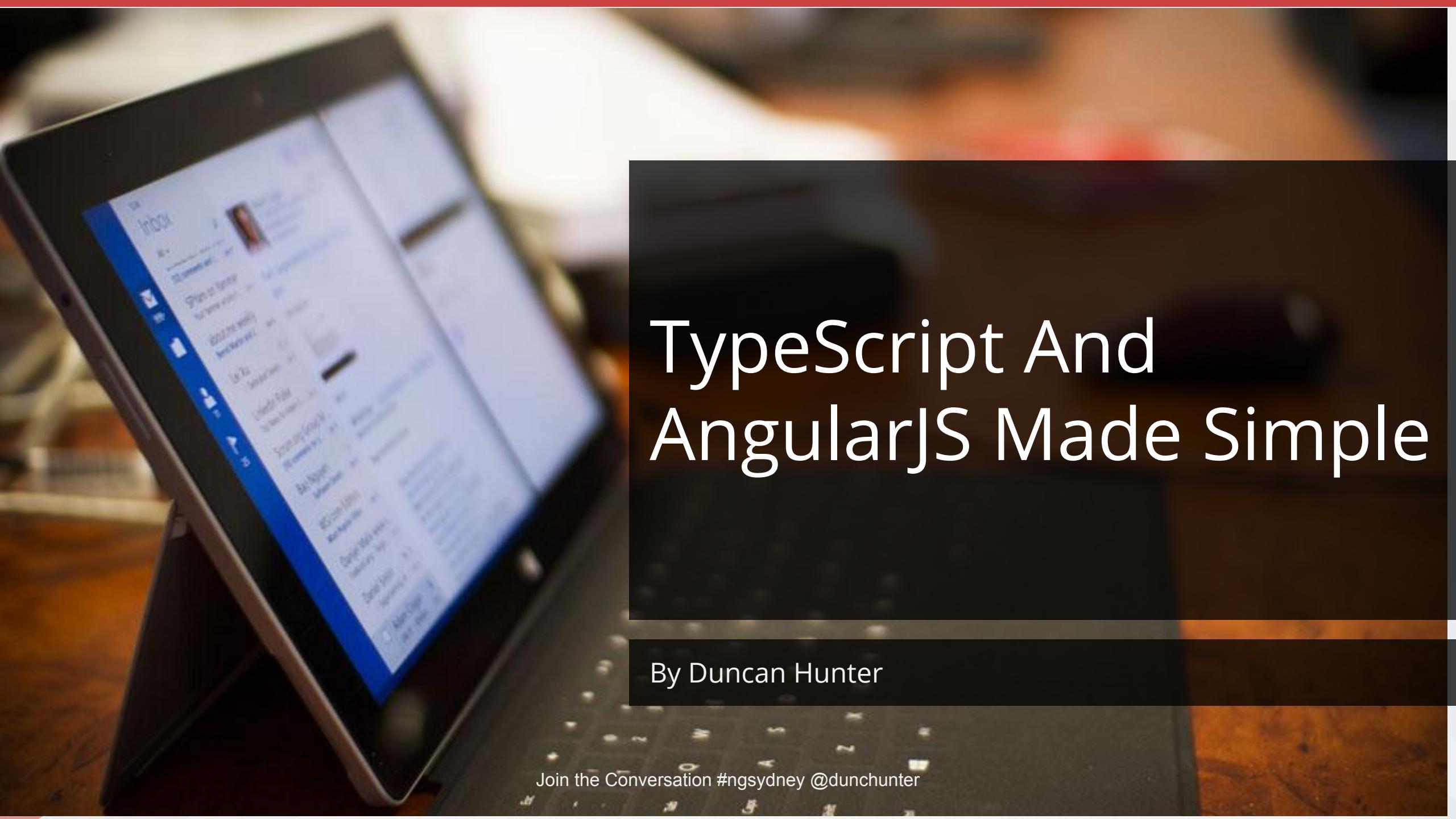




Enterprise Software Development



TypeScript And AngularJS Made Simple

By Duncan Hunter

Join the Conversation #ngsydney @dunchunter



Duncan Hunter



#dunchunter



www.duncanhunter.com.au

- .NET Developer
- Rock climber

Join the Conversation #ngsydney @dunchunter



ANGULARJS
by Google

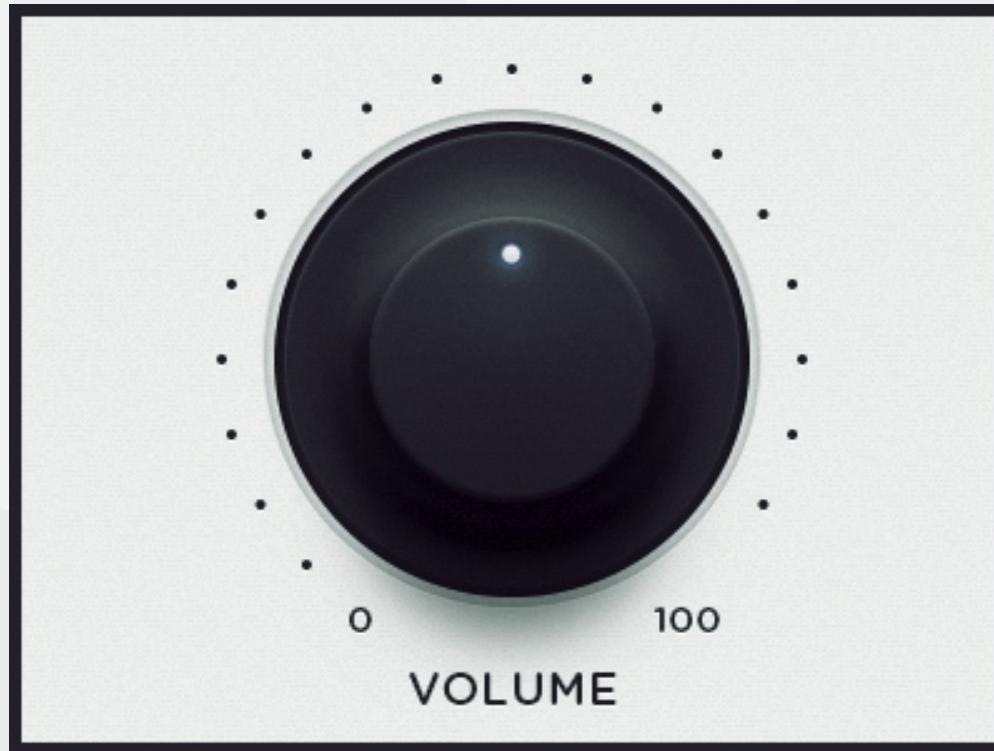
TypeScript



Microsoft

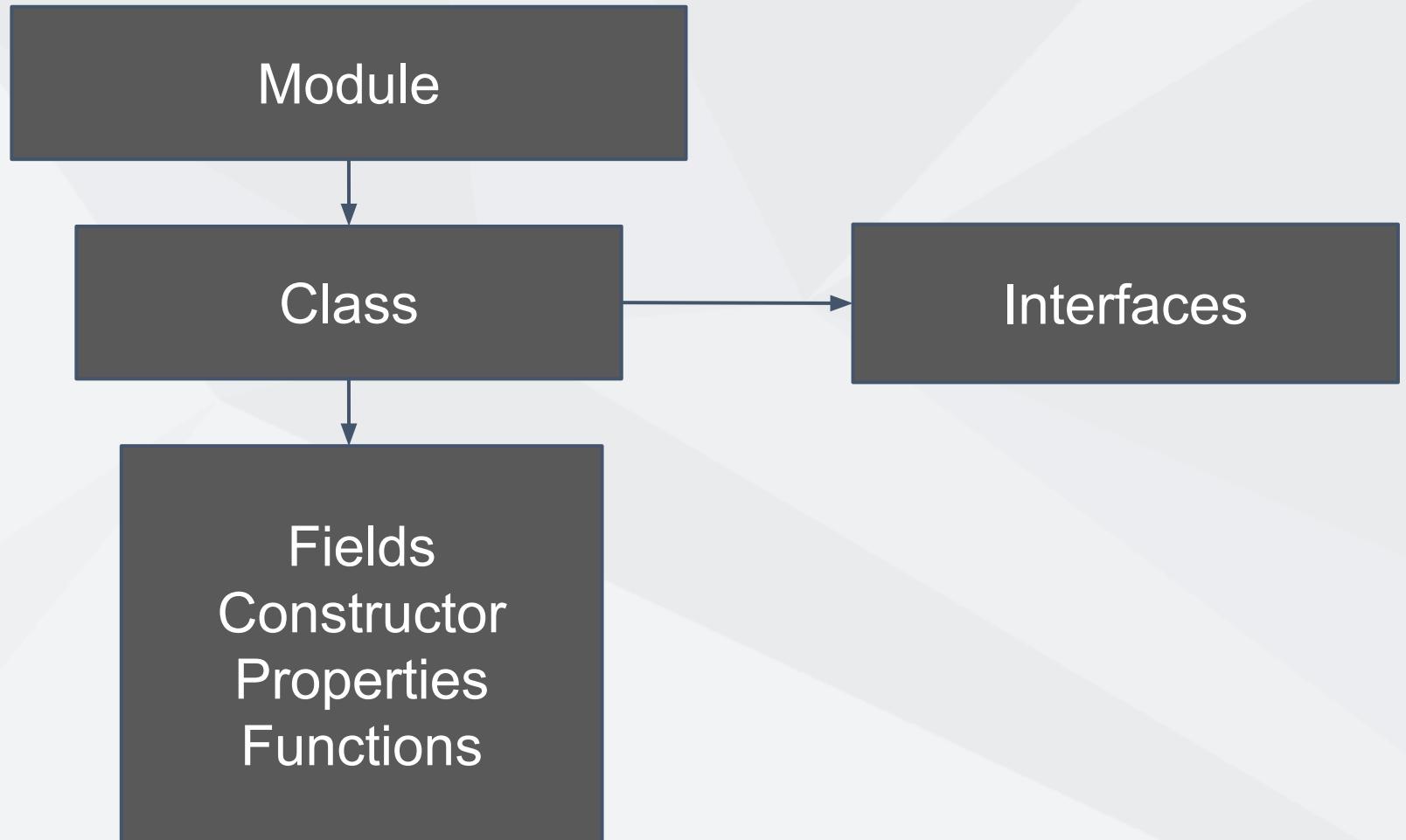
TypeScript

TypeScript is a typed superset of JavaScript that compiles to plain JavaScript.





TypeScript structure



The TypeScript Documentation and Playground

The screenshot shows the official TypeScript website. At the top, there's a blue header bar with the word "TypeScript" in white. Below it, the main content area has a light gray background. On the left side of this area, there's a large blue rectangular image featuring a silhouette of a city skyline against a blue sky with white clouds. Overlaid on this image is the word "Scalable" in a large, white, sans-serif font. To the right of the image, there's descriptive text about TypeScript's features: "TypeScript offers classes, modules, and interfaces to help you build robust components.", "These features are available at development time for high-confidence application development, but are compiled into simple JavaScript.", and "TypeScript types let you define interfaces between software components and to gain insight into the behavior of existing JavaScript libraries.". At the very bottom of the page, there's a footer with the text "Join the Conversation #ngsydney @dunchunter".

TypeScript

learn play download interact

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.

Get TypeScript Now →

Scalable

TypeScript offers classes, modules, and interfaces to help you build robust components.
These features are available at development time for high-confidence application development, but are compiled into simple JavaScript.
TypeScript types let you define interfaces between software components and to gain insight into the behavior of existing JavaScript libraries.

