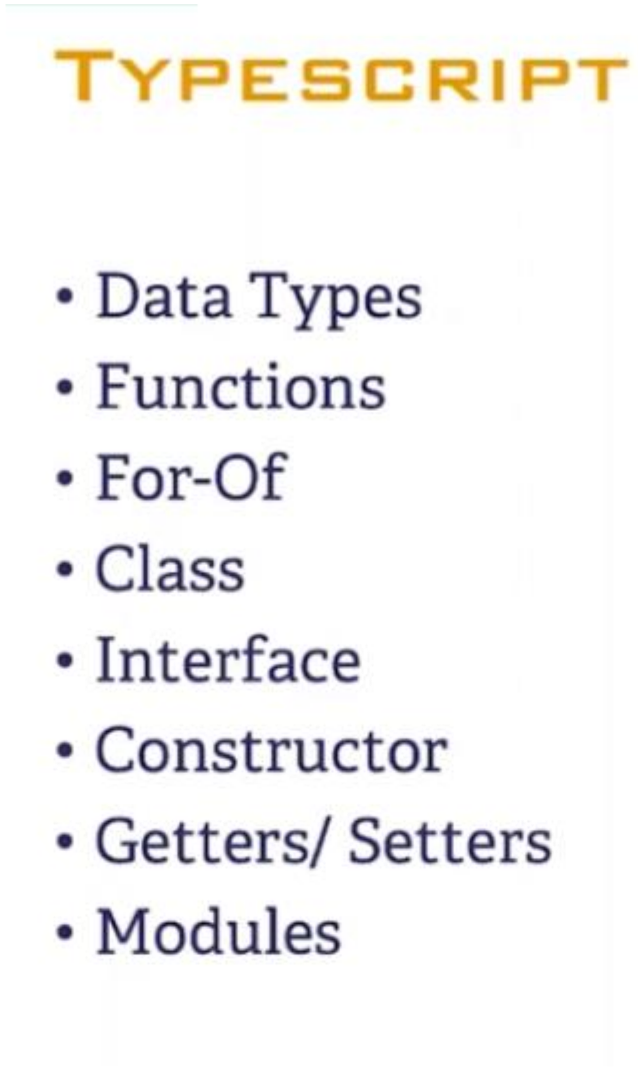
Typescript

Typescript content



TYPESCRIPT

- Data Types
- Functions
- For-Of
- Class
- Interface
- Constructor
- Getters/ Setters
- Modules

Installation

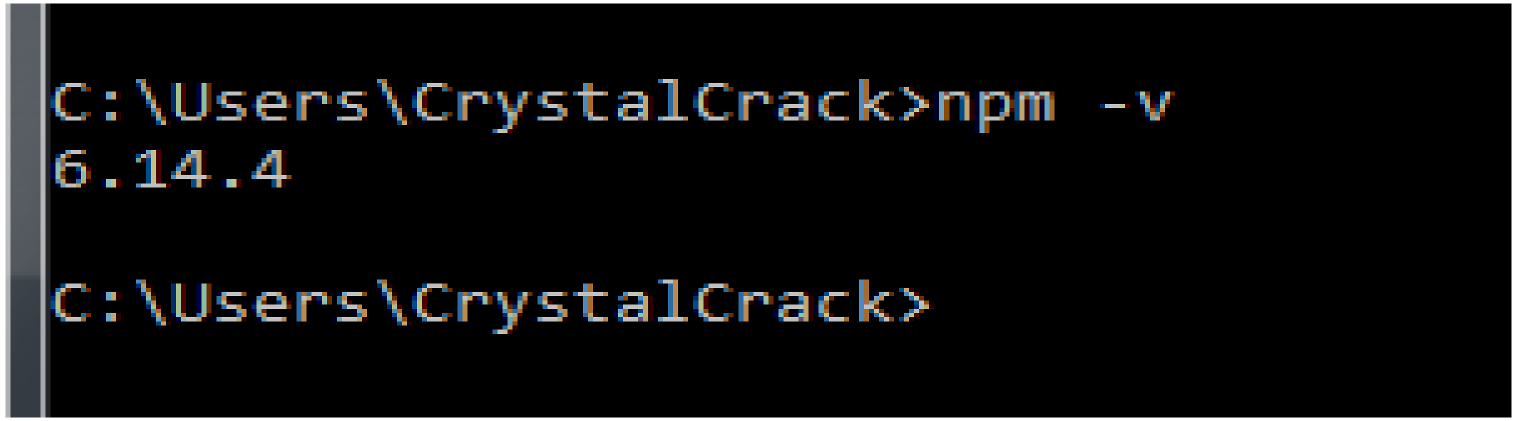
- NodeJS
- Typescript
- Angular CLI
- Visual Studio Code

Installation

- Install node.js from https://nodejs.org/en/download/
  - Check if node installed by typing node –v and npm -v on command prompt
- Installing TypeScript using npm
  - npm install –g typescript
- Installing the Angular CLI using npm
  - npm install -g @angular/cli
  - ng –v [to test if angular installed]
- Select editor of your choice to start creating angular apps
  - We will be using VSCode
  - Download from : https://code.visualstudio.com

Install node js For windows 7

- Install node js https://nodejs.org/download/release/v13.14.0/
- Click on the link
- node-v13.14.0-x64.msi 29-Apr-2020 19:58 29904896
- To check version open cmd prompt enter >>>npm –v >>enter

