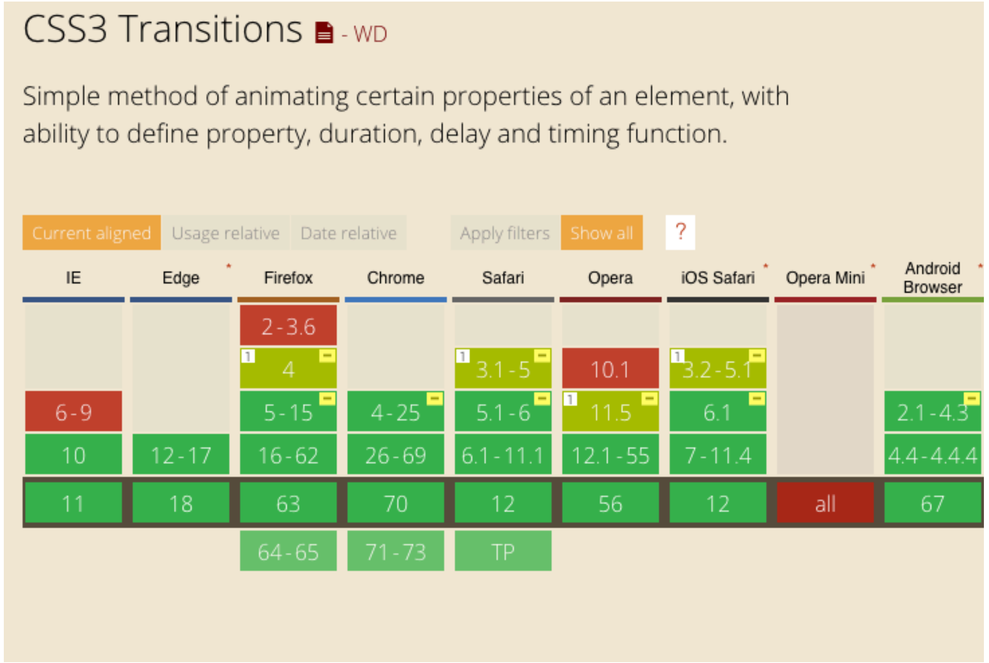
- Added animations to the web
  - Used for games and media playback
- Came with a lot of other bad things
  - Security vulnerabilities
  - Accessibility concerns
- Canvas is still in use,
  avoids some of the concerns
  - SVG has similar functionality



Microsoft®
Silverlight™

CSS animations

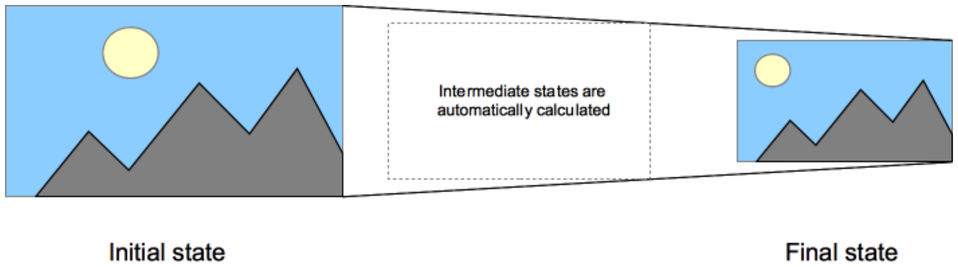
- Animations are still useful
  - Were added to the CSS standard around 2011
- Supported in all modern browsers
- Three types of animations
  - Transitions
  - Transforms
  - Animations



https://caniuse.com/#search=css%20transition

Transitions

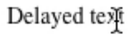
- Used to change a CSS property in response to an action
- Transitions can specify four values:
  - Property: what CSS properties should change
  - Duration: how long the transition should take
  - Timing function (easing): how the value should change (linearly, non-linearly, etc.)
  - Delay: when the transition should trigger



https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Transitions/Using_CSS_transitions

Transitions

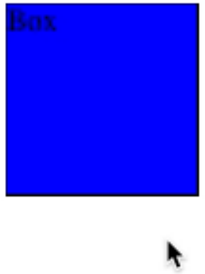
```css
#delay {
  font-size: 14px;
  transition-property: font-size, color;
  transition-duration: 2s;
  transition-delay: 1s;
  transition-timing-function: ease;
}

#delay:hover {
  font-size: 48px;
  color: #ff0000;
}
```



https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Transitions/Using_CSS_transitions

