Basic Typescript

hands on experince on basic typescript features

Saman Taghavi

What is Typescript?

TypeScript is JavaScript with syntax for types.

- 🐞 **Errors** - some of them can be noticed before deployment
- 🎨 **Tools** - most Editors support it and have tools for it
- 🧑‍💻 **Developer Experience** - if You know it well enough otherwise use any

  ```
  let s: any = "I don't care enough";
  ```

- 🧙 **Inference** - Typescript understands JS, it can infer when possible
- 🎥 **Progressive** - we can add it, incrementally and opt-in

```
// @ts-check
function oh(array) {
  if (orray.length > 0) {
    return array;
  }
}
```

Read more about Typescript

Basic Types

see if you can solve this simple problem from

```
// * 1. Boolean
var isDone = false;

// * 2. Number
var hex = 0xf00d;
var binary = 10;

// * 3. String
var color = "blue";
color = "red";

// * 4. Array
var list = [1, 2, 3];
var list2 = ["test", "not a test", "array of s
```

later on we will try to solve more problems from this repo, shout out and thanks to **@mattpocockuk**

ENUMS

```typescript
type ColorEnum = "red" | "blue" | "green";

let ENUM_COLOR: ColorEnum = "blue";
```

Types and Interfaces

```typescript
type BasicColor = string;

interface ColorInterface {
  // ! interface is mostly used for objects
  red: BasicColor;
  green: BasicColor;
  blue: number;
  colorGenerator: (colorName:string)=>string
}

type ColorByType = {
  red: BasicColor;
  green: string;
  blue: number;
  colorGenerator: (colorName:string)=>string
};
```

Typing Functions

```typescript
// inline typing
function add(x: number, y: number): number {
  return x + y;
}
// or use Interface and types
// a type
type minus = (firstNumber:number,secondNumber:number)=> number
const x :minus = (firstNumber:number,secondNumber:number)=> firstNumber - secondNumber
// an interface
interface IMinus {
  (firstNumber: string, secondNumber: string): number;
}
const y :IMinus = (firstNumber:number,secondNumber:number)=> firstNumber - secondNumber
```

Thank you