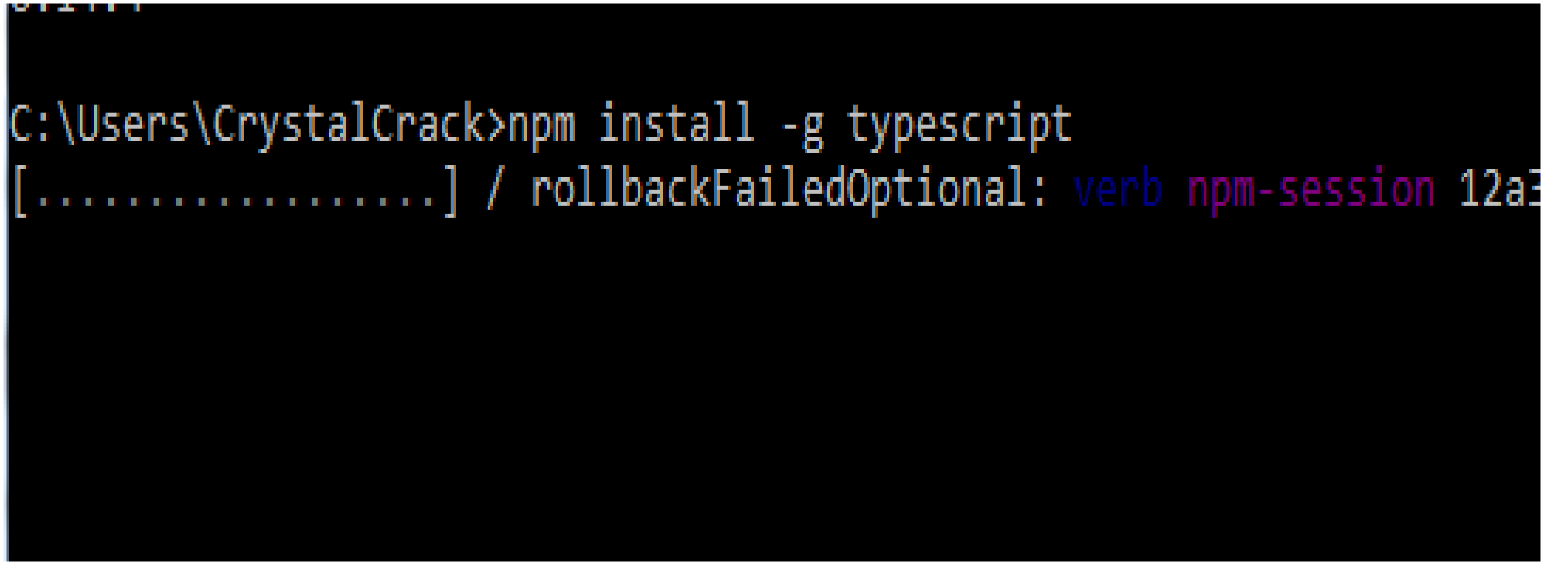


```
C:\Users\CrystalCrack>npm -v
6.14.4

C:\Users\CrystalCrack>
```

Install typescript

- Globally Installing TypeScript

- Npm install -g typescript >>>>>enter
-



```
C:\Users\CrystalCrack>npm install -g typescript
[...................] / rollbackFailedOptional: verb npm-session 12a3
```

Download visualstudio

- https://code.visualstudio.com/download
- Click on the icon windows 7,8,10 download.
- Install it.thats all!!!!!!!!

- The name typescript it indicates its based on types.
- Javascript is dynamic based type where as java .net c c++ which are static based types.
- Typescript supports full object oriented programming and principals.

- Two benefits
- It compiles source code into javascript.
- It can run on any operating system capable of executing javascript.

Why should we learn typescript.

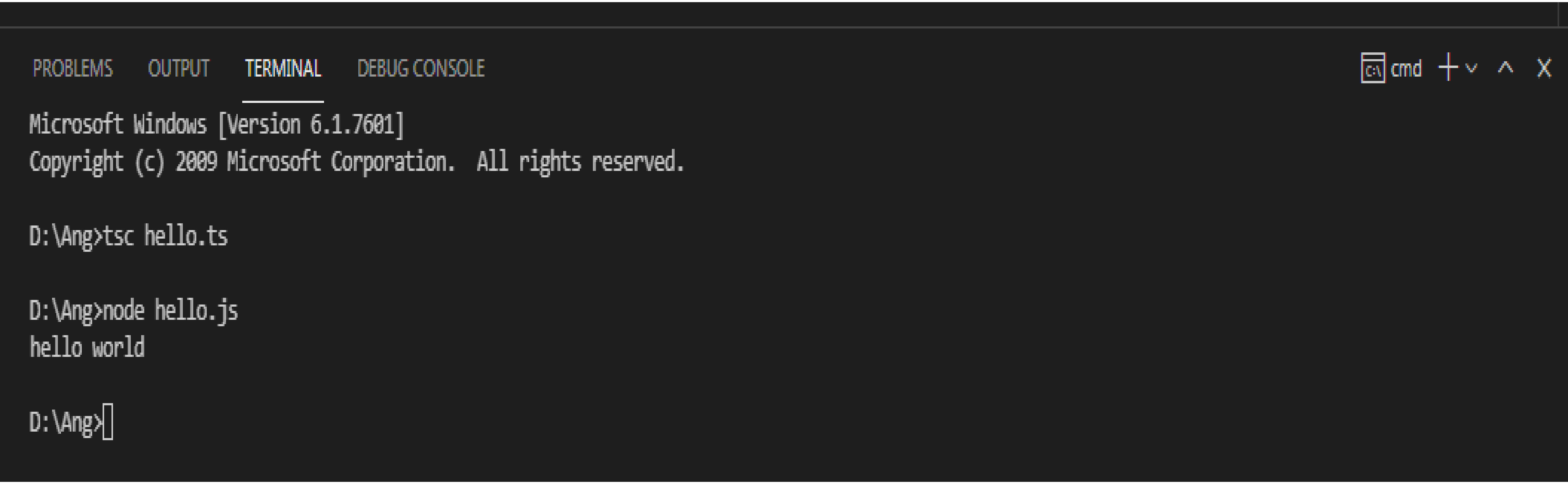
- Typescript the name indicates type safety,and it enhance code quality and understandability.
- Javascript is a typescript and vice versa.
- Types can be implicity.>what ever you assigned the type,strictly type based.
- Types can be explicity.>you want to store in a variable.
- Types are structural
- Type error do not prevent emit javascript code>means one typescript page have errors once you compile you wont identity in javascript,(because in javascript wont identity the errors)
- so make sure clear all the errors in typescript page and compile it ok.

