TypeScript, Now.

Suthep Sangvirotjanaphat

GreatFriends.Biz Founder | Microsoft MVP

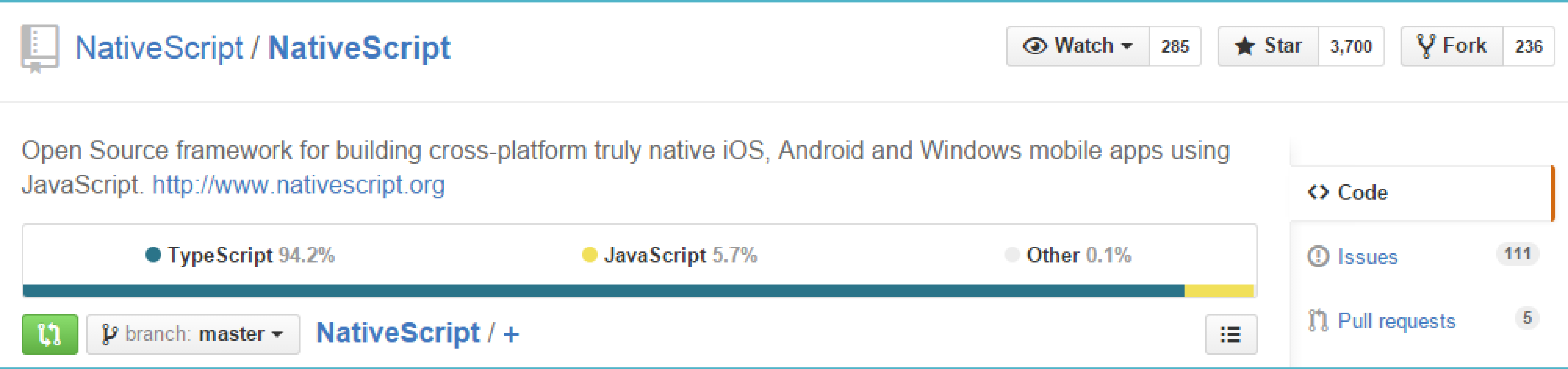
http:||Next.GreatFriends.Biz

facebook.com|suthep

Code Mania 10

GreatFriends.Biz

MVP Microsoft® Most Valuable Professional


NativeScript / NativeScript

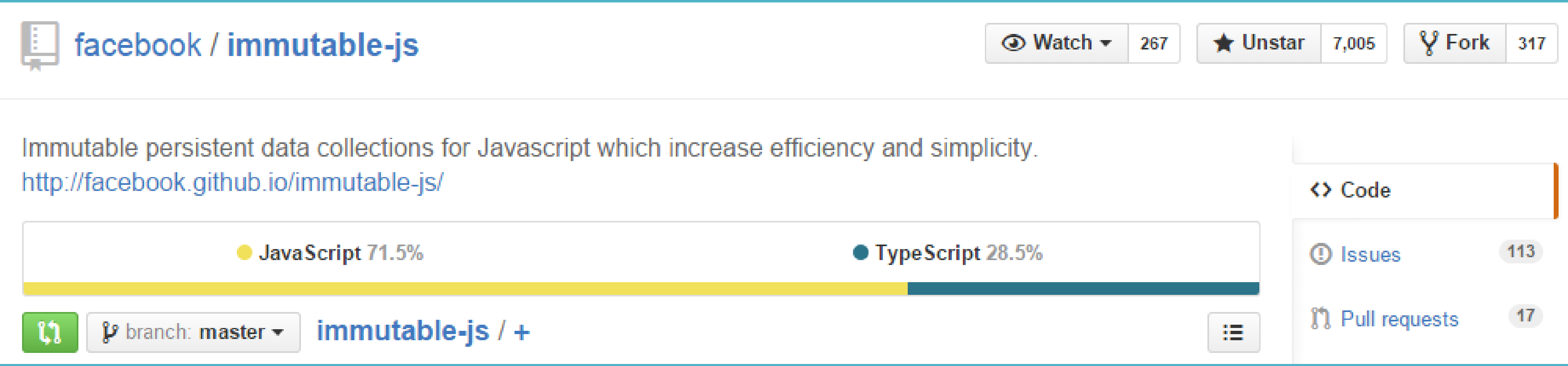
Watch 285 | Star 3,700 | Fork 236

Open Source framework for building cross-platform truly native iOS, Android and Windows mobile apps using JavaScript. http://www.nativescript.org

● TypeScript 94.2% ● JavaScript 5.7% ● Other 0.1%

branch: master NativeScript / +

Code
Issues 111
Pull requests 5


facebook / immutable-js

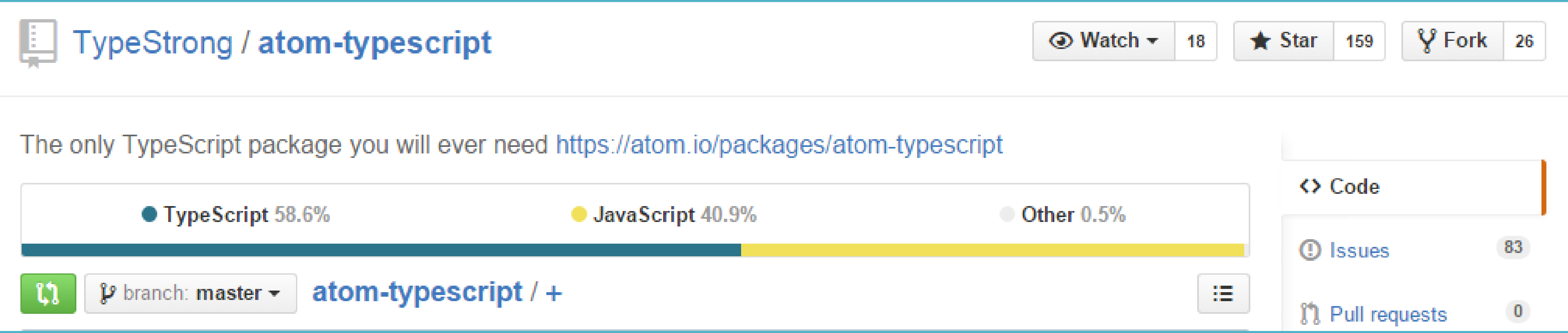
Watch 267 | Unstar 7,005 | Fork 317

Immutable persistent data collections for Javascript which increase efficiency and simplicity. http://facebook.github.io/immutable-js/

● JavaScript 71.5% ● TypeScript 28.5%

branch: master immutable-js / +

Code
Issues 113
Pull requests 17


TypeStrong / atom-typescript

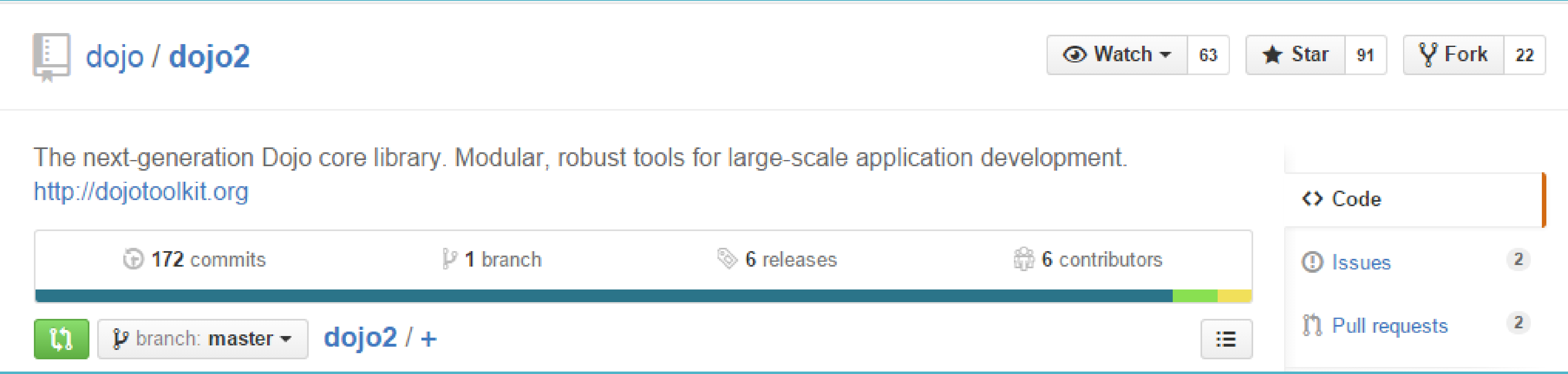
Watch 18 | Star 159 | Fork 26

The only TypeScript package you will ever need https://atom.io/packages/atom-typescript

● TypeScript 58.6% ● JavaScript 40.9% ● Other 0.5%

branch: master atom-typescript / +

Code
Issues 83
Pull requests 0

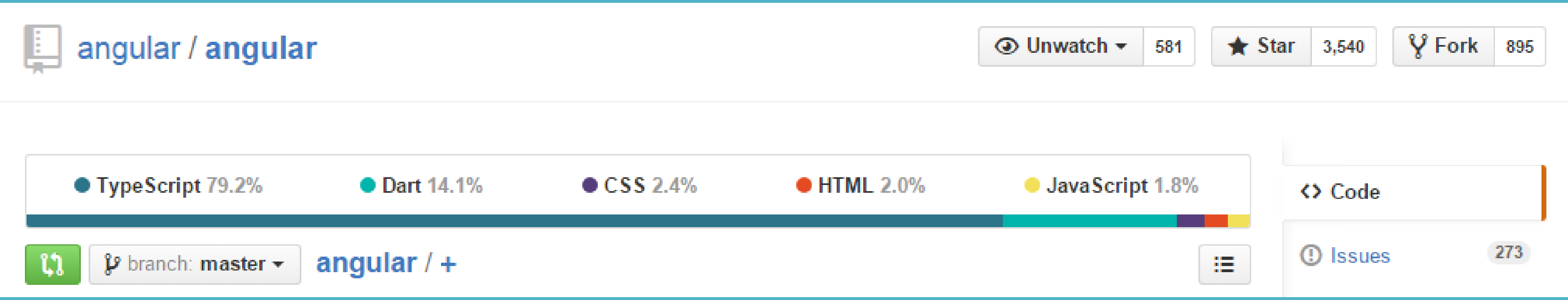


dojo / dojo2

The next-generation Dojo core library. Modular, robust tools for large-scale application development. http://dojotoolkit.org

