MODULES

MODULES

A container for the different parts of an app including controllers, services, filters, directives. Modules are the building blocks that apps are made of. A module's primary job is to serve as place for the pieces of an app to be registered.

MODULE EXAMPLE

We've seen this before. Let's look in more detail

```javascript
var app = angular.module('myCoolModule', []);
// A module with no dependencies.

var app = angular.module('myOtherCoolModule', ['ngRoute']);
// A module with one dependencies
```

MODULE EXAMPLE

The difference between creation vs. retrieval

```
var app = angular.module('myCoolModule', []);
// This syntax creates a module. It's sort of like a setter*.

var app = angular.module('myCoolModule');
// This syntax retrieves an already created module. It's a getter.

var app = angular.module('myOtherCoolModule');
// This will throw an error because this module hasn't been created.
```

MODULES

Once you have defined a module, you can add other pieces to it such as controllers, services, custom filters, or directives.

MODULES

```javascript
var app = angular.module('myCoolModule', []);

app.controller('myCoolController', function(){
  // cool controller stuff
});

app.factory('myCoolFactory', function(){
  // cool factory stuff
});
```

ROUTES

ROUTING

In a multipage site we can link to different pages using the same anchor tags and `href` attributes. For a single page application, we need a way to tell the browser to load different content. We can do this with routing.

ROUTING

Originally routing was built-in to Angular. Later, it was decided that it should be maintained as its own module and repository. So we must link to it and include it as a module dependency in order to use it.

IMPORTING NGROUTE

Like other stuff, you can download it or use a CDN

```html
<script src="lib/angular-route.js"></script>
```

```javascript
var app = angular.module('myModule', ['ngRoute']);
```

NGVIEW

`ngView` is a directive that goes with the `ngRoute` service. It will include a rendered template into the main layout (index.html). Every time the current route changes, the included view changes with it according to the configuration of the `$route` service.

ngView

```
<div ng-view=""></div>

<!-- or -->

<ng-view></ng-view>
```

CONFIGURING ROUTES

For our routes, we will add a `config` object to our module. Inside this config object we will inject the `$routeProvider` to define our routes and define which controllers are used with what views.

The `when()` function takes a route name as a string and an object with the route's properties such as `controller` and `templateUrl`.

The `otherwise` function defines what the router should do for unknown routes.

CONFIGURING ROUTES

```javascript
var app = angular.module('routingDemo', ['ngRoute']);

app.config(function($routeProvider) {
  $routeProvider
    .when('/greatness', {
      controller: 'routingController',
      templateUrl: 'greatness.html'
    })
    .when('/amazing', {
      controller: 'routingController',
      templateUrl: 'amazing.html'
    })
    .otherwise({ redirectTo: '/' });
});
```

CONFIGURING ROUTES

```html
<nav>
  <a href="#!/home">Home</a>
  <a href="#!/greatness">Greatness</a>
  <a href="#!/amazing">Amazing</a>
</nav>
```

VIEWS

VIEWS

A view is the visible part of the website (the DOM).
When using a router like `ngRoute`, views are usually
partial (incomplete) snippets of HTML that are
injected into the viewport as needed.

VIEWS

We can link to these snippets of html in our router configuation using the `templateUrl` property. We can also define templates inline using the `template` property but this is usually not recommended for anything but the simplest templates.

ONE LAST THING ABOUT ROUTES!

When using Angular's routing module, we need to run this application locally. To do so, we will use a very basic command-line server: http-server