If error while executing tsc hello.ts

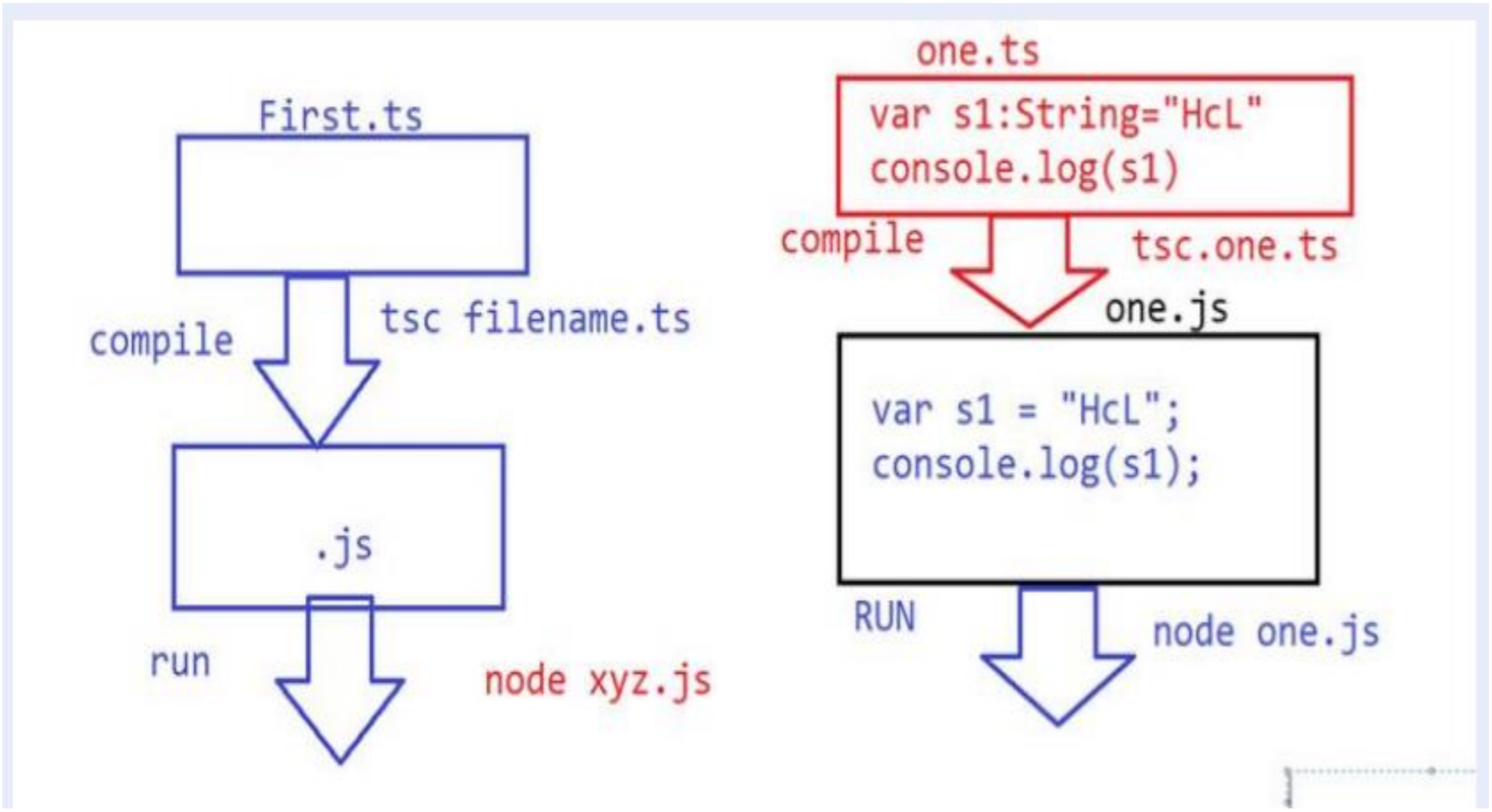
- Then problem with powershell.
- Kill the terminal
- Press Ctrl+Shift+P to show all commands.
- Type **profile** in the displayed text box to filter the list.
- Select **Terminal**: Select **Default** Profile.
- You will be prompted to Select your preferred terminal shell, you can change this later in your settings or follow the same process as we do now
- Select Command Prompt (cmd.exe)
- Go near explorer right click create new integrated termal >>default should be cmd otherwise do once again.

Execution of typescript.