172 commits · 1 branch · 6 releases · 6 contributors

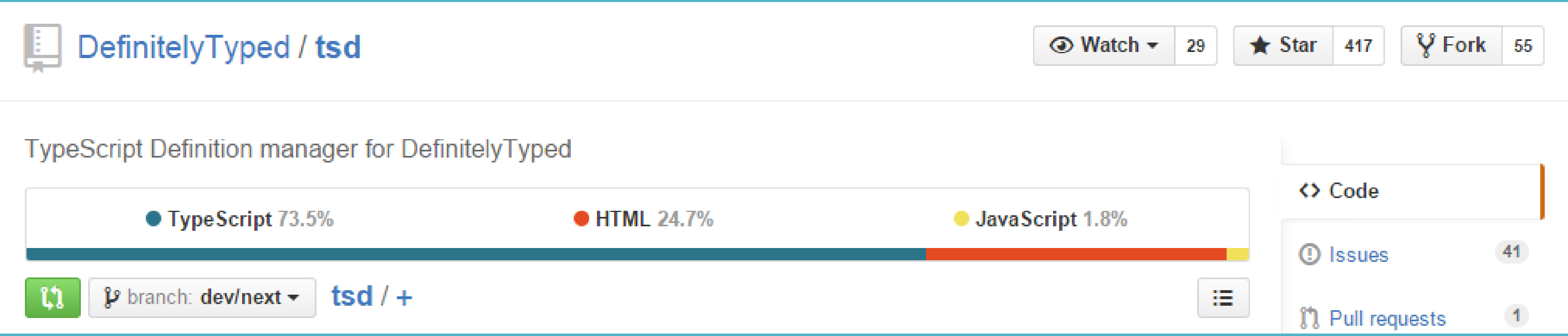
branch: master · dojo2 / +



angular / angular

● TypeScript 79.2% · ● Dart 14.1% · ● CSS 2.4% · ● HTML 2.0% · ● JavaScript 1.8%

branch: master · angular / +



DefinitelyTyped / tsd

TypeScript Definition manager for DefinitelyTyped

● TypeScript 73.5% · ● HTML 24.7% · ● JavaScript 1.8%

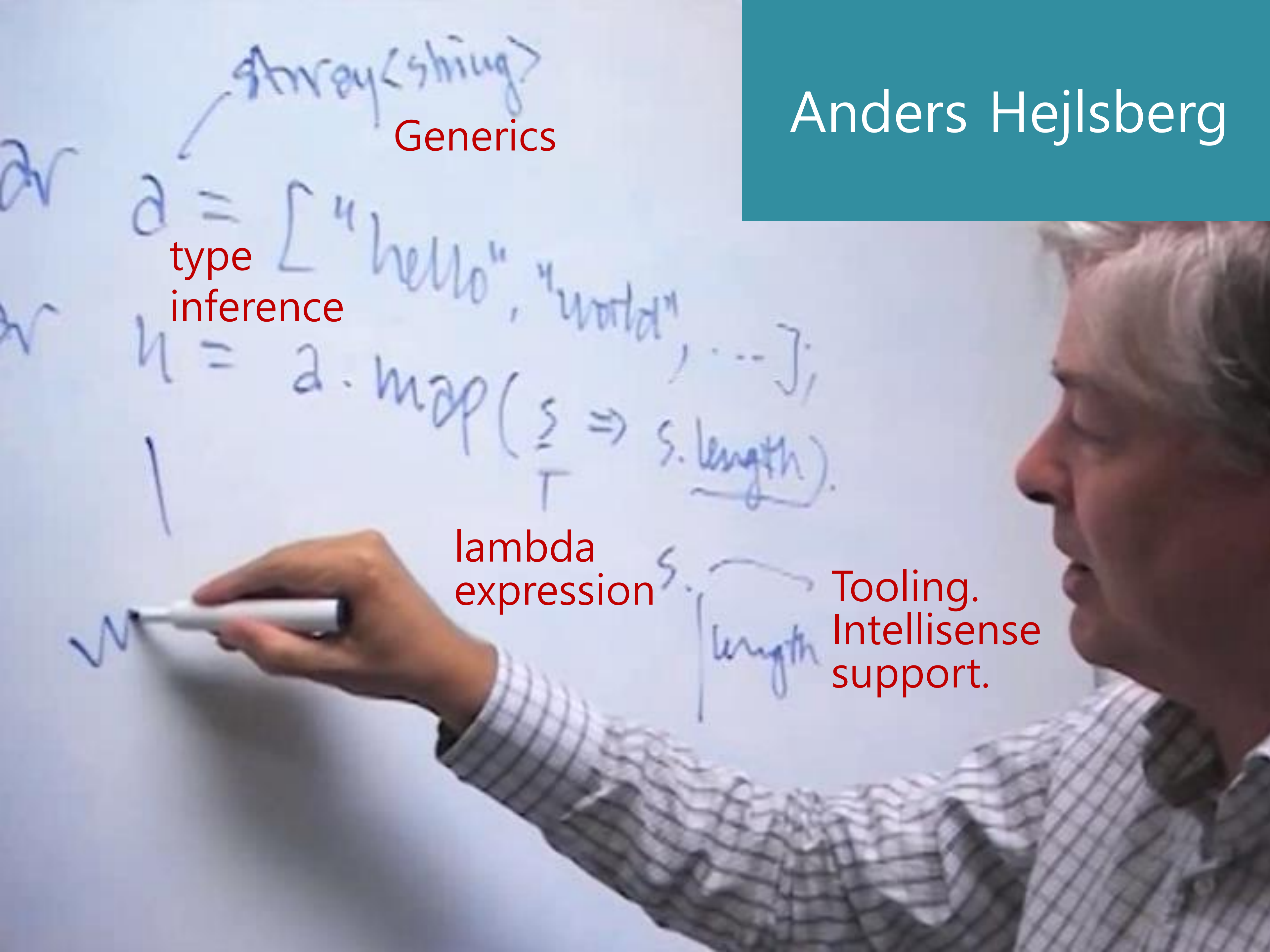
branch: dev/next · tsd / +

TypeScript

TypeScript is a superset of JavaScript
that compiles to
clean JavaScript output.

So, TypeScript is a programming language
that compiles to another programming language.

Anders Hejlsberg


Array<string>
ar a = ["hello", "world", ...];
ar n = a . map(s => s.length).
T
s.
length

Generics

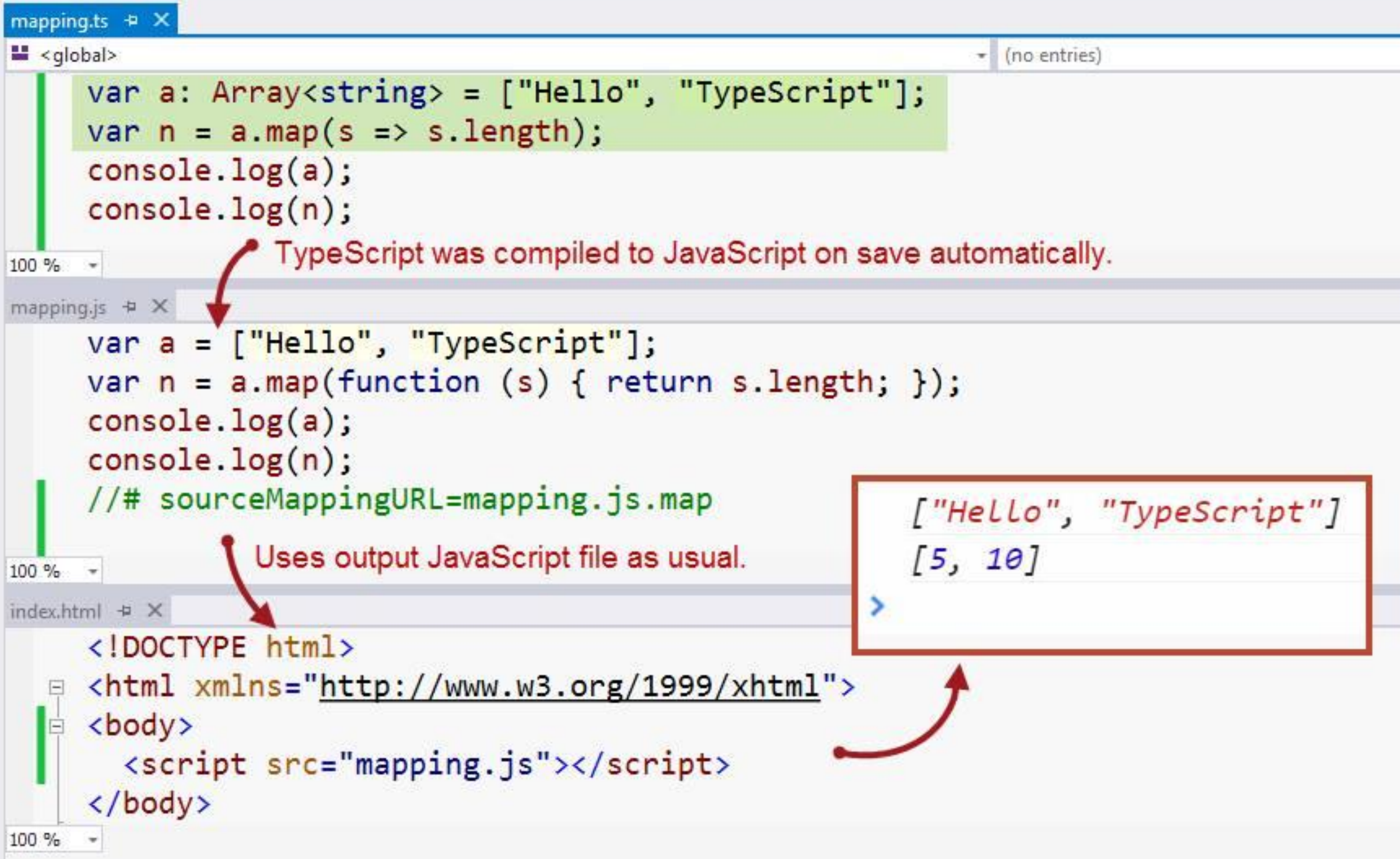
type inference

lambda expression

Tooling. Intellisense support.

