STAYING SHARP with Angular.js

Services Level 3

$resource Section 4

STAYING SHARP with Angular.JS

Our Previous Note Service

So far we have only been using the built-in $http service to grab or change data.

note.js

```
angular.module("NoteWrangler")
.factory("Note", function NoteFactory($http) {
  return {
   all: function() {
     return $http({method: "GET", url: "/notes"});
   },
   find: function(id){
     return $http({method:"GET", url: "/notes" + id});
   },
   update: function(noteObj) {
     return $http({method: "PUT", url:"/notes", data: noteObj});
   },
   create: function(noteObj) {
     return $http({method: "POST", url:"/notes", data: noteObj});
   };
});
```

$http

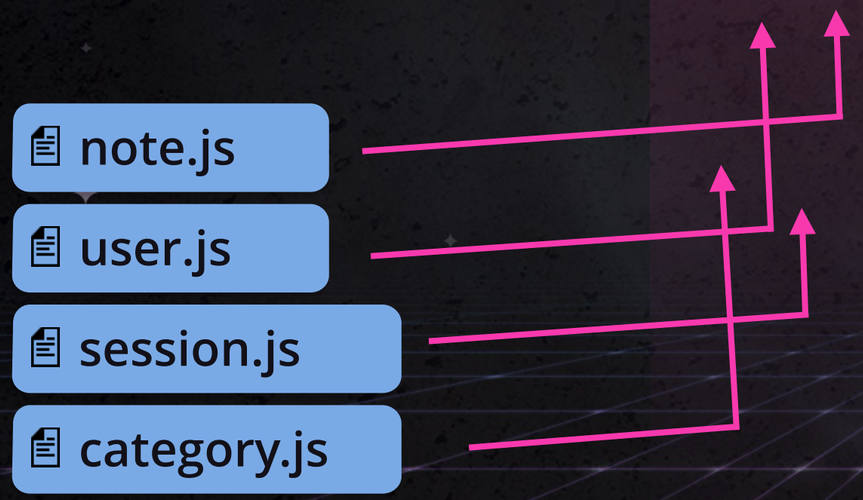
note.js

```
all:    function() {... }
find:   function(id){...}
update: function(noteObj) {...}
create: function(noteObj) {...}
```

Most Data Services Will Need These Methods

It's common that you'll have code that calls web services, and they will have these methods:

```
all:    function() { ... }
find:   function(id){...}
create: function(noteObj) {...}
                            ...}
```



Note Service Currently Using $http

Right now we are using the $http service — switching to $resource will clean this service up.

note.js

```
angular.module("NoteWrangler")
.factory("Note", function NoteFactory($http) {
  return {
   all: function() {
     return $http({method: "GET", url: "/notes"});
   },
   find: function(id){
     return $http({method:"GET", url: "/notes" + id});
   },
   update: function(noteObj) {
      return $http({method: "PUT", url:"/notes", data: note});
   },
   create: function(noteObj) {
      return $http({method: "POST", url:"/notes", data: note});
   };
});
```

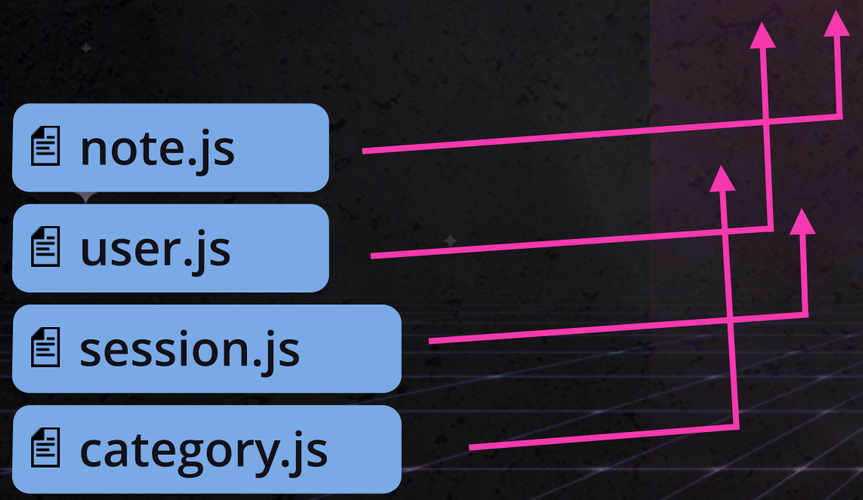
$http

The $resource Service Has These Built In

It's common that you'll have code that calls web services, and they will have these methods:

$resource includes these by default

```
all:    function() {... }
find:   function(id){...}
create: function(noteObj) {...}
                          ...}
```



Installing Angular Resource

The ngResource module is not included with the Angular core by default — we need to download it from code.angularjs.org and include it here: js/vendor/angular-resource.js

- Note Wrangler
  - index.html
  - > css
  - > js
  - > templates
  - vendor
    - angular-resource.js

Including ngResource in our app:

index.html

```
<script src="js/vendor/angular-resource.js"></script>
```

Including ngResource in Our App Module

We need to give our main application module access to ngResource so our whole app will have access to this service.

app.js

```
angular.module("NoteWrangler", ['ngRoute', 'Gravatar', 'ngResource'])
```

Injecting $resource Into Service

note.js

```
angular.module("NoteWrangler")
.factory("Note", function NoteFactory($resource) {
  return $resource("/notes/:id", {}, {});
});
```

We send in the URL of the API it will call for data.