- tsc hello.ts
- Once js file is created then execute it using node,because node is a server.
- Node will understand the js file ,becaz internally it has javascript engine.
- Node hello.js



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

D:\Ang>tsc hello.ts

D:\Ang>node hello.js
hello world

D:\Ang>
```



```
tsc filename.ts
```

```
node xyz.js
```

```
var s1:String="HcL"
console.log(s1)
```

```
tsc.one.ts
```

```
var s1 = "HcL";
console.log(s1);
```

```
node one.js
```

Hello.ts

- Sp document based concepts wont work
- Like alert(“hello world”);>>it wont work. It is not defined.

- `console.log("hello world");`
- `console.log("now its working fine");`
- `//alert("alert me baby");`
- `var x=10;`
- `console.log("x will work or no baba " + x);`

Main.ts

```
function Add(a,b)
{
    return a+b;
}
console.log("aree baba adding the number "+ Add(10,5));
```

Add 2 files in 1 file

- D:\Ang>tsc hello.ts main.ts --out app.js
- When you open app.js >>how you written exactly same will it will be add.

- But using wild its not working because of powershell
- D:\Ang>tsc *.ts --out hey.js>>it wont work: error

Standards output

- /**
- * standard outputs
- * tsc hello.ts
- * tsc hello.ts main.ts
- * tsc *.ts --out app.js >>not working alternate
- * tsc hello.ts main.ts --out app.js
- * tsc hello.ts --watch
- */

Alert add in ts file>> watch >>automatic added to ts>>add in index.html>>open index in browser

- Hello.ts
- `alert("alert me baby");`
- Automicatilly sameas in hello.js by watch applied
- Create index.html page add it

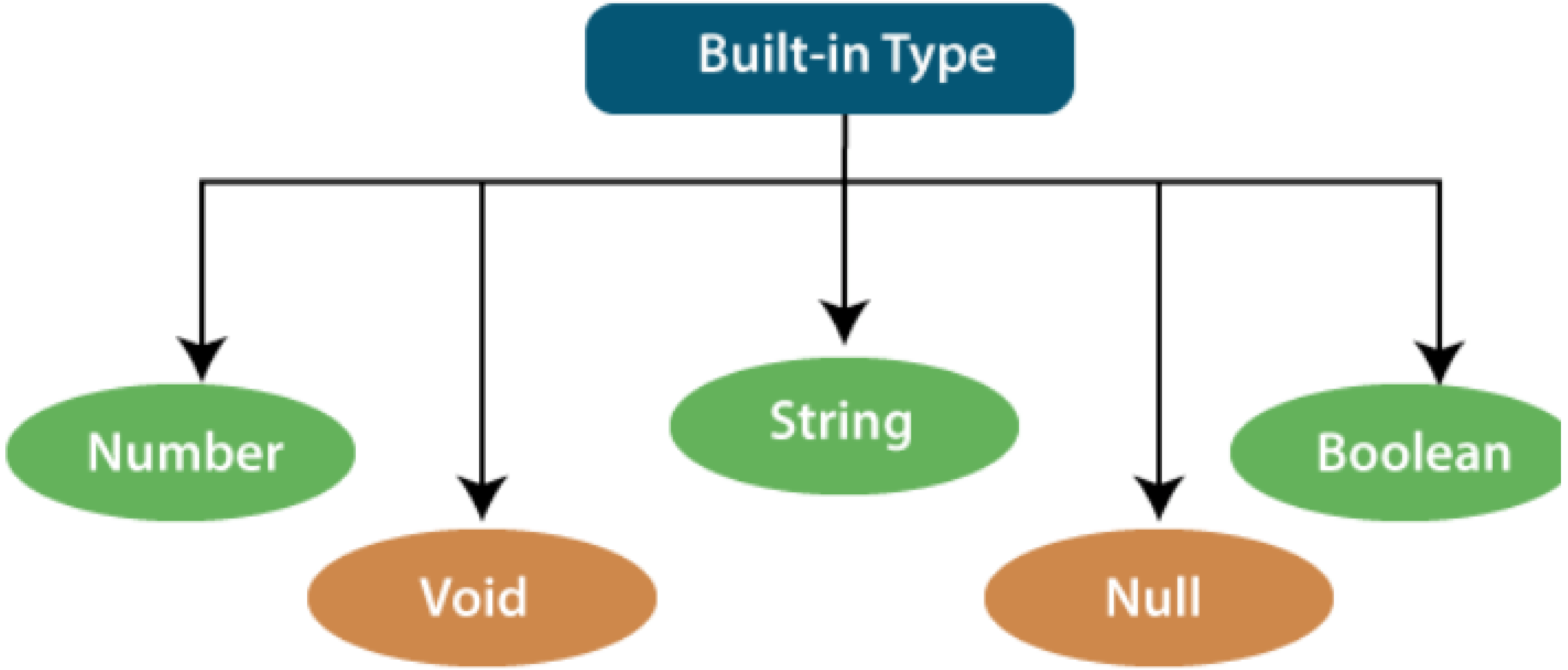
```
<!DOCTYPE html>
<html>