Join the Conversation #ngsydney @dunchunter

Why bother with TypeScript

POSITIVES

1. Type safety
2. Code completion
3. Great tooling
4. AngularJS 2.0 is built on TypeScript

NEGATIVES

1. You write more code
2. It's sometimes harder to read
3. It's another dependency
4. You can not always type everything

Definitely Typed Definitions

The screenshot shows the GitHub repository page for `borisyankov / DefinitelyTyped`. The top navigation bar includes links for `Explore`, `Gist`, `Blog`, and `Help`. The user profile of `duncanhunter` is shown on the right. The main repository information includes 7,985 commits, 7 branches, 0 releases, and 411 contributors. A green pull request button is visible. The repository description states: "The repository for high quality TypeScript type definitions. <http://definitelytyped.org/>". The commit history lists several recent contributions from users like `horiuchi`, `FileSaver`, `Finch`, `Headroom`, and `JSONStream`. On the right side, there is a sidebar with links for `Code`, `Issues`, `Pull Requests`, `Wiki`, `Pulse`, `Graphs`, and a `HTTPS clone URL` button.

This repository for high quality TypeScript type definitions. <http://definitelytyped.org/>

7,985 commits 7 branches 0 releases 411 contributors

Merge pull request #3562 from Asana/del ...

horiuchi authored 2 days ago latest commit 7bab855ae3

FileSaver	Update reference.	15 days ago
Finch	cleaned-up headers	8 months ago
Headroom	add definitions for Headroom	3 months ago
JSONStream	added definition for JSONStream	8 months ago

Join the Conversation #ngsydney @dunchunter

Definitely Typed Definitions

How to get the definitions

- Directly from the Github repos
- NuGet packages
- TypeScript Definition manager

```
C:\Demo\AngularAndTypeScriptMadeSimple\AngularAndTypeScriptMadeSimple> tsd install angular --save
```

How to get the definitions

Pluralsight courses

Using TypeScript for Large AngularJS Applications

This course examines how to use TypeScript with a large AngularJS application and uncovers the positives and negatives along the way.

by [Justin Schwartzenberger](#)