Transitions

```
.box {
    border-style: solid;
    border-width: 1px;
    display: block;
    width: 100px;
    height: 100px;
    background-color: #0000FF;
    transition: width 2s, height 2s, background-color 2s;
}

.box:hover {
    background-color: #FFCCCC;
    width: 200px;
    height: 200px;
}
```



https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Transitions/Using_CSS_transitions

Transforms

- Lets you rotate, scale, skew, or translate an element
- Can only be used with elements following the box model
  - e.g., cannot transform absolutely-positioned elements
- Can be used statically or in response to an action
  - (e.g., not necessarily animated)

https://www.w3schools.com/css/css3_2dtransforms.asp
https://developer.mozilla.org/en-US/docs/Web/CSS/transform

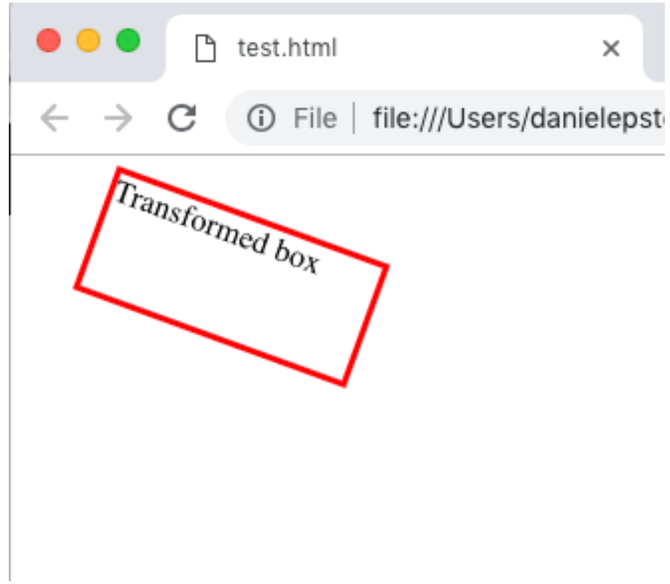
Transforms

- Four types of transforms
  - Translate: move around (x, y)
  - Rotate: spin around
  - Scale: change size
  - Skew: “lean” along an axis
- Matrix can be used to succinctly combine the above
  - Like in math (linear algebra)

https://www.w3schools.com/css/css3_2dtransforms.asp
https://developer.mozilla.org/en-US/docs/Web/CSS/transform

Transforms

```
.box {
  border: solid red;
  transform: translate(30px, 20px) rotate(20deg);
  width: 140px;
  height: 60px;
}
```



https://www.w3schools.com/css/css3_2dtransforms.asp

https://developer.mozilla.org/en-US/docs/Web/CSS/transform

Transforms

```
.box {
    border-style: solid;
    border-width: 1px;
    display: block;
    width: 100px;
    height: 100px;
    background-color: #FFFFFF;
    transition: width 2s, height 2s,
    background-color 1s, transform 1s;
}

.box:hover {
  transform: translate(30px, 20px) rotate(20deg);
    background-color: #FFCCCC;
    width: 200px;
    height: 200px;
}
```

https://www.w3schools.com/css/css3_2dtransforms.asp

https://developer.mozilla.org/en-US/docs/Web/CSS/transform

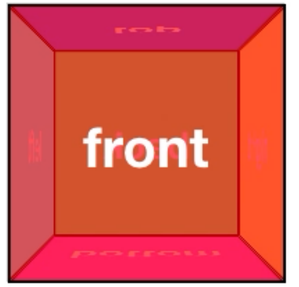
Transforms

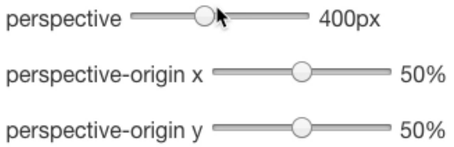
https://www.w3schools.com/css/css3_2dtransforms.asp

https://developer.mozilla.org/en-US/docs/Web/CSS/transform

3D Transforms

- The same as 2D, but in 3D (flat projection, not virtual reality)
- Key CSS property: `perspective`
  - How far away from the screen is the object
  - “Into the screen” (z-axis)
  - `perspective-origin`: vanishing point
- 3D transforms get math-y quickly
  - The demo is extremely helpful

