

Advanced Angular

If you don't have a FP group, let
us know ASAP

Angular: Objects and Services



Specialized Objects

Specialized objects conform to a specific Angular framework API.

*These objects are one of **Controllers, Directives, Filters or Animations**.*

The injector needs to know how to create these objects. You tell it by registering a "recipe" for creating your object with the injector.

Services


Services are objects whose API is defined by the developer writing the service.

*The most verbose, but also the most comprehensive one is a **Provider** recipe. The remaining four recipe types — **Value**, **Factory**, **Service** and **Constant** — are just syntactic sugar on top of a provider recipe.*

Services

*Use services to organize and share code across your app.
Think of services as constructors that we run “new” on.*

```
app.service('myService', function() {  
  // service is just a constructor function  
  // that will be called with 'new'  
  this.sayHello = function(name) {  
    return "Hi " + name + "!";  
  };  
});
```



```
▼ Object {} ⓘ  
  ► sayHello: function (name)  
  ► __proto__: Object
```

Factories

*Very similar to services. Provides shared code across app.
Think of factories as functions we run and return from.*

```
app.factory('myFactory', function() {  
  
    // factory returns an object  
    // you can run some code before  
  
    return {  
        sayHello : function(name) {  
            return "Hi " + name + "!";  
        }  
    }  
});
```

▼ Object {} *i*

- ▶ sayHello: function (name)
- ▶ __proto__: Object

Custom Directives and Filters



<https://angularjs.org/>

A little detour through JS and
framework land...

ECMAScript 6

A bright new future is coming...

Constants

Constants

Scoping

- Block-Scoped Variables
- Block-Scoped Functions

Arrow Functions

- Expression Bodies
- Statement Bodies
- Lexical `this`

Extended Parameter Handling

- Default Parameter Values
- Rest Parameter
- Spread Operator

Template Literals

- String Interpolation
- Custom Interpolation
- Raw String Access

Extended Literals

- Binary & Octal Literal
- Unicode String & RegExp Literal

Enhanced Regular Expression

- Regular Expression Sticky Matching

Enhanced Object Properties

- Property Shorthand
- Computed Property Names
- Method Properties

Destructuring Assignment

- Array Matching
- Object Matching, Shorthand Notation
- Object Matching, Deep Matching
- Parameter Destructuring

Constants Constants

Support for constants (also known as "immutable variables"), i.e., variables which cannot be re-assigned new content. Notice: this only makes the variable itself immutable, not its assigned content (for instance, in case the content is an object, this means the object itself can still be altered).

ECMAScript 6 — syntactic sugar: [reduced](#) | [traditional](#)

```
const PI = 3.141593
PI > 3.0
```

ECMAScript 5 — syntactic sugar: [reduced](#) | [traditional](#)

```
// only in ES5 through the help of object properties
// and only in global context and not in a block scope
Object.defineProperty(typeof global === "object" ? global : window, "PI", {
  value:      3.141593,
  enumerable: true,
  writable:   false,
  configurable: false
})
PI > 3.0;
```





TodoMVC

Helping you **select** an MV* framework

<http://todomvc.com/>

MP4 is coming...

QUESTIONS?

NEXT CLASS: Node, Express

courses.engr.illinois.edu/cs498rk1/