    <script src="hello.js"></script>
<body>
    <h1>this refer to js file</h1>
</body>
</html>
```

Want to increase the font size

- Open VS Code.
- Type command CTRL + SHFT + P.
- Type Settings.>>indicates >>preference:open user settings>>click on it
- In user>>text editor>>font>>increase the size and close it.

Using type ANY,then apply number,then String

- //we can pass anything using any for vaiable
- var no:any=10;
- console.log(no);
- no="cs;"
- console.log(no);
- --
  var no1=10;
- console.log(no1);
- //no1="cs";//already occupied with number
- ---
  var no2="cs";
- console.log(no2);
- no2=10;// already occupied with String

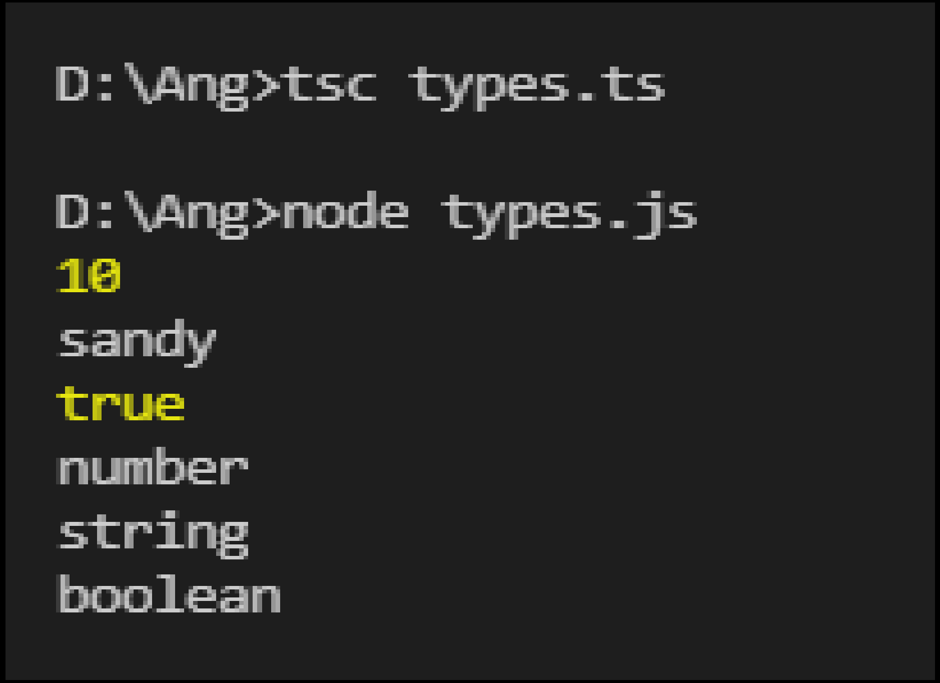

Built-in Type
Number
Void
String
Null
Boolean

Data type	Keyword	Description
Number	number	It represents a double precision 64-bit floating point values which can be used to represent both integers and fractions.
String	string	It represents a sequence of characters.
Boolean	boolean	It represents logical values true and false.
Void	void	It is used on function return types to represent non-returning functions.
Null	null	It is used to represent an intentional absence of an object value.
Undefined	undefined	It represents uninitialized variables.

Number,String,boolean

- Javascript doesnot support types.they are just normal variables.
- `var no:number=10;`
- `var uname:String=“cs";`
- `var choice:boolean=true;`
-

  `console.log(no);`
- `console.log(uname);`
- `console.log(choice);`
- `console.log(typeof(no));`
- `console.log(typeof(uname));`
- `console.log(typeof(choice));`



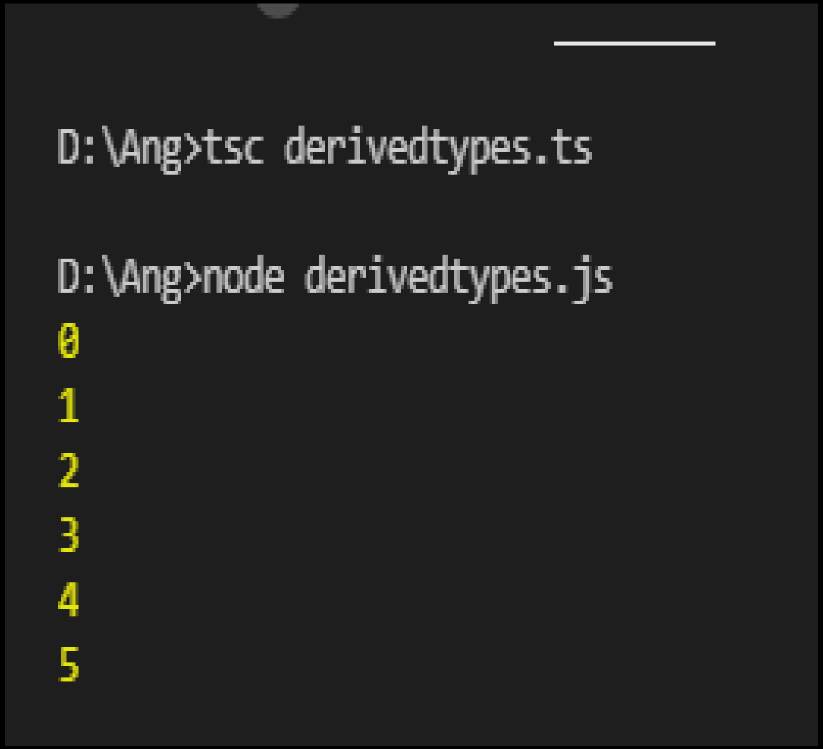
```
D:\Ang>tsc types.ts