So, we'll try the same code in the playground

Replay what Anders had written on the whiteboard



```
var a: Array<string> = ["Hello", "TypeScript"];
var n = a.map(s => s.length);
console.log(a);
console.log(n);
```

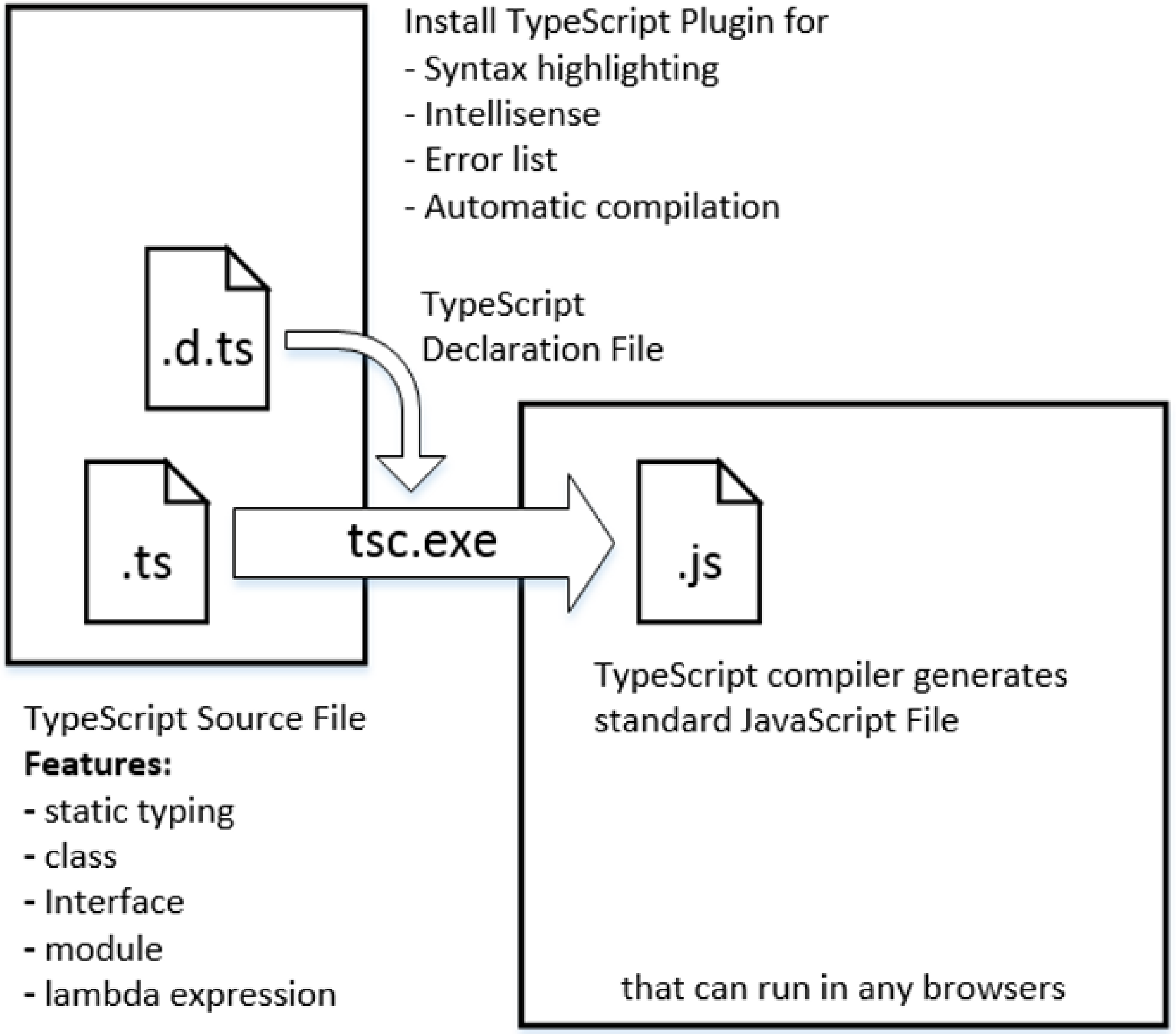
TypeScript was compiled to JavaScript on save automatically.

```
var a = ["Hello", "TypeScript"];
var n = a.map(function (s) { return s.length; });
console.log(a);
console.log(n);
//# sourceMappingURL=mapping.js.map
```

Uses output JavaScript file as usual.

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<body>
  <script src="mapping.js"></script>
</body>
```

