



+



ANGULARJS AND TYPESCRIPT TWO GREAT TOOLS TOGETHER

KURT WIERSMA

@KWIERSMA

Who am I?

- * From Minneapolis, MN
- * Work for the American Academy of Neurology
- * Have been developing web apps for over 14 years
- * Have used Python, Groovy, Java, C#, and CFML
- * Favorites: C#/MVC, Python/Django, AngularJS, TypeScript

Agenda

- * TypeScript intro and getting started
- * AngularJS basic knowledge assumed
- * How to combine two great tools
 - * App config, routes, references.ts ordering
 - * Controllers
 - * Services
 - * Authentication with API's
 - * Authorization and routes



PROJECT

TypeScript

A RISING STAR

DATE

8/1/2015

CLIENT

AWESOME, INC

TypeScript

- * <http://typescriptlang.org>
- * TypeScript lets you write JavaScript the way you really want to.
- * TypeScript is a typed superset of JavaScript that compiles to plain JavaScript.
- * Any browser. Any host. Any OS. Open Source.
- * AngularJS 2 is implemented in TypeScript



TypeScript

Select...

Share

```
1 function greeter(person) {  
2     return "Hello, " + person;  
3 }  
4  
5 var user = "Jane User";  
6  
7 document.body.innerHTML = greeter(user);  
8  
9
```

Run

JavaScript

```
1 function greeter(person) {  
2     return "Hello, " + person;  
3 }  
4  
5 var user = "Jane User";  
6  
7 document.body.innerHTML = greeter(user);  
8
```

JAVASCRIPT IS VALID TYPESCRIPT

TypeScript Syntax

```
1  /// <reference path='../_all.ts' />
2
3  module djleague {
4
5      export class FantasyTeamService {
6
7          public teams: FantasyTeam[];
8
9          private httpService: ng.IHttpService;
10
11         constructor ($http: ng.IHttpService) {
12             this.httpService = $http;
13         }
14
15         getTeams(): ng.IPromise<FantasyTeam[]> {
16             return this.httpService.get('/api/teams')
17                 .then(function (response) {
18                     var data = response.data;
19                     this.teams = new Array<FantasyTeam>();
20
21                     for (var i = 0; i < data.length; i++) {
22                         var team: FantasyTeam = new FantasyTeam();
23                         team.id = data[i].id;
24                         team.name = data[i].name;
25                         team.draftorder = data[i].draftorder;
26                         team.owner = data[i].owner;
27                         this.teams.push(team);
28                     }
29
30                     return this.teams;
31                 });
32         }
33     }
34 }
35
```

```
1  /// <reference path='../_all.ts' />
2  var djleague;
3
4  (function (djleague) {
5      var FantasyTeamService = (function () {
6
7          function FantasyTeamService($http) {
8              this.httpService = $http;
9          }
10
11         FantasyTeamService.prototype.getTeams = function () {
12             return this.httpService.get('/api/teams').then(function (response) {
13                 var data = response.data;
14                 this.teams = new Array();
15
16                 for (var i = 0; i < data.length; i++) {
17                     var team = new djleague.FantasyTeam();
18                     team.id = data[i].id;
19                     team.name = data[i].name;
20                     team.draftorder = data[i].draftorder;
21                     team.owner = data[i].owner;
22                     this.teams.push(team);
23                 }
24
25                 return this.teams;
26             });
27         };
28
29         return FantasyTeamService;
30     })();
31
32     djleague.FantasyTeamService = FantasyTeamService;
33
34 })(djleague || (djleague = {}));
```

The TypeScript logo, featuring the word "TypeScript" in a blue, sans-serif font. The "T" is significantly larger and more prominent than the other letters. Above the "T" is a solid blue horizontal bar.

JavaScript

Features

- * Classes
- * Modules
- * Interfaces
- * Generics
- * Arrow Functions
- * References
- * Type Definitions
- * Better “this” by default

```
1  /// <reference path='../_all.ts' />
2
3  module dleague {
4
5      export class FantasyTeamService {
6
7          public teams: FantasyTeam[];
8
9          private httpService: ng.IHttpService;
10
11         constructor ($http: ng.IHttpService) {
12             this.httpService = $http;
13         }
14
15         getTeams(): ng.IPromise<FantasyTeam[]> {
16             return this.httpService.get('/api/teams')
17                 .then(function (response) {
18                     var data = response.data;
19                     this.teams = new Array<FantasyTeam>();
20
21                     for (var i = 0; i < data.length; i++) {
22                         var team: FantasyTeam = new FantasyTeam();
23                         team.id = data[i].id;
24                         team.name = data[i].name;
25                         team.draftorder = data[i].draftorder;
26                         team.owner = data[i].owner;
27                         this.teams.push(team);
28                     }
29
30                     return this.teams;
31                 });
32         }
33     }
34 }
35
```


Why would you want types?