D:\Ang>node types.js
10
sandy
true
number
string
boolean
```

Var and let

- var=globally
- Let=locally
- Constant value cannot be changed.
- `var x=10;`
- `let y=20;`
- `if(x==10){`
- `    var i=y+89;//if you replace var to let will have error`
- `    }`
- `console.log(j);`
- `const c=100;`
- `c=250;//again assign it wont work`

Derived types means arrays concept

- `var nu:number[]=[1,2,3,4,5];`
- `for(var i=0;i<=nu.length;i++)`
- `console.log(i);`



```
D:\Ang>tsc derivedtypes.ts

D:\Ang>node derivedtypes.js
0
1
2
3
4
5
```

TemplateString its interpolation

```
• //above the tab key
• // ` backticks  …will work as it is like space.
• var cname = ‘hcl tss';
• // ${} -> string interpolation
• let description = `How are you? ${cname}
•     Hope having fun learning angular




• ;
• console.log(description);
```

Generics ,before generics ,this page will get error

```
• function reverse(items:number[]) //strict type number
• {
•     var revnos = [];
•     for (var i = items.length-1 ; i>=0 ;i--)
•     {
•         revnos.push(items[i]);
•     }
•     return revnos;
• }
• var sample = [ 1,2,3,4,5];
• var reversenos = reverse(sample);
• console.log(reversenos);
• var names =["shalini","navin","vihaan"];
• var revnames = reverse(names); //cant pass ,its already number type
• console.log(revnames);
```

Generics,   That is the reson we came up with concept generics

```
• function reverse<T>(items:T[])//now it’s a type>>either num or String
• {
•       var revnos = [];
•       for (var i = items.length-1 ; i>=0 ;i--)
•       {
•           revnos.push(items[i]);
•       }
•       return revnos;
• }
• var sample = [ 1,2,3,4,5];
• var reversenos = reverse(sample);
• console.log(reversenos);
• var names =["shalini","navin","vihaan"];
• var revnames = reverse(names);//now no problem with String concept
• console.log(revnames);
```

Functions (return and void)

```
function display(name:string):string //return a value
{
    return "Welcome "+name;
}
console.log(display('Shalini'));
function show():void{                //wont return a value
   // return "hello";
}
```

•

Function>>optional agruments

```
//optional arguments
// required
//n3 -> optional
function add(n1:number,n2:number,n3?:number)
{
    if(n3 === undefined)
    {
        console.log(n1+n2);
    }
    else
        console.log(n1+n2+n3);
}
add(1,2);
add(1,2,3);
```

Functions Default arguments

```
//default arguments
function message(food:string, drinks:string = 'pepsi')
{
    console.log(`Have this tasty ${food} along with ${drinks}
`);
}
message('pizza');  //default it will load pepsi
message('noodles','lemonade');//explicity loaded.
```

Functions Rest Parameters

```
• //rest parameters
• function greet(company, ...names)
• {
•       console.log(names.length);
•       console.log(`${company} welcomes you ${names[3]}`);
• }
• 
  greet(‘MyTraining');
• greet(‘CS',‘Ram Krishna',‘babu','riya','sandy');
```

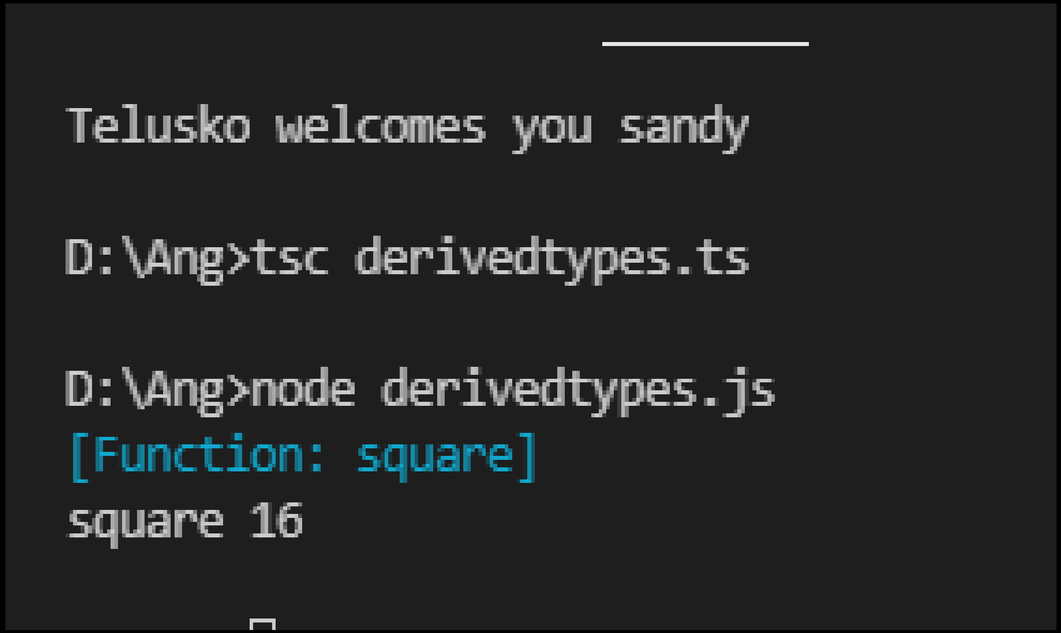
Arrow Functions

- Normal function

```
function sq(x)
{
    console.log(x*x);
}
//arrow
var square = (p) => {
    console.log("square "+ p*p);
  //  return p*p;
}
square(4);
```

Arrow function return and void

```
//arrow
var square = (p:number):number => {
    console.log("square "+ p*p);
    return p*p;
}
console.log(square);
square(4);
```

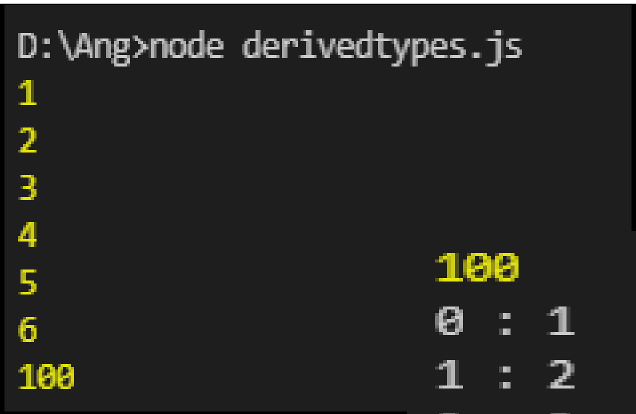