```
["Hello", "TypeScript"]
[5, 10]
>
```



Install TypeScript Plugin for
- Syntax highlighting
- Intellisense
- Error list
- Automatic compilation

TypeScript
Declaration File

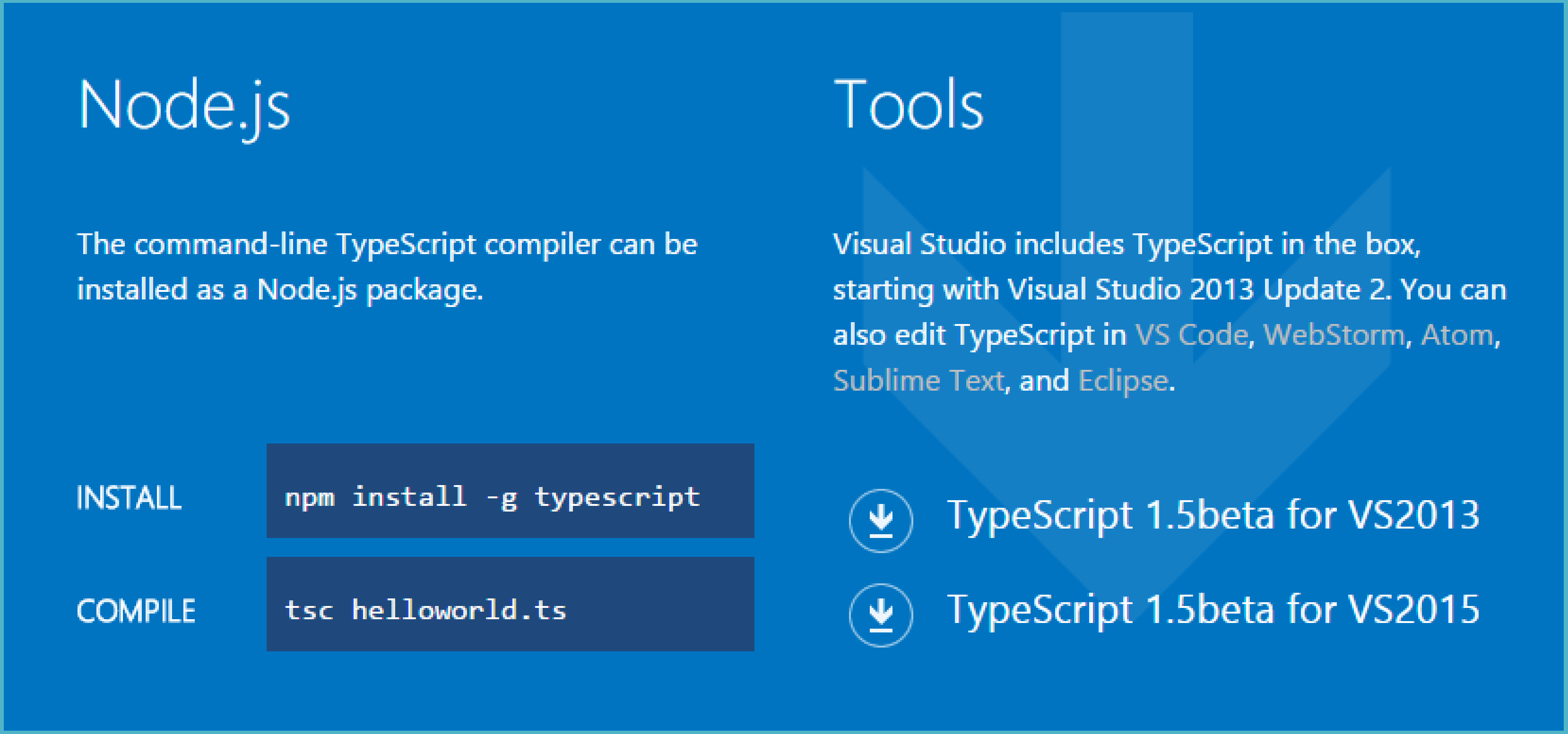
TypeScript compiler generates
standard JavaScript File

that can run in any browsers

TypeScript Source File

Features:
- static typing
- class
- Interface
- module
- lambda expression

Get TypeScript



Node.js

The command-line TypeScript compiler can be installed as a Node.js package.

INSTALL
```
npm install -g typescript
```

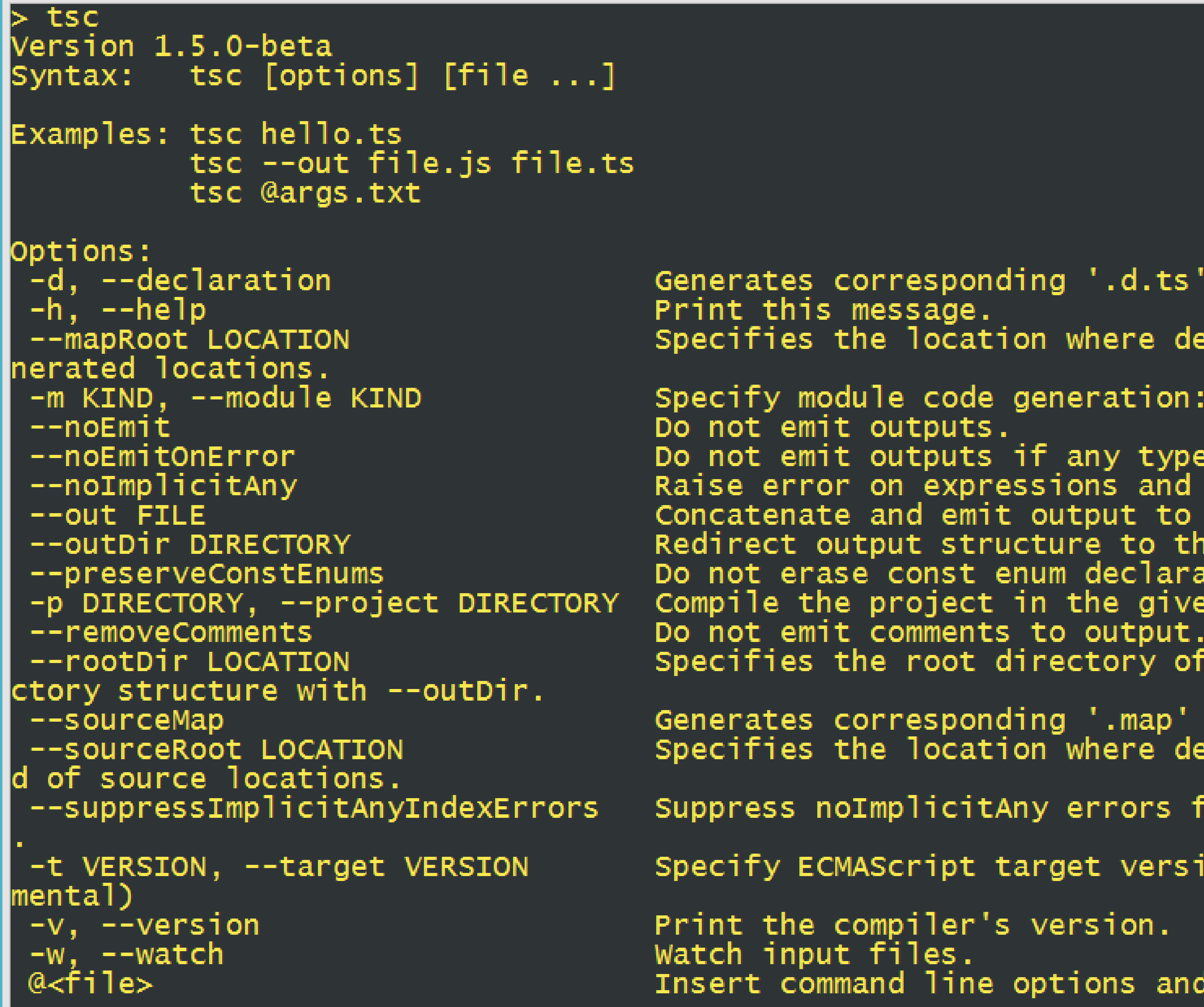
COMPILE
```
tsc helloworld.ts
```

Tools

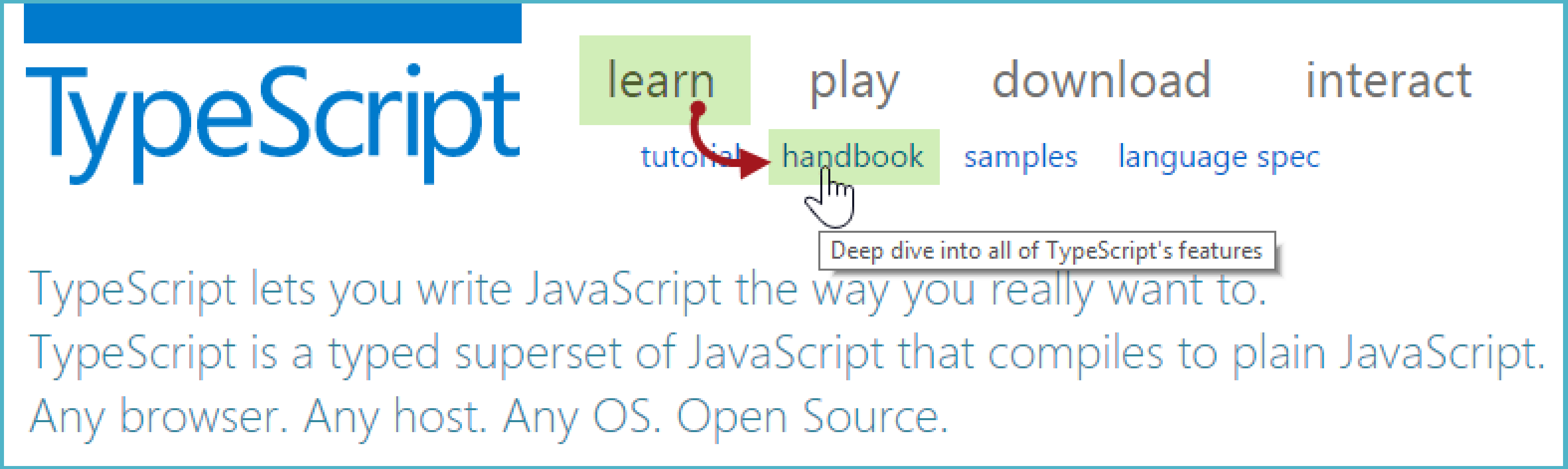
Visual Studio includes TypeScript in the box, starting with Visual Studio 2013 Update 2. You can also edit TypeScript in VS Code, WebStorm, Atom, Sublime Text, and Eclipse.

TypeScript 1.5beta for VS2013

TypeScript 1.5beta for VS2015



```
> tsc
Version 1.5.0-beta
Syntax:   tsc [options] [file ...]

Examples: tsc hello.ts
          tsc --out file.js file.ts
          tsc @args.txt

