Type Script

Saturday, March 23, 2024 4:23 PM

Easy

Type

What

Type Safe

JS++

Easy

```
var x:number = 10;
x="Hello";
```

TS —Compiled→ JS (TSC)

Few lines of code

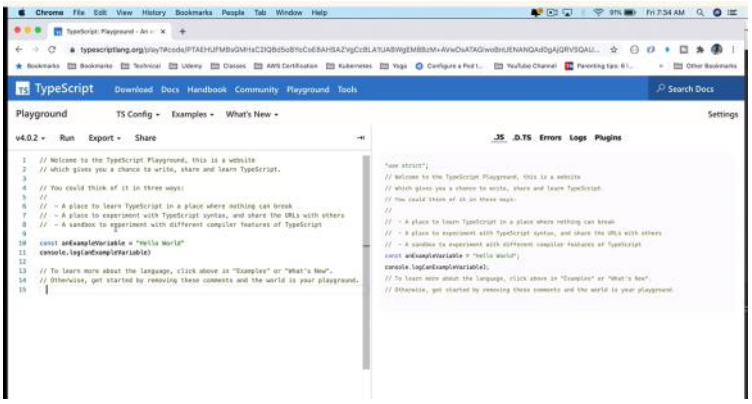
```
class Employee{
}
```

OOPS

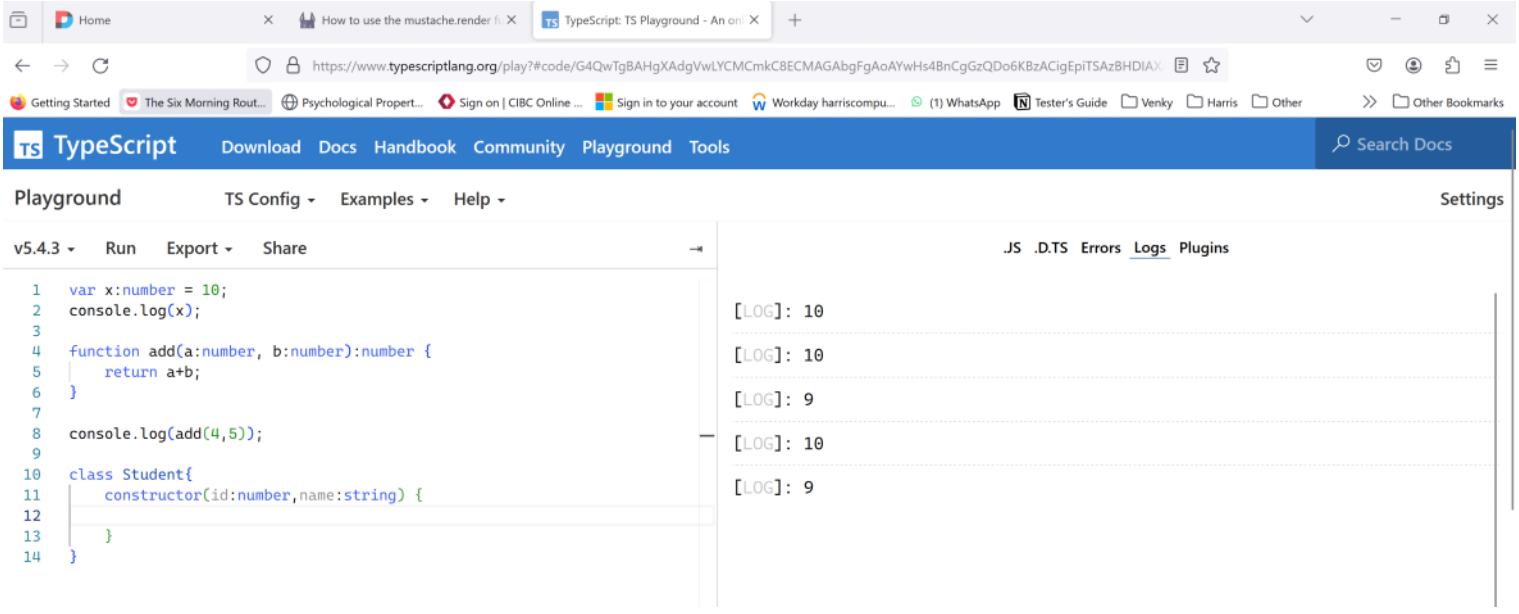
interface

inheritance

encapsulation



https://www.typescriptlang.org/play?




```
var x:number = 10;
console.log(x);

function add(a:number, b:number):number {
    return a+b;
}

console.log(add(4,5));

class Student{
    constructor(id:number,name:string) {

    }
}
```