```
Telusko welcomes you sandy

D:\Ang>tsc derivedtypes.ts

D:\Ang>node derivedtypes.js
[Function: square]
square 16
```

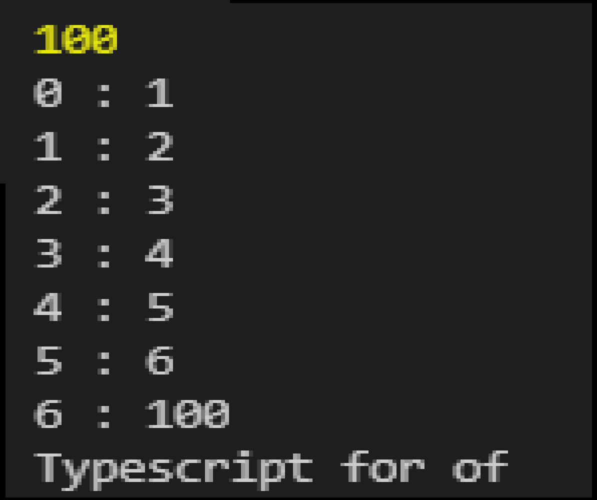
For of

```
var nos = [1,2,3,4,5,6,100];
for(var i =0; i< nos.length;i++)
{
    console.log(nos[i]);
}
for(var j in nos)
{
    console.log(j+" : "+nos[j]);
}

//typescript -> for -of
console.log("Typescript for of");
for(var n of nos)
{
    console.log(n);
}
```

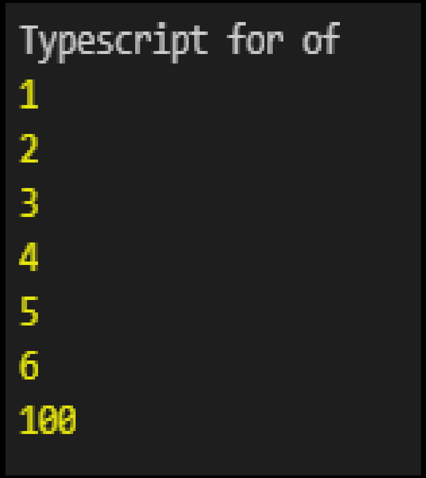
- //its new type no need to call index directly it will be load the data



```
D:\Ang>node derivedtypes.js
1
2
3
4
5
6
100
```



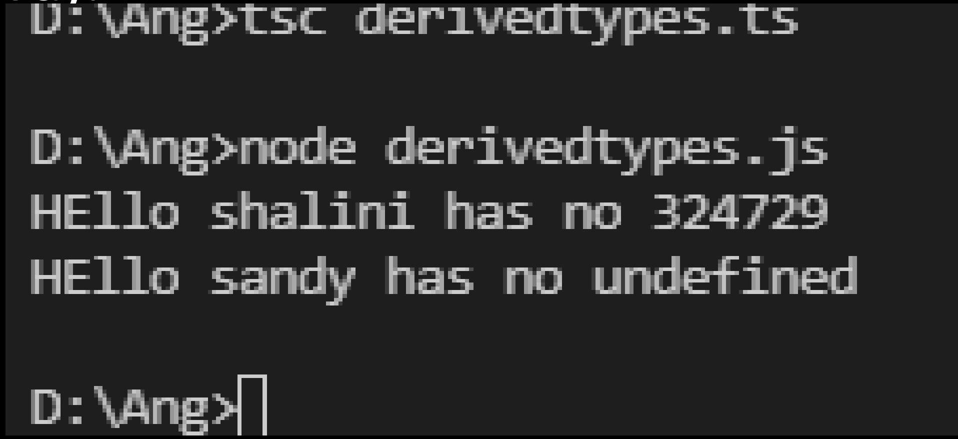
```
100
0 : 1
1 : 2
2 : 3
3 : 4
4 : 5
5 : 6
6 : 100
Typescript for of
```



```
Typescript for of
1
2
3
4
5
6
100
```

interface

- Interface just refer to data store(don't compare with java).
- To mark a structure of particular data.



```
D:\Ang>tsc derivedtypes.ts

D:\Ang>node derivedtypes.js
HEllo shalini has no 324729
HEllo sandy has no undefined

D:\Ang>
```

```
interface Person{
    name:string,
    phone?:number
}
function displaydetails(person:Person)//structure
{

    console.log("HEllo "+person.name + " has no "+ person.phone);
}

var p1 = {name : 'shalini',phone :324729};//data store
displaydetails(p1);
displaydetails({name :'sandy'});
```

interface

- An interface is a way to define a contract on a function with respect to the arguments and their type.
- Javascript does not support interface

Class

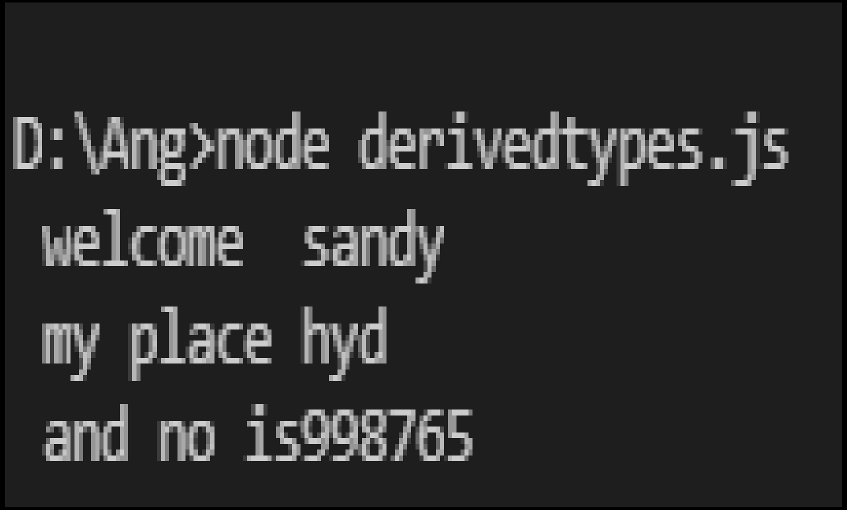
```
class User{
    //data members of the class user
    name:string;
    city:string;
    phone:number;
}

