

Introduction to TypeScript

Rajeev Gupta

-Trainer & consultant

What is TypeScript

- Free and open source, strongly supported by Microsoft
- Based on ecmascript 4 + ecmascript 6
- Created by the father of C# Anders Hejlsberg
- A superset of JavaScript
-
-
- To answer why we need JavaScript+, we need to understand what's wrong with vanilla JavaScript

JavaScript

1995
20 years ago

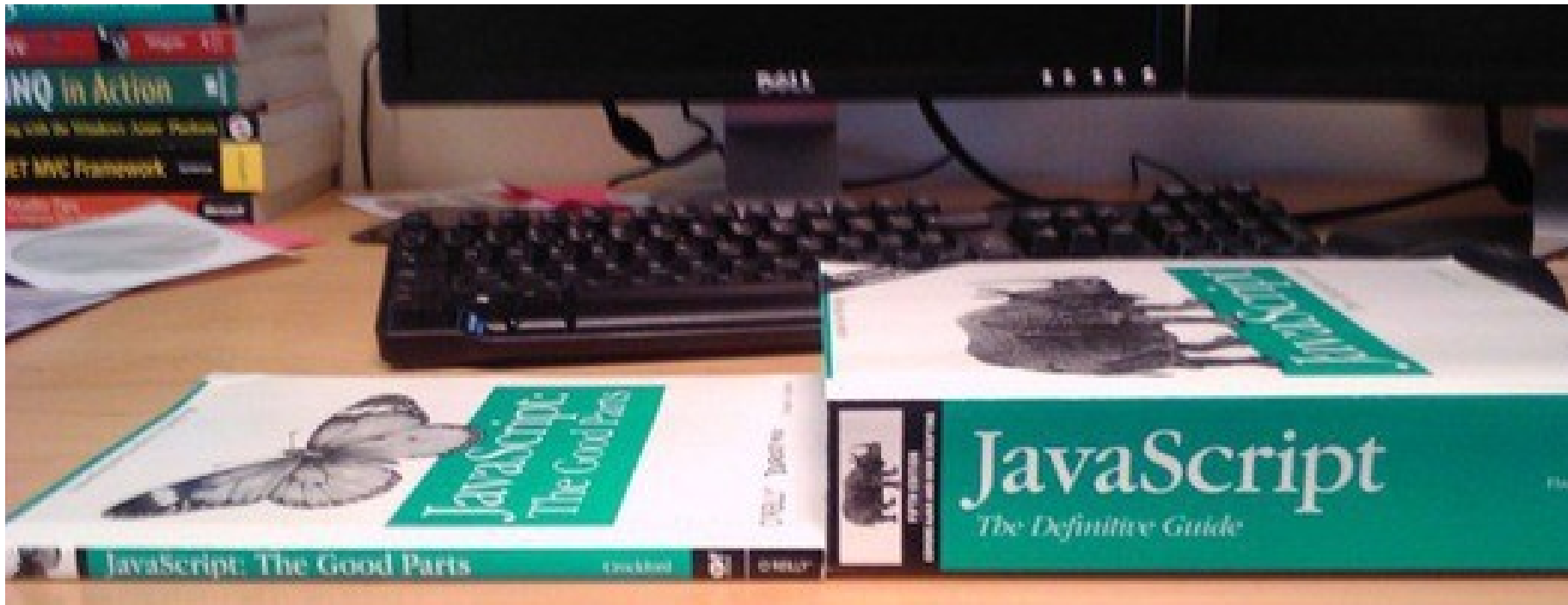
Dynamic
computer
programming
language

Brendan Eich
Netscape
Communica.,
Mozilla Foundation

Client side
Server side
(Node.js)
Single Page App

What is the problem

- Why do people hate working in JavaScript?



