TypeScript Variables

TypeScript Variables

TypeScript Variables are named storage locations to store a type of or dynamic type of value and can be referenced by a name called identifier. In this tutorial, we shall learn about variables, rules in writing an identifier for a variable, how a variable can be declared, initialized, and we shall also discuss about the scope of variables based on their placement in the program.

Syntax of variable declaration

Following is the syntax of variable declaration :

```
var <identifier> [: type-annotation] [= value] ;
```

- **var** keyword, **<identifier>** and the semicolon at the end are mandatory.
- **type-annotation** followed by colon is optional.
- initialization of the variable is also optional. To initialize or load a variable, **equal to** operator is used.

Following are some of the different types of declaring a variable :

- Only variable declaration Examples: `var n1` `var a`
- Variable declaration along with type-annotation specified. Examples: `var n1:number` `var a:string`
- Variable declaration along with value initialization. Examples: `var n1 = 25` `var a = 'hello'`
- Variable declaration along with type annotation and value initialization. Examples: `var n1:number = 25` `var a:string = 'hello'`

Rules to be followed in naming an identifier

- Alphabets and numbers are allowed.
- First character of variable name cannot be a number.
- Underscore(_) and Dollar($) sign are the only allowed special characters. Variable names cannot contain any other special characters or spaces.

`name` , `a1` , `a2` , `ght6$` : are some of the allowed identifiers for a variable.

`34shf` , `sdf#lsk` , `num@` : are some of the identifiers that are not allowed.

Scope of Variables

Scope is specific block of the code where a variable can be accessed. There are three types of variables based on the scope. They are :

1. **Global scope** : Variables that do not belong to any class and can be accessed from any part of the code.
2. **Class scope** : These variables are members of class and can be accessed by the member functions of that class.

3. **Local scope** : Variables that are declared insides a construct and are destroyed once the control is out of that construct. The construct can be a loop, function, if block, case block, etc.

Static Variables

Static variables are like in Java, not bound to any object and can be accessed using class name. Their value can be modified and the change in state is saved across the program life-cycle.

```
var n1:number = 569          // global variable

class Person {
   name:string              //class variable

   static sval = 10         //static property

   printSomething():void {
      var message:string = 'hello!'    //local variable
   }
}
```

Conclusion

In this TypeScript Tutorial – **TypeScript Variables**, we have learnt the role of variables in TypeScript, their syntax for declaration, initialization and also the different scopes of variables with examples.

TypeScript

- TypeScript Tutorial
- TypeScript Variables
- TypeScript if
- TypeScript if-else
- TypeScript Switch
- TypeScript For Loop
- TypeScript While Loop
- TypeScript Do-While Loop
- TypeScript Arrays
- TypeScript Tuples
- TypeScript Unions
- TypeScript Functions
- TypeScript Anonymous Functions

TypeScript Object Oriented Concepts

- TypeScript Class
- TypeScript Inheritance
- TypeScript Method Overriding
- TypeScript Interface

TypeScript String Operations

- TypeScript String Length
- TypeScript String Split