//user1 -> object
var user1 = new User();
user1.name='sandy';
user1.city='hyderabad';
user1.phone=1234512345;
console.log(" welcome  "+user1.name +"\n my place "+user1.city+"\n
 and no is" +user1.phone)
```

•

Constructor

```
class User{
    //data members of the class user
    name:string;
    city:string;
    phone:number;
    constructor(uname:string,city:string,phone:number)
{
    this.name = uname;
    this.city = city;
    this.phone = phone;
}
}

//user1 -> object
var user1 = new User('sandy','hyd',998765);
console.log(" welcome  "+user1.name +"\n my place "+user1.city+"\n and no is" +user1.
phone)
```

•



```
D:\Ang>node derivedtypes.js
 welcome  sandy
 my place hyd
 and no is998765
```

Getter and setter (inside user class constructor and getter method)

```
class User{
    constructor(private uname:string,private city:string,private phone:String) //private
{
    this.phone='+91-'+this.phone;
}
 //getters or accessors
 public get Name()
 {
     return this.uname; //constructor name
 }
 public get City()
 {
     return this.city; //constructor name
 }
 public get Phone()
 {
     return this.phone; //constructor name
 }
}
```

Calling the getter method

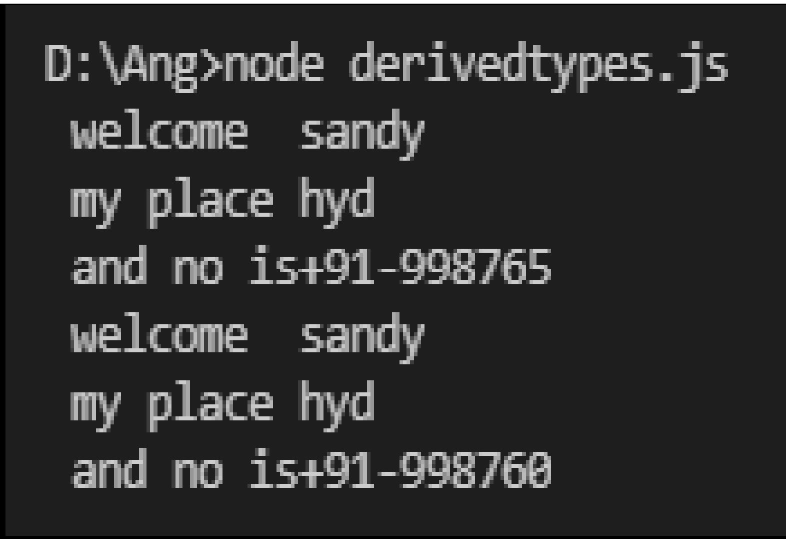
```
/user1 -> object
var user1 = new User('sandy','hyd','998765');
console.log(" welcome  "+user1.Name +"\n my place "+user1.City+"\n and no is" +user1.Phone)


//execution using ec5 without it will show error.
```

- `tsc derivedtypes.ts --target es5`

- `node derivedtypes.js`

Calling setter method

```
class User{
    constructor(private uname:string,private city:string,private phone:String) //private
{
    this.phone='+91-'+this.phone;
}
 //getters or accessors
 public get Name()
 {
     return this.uname; //constructor name
 }
 public get City()
 {
     return this.city; //constructor name
 }
 public get Phone()
 {
     return this.phone; //constructor name
 }
 public set Phone(ph:String){
        this.phone='+91-'+ph;
 }
}

//user1 -> object
var user1 = new User('sandy','hyd','998765');
console.log(" welcome  "+user1.Name +"\n my place "+user1.City+"\n
 and no is" +user1.Phone)
user1.Phone='998760'
console.log(" welcome  "+user1.Name +"\n my place "+user1.City+"\n
 and no is" +user1.Phone)
```



```
D:\Ang>node derivedtypes.js
 welcome  sandy
 my place hyd
 and no is+91-998765
 welcome  sandy
 my place hyd
 and no is+91-998760
```

- //output   >tsc derivedtypes.ts
-            >node derivedtypes.js

Functions in class

- Every time am writing console.log
- Now I want to use function,using function I will display the records.
- Inside the class we are calling so using this. Will use.
- Functions can also appied for passing arguments and include void.

Functions….

```
class User{
    constructor(private uname:string,private city:string,private phone:String)
{
    this.phone='+91-'+this.phone;
}
 //getters or accessors
 public get Name()
 {
     return this.uname;
 }
 public get City()
 {
     return this.city;
 }
 public get Phone()
 {
     return this.phone;
 }
public set Phone(ph:String){
        this.phone='+91-'+ph;
 }
 public display():void
 {
   console.log(this.Name+" welcome here, details\n city : "+ this.
City+" \nPhone "+ this.Phone);
 }
}

//user1 -> object
var user1 = new User('sandy','hyd','998765');
user1.display();
user1.Phone='0098765';
user1.display();
```

Modules

- If I want to access the user class in other module then we use module.
- Which ever class I want to use user class then we need use import
- We need to write export in user class.

Modules user class export and module file import

```
• export class User{
•       constructor(private uname:string,private city:string,pri
  vate phone:String)
• {
•       this.phone='+91-'+this.phone;
• }
```

- ....go on
- Module.ts file

```
• import {User} from './derivedtypes'
• var obj = new User('josmine','trivendram','78956412');
• obj.display();
```

- //output tsc modules.ts --target es5