Options:
 -d, --declaration                 Generates corresponding '.d.ts'
 -h, --help                        Print this message.
 --mapRoot LOCATION                Specifies the location where de
nerated locations.
 -m KIND, --module KIND            Specify module code generation:
 --noEmit                          Do not emit outputs.
 --noEmitOnError                   Do not emit outputs if any type
 --noImplicitAny                   Raise error on expressions and
 --out FILE                        Concatenate and emit output to
 --outDir DIRECTORY                Redirect output structure to th
 --preserveConstEnums              Do not erase const enum declara
 -p DIRECTORY, --project DIRECTORY Compile the project in the give
 --removeComments                  Do not emit comments to output.
 --rootDir LOCATION                Specifies the root directory of
ctory structure with --outDir.
 --sourceMap                       Generates corresponding '.map'
 --sourceRoot LOCATION             Specifies the location where de
d of source locations.
 --suppressImplicitAnyIndexErrors  Suppress noImplicitAny errors f
.
 -t VERSION, --target VERSION      Specify ECMAScript target versi
mental)
 -v, --version                     Print the compiler's version.
 -w, --watch                       Watch input files.
 @<file>                           Insert command line options and
```



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.

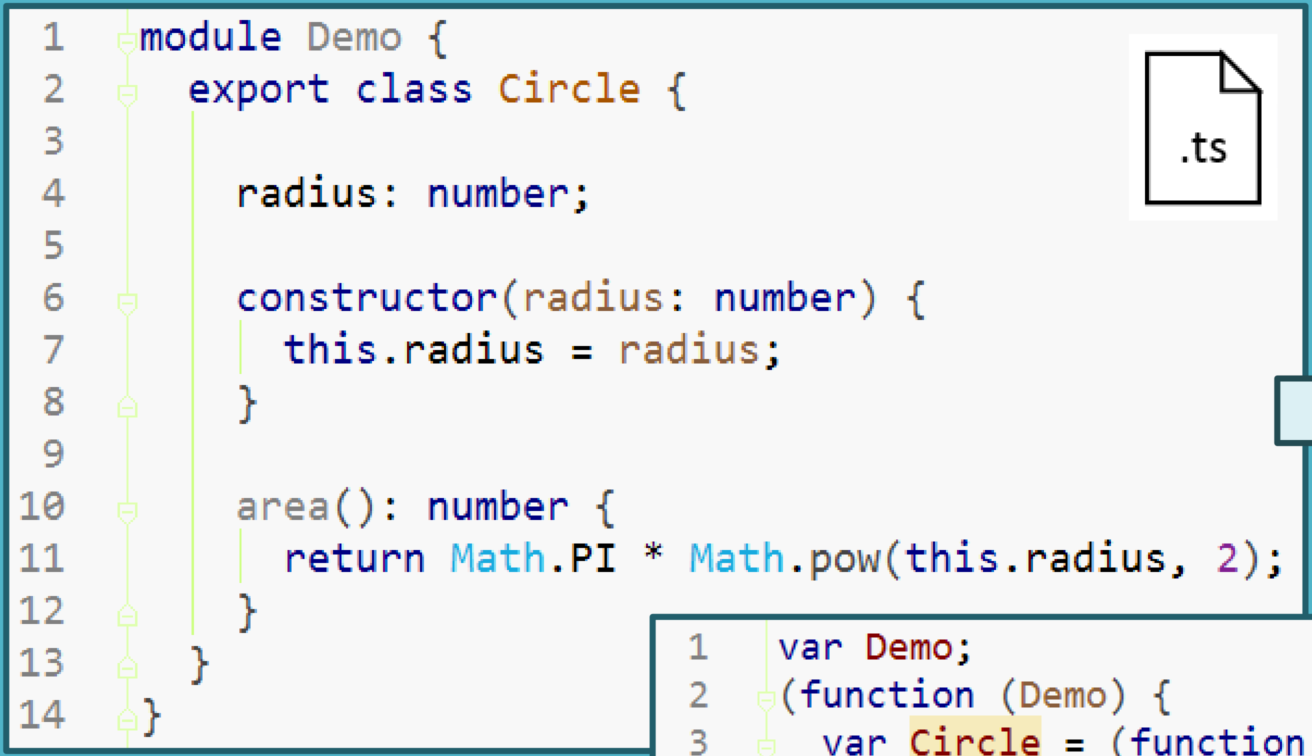
TypeScript Handbook

- Basic Types
  - Boolean
  - Number
  - String
  - Array
  - Enum
  - Any
  - Void
- Interfaces
- Classes
- Modules
- Functions
- Generics
- Common Errors
- Mixins
- Declaration Merging
- Type Inference
- Type Compatibility

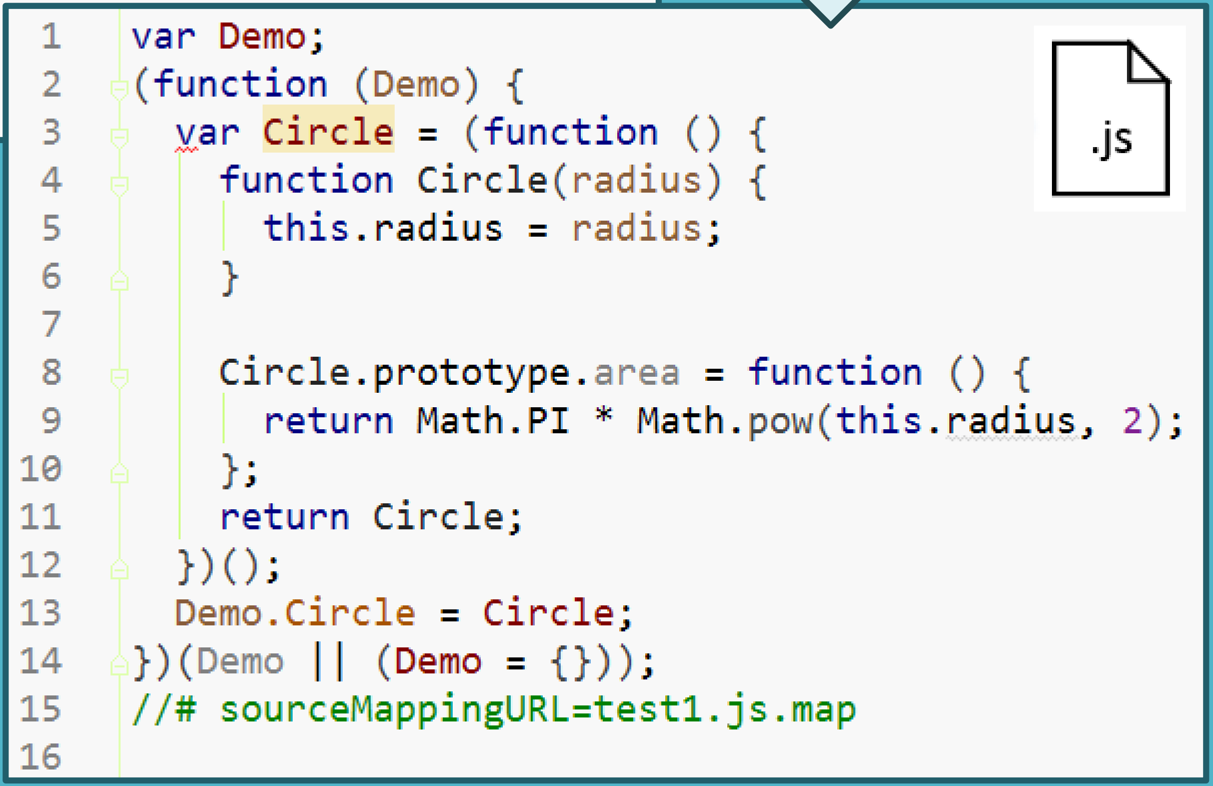
เริ่มหัวข้อแรก
ก็เห็นได้ชัดว่าเป็นเรื่อง Type

ความจริงแล้ว เรื่องถัดๆไป ก็ยังคงเป็นเรื่อง Type นะ

อ่านดูไปเรื่อยๆ แล้วจะพบว่า
ทั้งหมดนี้มันเป็นเรื่องของ Type ทั้งนั้นเลยนี่ :-^) เฮ!



```
module Demo {
  export class Circle {

    radius: number;

    constructor(radius: number) {
      this.radius = radius;
    }

    area(): number {
      return Math.PI * Math.pow(this.radius, 2);
    }
  }
}
```



```
var Demo;
(function (Demo) {
  var Circle = (function () {
    function Circle(radius) {
      this.radius = radius;
    }

    Circle.prototype.area = function () {
      return Math.PI * Math.pow(this.radius, 2);
    };
    return Circle;
  })();
  Demo.Circle = Circle;
})(Demo || (Demo = {}));
//# sourceMappingURL=test1.js.map

```

Type annotation

```
var x : number;
x = true; // error
x = 100; // ok

function print(s : string) : void {
   console.log(s);
}

print(x);               // error
print(x.toString());    // ok
```

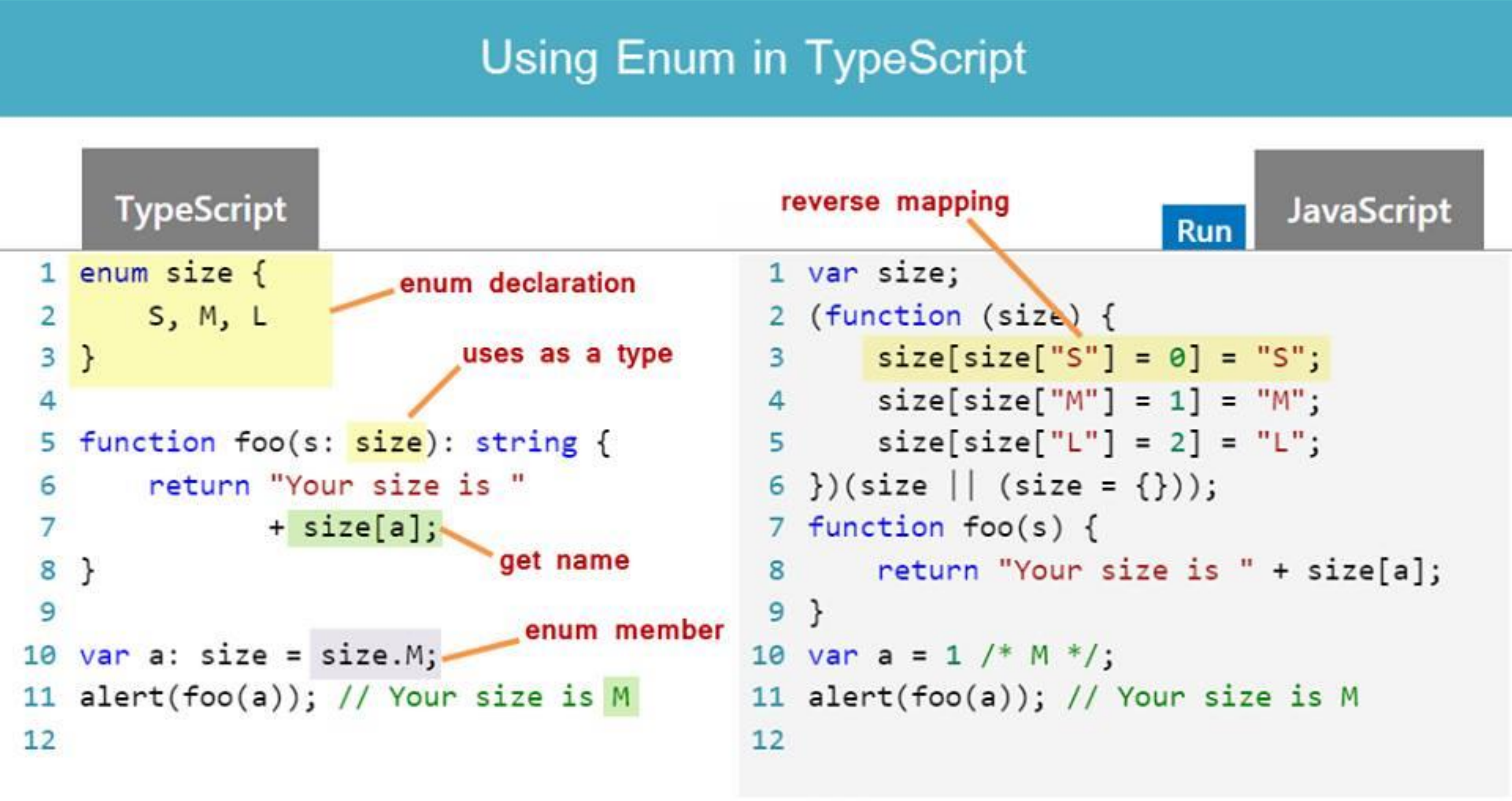
Type inference