Application scale JavaScript development is hard

- not design as a programming language for big application
- does not have static typing
- lack structuring mechanisms like classes, modules, interfaces

Problem - dynamic types

- Variables are untyped and dynamic. They are flexible
- Bad because it is so easy to get wrong
- `var x = 1; var y = x + 1;`
- `// OK, type is inferred. can assume x and y are both numbers.`
-
- `var x = 1; x = "hello";`
- `// NOT OK, type is mixed up. We can't assume what type is x.`
-
- `// JS is interpreted. There are no design-time intellisense or compile-time assistance to help you point out errors`

Problem - scope

- JavaScript's scope looks like C#, but does not work at a block level. It is at the function level.
- It is so easy to get wrong

```
var i = 1;  
if (i == 1) {  
    var i = 2;  
}  
var y = function { var i = 3; }  
•
```

Problem - object inheritance is hard

- Based on object extension. Not class inheritance (at a syntax level)

```
var animal = {  
  var name;  
};
```

```
var cat = jQuery.extend( animal,  
  var claw = function() { /*claw*/ };  
• });
```

- //Syntax complicated, so nobody really does it.

Expert

“JavaScript is the assembly language of the Web.”

Erik Meijer (software architect)

Expert

“You can write large programs in JavaScript. You just can’t maintain them.”

Anders Hejlsberg (father of C#)

The Alternatives

- Hard core JavaScript development
- JavaScript preprocessors that compiles to JavaScript:



CoffeeScript
coffeescript.org
custom lang.



Dart
<http://dartlang.org>
custom lang.
by Google



Clojurescript
github.com
custom lang.



Script#
github.com
C#

TypeScript

- TypeScript is a typed superset of JavaScript that compiles to plain JavaScript.
- TypeScript is a language for application scale JavaScript development.
- Any browser. Any host. Any OS.
- Open Source.

~typescriptlang.org



Copy JS code



Into TS file



Compile to
JavaScript

TypeScript Key Features

- Support standard JavaScript code with static typing
- Zero cost: Static types completely disappear at run-time
- Encapsulation through classes, modules and interfaces
- Constructors, properties and functions (public, private)
- Enums
- Lambda and generics support
- Intellisense and syntax checking
- IDE support
 - Visual Studio
 - Sublime Text, Vi, Emacs
 - Eclipse, WebStorm
- Preview Pane – Web Essentials

What's Typescript?

- JavaScript is not originally designed for large complex applications (mostly a scripting language, with functional programming constructs), lacks structuring mechanisms like Class, Module, Interface.
- Typescript is a typed superset of JavaScript that compiles to plain JavaScript.
- Adds additional features like Static Type (optional), Class, Module etc. to JavaScript
- Microsoft technology.
- Open Source.
- Versions.
 - First made public in October 2012.
 - Latest version - Typescript 1.7.

Type Annotation

- Any
 - Any Type is a super set of all types
 - `var x : any;`
 - `var y;`
- Primitive
 - Number
 - Does not have separate integer and float/double type.
 - `var num : number = 20;`
 - `var num = 20;`
 - String
 - Both single quote or double quotes can be used.
 - `var name : string = "hello";`
 - `var name = 'hello';`
 - Bool
 - `var isOpen =true;`

Type Annotation

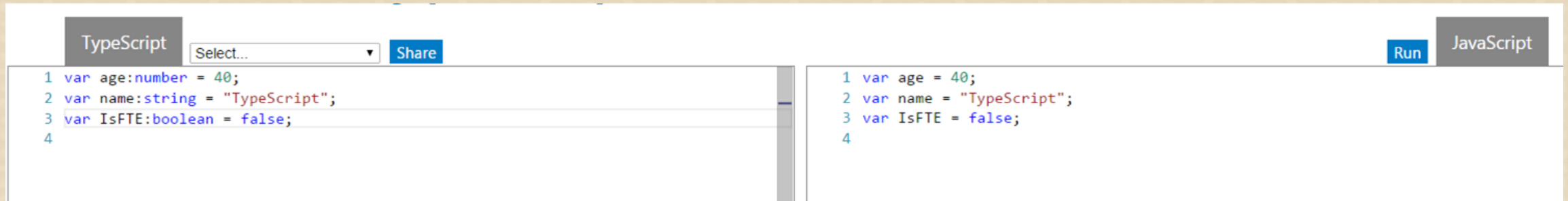
- Void
 - Used as the return type of functions that don't return any value
- Object Types
 - class, interface, module.
- Array
 - Array types can be written in:
 - `var list: number[] = [1, 2, 3];`
 - `var list: Array<number> = [1, 2, 3];`
 - `var list:any[] = [1, true, "free"]`
- Enum
 - `enum Color { Red, Green, Blue };`
 - `var color = Color.Blue;`

Type Annotation

- Tuple
 - Tuple types allow you to express an array where the type of a fixed number of elements is known.
 - `var x: [string, number];`
 `x = ['hello', 10];`

Type Annotation

- Design time feature. No additional code is emitted in the final JavaScript that TypeScript compiler produces.