Join the Conversation #ngsydney @dunchunter

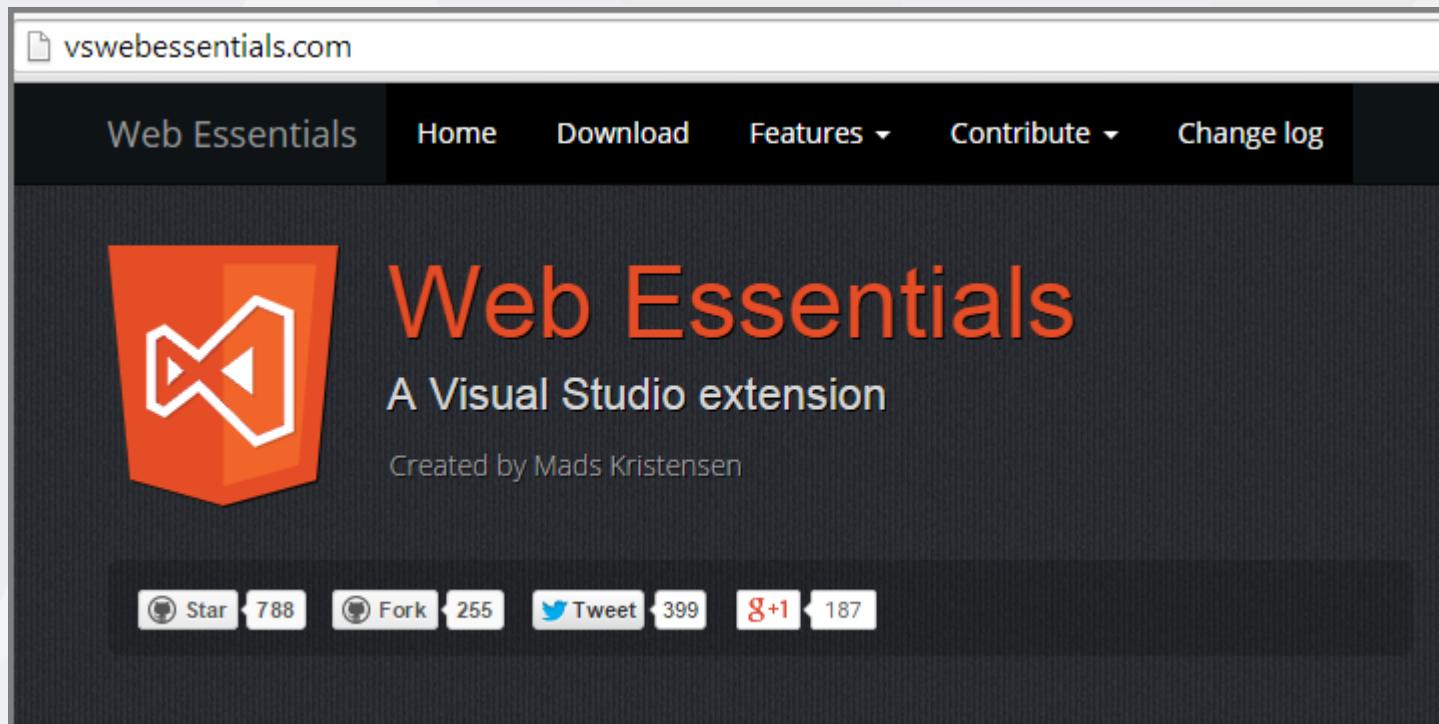
TsLint

TsLint - <https://github.com/palantir/tslint>

Error List				
	Description	File	Line	Column
	Project			
1	TsLint: expected parameter: 'endDate' to have a typedef	rates.ts	2	32
2	TsLint: missing 'use strict'	rates.ts	2	3
3	TsLint: ' should be "	rates.ts	6	17
4	TsLint: expected parameter: 'rates' to have a typedef	rates.ts	10	29
5	TsLint: missing 'use strict'	rates.ts	10	2
6	TsLint: ' should be "	rates.ts	14	74
7	TsLint: ' should be "	rates.ts	14	17
8	TsLint: missing semicolon	rates.ts	14	78
9	TsLint: ' should be "	rates.ts	15	17
10	TsLint: ' should be "	rates.ts	15	42
11	TsLint: missing semicolon	rates.ts	15	50
12	TsLint: ' should be "	rates.ts	16	49

Web Essentials

Web Essentials - <http://vswebessentials.com/>



Join the Conversation #ngsydney @dunchunter

The screenshot shows a Visual Studio Code interface with the following components:

- Left Panel:** A code editor with two tabs: `beer.controller.ts` and `beer.service.ts`. The `beer.controller.ts` tab is active, displaying the following code:

```
1 app.beer {
2   'use strict';
3
4   import IBeer = app.services.IBeer;
5
6   interface IBeerScope {
7     beers: IBeer[];
8     getBeers(): ng.IPromise<IBeer[]>;
9   }
10
11 class BeerController implements IBeerScope {
12   beers: IBeer[];
13
14   static $inject = ['app.services.BeerService'];
15   constructor(private beerService: services.IBeerService) {
16     this.getBeers();
17   }
18
19   getBeers(): angular.IPromise<IBeer[]> {
20     return this.beerService.getBeers()
21       .then((data: ng.IHttpPromiseCallbackArg<IBeer[]>): any =>
22         return this.beers = <IBeer[]>data;
23       );
24   }
25
26   angular
27     .module('app')
28     .controller('app.beer.BeerController', BeerController);
29
30
31
32
33
34 }
```