TS 1 Installation

Saturday, March 23, 2024 6:03 PM

- Node.js
- npm
- Visual Studio

https://nodejs.org/en/

https://docs.npmjs.com/cli/v8/commands/npm-install

https://code.visualstudio.com/download

```
node -v

npm install -D typescript

npm install -g typescript

tsc -v
```

Data Types

number	var x:number = 123
string	var s:string = “Bharath”
boolean	var b:boolean = true
any	var a:any = “Bharath”
enum	var e ={Male,Female}

Object Types

Classes

Interfaces

Ctrl + Shift + i : To open console of a browser

Ctrl + k + c : To comment lines in Visual Studio

Ctrl + k + u : To un-comment lines in Visual Studio

TS 2 Variables

Sunday, March 24, 2024 1:12 AM

1_variables.ts:

```
//-- number --------
var n1:number = 10;
console.log(n1);


//-- string --------
var s1:string = "you are the master of your destiny";   // double quotes "
var s2:string = 'Rama Rama Rama' // single quotes '
var s3:string = `all power is within you, you can do anything and
everything`//back tick symbol `
console.log(s1);
console.log(s2);
console.log(s3);


//-- boolean --------
var b1 = true;
var b2 = false;
console.log(b1);
console.log(b2);


//-- JSON Object --------
var a1 = {
    productId: 1,
    productName: "iPhone 15",
    productPrice: 599
};
console.log(a1);


//--- Array ----------
var array1 = ["AngularJS", "ReactJS", "NodeJS"];
console.log(array1);
console.log(array1[0]);
console.log(array1.length);

var array2 = [123, "AngularJS", true];
console.log(array2);

var array3:Array<string> = ["Rama", "Krishna", "Siva"];
console.log(array3);

var array4:Array<any> = [123, "Angular JS", true, {name: "krishna", id: 973},
a1];
```

```
console.log(array4);
```

==========

tsc 1_variables.ts

==========

1_variables.html:

```
<html>
    <script src="1_variables.js"></script>
</html>
```

2_popoups.ts:

```
console.log("Hello");
alert("Hello");
confirm("Do you really want to do this?");
var data = prompt("Please enter you name");
console.log(data);
```

==========

tsc 2_popups.ts

==========

2_popoups.html:

```
<html>
    <script src="2_popups.js"></script>
</html>
```

Comments

// single line comment

/*

Multiline comments

***/**

2_popoups.ts:

```
console.log("Hello");
alert("Hello");
confirm("Do you really want to do this?");
var data = prompt("Please enter you name");
console.log(data);
```

==========

tsc 2_popups.ts

==========

2_popoups.html:

```
<html>
    <script src="2_popups.js"></script>
</html>
```

3_enum.ts:

```
console.log("Hello");
alert("Hello");
confirm("Do you really want to do this?");
var data = prompt("Please enter you name");
console.log(data);
```

==========

tsc 3_enum.ts

==========

3_enum.html:

```
<html>
    <script src="3_enum.js"></script>
</html>
```

4_stringtype.ts:

```
var s1:string = "<a href='' />";
var myName = 'Raja';
var s2:string = `My name is ${myName}`;
console.log(s2);
console.log(s2.length);
console.log(s2.charAt(0));
console.log(s2.indexOf('n'));
console.log(s2.lastIndexOf('a'));
```

==========

tsc 4_stringtype.ts

==========

4_stringtype.html:

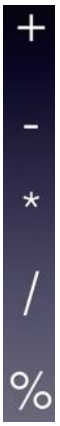
```
<html>
    <script src="4_stringtype.js"></script>
</html>
```

5_uniontype.ts:

```
var sn: string | number;
sn = 'Raja Ram';
sn = 5.4;
```

TS 3 Operators

Sunday, March 24, 2024 10:38 AM



1_arithmetic.ts:

```
var x:number = 10;
var y:number = 5;

console.log('x =' +  '  y = ' + y);
console.log('x - y = ' + (x+y));
console.log('x + y = ' + (x-y));
console.log('x * y = ' + (x*y));
console.log('x / y = ' + (x/y));
console.log('x % y = ' + (x%y));

==================
```

tsc 1_arithmetic.ts

```
================
```

1_arithmetic.html:

```
<html>
    <script src="1_arithmetic.js"></script>
</html>

=================
```

