Typescript in BigPanda

Why

- Types reduce amount of errors by 15%
- Developer Time is divided between Reading \ Refactoring \ Creating.
- By the end of this talk, you will know what typescript and how it helps you be more productive.

What

» What is typescript

A supert set of javascript!

Some basic types

```
let first_name: string = "dor"
const my_age: number = 28;
const isHungry: boolean = false;
interface Person {
    name: string,
    age: number
}

const person: Person = { name: first_name, age: my_age }
```

Arrays

```
const names : string[] = ["dor","noa"];
const namesArray : Array<string> =  ["dor","noa"];
```

Tuples

```
let x: [string, number] = ["hello", 10];
let name = x[0]; //"hello"
//let unknown = x[4];
//yields 'Tuple type '[string, number]' of length '2' has no element at index '4'.ts' !!!!
```

Intersection Types

» Allows to define conditional types on values

```
const id : string | string[] = null;
if(typeof(id)==="string"){
    //id is string here. compiler figures this out alone!
    id
}else{
    //id is string[] here. compiler figures this out alone!
    id
}
```

Algebric data types!!!

```
sealed trait Shape
case class Square(kind:String = "square", size:Int);
case class Rectangle(kind:String = "rectangle", height:Int, weight:Int);
case class Circle(kind:String = "circle", radius:Int)
def area(s:Shape):Int = {
  s match {
    case Square(k,size) => size
    case Rectangle(k,h,w) => h * w;
    case Circle(k,r) => Math.PI * r * 2;
  }
}
```

Algebric data types in type script!

```
interface Square {
    kind: "square";
    size: number;
}
interface Rectangle {
    kind: "rectangle";
    width: number;
    height: number;
}
interface Circle {
    kind: "circle";
    radius: number;
}

type Shape = Square | Rectangle | Circle ;

function area(s: Shape): number {
    switch (s.kind) {
        case "square": return s.size * s.size;
        case "rectangle": return s.height * s.width;
        case "circle": return Math.PI * s.radius ** 2;
    }
}
```

Algebric data types in type script!

```
interface Square {
    kind: "square";
    size: number;
}
interface Rectangle {
    kind: "rectangle";
    width: number;
    height: number;
}
interface Circle {
    kind: "circle";
    radius: number;
}
interface Triangle {
  kind: "triangle";
  height: number;
  length: number;
}
type Shape = Square | Rectangle | Circle| Triangle ;

//This will not compile!!!!
function area(s: Shape): number {
    switch (s.kind) {
        case "square": return s.size * s.size;
        case "rectangle": return s.height * s.width;
        case "circle": return Math.PI * s.radius ** 2;
    }
}
```

Union types

```
type LinkedList<T> = {current:T} & { next: LinkedList<T> } | undefined;
const people: LinkedList<string> = {current:"dor",next:{current:"bar",next:undefined}}
```

How to start using it

- TSC - the typescript compiler
- tsc --init
- live example

Quick intro to the benefits

» In vs-code

```
typescript
\\@ts-check
```

» JsDoc

ts.d files

» Contains function defenitions to interact with js code in typescript code.

@Types packages

```
"@types/bluebird": "3.5.29",
"@types/highland": "2.12.9",
"@types/lodash": "4.14.149",
"@types/node": "12.12.6",
```

» Just contains the original code, and the header files of the library

Imports change

```
const metrics = require('setup/statsd');
```

» Now turns to:

```
import metrics from '../setup/statsd'
```

More work

- Norm work! (thank you @adam)
- Labels work!

Missing

- Converting express requests to interfaces
- Some mongodb examples

Call to action

- When reviewing code!
- Incremental work!

Some videos

» https://www.youtube.com/channel/UCtxCXg-UvSnTKPOzLH4wJaQ