https://www.npmjs.com/package/http-server
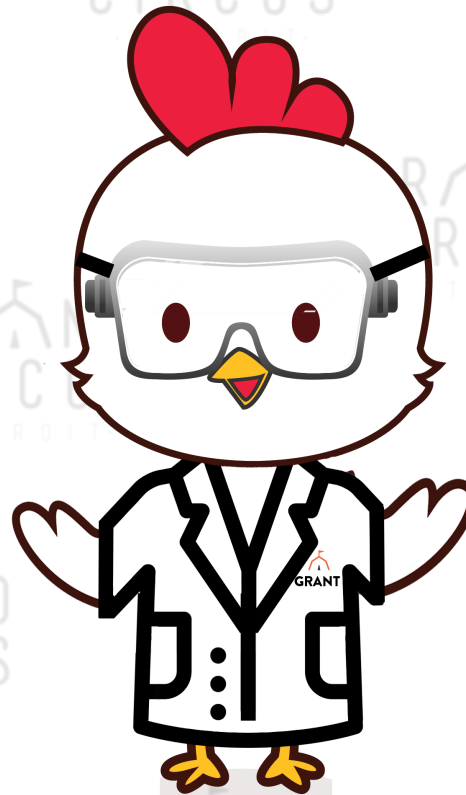
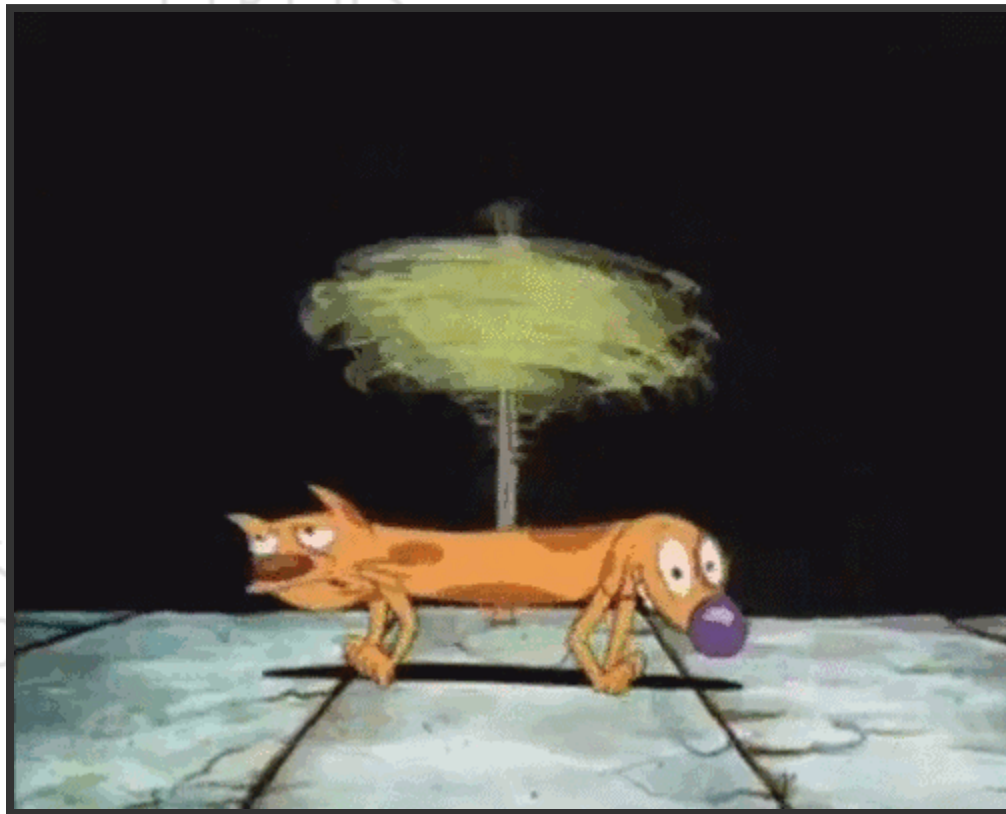We will walk through setting this up now.

HOMEWORK

From ng-book:

- Multiple Views and Routing 139-152
- Services: 161-176

LAB 14

CAT-DOG

CAT-DOG

YOU WILL NEED TO RUN THIS APP ON A SERVER!

```
// OSX
python -m SimpleHTTPServer [port]

// OSX or Windows
npm install http-server -g
http-server

// browse to localhost:8080 to view your content!
```

INSTRUCTIONS

Create a single page app with two views.

- Create a new project. (index.html, script.js, two partial views)
- Define a module & controller for this project.
- Remember to download and include `ngRoute`.
- Define two separate routes, and an `otherwise()` route.
- Set up a shell page with navigation for moving between both views.
- In one route, show a picture of a dog. The other should show a cat.

BONUS

- Bonus! Add two more routes that feature two more images of two more different species. i.e. cat-dog-ferret-walrus
- Bonus++ Add the ability to catch errors in your routes. If they user tries to visit a nonexistant page, they are redirected to the home page (cat).
- Double Bonus++ Add a custom 404 page. Suggestion: Fail Whale

PRE-GAME