2_assignment.ts:

```
var num1:number = 10;
var num2:number = 2;
var num3:number = num2;
num3 += num1;
console.log(num3);

==================
```

tsc 1_arithmetic.ts

```
================
```

2_assignment.html:

```
<html>
    <script src="2_assignment.js"></script>
</html>

=================
```

3_comparison.ts:

```
var x:number = 40;
var y:number = 50;
console.log('x = ' + x + ' y = ' + y );
console.log('x === 30 : ' + (x === 30));
console.log('x !== 30 : ' + (x !== 30));
console.log('x < y    : ' + (x < y));

==================
```

tsc 3_comparison.ts

```
================
```

3_comparison.html:

```
<html>
    <script src="3_comparison.js"></script>
</html>

=================
```

4_logical.ts:


```
console.log((10>20) && (20>5));
console.log((10>20) || (20>5));
console.log((10>20) || (4>5));
console.log(!((10>20) || (20>5)));
```

==========

```
tsc 4_logical.ts
```

==========

4_logical.html:

```
<html>
    <script src="4_logical.js"></script>
</html>
```

==========

5_teranary.ts:


```
var x:number = 8;
var y:number = 10;
console.log('x =' + x + '  y = ' + y);
console.log((x > y) ? "x is greater than y" : "y is greater than x");
```

==========

```
tsc 5_teranary.ts
```

==========

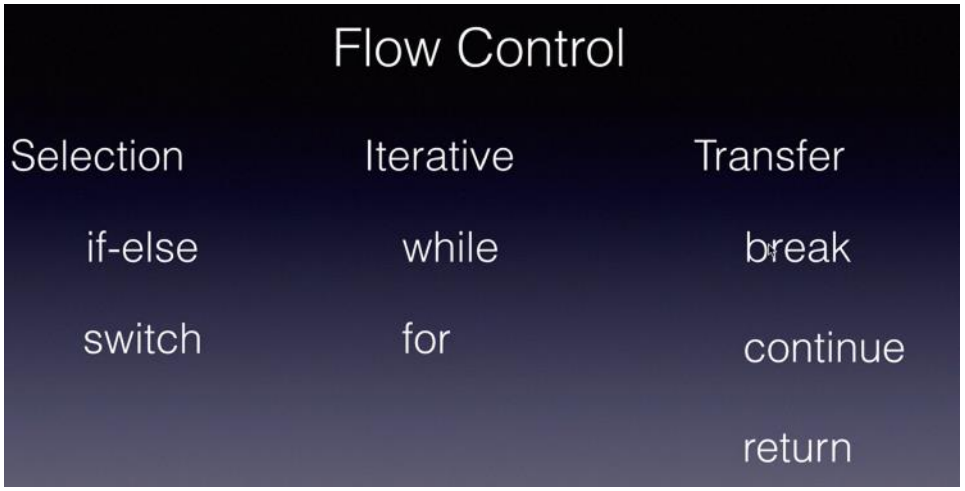
5_teranary.html:

```
<html>
    <script src="5_teranary.js"></script>
</html>
```

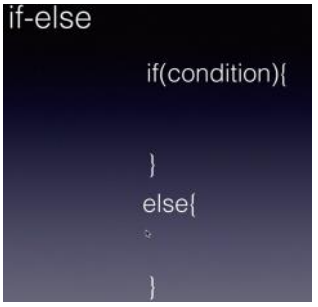
==========

TS 4 Flow Control

Monday, March 25, 2024 12:11 AM



1_ifelse.ts:




```
var x:number = 10;
var y:number = 20;
var z:number = 30;
console.log('x = ' + x + ' y = ' + y + ' z = ' + z );
if(x > y && x > z){
    console.log("x is greater");
         }else if(y > x && y > z){
    console.log("y is greater");
}else if(z > x && z > y){
    console.log("z is greater");
}else{
    console.log("Given numbers are equal");
}
```

==========

tsc 1_ifelse.ts

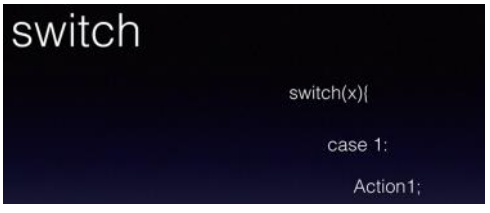
==========

1_ifelse.html:

```
<html>
    <script src="1_ifelse.js"></script>
</html>
```

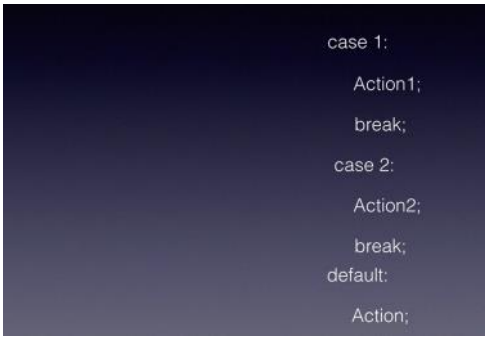
==========

2_switch.ts:




```
var x:number = 3;
switch(x){
    case 1:
```

```
var x:number = 3;
switch(x){
    case 1:
    case 2:
    console.log("Common Logic");
    break;
    case 3:
    console.log("Case 3");
    break;
    default:
    console.log("Default");
}
```