Using $resource Instead of $http

note.js

```
angular.module("NoteWrangler")
.factory("Note", function NoteFactory($resource) {
  return $resource("/notes/:id", {}, {});
});
```

$resource

Provides similar functionality to all these lines

```
return {
  all: function() {
    return $http({method: "GET", url: "/notes"});
  },
  find: function(id){
    return $http({method:"GET", url: "/notes" + id});
  },
  create: function(noteObj) {
    return $http({method: "POST", url:"/notes", data: note});
  }
};
```

$http

Note Show With $resource

notes-show-controller.js

```
angular.module("NoteWrangler")
.controller("NotesShowController", function($scope, $routeParams, Note) {
  Note.find($routeParams.id).success(function(data) {
    $scope.note = data;
  });
});
```

old code

Replaced by this

```
$scope.note = Note.get({id: $routeParams.id})
```

new code

Note Show With $resource

notes-show-controller.js

```
angular.module("NoteWrangler")
.controller("NotesShowController", function($scope, $routeParams, Note) {
  $scope.note = Note.get({id: $routeParams.id})
});
```

new code

Note Index With $resource

To fetch all of the notes, we need to use the query method.

notes-index-controller.js

```
angular.module("NoteWrangler")
.controller("NotesIndexController", function($scope, $routeParams, Note) {
  Note.all().success(function(data) {
    $scope.notes = data;
  });

});
```

old code

```
$scope.notes = Note.query();
```

new code

Replaced by this

Note Index With $resource

To fetch all of the notes, we need to use the query method.

notes-index-controller.js

```
angular.module("NoteWrangler")
.controller("NotesIndexController", function($scope, $routeParams, Note) {
  $scope.notes = Note.query();
});
```

new code

Note Create With $resource

note-create-controller.js

```
angular.module("NoteWrangler")
.controller("NoteCreateController", function($scope, Note) {
  $scope.saveNote = function(note) {
    $scope.errors = null;
    $scope.updating = true;
    Note.create(note)
    .catch(function(note) {
        $scope.errors = [note.data.error];
    }).finally(function() {
        $scope.updating = false;
    });
  };
});
```

old code

```
$scope.note = new Note();
note.$save(note)
```

new code

Replaced by this

Note Create With $resource

note-create-controller.js

```
angular.module("NoteWrangler")
.controller("NoteCreateController", function($scope, Note) {
  $scope.note = new Note();
  $scope.saveNote = function(note) {
    $scope.errors = null;
    $scope.updating = true;
     note.$save(note)
     .catch(function(note) {
         $scope.errors = [note.data.error];
     }).finally(function() {
         $scope.updating = false;
     });
  };
});
```

Notice our 'catch' and 'finally' from earlier can still be used

Note Delete With $resource

The $resource DELETE method is used to DELETE a resource from a server.

note-delete-controller.js

```
angular.module("NoteWrangler")
.controller("NoteDeleteController", function($scope, Note) {
  $scope.deleteNote = function(note) {
    Note.$delete(note);
  };
});
```

```
<note-delete-button ng-click="deleteNote(note)">
```

There is also a substitute $remove method that does the same thing, but is helpful for IE browsers because delete is a reserved word.

$resource shortened our code

$resource has allowed us to simplify getting all notes, fetching one note, and creating new notes.

To get a single resource

```
Note.get({id: $routeParams.id})
```

To get all resources

```
Note.query();
```

To delete resources

```
Note.$delete()
```

To create a new resource

```
$scope.note = new Note();
note.$save();
```

What about updating resources?

Creating Custom $resource Methods

We can add our own custom functions to the object resource returns.

note.js

```
angular.module('NoteWrangler')
.factory('Notes', function NotesFactory($resource) {
  return $resource('/notes/:id', {}, {
    update: {
      method: "PUT"
    }
  });
});
```

Calling a Custom $resource Function

Using a custom function off of $resource is exactly the same as using a built-in function.

note-edit-controller.js

```
angular.module("NoteWrangler")
.controller("NoteEditController", function($scope, $routeParams, Note) {
  $scope.note = Note.get({id: $routeParams.id})
  $scope.updateNote = function(note) {
    $scope.errors = null;
    $scope.updating = true;

    note.$update().catch(function(note) {
      $scope.errors = [note.data.error];
    }).finally(function() {
      $scope.updating = false;
    });
 };
});
```