- * Structure for large code bases and/or teams
- * Catch errors early
- * Provide a structured API
- * Tooling can provide better code completion & refactoring

What about existing JavaScript Libraries?

- * DefinitelyTyped provides TS definitions for tons of JS libraries
 - * JQuery, Angular, Node, Ember, Backbone, ect.
 - * <http://definitelytyped.org>
- * You can write you own definitions easily
- * TSD tool to manage definitions



Custom Definitions

```
1  declare class Pusher {
2      constructor (key: string);
3      subscribe(channelName: string): Channel;
4  }
5
6  interface Channel {
7      bind(eventName: string, callback: Function);
8  }
```

pusher.d.ts

```
102  $scope.PUSHER_ENABLED = true;
103  if ($scope.PUSHER_ENABLED) {
104      this.pusher = new Pusher('|');
105      this.channel = this.pusher.subscribe('draftedPlayers');
106      this.channel.bind('playerDrafted', function(data) {
107          //console.log("playerDraft notification received");
108          //console.dir(data);
109
110          $scope.$apply(function(scope) {
111              $scope.picks = data;
112              $scope.vm.processPicks();
113          });
114      });
115  }
```

Useage

Getting Started

- * Install:

- * `npm install -g typescript`

- * Compile:

- * `tsc --sourcemap --out js/Application.js js/_all.ts`

Tooling

- * CLI: grunt with grunt-ts or gulp
- * TSD: managing definitions for JS libraries
 - * `tsd install angular --resolve --overwrite --save`
- * IDEs:
 - * WebStorm / IntelliJ (Mac & Win) [\$]
 - * Visual Studio 2012+ (Win) [\$]
 - * Visual Studio Code (Mac & Win) [Free]
 - * Eclipse (Mac & Win) [Free]
- * Editors:
 - * Sublime [\$]
 - * Atom [Free]

TYPESCRIPT REFERENCES

`_all.ts`

```
1 // <reference path='typings/angularjs/angular.d.ts' />
2 /// <reference path='typings/angularjs/angular-route.d.ts' />
3 /// <reference path='libs/bootstrap.d.ts' />
4 /// <reference path='libs/pusher.d.ts' />
5
6 // Application.ts must appear first since it as the angular module definition
7 /// <reference path='Application.ts' />
8
9 /// <reference path='models/FantasyTeam.ts' />
10 /// <reference path='models/Player.ts' />
11 /// <reference path='models/Pick.ts' />
12
13 /// <reference path='services/FantasyTeamService.ts' />
14 /// <reference path='services/PlayerService.ts' />
15
16 /// <reference path='controllers/DraftCtrl.ts' />
17 /// <reference path='controllers/TeamCtrl.ts' />
18
```

TYPSCRIPT COMPILE

GruntFile.JS

```
1  module.exports = function(grunt) {
2
3      grunt.initConfig({
4          exec: {
5              tsPublic: {
6                  cmd: 'tsc --sourcemap --out dleague/static/js/Application-all.js dleague/static/js/_all.ts'
7              }
8          },
9
10         watch: {
11             public: {
12                 files: ["dleague/static/js/**/*.ts"],
13                 tasks: ["exec:tsPublic"],
14                 options: { livereload: true }
15             }
16         }
17     });
18
19     grunt.loadNpmTasks('grunt-exec');
20     grunt.loadNpmTasks('grunt-contrib-watch');
21
22     grunt.registerTask('default', ['exec:tsPublic', 'watch'])
23
24 }
```



PROJECT

AngularJS

A SUPER HEROIC FRAMEWORK

DATE

4/1/2010

CLIENT

AWESOME, INC

APP CONFIG & ROUTES

Application.ts

```
19
20 module djleague {
21
22     var djleagueAngular = angular.module('players',
23         ['ngRoute', 'angular-table']);
24
25     angular.module('players')
26         .config(['$routeProvider', function($routeProvider: ng.route.IRouteProvider) {
27             $routeProvider.
28                 when('/players', {
29                     templateUrl: '/static/partials/players.html',
30                     controller: 'DraftCtrl'
31                 }).
32                 when('/teams', {
33                     templateUrl: '/static/partials/teams.html',
34                     controller: 'TeamCtrl',
35                     controllerAs: 'vm'
36                 }).
37                 otherwise({redirectTo: '/players'});
38         });
39     });
40 }
```

