TypeScript Introduction

By Dmitry Sheiko

I'm Dmitry Sheiko,
a web developer,
blogger, open source
contributor.

@sheiko
http://dsheiko.com

TypeScript
is a new language
developed by Microsoft

Why on earth?!

TypeScript is free and open source

TypeScript
syntax is based on
ECMAScript 4 and
ECMAScript 6 proposals

TypeScript
is first and foremost
a superset of JavaScript

So what do we gain?

We still write JavaScript, but augmented by class-based OOP of ES6 and structural type system of ES4. Our code is being compiled to regular JavaScript and supposed to work on any browser

Microsoft's TypeScript may be the best of the many JavaScript front ends. It seems to generate the most attractive code.

Douglas Crockford

..one thing TypeScript got right: local type inference.

Brendan Eich

What impressed me the most is what TypeScript doesn't do. It doesn't output type checking into your JavaScript code.

Nicholas C. Zakas

The Language

Type Annotations and Type Checking

```
var inx: number = 1,
    text: string = “Lorem”,
    isReady: bool = false,
    arr: string[],
    obj: ObjInterface = factory.getObj(),
    mixedVal: any;
```

```
var obj: { x: number, y: number },
    fn: ( x: number, y?: any ) => number,
    constr: new() => ObjInterface;
```

```
var treatItems = function( arr,
    callback ) {
   // do something
   return arr;
};
```

```
var treatItems = function(
    arr: string[],
    callback: (value: string,
                inx: number,
                arr: string[]) => string[]) {
        // do something
        return arr;
};
```

```
var treatItems = function( arr,
    callback ) {
   // do something
   return arr;
};
```

Classes

```
class Mamal
{
    private nickname: string;

    constructor( nickname: string = "Noname" ) {
        this.nickname = nickname;
    }

    public getNickname():string {
        return this.nickname;
    }
}
```

```
class Cat extends Mamal
{
    private family: string = "Felidae";

    constructor( nickname: string ) {
        super( nickname );
    }

    public getFamily():string {
        return this.family;
    }
}
```

```
// Generated JavaScript
var __extends = ...
var Mamal = (function () { ... })();
var Cat = (function (_super) {
    __extends(Cat, _super);
    function Cat(nickname) {
        _super.call(this, nickname);
        this.family = "Felidae";
    }
    Cat.prototype.getFamily = function () {
        return this.family;
    };
    return Cat;
})(Mamal);
```

Interfaces

```
interface Point {
    x: number;
    y: number;
}

function getDistance( pointA: Point, pointB: Point ) {
    return Math.sqrt(
            Math.pow( pointB.x - pointA.x, 2 ) +
            Math.pow( pointB.y - pointA.y, 2 )
        );
}

var result = getDistance(
            { x: - 2, y: - 3 }, { x: - 4, y: 4 })
```

```
interface Mover
{
    move(): void;
}
interface Shaker
{
    shake(): void;
}
interface MoverShaker extends Mover, Shaker
{
}
```

Modules

```
module graphic
{
        export class Point
        {
                x: number;
                y: number;
                constructor( x: number = 0, y: number = 0 )
                {
                };
        }
}
var point = new graphic.Point( 10, 10 );
```

```
// File main.ts:
import log = module ( "log" );
log.message( "hello" );

// File  log.js:
export function message(s: string) {
    console.log(s);
}
```

Arrow Expressions

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

```
var messenger = {
    message: "Hello World",
    start: function() {
        setTimeout( () =>
            { alert( this.message ); }, 3000 );
    }
};
messenger.start();
```

Type Assertions

```
class Shape { ... }
class Circle extends Shape { ... }

function createShape( kind: string ): Shape {
        if ( kind === "circle" ) return new Circle(); ...
}

var circle = <Circle> createShape( "circle" );
```

Ambient Declarations

```
declare var document:Document;
declare var screen;
declare var console;
```

```
interface JQuery
{
        text(content: string);
}
interface JQueryStatic {
        get(url: string, callback: (data: string) => any);
        (query: string): JQuery;
}
declare var $: JQueryStatic;
```

TypeScript definition file for YUI3

gist.github.com/3845543

Environment Setup

Install NodeJs

https://github.com/joyent/node/wiki/Installing-Node.js-via-package-manager

Install TypeScript compiler

```
npm install -g typescript
```

Compile a file

```
tsc example.ts
```

Compile as ECMAScript 5

```
tsc --target ES5 example.ts
```

Do you want it run-time?

https://github.com/niutech/typescript-compile

Include compilation task into Ant build script

```
<?xml version="1.0"?>
<!DOCTYPE project>
<project name="tsc"  basedir="." default="build">
    <target name="build">
        <!-- Compile all .ts files -->
        <apply executable="tsc" parallel="true">
            <srcfile/>
            <fileset dir="." includes="**/*.ts"/>
        </apply>
        <!-- Lint all required CSS, JS files -->
        <!-- Concatenate all required CSS, JS files -->
        <!-- Compress built CSS, JS files -->
    </target>
</project>
```

Build project

ant

Thank you!