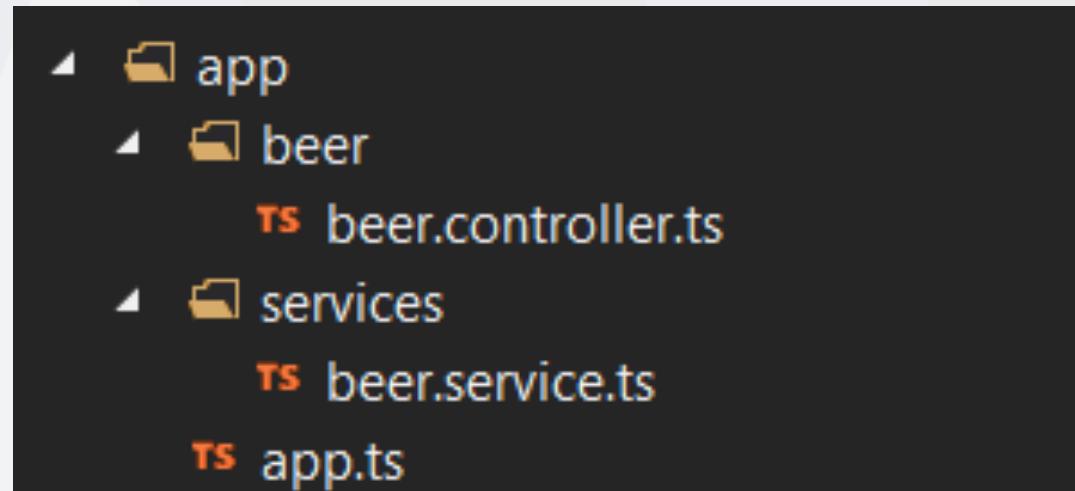
- Middle Panel:** A code editor showing the generated JavaScript code for `beer.controller.ts`, which is highlighted with a red border.

```
1 var app;
2 (function (app) {
3   var beer;
4   (function (beer) {
5     'use strict';
6     var BeerController = (function () {
7       function BeerController(beerSe) {
8         this.beerService = beerSe;
9         this.getBeers();
10      }
11      BeerController.prototype.getBeers = function () {
12        var _this = this;
13        return this.beerService.getBeers().then(function (data) {
14          _this.beers = data;
15        });
16      };
17      BeerController.$inject = ['app.services.BeerService'];
18      return BeerController;
19    })();
20    angular.module('app').controller('app.beer.BeerController', beer);
21  })(app || (app = {}));
22 // # sourceMappingURL=beer.controller.js.map
```

- Right Panel:** The `Solution Explorer` sidebar, also with a red border around the `App` folder. It lists the project structure:

 - Solution:** AngularAndTypeScriptMadeSimple
 - AngularAndTypeScriptMadeSimple**
 - App** (selected)
 - beer**
 - `beer.controller.js`
 - `beer.controller.js.map`
 - `beer.controller.ts` (highlighted with orange)
 - modals**
 - services**
 - `beer.service.js`
 - `beer.service.js.map`
 - `beer.service.ts` (highlighted with orange)
 - `app.js`
 - `app.js.map`
 - `app.ts` (highlighted with orange)
 - `demo.js`
 - `demo.js.map`
 - App_Data**
 - App_Start**
 - bin**
 - bower_components**
 - Content**
 - Controllers**
 - fonts**