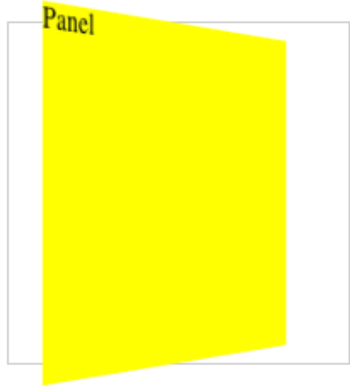




https://3dtransforms.desandro.com/

3D Transforms

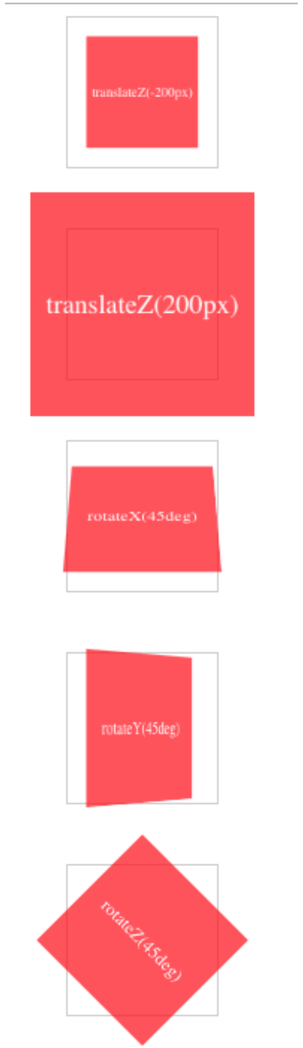
```
.scene {
  width: 200px;
  height: 200px;
  border: 1px solid #CCC;
  margin: 40px;
  /* perspective property */
  perspective: 600px;
}

.panel {
  width: 100%;
  height: 100%;
  background: yellow;
  transform: rotateY(45deg);
}
```



https://3dtransforms.desandro.com/

3D Transforms

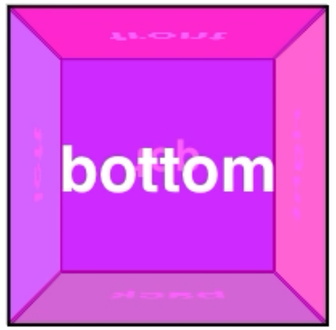
```
.panel--translate-neg-z {
  transform: translateZ(-200px);
}

.panel--translate-pos-z {
  transform: translateZ(200px);
}

.panel--rotate-x {
  transform: rotateX(45deg);
}

.panel--rotate-y {
  transform: rotateY(45deg);
}

.panel--rotate-z {
  transform: rotateZ(45deg);
}
```



https://3dtransforms.desandro.com/

3D Transforms

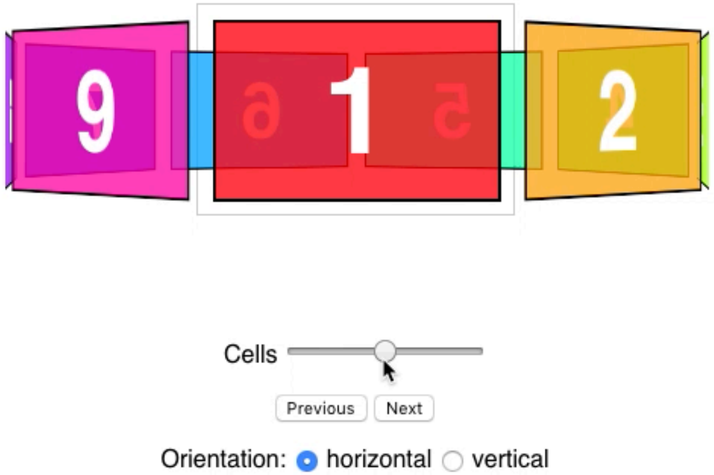
```
.cube.show-front  { transform: translateZ(-100px) rotateY(   0deg); }
.cube.show-right  { transform: translateZ(-100px) rotateY( -90deg); }
.cube.show-back   { transform: translateZ(-100px) rotateY(-180deg); }
.cube.show-left   { transform: translateZ(-100px) rotateY(  90deg); }
.cube.show-top    { transform: translateZ(-100px) rotateX( -90deg); }
.cube.show-bottom { transform: translateZ(-100px) rotateX(  90deg); }


.cube__face--front  { transform: rotateY(  0deg) translateZ(100px); }
.cube__face--right  { transform: rotateY( 90deg) translateZ(100px); }
.cube__face--back   { transform: rotateY(180deg) translateZ(100px); }
.cube__face--left   { transform: rotateY(-90deg) translateZ(100px); }
.cube__face--top    { transform: rotateX( 90deg) translateZ(100px); }
.cube__face--bottom { transform: rotateX(-90deg) translateZ(100px); }
```