The screenshot shows the TypeScript Playground interface. On the left, the 'TypeScript' tab is active, displaying the following code:

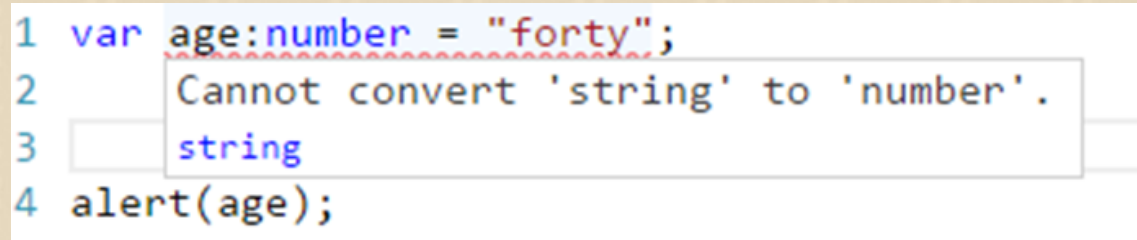
```
1 var age:number = 40;  
2 var name:string = "TypeScript";  
3 var IsFTE:boolean = false;  
4
```

On the right, the 'JavaScript' tab is active, displaying the compiled output:

```
1 var age = 40;  
2 var name = "TypeScript";  
3 var IsFTE = false;  
4
```

Buttons for 'Share' and 'Run' are visible between the two code editors.

If there's a type mismatch TypeScript shows a warning.



The screenshot shows a TypeScript error warning. The code in the editor is:

```
1 var age:number = "forty";  
2  
3 string  
4 alert(age);
```

A red squiggly line is under the string "forty" on line 1. A tooltip box displays the error message: "Cannot convert 'string' to 'number'." Below the message, the word "string" is listed, indicating the type that caused the error.

Functions

- Type Annotation for parameter and return type.
- Optional and Default Parameter.
- Function Overloads.
- Fat Arrow functions.
- Rest parameters.
 - denoted by '...argumentName' for the last argument allow you to quickly accept multiple arguments in your function and get them as an array.

Class

- Properties and fields to store data
- Methods to define behavior

TypeScript [Share](#)

```
1 class Employee{
2     private name:string
3     private basic:number
4     private allowance:number
5
6     constructor(name:string, basic:number, allowance:number){
7         this.name = name
8         this.basic = basic
9         this.allowance = allowance
10    }
11
12    public getSalary():number{
13        return this.basic + this.allowance
14    }
15 }
16
17 var emp = new Employee("Aniruddha",100,20)
18 alert(emp.getSalary())
```

[Run](#) JavaScript

```
1 var Employee = (function () {
2     function Employee(name, basic, allowance) {
3         this.name = name;
4         this.basic = basic;
5         this.allowance = allowance;
6     }
7     Employee.prototype.getSalary = function () {
8         return this.basic + this.allowance;
9     };
10    return Employee;
11 })();
12
13 var emp = new Employee("Aniruddha", 100, 20);
14 alert(emp.getSalary());
15
```

Inheritance

- Typescript supports inheritance of class through extends keyword
- super keyword.

Module

- Modules can be defined using module keyword.
 - A module can contain sub module, class, interface or enum.
 - Class, Interfaces , functions can be exposed using export keyword.
-
- Adding file references. - `/// <reference path="filename.ts" />`

Interface

- Declared using interface keyword
- TS compiler shows error when Interface signature and implementation does not match
- Optional properties can be declared for an interface (using ?)

Generics

- Able to create a component that can work over a variety of types rather than a single one.

```
function identity<T>(arg: T): T {  
    return arg;  
}
```

- type argument inference - we want the compiler to set the value of T for us automatically based on the type of the argument we pass in.

New features

- await & async
- for..of – iteration.
- Exponentiation operators - ** and **=
- Spread operator