TYPE SCRIPT ALL EXAMPLES

// Hello World

let message="Welcome To TypeScript";

console.log(message)

//TypeScript if

ifExample.ts

```
var a:number = 1
var b:number = 3

if(a == 1){
    console.log("value of a is 1.")
}

if(a == b){
    console.log("a and b are equal.")
}
```

Example 1 – TypeScript If Else

Following is an example TypeScript code to demonstrate if conditional statement.

example.ts

```
var a:number = 1
var b:number = 3

if(a == 1){
    console.log("value of a is 1.")
} else {
    console.log("value of a is not 1.")
}

if(a == b){
    console.log("a and b are equal.")
} else {
    console.log("a and b are not equal.")
}
```

When the above code is compiled using typescript compiler, `tsc example.ts` , following JavaScript code is generated.

example.js

```
var a = 1;
var b = 3;
if (a == 1) {
    console.log("value of a is 1.");
}
else {
    console.log("value of a is not 1.");
}
if (a == b) {
    console.log("a and b are equal.");
}
else {
    console.log("a and b are not equal.");
}
```

For this example, you might notice that there is no difference between TypeScript and JavaScript **if-else** code blocks.

TypeScript If Else If

Instead of a code block following else keyword, another if statement could be placed forming an if-else-if ladder statement.

example.ts

```
var a:number = 1
var b:number = 3

if(a == b){
    console.log("a and b are equal.")
} else if (a>b) {
    console.log("a is greater than b.")
} else if (a<b) {
    console.log("a is less than b.")
}
```

TypeScript If Else If

Instead of a code block following else keyword, another if statement could be placed forming an if-else-if ladder statement.

example.ts

```
var a:number = 1
var b:number = 3

if(a == b){
    console.log("a and b are equal.")
} else if (a>b) {
    console.log("a is greater than b.")
} else if (a<b) {
    console.log("a is less than b.")
}
```

Example – TypeScript Switch

Following is a simple typescript switch statement example, where the expression is addition of two variables, and the cases has some of the possible values. Only the case block that matches its value with the expression gets executed.

example.ts

```
var a:number = 1
var b:number = 3

switch (a+b){
    case 1 : {
        console.log("a+b is 1.")
        break
    }
    case 3 : {
        console.log("a+b is 3.")
        break
    }
    case 4 : {
        console.log("a+b is 4.")
        break
    }
    case 5 : {
        console.log("a+b is 5.")
        break
    }
    default : {
        console.log("a+b is 5.")
        break
    }
}
```

Typical Example of TypeScript for loop

Following is a typical example of for loop, where a counter is initialized, and incremented by 1 each time the loop is executed and the condition is such that the counter does not exceed a limit value.

example.ts

```
for(var counter:number = 1; counter<10; counter++){
    console.log("for loop executed : " + counter)
}
```

Output

```
for loop executed : 1
for loop executed : 2
for loop executed : 3
for loop executed : 4
for loop executed : 5
for loop executed : 6
for loop executed : 7
for loop executed : 8
for loop executed : 9
```

Example for-in loop with array

A working example is provided below where for-in loop is applied on an array of numbers :

example.ts

```
var arr:number[] = [10, 65, 73, 26, 44]

for(var item in arr){
    console.log(arr[item])
}
```

Output

```
10
65
73
26
44
```

example.ts

```
var student = [10, "John", "Spanish"]

for(var item in student){
    console.log(student[item])
}
```

Output

```
10
John
Spanish
```

Example 1 – Simple While Loop

Following is a simple while loop iterating for N times.

example.ts

```
var N = 4
var i = 0

while(i<N){
    // set of statement in while loop
    console.log(i)
    i++ // updating control variable
}
```

Output

```
0
1
2
3
```

Example – do-while loop

Following is an simple do-while loop example.

example.ts

```
var N = 6
var i = 1

do {
    console.log(i)
    i++ // updating control variable
} while(i<N)
```

Output

```
1
2
3
4
5
```

//USING FUNCTIONS

example.js

```
function sum(a, b) {
    var result = a + b;
    return result;
}
var c = sum(10, 12);
console.log(c);
```