○ front ○ right ○ back ○ left ○ top ◉ bottom

https://3dtransforms.desandro.com/

3D Transforms



https://3dtransforms.desandro.com/

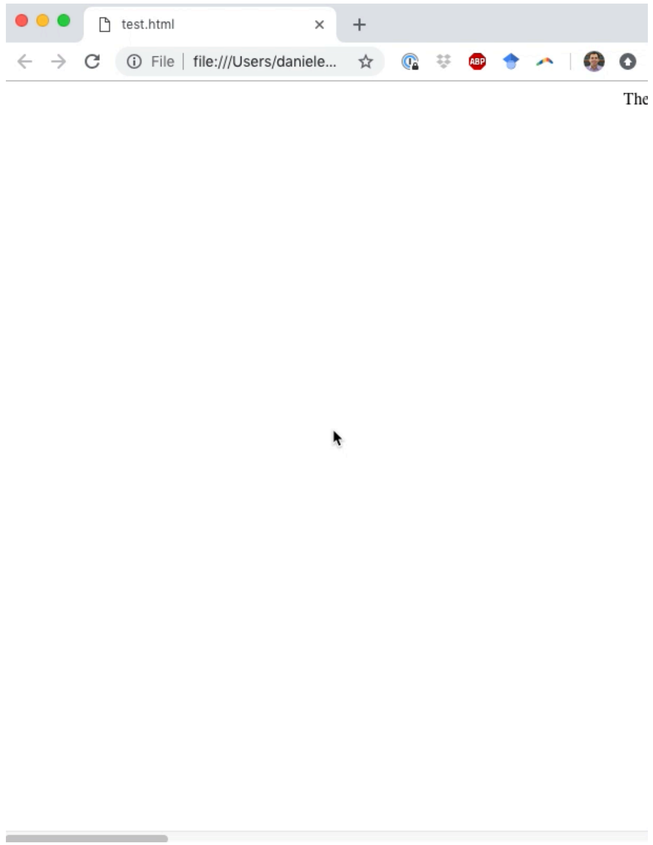
Animations

- Animations consist of two components:
  - A style describing the animation (e.g., name, duration, timing function)
  - Keyframes that indicate different states of the animation style
- Could instead be done in JavaScript
  - But CSS animations are optimized for graphics cards
  - Usually means smoother animations
  - But there are good JavaScript libraries

https://blog.bitsrc.io/11-javascript-animation-libraries-for-2018-9d7ac93a2c59
https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations

Animations

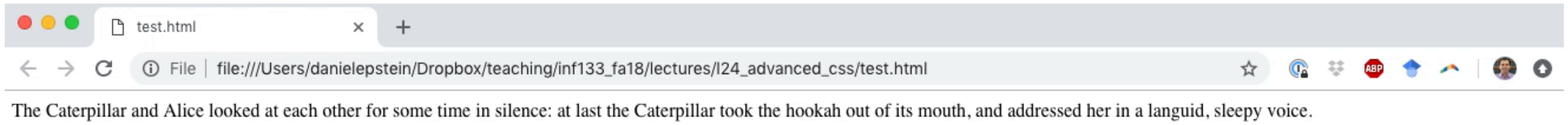
```
p {
  animation-duration: 3s;
  animation-name: slidein;
}

@keyframes slidein {
  from {
    margin-left: 100%;
    width: 300%;
  }

  to {
    margin-left: 0%;
    width: 100%;
  }
}
```