==========

tsc 2_switch.ts

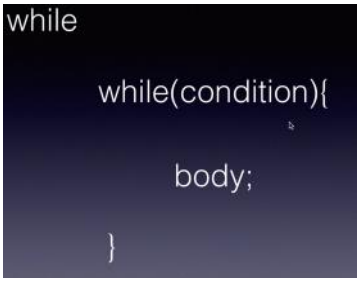
==========

2_switch.html:

```
<html>
    <script src="2_switch.js"></script>
</html>
```

==========

3_while.ts:




```
var n:number = 10;
var i = 1;
while (i <= n) {
    console.log(i++);
}
```

==========

tsc 3_while.ts

==========

3_while.html:

```
<html>
    <script src="3_while.js"></script>
</html>
```

==========

4_emailvalidator.ts:

```
    var email:string = "test@test.com";
    var atPosition:number = email.indexOf('@');
    var dotPosition:number = email.indexOf('.');
```

```
    if ((atPosition == -1) || (dotPosition == -1)) {
        console.log("Invalid email ID :" + email)
    } else {
        console.log("Valid email ID :" + email)
    }
```

==========

tsc 4_emailvalidator.ts

==========

4_emailvalidator.html:

```
<html>
    <script src="4_emailvalidator.js"></script>
</html>
```

==========

5_pwdvalidator.ts:

```
// Rule: Password should begin with a capital letter.
var pwd:string = "Test@123";
if ((pwd.charCodeAt(0) >= 65) && (pwd.charCodeAt(0) >= 90)) {
    console.log("Valid password");
} else {
    console.log("Invalid password");
}
```

==========

tsc 5_pwdvalidator.ts

==========

5_pwdvalidator.html:

```
<html>
    <script src="5_pwdvalidator.js"></script>
</html>
```

==========

TS 5 Objects Arrays

Monday, March 25, 2024 4:16 PM

1_object.ts:

```
var student = {
    firstName:"John",
    lastName:"Bailey",
    score:90
};
console.log(student.firstName);
console.log(student.score);
for(var item in student){
    console.log(item + " --- " + student[item]);
}
var {firstName,lastName} = student;
console.log(firstName+" "+lastName);
```

==========

tsc 1_object.ts

==========

1_object.html:

```
<html>
    <script src="1_object.js"></script>
</html>
```

==========

2_arrays.ts:

```
var courses:any = ["Angular","React","ES6","JMS"];
courses.push("Spring Security");
courses.push(20);
for(var i=0;i<courses.length;i++){
    console.log(courses[i]);
}
var x = courses[0];
var y = courses[1];
var[a,b,c] = courses;
console.log(a);
console.log(b);
console.log(c);
```

==========

tsc 2_arrays.ts

==========

2_arrays.html:

```
<html>
    <script src="2_arrays.js"></script>
</html>
```

==========

2_arraymethods.ts:

```
var levels:number[] = [20, 30, 12, 30, 100, 20];
console.log(levels.toString());
console.log(levels.join(" "));
console.log(levels.join(" | "));
console.log(levels.slice(3,5));
console.log(levels);
console.log(levels.slice(3));
console.log(levels);
levels.splice(2, 88, 99);
console.log(levels);
levels.push(2, 5, 11);
console.log(levels);
console.log(levels.pop());
console.log(levels);
```

==========

```
tsc 2_arraymethods.ts
```

==========

2_arraymethods.html:

```
<html>
    <script src="2_arraymethods.js"></script>
</html>
```

==========

TS 6 Functions

Monday, March 25, 2024 9:46 PM

Functions

Reuse placeOrder()

Call or Event displayPage()

```
function functionName(){
    body
}
```

```
function add(num1:number,num2:number):number{
    return num1+num2;
}
```

