Custom Services with .factory()

Factory Design Pattern

Central place that produces new objects or functions

- Can produce any type of object, not just a singleton
- Can be used to produce dynamically customizable services

Factory vs. Service Confusion

.factory() is NOT just another way of creating the same service you can create with .service() but it CAN BE.

.service() is also a factory, but a much more limited one compared to .factory(). It's a factory that always produces the same type of service - a singleton, without an easy way to configure its behavior.

Register Service Factory Function

```
angular.module('app', [])
.controller('ctrl', Ctrl)
.factory('CustomService', CustomService);
```

Use this name to inject it into other services, controllers, etc.

Function that's expected to *produce* a service

Service Factory Function – Return Function

```
function CustomService() {
  var factory = function () {
    return new SomeService();
  };

  return factory;
}
```

Service Factory Function – Return Object Literal

```
function CustomService() {
  var factory = {
    getSomeService: function () {
      return new SomeService();
    }
  };

  return factory;
}
```

Differences between Approaches

```
…
var factory =
        {…};

return factory;
```

Object literal with a prop method that calls new SomeService()

```
…
var factory =
        function () {…};

return factory;
```

Function that returns a reference to new SomeService()

Using Object Literal Approach

```
…
var factory =
        {…};

return factory;
```

```
…
var someSrv =
    CustomService
    .getSomeService ();

someSrv.method();
```

CustomService refers to object literal, accessing its property

Property is a function, so invoke it with parens

Using Object Function Approach

```
…
var factory =
      function () {…};

return factory;
```

```
…
var someSrv =
    CustomService();

someSrv.method();
```

CustomService refers to a function value. So, parens would invoke it

Summary

- ✧ .factory() allows us to produce any type of object or function
  - That includes a service (even a singleton), but is NOT limited to
  - .service() is just a more limited factory
- ✧ .factory('name', FactoryFunction) – name is what's injected
- ✧ Injected factory function refers to whatever is returned in the factory function
  - Can be object literal with a prop that's a function that creates something
  - Can be a function that creates something