Rest Resources in some more depth

Databases Three

Otago Polytechnic
Dunedin, New Zealand

We know so far

- We can create a $resource object with a RESTful URL scheme.
- This lets us access a RESTful resource that presents a typical REST API.
- We can add custom methods to access other parts of our API.
- We need be be attentive to *Cross-origin resource sharing* (CORS) issues.

So, we can do this:

```
app.factory('User', function($resource) {
   return $resource('http://someurl/users/:id.json')});

 app.controller('UserController' function(User) {
 user = User.get({id: 1});

 this.getUser = function() {
     this.user = User.get(this.data.id);
 };
```

Creating a new user

```
this.createUser = function() {
    user = new User({name: this.data.name,
    email: this.data.email,
    password: this.data.password
    });
    user.$save();
};
```

Custom method for updating

We want an update method that performs an HTTP PUT. This is not a standard ngResource method.

```
app.factory('User', function($resource) {
   return $resource('http://someurl/users/:id.json', {},
     {update: {method:'PUT', url:'http://someurl/users/:id.json'}}
   )});
```

Then, in our controller, we can call the update method.

```
this.updateUser = function() {
    this.user.name = form.data.name;
    this.user.blurb = form.data.blurb;
    User.update({id: this.user.id}, this.user });
}
```

Deleting a user

```
this.deleteUser = function() {
    User.delete({id: this.user.id});
};
```