Integrating CSS Animation

Due: By next class

Grading: 20 points possible

This exercise focuses on integrating animated effects into Breakroom driven by CSS3 animation properties. Animated effects like transitions and motion have traditionally been the domain of Flash or JavaScript. But with CSS3, there a slew of animation related properties available. The CSS3 animation properties used in this exercise are explained in detail here. The last part of this exercise will have you add transitions between page views to soften the speed with which AngularJS loads partials, which can be kind of jarring! This will require the addition of a new AngularJS resource, an animation library documented below, and CSS-driven transitions between page views.

Exercise details:

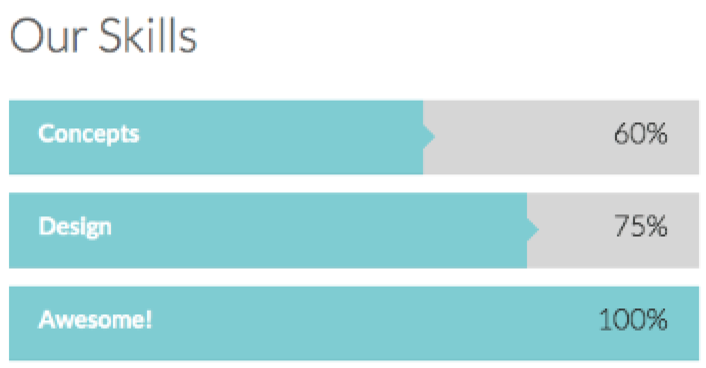
- Update markup with new classes
- Add CSS-driven animations
- Add new Sass partial
- Add transitions to page views
- Sass challenge!

Directions

1. Launch Scout and pressing the play button.

CSS animated bar graph

You will begin by adding the markup and CSS to create an animated bar graph that gradually fills according to the stated percentage value.



2. Replace your current markup for the 3 divs under “Our Skill” with the following markup:

```
<h2>Our Skill</h2>
<div class="progress-bar bar-one">
  <div class="progress">
    <span class="field">Concepts</span>
    <span class="field-value fade-in">60%</span>
    <div style="width: 60%;" class="bar"></div>
  </div>
</div>
  <div class="progress-bar bar-two">
    <div class="progress">
      <span class="field">Design</span>
      <span class="field-value fade-in">75%</span>
      <div style="width: 75%;" class="bar"></div>
  </div>
</div>
<div class="progress-bar bar-three">
  <div class="progress">
    <span class="field">Awesome!</span>
    <span class="field-value fade-in">100%</span>
      <div style="width: 100%;" class="bar"></div>
  </div>
</div>
```

Note the inline *style* attribute that sets a width value for the “bar” div. You can adjust that value to create different bar widths.

Create new partial

3. Create a new blank file and save it as “_progress-bar.scss” in your local *scss* folder.

Download Gist

4. Go to the following URL and grab the Github Gist:
https://gist.github.com/modified65/79db674c3d5751cf3638

5. Next, paste the code into the blank partial and save it.

6. With *_progress-bar.scss* open, and take a look at the CSS rules. Most define progress bar styling but there a series of CSS3 animation properties that are explained in detail here: https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Using_CSS_animations

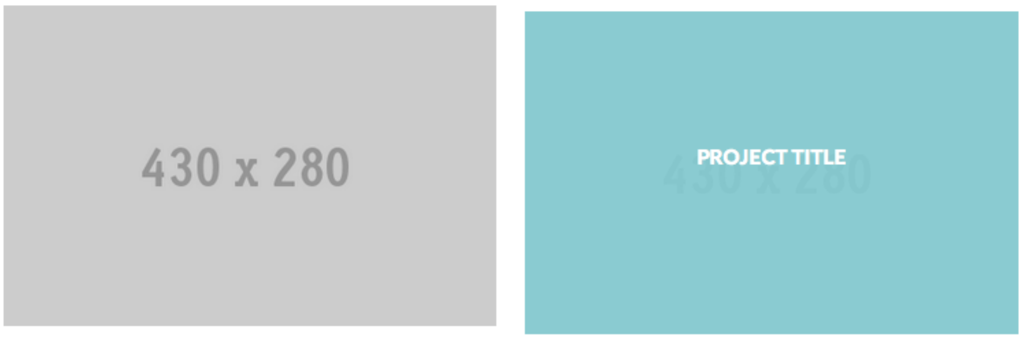
Include progress-bar partial

7. Open *style.scss* and add *_progress-bar.scss* to the bottom of the list of partials.

```
@import url(http://fonts.googleapis.com/css?family=Lato:100,300,400,700,900);
@import "variables";
@import "mixins";
@import "typography";
@import "patterns";
@import "progress-bar";
```

Portfolio rollovers

For the Portfolio page you are going to set up an interaction where rolling over a thumbnail image will hide the image and replace it with a semi-transparent turquoise background with white text. The background should transition into view with the text rising up into place.