Code



```
module app.beer {  
  
    class BeerController {  
        beers = null;  
  
        static $inject = ['app.services.BeerService'];  
        constructor(private beerService) {  
            this.getBeers();  
        }  
  
        getBeers() {  
            return this.beerService.getBeers()  
                .then((response) => {  
                    this.beers = response.data;  
                });  
        }  
    }  
  
    angular  
        .module('app')  
        .controller('app.beer.BeerController', BeerController);  
}
```

```
module app.services {

    export class BeerService {
        beers: any;

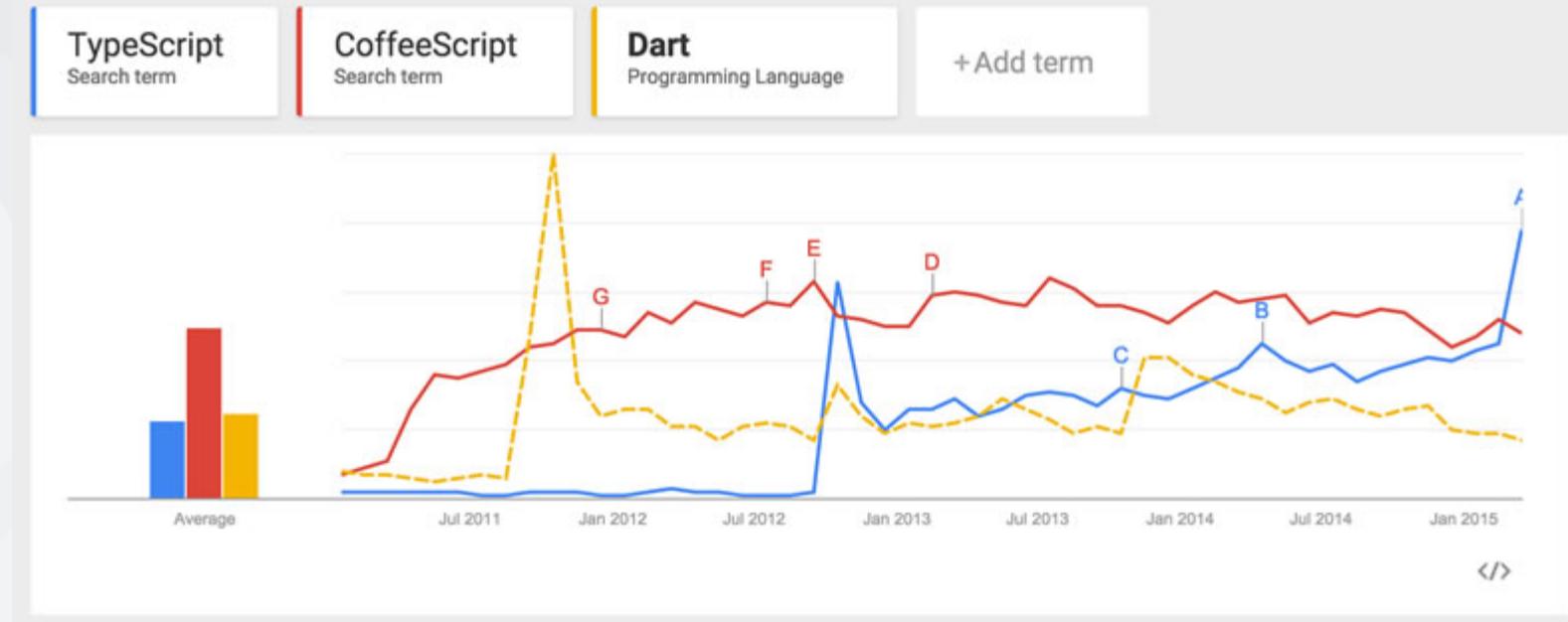
        static $inject = ['$http'];
        constructor(private $http: ng.IHttpService) {
        }

        getBeers() {
            return this.$http.get('/api/beer');
        }
    }

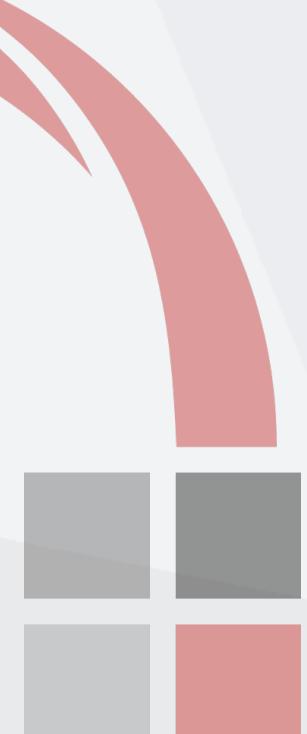
    angular
        .module('app')
        .service('app.services.BeerService', BeerService);
}
```

TypeScript's future?

1. ES6
2. async/await
3. AngularJS
4. Community adoption



Join the Conversation #ngsydney @dunchunter



Thank you!

duncanhunter@ssw.com.au

www.ssw.com.au

Sydney | Melbourne | Brisbane | Adelaide