```
var x = 10;  // infer x as a number
var x : number = 10;

// infer this function return type as string
function Foo(n: number) { return n.toFixed(2); }
function Foo(n: number) : string { … }
```

Basic types

boolean	// var b = true, c = false;
number	// var n = 100, m = 10.0;
string	// var s = 'Hello';
array	// var a1: number[] = [1, 3, 5];
array	// var a2: Array<number> = [ ];
enum	// var c = Color.red;
any	// var x;
void	// foo(): void { }

enum

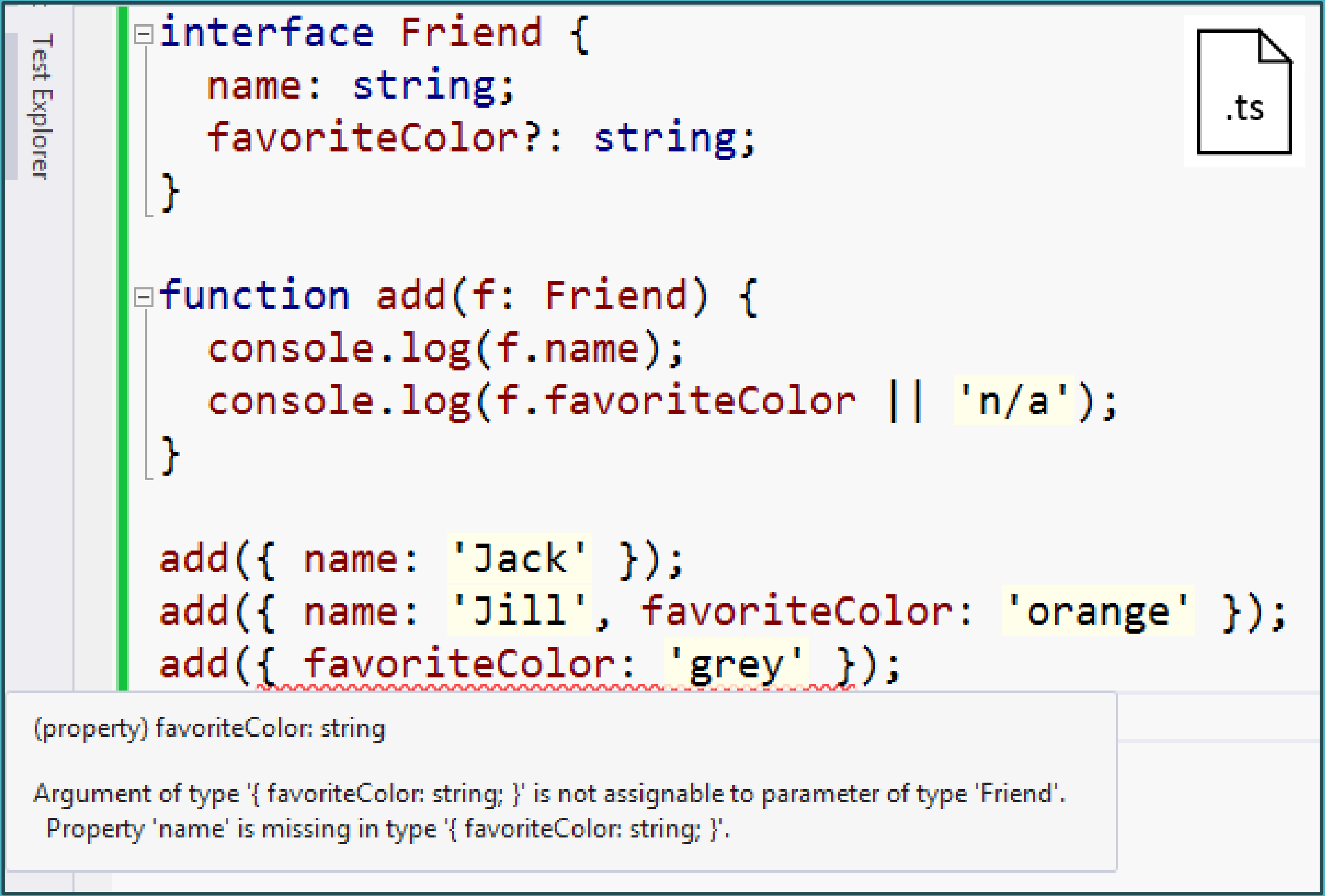


```typescript
enum size {
    S, M, L
}

function foo(s: size): string {
    return "Your size is "
           + size[a];
}

var a: size = size.M;
alert(foo(a)); // Your size is M
```

```javascript
var size;
(function (size) {
    size[size["S"] = 0] = "S";
    size[size["M"] = 1] = "M";
    size[size["L"] = 2] = "L";
})(size || (size = {}));
function foo(s) {
    return "Your size is " + size[a];
}
var a = 1 /* M */;
alert(foo(a)); // Your size is M
```

*Typo: please change size[a] to size[s]

Interface and duck typing



Test Explorer

.ts

```
interface Friend {
  name: string;
  favoriteColor?: string;
}

function add(f: Friend) {
  console.log(f.name);
  console.log(f.favoriteColor || 'n/a');
}

add({ name: 'Jack' });
add({ name: 'Jill', favoriteColor: 'orange' });
add({ favoriteColor: 'grey' });
```

```
(property) favoriteColor: string

Argument of type '{ favoriteColor: string; }' is not assignable to parameter of type 'Friend'.
  Property 'name' is missing in type '{ favoriteColor: string; }'.
```

Class

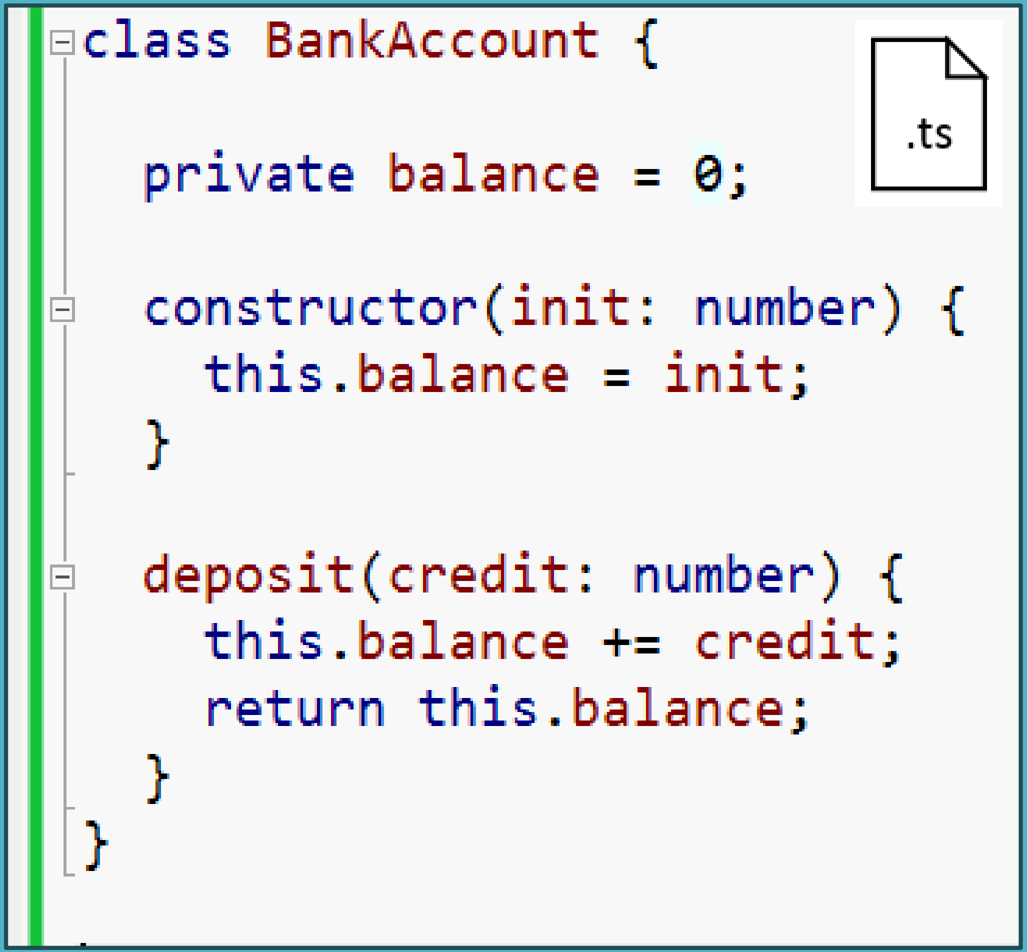
```
class BankAccount {
    balance = 0;
    deposit(credit: number) {
        this.balance += credit;
        return this.balance;
    }
}
```

.ts