8. Open *portfolio.html*. Wrap the *img* and link tags with a new div and add a *.project* class.

9. Within each portfolio link, wrap the title with nested *span* tags. The outer *span* tag gets a “project-title” class.

```
<div class="col-sm-4">
  <div class="project">
    <img src="images/portfolio1.jpg" alt="..." class="img-responsive">
    <a href="#"><span class="project-title"><span>Project Title</span></span></a>
  </div>
```

```
</div>
```

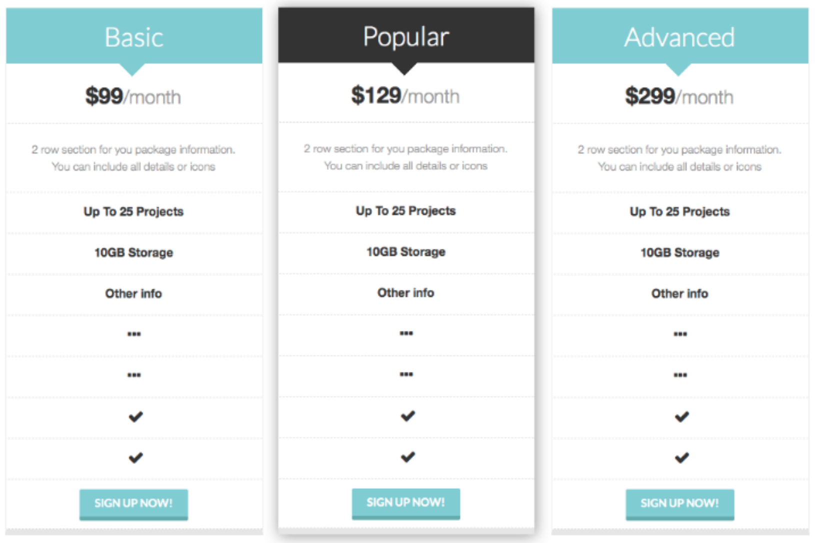
10. Open *_patterns.scss* and add the following:

```
.project {
  overflow: hidden;
  position: relative;
  margin-bottom: 30px;
  a {
    @include contentbox-style($turquoise, $white);
    text-align: center;
    text-transform: uppercase;
    font-weight: bold;
    opacity: 0;
    position: absolute;
    top: 0;
    left: 0;
    bottom: -200px;
    right: 0;
    z-index: 100;
    display: table-cell;
    vertical-align: middle;
  }
  a:hover {
    bottom: 0;
    opacity: .9;
    transition: opacity 0.3s ease-in-out 0s, bottom 0.4s cubic-bezier(0.25,0.5,0,1)
0s;
  }
}

.project-title {
    display: table;
    width: 100%;
    height: 90%;
    span {
      vertical-align: middle;
      display: table-cell;
    }
    }
```

Services

The Services page contains three price plans that will each generate a box shadow when hovering over any area of a given price plan.


Basic
$99/month
2 row section for you package information. You can include all details or icons
Up To 25 Projects
10GB Storage
Other info
...
...
✓
✓
SIGN UP NOW!
Popular
$129/month
2 row section for you package information. You can include all details or icons
Up To 25 Projects
10GB Storage
Other info
...
...
✓
✓
SIGN UP NOW!
Advanced
$299/month
2 row section for you package information. You can include all details or icons
Up To 25 Projects
10GB Storage
Other info
...
...
✓
✓
SIGN UP NOW!

11. Open *services.html*. Within each of the 3 columns, wrap the *h2* and *ul* tags with a new div and add a *.price-plan* class.

```
<div class="price-plan">
  <h2 class="titlebox text-center">Basic</h2>
  <ul class="list-unstyled text-center price-list">
  ...
```

12. Next, open *_patterns.scss* and add the following:

```
.price-plan:hover {
  -moz-box-shadow: 0 0 20px $gray;
  -webkit-box-shadow: 0 0 20px $gray;
  box-shadow: 0 0 20px $gray;
}
.price-plan:hover .titlebox {
  background-color: $black;
  &:after {
    border-color: $black transparent;
  }
}
```

AngularJS transitions between pages

If you test your app at this point, the partials should load really quick. In order to soften the page loads, we can add transitions using AngularJS. AngularJS adds animation options via its animation library. Our use of the animations options available to us will be pretty straightforward but there's a lot you can do with it.

13. Open *index.html*. At the bottom of the document, add the following new link to the AngularJS animation resource. Place it right above the link to *scripts.js*.

```
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.25/angular-animate.min.js"></s
cript>
    <script type="text/javascript" src="js/app.js"></script>
```

14. Next, open *app.js* and add the following reference to the "ngAnimate" module:

```
    angular.module('breakroom', ['ngRoute','ngAnimate'])
    .config(breakroomRouter);
```

15. Next, create a new Sass file and save it as "_ngview.scss".

16. In *_ngview.scss*, add the following Sass:

```
    .ng-enter {
      -webkit-transition: 1s linear all;
      -moz-transition: 1s linear all;
      transition: 1s linear all;
      opacity: 0;
    }

    .ng-enter-active,
    .ng-leave {
      opacity: 1;
```

```
}

.ng-leave {
  opacity: 1;
}

.ng-leave-active {
  opacity: 0;
}
```

17. Next, open *style.scss* and add the following at the bottom of the file:

```
@import "ngview";
```

Update markup

18. Open *index.html* and add a class attribute with value “view”.

```
<main class="view" ng-view>
```

Test using local server

19. Use Brackets preview to launch Chrome in a local server environment or by running an alternative local server.

Sass Challenge!

Throughout this exercise browser prefixes (-moz-, -webkit-) are used repeatedly for the CSS3 properties. That’s exactly the kind of repetition Sass is designed to address!

20. Leveraging what you have learned thus far about Sass, figure out a way to integrate browser prefixes more efficiently into your CSS and make your CSS DRYer!

Transfer project folder to web host

21. Use an FTP client like Fugu or Filezilla to transfer your project folder to your remote account. You don’t need to upload your *scss* folder. That can stay on your local drive.

Test after uploading

22. Launch a web browser, go to your project URL, and make sure that your loads as expected.

Notify instructor when exercise is complete

23. Email me the URL to your remote *breakroom* project at chris.clark@mail.ccsf.edu.

For next class

- Bootstrap Navbar (familiarize yourself with it)
- Bootstrap JavaScript (we'll use the Carousel in particular)