https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations

Animations

```
p {
  animation-duration: 3s;
  animation-name: slidein;
}

@keyframes slidein {
  from {
    margin-left: 100%;
    width: 300%;
  }

  75% {
    font-size: 300%;
    margin-left: 25%;
    width: 150%;
  }

  to {
    margin-left: 0%;
    width: 100%;
  }
}
```



https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations

When should animations be used?

Animations

- Drawing attention to specific items on the page
  - Create a subtle focal point
- Bringing delight to users without impacting usability
- Subtle effects to alert users to changes on the page

Animations

Good uses for animations

- Simple hover effects for links
- Bring in content while scrolling
- Enlarging images or buttons on hover
- Shaking an input field when there’s an error
- Creative loading bars or pages

https://blog.prototypr.io/6-animation-guidelines-for-ux-design-74c90eb5e47a

Animations

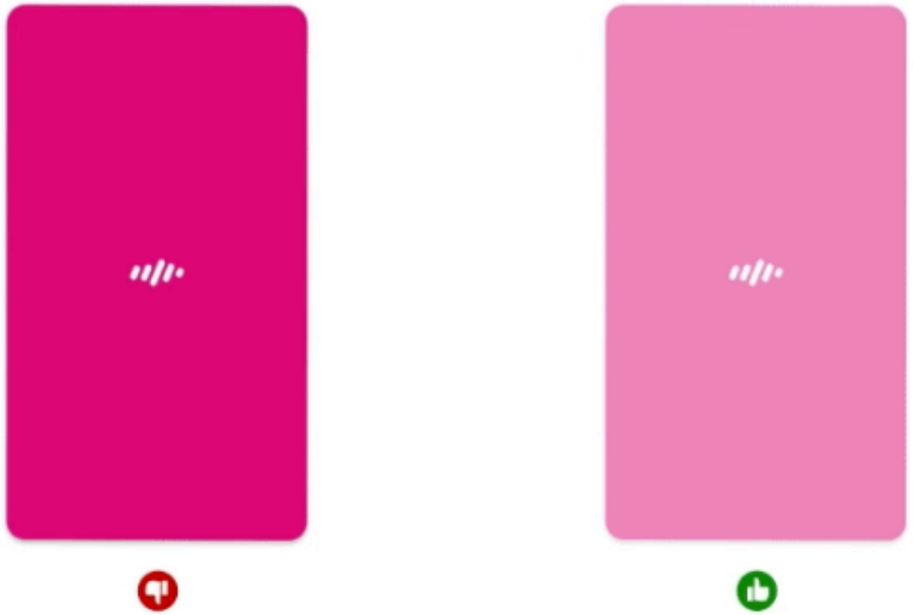
Soften harsh cuts

https://uxdesign.cc/the-ultimate-guide-to-proper-use-of-animation-in-ux-10bd98614fa9

https://blog.prototypr.io/6-animation-guidelines-for-ux-design-74c90eb5e47a

Animations

Provide context



https://uxdesign.cc/the-ultimate-guide-to-proper-use-of-animation-in-ux-10bd98614fa9

https://blog.prototypr.io/6-animation-guidelines-for-ux-design-74c90eb5e47a

Animations

Provide orientation

https://uxdesign.cc/the-ultimate-guide-to-proper-use-of-animation-in-ux-10bd98614fa9

https://blog.prototypr.io/6-animation-guidelines-for-ux-design-74c90eb5e47a

Animations

Make content feel alive

https://uxdesign.cc/the-ultimate-guide-to-proper-use-of-animation-in-ux-10bd98614fa9

https://blog.prototypr.io/6-animation-guidelines-for-ux-design-74c90eb5e47a

Question

Which animation, if either, is preferable?