```
var BankAccount = (function () {
    function BankAccount() {
        this.balance = 0;
    }
    BankAccount.prototype.deposit = function(credit) {
        this.balance += credit;
        return this.balance;
    };
    return BankAccount;
})();
```

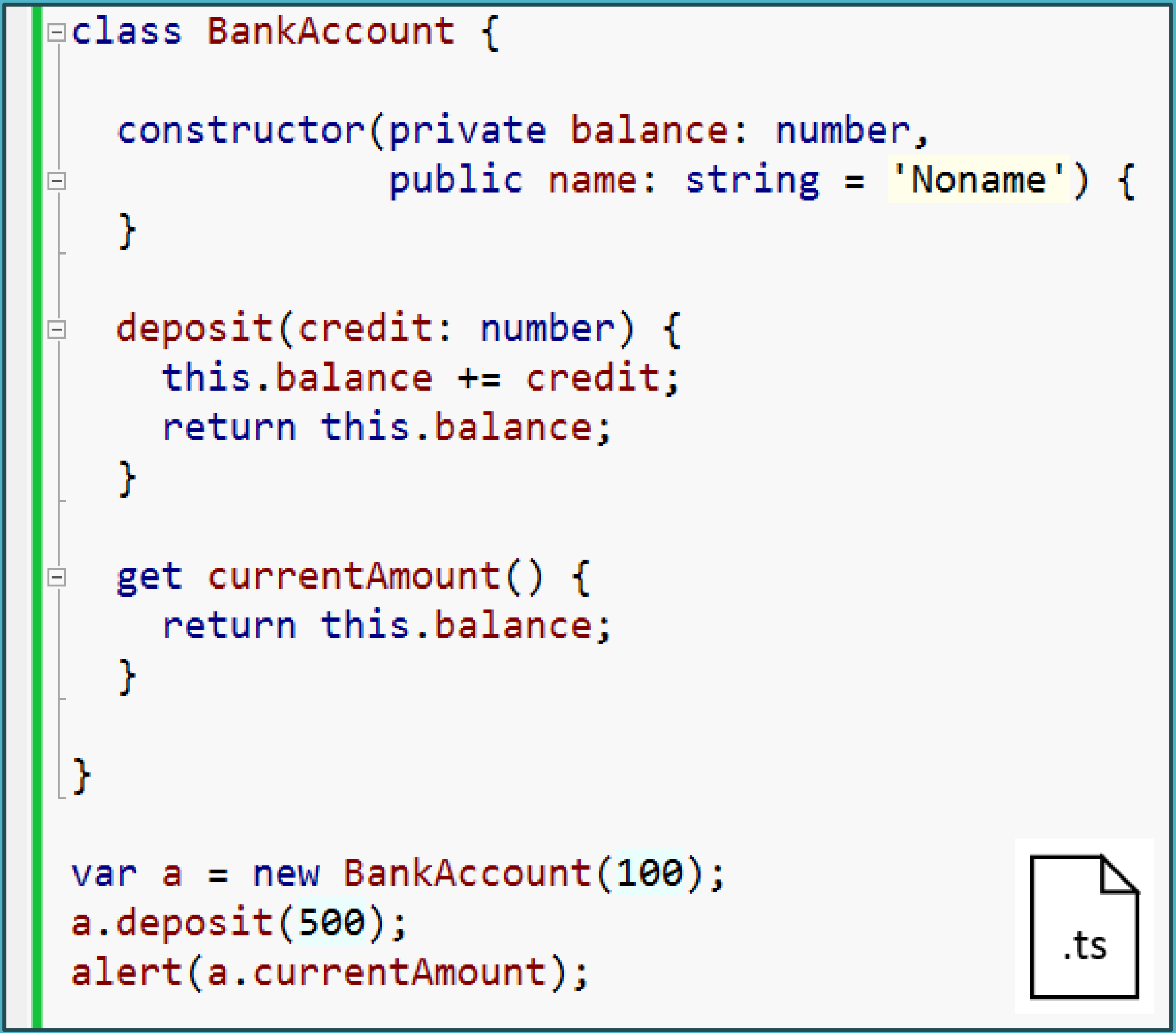
.js

Constructor & Private Members



```
class BankAccount {

  private balance = 0;

  constructor(init: number) {
    this.balance = init;
  }

  deposit(credit: number) {
    this.balance += credit;
    return this.balance;
  }
}
```

Parameter Properties & Accessors



```
class BankAccount {

  constructor(private balance: number,
              public name: string = 'Noname') {
  }

  deposit(credit: number) {
    this.balance += credit;
    return this.balance;
  }

  get currentAmount() {
    return this.balance;
  }

}

var a = new BankAccount(100);
a.deposit(500);
alert(a.currentAmount);
```

.ts

Note that accessor
(get & set)
requires ES5 output

Class Inheritance and the "super" calls

```
class CheckingAccount extends BankAccount {
    constructor(balance: number) {
        super(balance);
    }
    writeCheck(debit: number) {
        this.balance -= debit;
    }
}
```

Module

```
module M {
    var s = "hello";
    export function f() {
        return s;
    }
}

M.f();
M.s;  // Error, s is not exported
```

.ts

```
var M;
(function(M) {
    var s = "hello";
    function f() {
        return s;
    }
    M.f = f;
})(M||(M={}));
```

.js

Arrow functions

```
(x) => { return Math.sin(x); }
(x) => Math.sin(x)
x => { return Math.sin(x); }
x => Math.sin(x)
```

.ts

Arrow functions

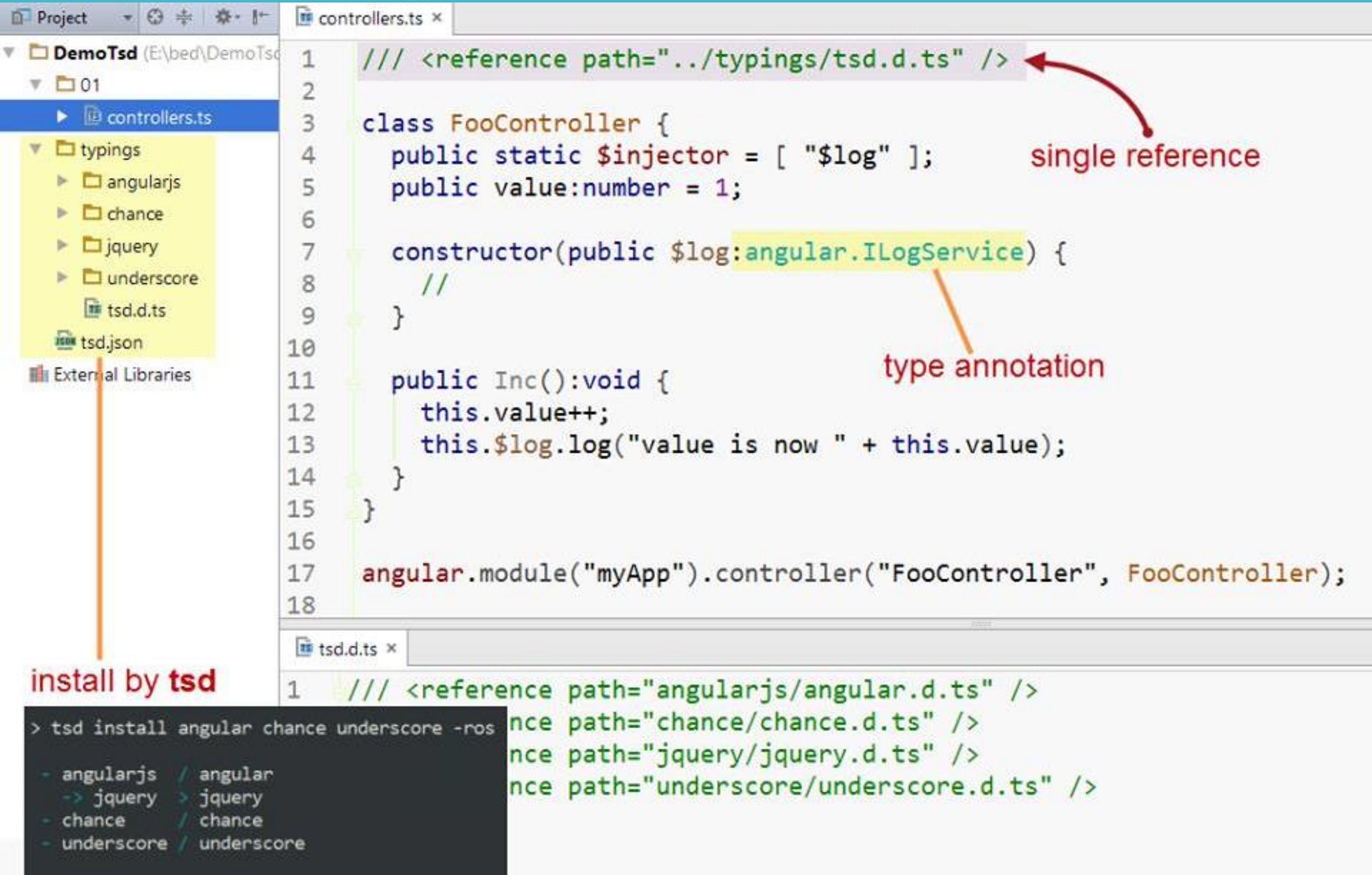
```
var bmi = (w, h) => w / Math.pow(h / 100, 2);

alert(bmi(65, 170));
```

.ts

```
var bmi = function (w, h) { return w / Math.pow(h / 100, 2); };

