

**var g = "Sam"**

**TypeScript => JS + types**

**Browser donot understand it => it compiles to JS**

**Helps catch error before running the code**

**Dynamically typed language**

**Statically typed language**

**int a=45**

**var b=45**

**let age = 23**

**age="Sam"**

**let age:number=23**

**age="Sam"**

## WHY Typescript?

- 1. catch errors /prevent bugs**
- 2. API response structures are predictable**
- 3. Prop types become safer**
- 4. Teams avoid the runtime errors**

```
let price:number=54
let name:string ="Sam"
let isActive:boolean =True
let marks:number[] =[12,13,12,14,15,16,23]
```

```
function add(x:number,y:number):number{
    return x+y
    // Type Aliase
    // Interface
    // Object
}
```

The image shows a presentation slide with a light gray background. At the top, there is a toolbar with various icons: a close button (X), a refresh button, a download button, a dropdown arrow, a font style icon (F), a color palette icon, a minus sign, a plus sign, and an equals sign. On the right side of the slide, there is a circular arrow icon.

**Todo App**

**Form =>**

**one input (enter todo)**

**one button (Add todo)**

**todos ["learning React", "Going Gym", "Learn Python"]**

**completedTodos[""]**

**Map the todos**

**- learning React button1 button2**

**button1 => completed task**

**button2 => delete task**