CONTROLLERS

```
1  /// <reference path='../_all.ts' />
2
3  module djleague {
4
5      export class TeamCtrl {
6
7          teams: FantasyTeam[] = [];
8          selectedTeam: FantasyTeam = new FantasyTeam();
9
10         static $inject = [
11             '$log',
12             'FantasyTeamService'
13         ];
14
15         constructor(
16             private $log: ng.ILogService,
17             private teamService: FantasyTeamService)
18         {
19             this.fetchTeams();
20         }
21
22         teamSelected(team: FantasyTeam): void {
23             this.selectedTeam = angular.copy(team);
24             this.$log.debug('selectedTeam:', this.selectedTeam);
25
26             $('#teamModal').modal('show');
27         }
28
29         saveSelectedTeam(): void {
30             this.teamService.saveTeam(this.selectedTeam).then(() => {
31                 this.fetchTeams();
32                 $('#teamModal').modal('hide');
33             });
34         }
35
36         fetchTeams(): void {
37             this.teamService.getTeams().then((data) =>{
38                 this.$log.debug('teams data:', data);
39                 this.teams = data;
40             });
41         }
42     }
43 }
44
45 angular.module('players').controller('TeamCtrl', TeamCtrl);
46
47 }
```

SERVICES

```
1  /// <reference path='../_all.ts' />
2
3  module djleague {
4
5      export class FantasyTeamService {
6
7          teams: FantasyTeam[];
8
9          static $inject = ['$http', '$q'];
10         constructor (
11             private $http: ng.IHttpService,
12             private $q: ng.IQService) {
13
14         }
15
16         getTeams(): ng.IPromise<FantasyTeam[]> {
17             return this.$http.get('/api/teams')
18                 .then(function (response) {
19                     var data = <Array<FantasyTeam>> response.data;
20                     this.teams = data;
21                     return this.teams;
22                 });
23         }
24
25         saveTeam(team: FantasyTeam): ng.IPromise<FantasyTeam> {
26             // TODO: implement saving a team
27             return this.$q.when({});
28         }
29     }
30 }
31
32 angular.module('players').service('FantasyTeamService', FantasyTeamService);
33 }
```

AUTHENTICATION WITH API'S

```
109 angular.module('myApp'  
110   .factory('authInterceptor', ['$rootScope', '$q', '$window', '$location',  
111     function ($rootScope, $q, $window, $location) {  
112       return {  
113         request: function (config) {  
114           config.headers = config.headers || {};  
115           if ($window.sessionStorage.token) {  
116             config.headers.Authorization = 'Token ' + $window.sessionStorage.token;  
117           }  
118           return config;  
119         },  
120         response: function (response) {  
121           if (response.status === 401) {  
122             // user is not authenticated according to the server  
123             delete $window.sessionStorage.token;  
124             $location.path('/start');  
125           }  
126           return response || $q.when(response);  
127         }  
128       };  
129     })  
130   .config(['$httpProvider', function ($httpProvider) {  
131     $httpProvider.interceptors.push('authInterceptor');  
132   }]);
```

AUTHORIZATION & ROUTES

Application.ts

```
98 angular.module('myApp'  
99   .run(['$rootScope', '$location', 'SecurityService',  
100     function ($rootScope, $location, SecurityService) {  
101       $rootScope.$on('$routeChangeStart', function (event, next) {  
102         if (next.requiresLogin !== undefined && next.requiresLogin)  
103           SecurityService.handleNotAuthenticated();  
104       }  
105     });  
106   }  
107 ])
```

```
12   .config(['$routeProvider', '$httpProvider',  
13     function ($routeProvider, $httpProvider) {  
14       $routeProvider  
15         .when('/start', {  
16           templateUrl: '/static/app/views/login.html',  
17           controller: 'LoginController',  
18           controllerAs: 'vm'  
19         })  
20         .when('/welcome', {  
21           templateUrl: '/static/app/views/welcome.html',  
22           controller: 'WelcomeController',  
23           controllerAs: 'vm',  
24           requiresLogin: true  
25         })  
26     })
```

Services.js

```
189 .factory('SecurityService', ['WebAssessService', '$location',  
190   function (WebAssessService, $location) {  
191     var service = {  
192       handleNotAuthenticated: function () {  
193         if (!WebAssessService.isAuthenticated()) {  
194           $location.path('/start');  
195         }  
196       }  
197     };  
198     return service;  
199   }  
200 ]  
201 )
```

AngularJS Tools

- * Debugging
 - * Batarang
 - * ng-inspector
- * AngularUI & Bootstrap UI
- * Angular Formly
- * Testing
 - * Protractor
 - * ng-describe

Resources

- * [Using Visual Studio Code with Typescript and AngularJS](#)
- * [John Papa's AngularJS Style Guide](#)
- * Real World Example Apps
 - * [Angular In 20 Minutes](#)
 - * [Expense Manager](#)

QUESTIONS?

KURT WIERSMA (KWIER SMA@MAC.COM)

[@KWIER SMA](#)

[HTTP://GITHUB.COM/KWIER SMA](http://GITHUB.COM/KWIER SMA)