alert(bmi(65, 170));
```

.js

Using TypeScript with AngularJS



controllers.ts

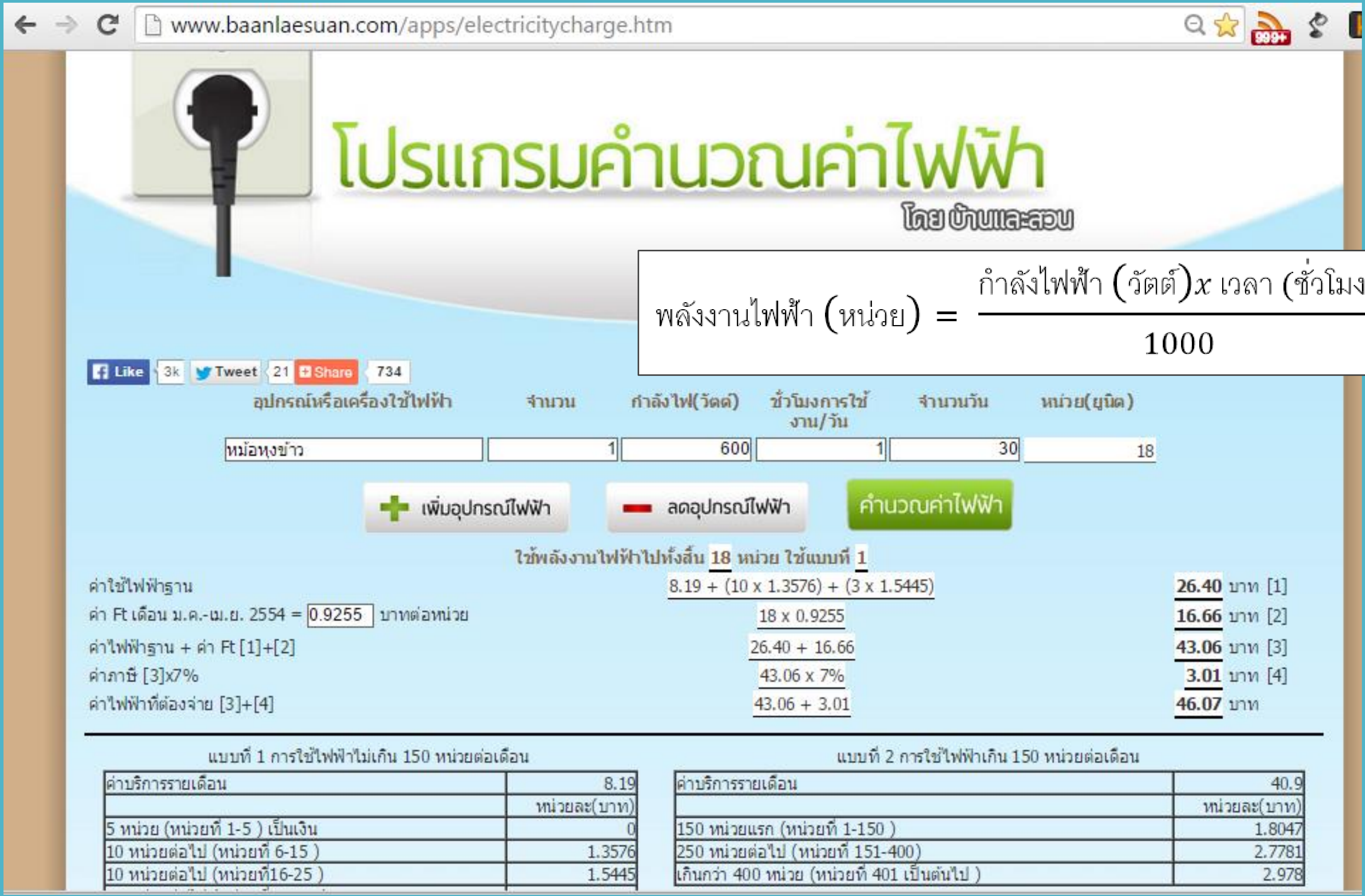
```
/// <reference path="../typings/tsd.d.ts" />

class FooController {
  public static $injector = [ "$log" ];
  public value:number = 1;

  constructor(public $log:angular.ILogService) {
    //
  }

  public Inc():void {
    this.value++;
    this.$log.log("value is now " + this.value);
  }
}

angular.module("myApp").controller("FooController", FooController);

```

tsd.d.ts

```
/// <reference path="angularjs/angular.d.ts" />
nce path="chance/chance.d.ts" />
nce path="jquery/jquery.d.ts" />
nce path="underscore/underscore.d.ts" />
```

```
> tsd install angular chance underscore -ros

 - angularjs  / angular
   -> jquery  > jquery
 - chance     / chance
 - underscore / underscore
```

http://www.baanlaesuan.com/apps/electricitycharge.htm



$$\text{พลังงานไฟฟ้า (หน่วย)} = \frac{\text{กำลังไฟฟ้า (วัตต์)} x \text{ เวลา (ชั่วโมง)}}{1000}$$

อุปกรณ์หรือเครื่องใช้ไฟฟ้า	จำนวน	กำลังไฟ(วัตต์)	ชั่วโมงการใช้งาน/วัน	จำนวนวัน	หน่วย(ยูนิต)
หม้อหุงข้าว	1	600	1	30	18

ใช้พลังงานไฟฟ้าไปทั้งสิ้น 18 หน่วย ใช้แบบที่ 1

ค่าใช้ไฟฟ้าฐาน	8.19 + (10 x 1.3576) + (3 x 1.5445)	26.40 บาท [1]
ค่า Ft เดือน ม.ค.-เม.ย. 2554 = 0.9255 บาทต่อหน่วย	18 x 0.9255	16.66 บาท [2]
ค่าไฟฟ้าฐาน + ค่า Ft [1]+[2]	26.40 + 16.66	43.06 บาท [3]
ค่าภาษี [3]x7%	43.06 x 7%	3.01 บาท [4]
ค่าไฟฟ้าที่ต้องจ่าย [3]+[4]	43.06 + 3.01	46.07 บาท

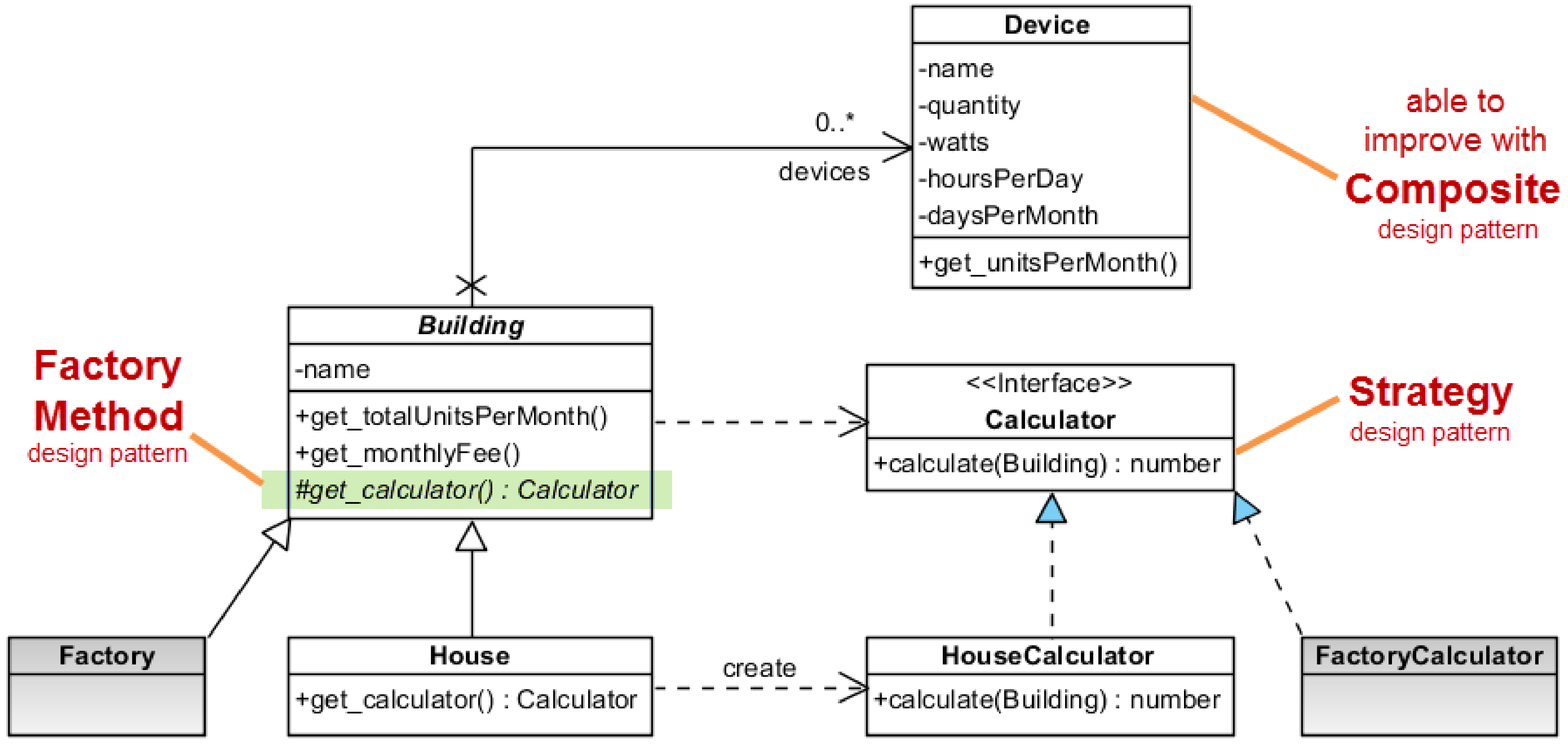
แบบที่ 1 การใช้ไฟฟ้าไม่เกิน 150 หน่วยต่อเดือน

ค่าบริการรายเดือน	8.19
	หน่วยละ(บาท)
5 หน่วย (หน่วยที่ 1-5) เป็นเงิน	0
10 หน่วยต่อไป (หน่วยที่ 6-15)	1.3576
10 หน่วยต่อไป (หน่วยที่16-25)	1.5445

แบบที่ 2 การใช้ไฟฟ้าเกิน 150 หน่วยต่อเดือน

ค่าบริการรายเดือน	40.9
	หน่วยละ(บาท)
150 หน่วยแรก (หน่วยที่ 1-150)	1.8047
250 หน่วยต่อไป (หน่วยที่ 151-400)	2.7781
เกินกว่า 400 หน่วย (หน่วยที่ 401 เป็นต้นไป)	2.978

Developing with TypeScript



Resources

http://facebook.com/groups/typescript.thailand

http://www.typescriptlang.org

http://www.typescriptlang.org/Handbook

http://blogs.msdn.com/b/typescript

http://definitelytyped.org

http://definitelytyped.org/tsd

https://github.com/microsoft/typescript

https://github.com/microsoft/typescript/wiki