//CLASS EXAMPLES

example.ts

```
class Student{
    // variables
    rollnumber:number
    name:string

    // constructors
    constructor(rollnumber:number, name:string){
        this.rollnumber = rollnumber
        this.name = name
    }

    // functions
    displayInformation():void{
        console.log("Name : "+this.name+", Roll Number :
"+this.rollnumber)
    }
}

var student1 = new Student(2, "Rohit")
var student2 = new Student(4, "Kohli")

// accessing variables
console.log(student1.name)
console.log(student2.rollnumber)

// calling functions
student1.displayInformation()
student2.displayInformat
```

Example – TypeScript Inheritance

In this example, we shall consider Person as Parent class and Student as Child class. Justification is that, a Person can be a Student and Student class can inherit the properties and behaviors of Person Class, but Student class has additional properties and functionalities.

example.ts

```
class Person{
    name:string

    speak():void{
        console.log(this.name+" can speak.")
    }
}

class Student extends Person{
    // variables
    rollnumber:number

    // constructors
    constructor(rollnumber:number, name1:string){
        super(); // calling Parent's constructor
        this.rollnumber = rollnumber
        this.name = name1
    }

    // functions
    displayInformation():void{
        console.log("Name : "+this.name+", Roll Number :
"+this.rollnumber)
    }
}

var student1 = new Student(2, "Rohit")
var student2 = new Student(4, "Kohli")

// accessing variables
console.log("Student 1 name is : "+student1.name)
console.log("Student 2 roll number is : "+student2.rollnumber)

console.log("\n---Student 1---")
// calling functions
student1.displayInformation()
// calling funciton of parent class
student1.speak()
```

```
console.log("\n---Student 2---")
student2.displayInformation()
student2.speak()
```

name variable is declared in Person Class. But as the Student Class inherits Person Class, the name variable could be accessed by the object of Student Class. The same explanation holds for the function speak() which is defined in Person Class, and could be accessed by the object of Student Class.

Compile the above code to JavaScript code using **tsc**, and execute the javascript online.

Output

```
Student 1 name is : Rohit
Student 2 roll number is : 4

---Student 1---
Name : Rohit, Roll Number : 2
Rohit can speak.

---Student 2---
Name : Kohli, Roll Number : 4
Rohit can speak.
```

TypeScript – Method Overriding

TypeScript – Method Overriding

Method Overriding is a process of overthrowing a method of super class by method of same name and parameters in sub class.

Method Overriding is useful when sub class wants to modify the behavior of super class for certain tasks.

Example

Following is a simple example of method overriding where eat() method of Student class overrides the eat() method of Person class.

```
class Person{
    name:string

    eat():void{
        console.log(this.name+" eats when hungry.")
    }
}

class Student extends Person{
    // variables
    rollnumber:number;

    // constructors
    constructor(rollnumber:number, name1:string){
        super(); // calling Parent's constructor
        this.rollnumber = rollnumber
        this.name = name1
    }

    // functions
    displayInformation():void{
        console.log("Name : "+this.name+", Roll Number :
"+this.rollnumber)
    }

    // overriding super class method
    eat():void{
        console.log(this.name+" eats during break.")
    }
}
```

```
var student1 = new Student(2, "Rohit")


student1.displayInformation()

student1.eat()
```

INTERFACE EXAMPLE

Example

Following is a simple Interface called Student, which has two variables declared: name and rollNumber, and one method : displayInformation().

example.ts

```
interface Student{
    // variables
    name:string
    rollnumber:number

    // functions
    displayInformation: () => void
}

var student1: Student = {
    name:"Rohit",
    rollnumber:2,
    displayInformation: ():void => {
        console.log("\n---- Student Information ----")
        console.log("Name is : " + student1.name)
        console.log("Roll Number is : " + student1.rollnumber)
    }
}

console.log(student1.name)
console.log(student1.rollnumber)
student1.displayInformation()
```

Output

```
Rohit
2
```

```
---- Student Information ----
Name is : Rohit
Roll Number is : 2
```

JavaScript file when the above .ts while is compiled using **tsc** :

e