A. Both animations are fine

B. Neither animation is good

C. The animation on the left is better

D. The animation on the right is better

E. I can see arguments for either

Animations

Use proper speed (200-500 ms)

https://uxdesign.cc/the-ultimate-guide-to-proper-use-of-animation-in-ux-10bd98614fa9

https://blog.prototypr.io/6-animation-guidelines-for-ux-design-74c90eb5e47a

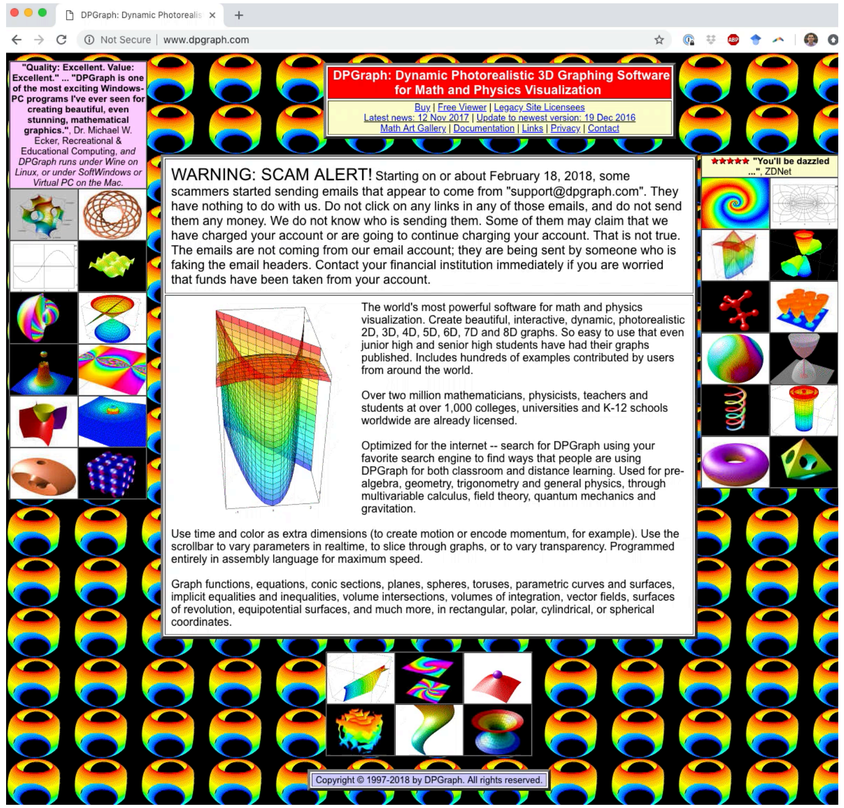
Animations

Bad uses for animations

- 360 degree rotating link text
- Continual movement of multiple objects
- Moving objects away from the mouse on hover
- Hiding important content in an animation
- Anyone remember `<marquee>` or `<blink>` tags?
  - These also violated code separation principles

Animations

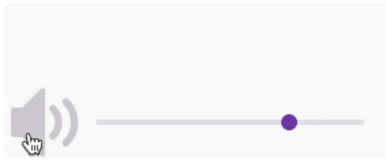
DPGraph



http://www.dpgraph.com/

Animations

Bad volume sliders



https://www.theverge.com/tldr/2017/6/9/15768800/reddit-worst-volume-sliders-ui-design

Vendor prefixes

- Used to specify browser-specific implementation of a feature that hasn't (yet) been fully adopted
- Growing to be less of an issue, but they show up in many animation examples
  - Mozilla says they're necessary for pre-2016 browsers

```
.rounded {
   -moz-border-radius: 10px; /* Mozilla (Firefox) */
   -webkit-border-radius: 10px; /* Webkit (Chrome, Safari) */
   -o-border-radius: 10px;  /* Opera */
   /* -ms-property for Internet Explorer */
   border-radius: 10px; /* the spec! */
}

/* NOTE: Outdated example! Border-radius is supported */
```

https://github.com/postcss/autoprefixer

Today's goals

By the end of today, you should be able to…

- Follow a few high-level principles for good visual design
- Implement transitions, transforms, and animations in CSS
- Describe situations where these advanced features both add and detract from the user experience

IN4MATX 133: User Interface Software

Lecture 24: Advanced CSS and Visual Design

Professor Daniel A. Epstein
TA Jamshir Goorabian
TA Simion Padurean