1_functions.ts:

```
var hello = function (name:string):string{
    return "Hello " + name;
}
function add(num1:number, num2:number):number{
    return num1+num2;
}

function calculator1(fun:any):void{
  console.log(fun(10, 20));
}

function calculator2():any{
    function subtract(num1:number, num2:number):number{
        return num1-num2;
    }
    return subtract;
}
function display(id:number, name:string, role:string="Normal"){
    console.log("Id",id);
    console.log("Name",name);
    if(role!=undefined) {
      console.log("Role",role);
    }
}
console.log(hello("Rama Raj"));
console.log(hello("Vishnu"));
console.log("Sum is: " + add(10,20));
display(1,"Sri Ranga","Admin");
console.log(calculator1(add));
console.log(calculator2()(20,5));
```

==========

tsc 1_functions.ts

==========

1_functions.html:

```
<html>
    <script src="1_functions.js"></script>
</html>
```

==========

2_anonymous.ts:

Arrow Functions

```
var doubleMe =  (num:number) => num*2;

(num:number) => { return num*2;}
```

```
var hello = (name:string):string=>{
    return "Hello "+name;
};
var multiply = (num1:number,num2:number):number=>{
    return num1*num2;
}
var myarray:Array<any> = [];
for(var i = 0;i<10;i++){
    myarray.push(():number=>{return i});
}
for(var i = 0;i<10;i++){
   console.log(myarray[i]());
}
console.log(hello("Sri Ranga"));
console.log("Product is",multiply(5,8));
```

==========

tsc 2_anonymous.ts

==========

2_anonymous.html:

```
<html>
    <script src="2_anonymous.js"></script>
</html>
```

==========

3_overloading.ts:

```
function doubleMe(x) {
    if (x && typeof x === "number") {
        console.log(x * 2);
    }
    else if (x && typeof x === "string") {
        console.log(x + " " + x);
    }
}
doubleMe(90);
doubleMe("Raja");
//doubleMe([1,2,3]); --> error
```

==========

```
tsc 3_overloading.ts
```

==========

3_overloading.html:

```
<html>
    <script src="3_overloading.js"></script>
</html>
```

==========

```
tsc --init
// Use the above command if you see "error TS2393: Duplicate function implementation."
// Because you open both Typescript and JavaScript file in the same window in IDE
// The above command will create tsconfig.json file for typescript configuration
```

rest params variadic functions

==========

4_varargs.ts:

```
var product = function(x:number,y:number,...nums:number[]){
    console.log("----------------------");
    var result = 1;
    for(var i=0; i < nums.length; i++){
        console.log(i + ")  " + nums[i]);
        result *= nums[i];
    }
    return result;
}
console.log(product(2,3,4,5));
console.log(product(2,3,1,2,3,4,5));
console.log(product(1,2,9,9, 19, 19, 2,2, -2, 2, 2));
```

==========

```
tsc 4_varargs.ts
```

==========

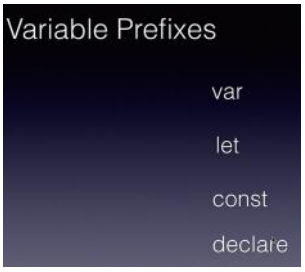
4_varargs.html:

```
<html>
    <script src="4_varargs.js"></script>
</html>
```

==========

TS 7 Variable Scopes

Wednesday, March 27, 2024 1:29 PM



1_let.ts:

```
function myFun():void{
    var i = 44;
    console.log(i);
    for(let i=0; i < 10; i++){
        console.log(i);
        //i has the scope only in this block because of let prefix.
    }
    //console.log(i);  //Error: Can not find name i, scope of is not effective here because of let prefix
    console.log(i);
    var i = 33;
    console.log(i);
    //let : block scope
    //var : function scope
}
myFun();
```

==========

```
tsc 1_let.ts
```

==========

1_let.html:

```
<html>
    <script src="1_let.js"></script>
</html>
```

==========

2_const.ts:

```
const product = function(x:number,y:number):number {
    return x*y;
}

console.log(product(2,3));
const pi = 3.14;
//pi = 4.5 // This is a warning because pi is declared as a constant

//const in C is constant
//const in Java is final
```

==========

```
tsc 2_const.ts
```

==========

2_const.html:

```
<html>
    <script src="2_const.js"></script>
</html>
```

==========

TS 8 Interfaces

Wednesday, March 27, 2024 2:22 PM

Interface

```
interface Student {
    firstName:string;
    lastName:string;
    score:number;
    display():void;
}
```

Rules

1_Product.ts:

```
interface Product{
    id:number;
    name:string;
    description: string;
    price?:number;

    display():void;
}
var product1:Product = {
    id:123,
    name:"iPhone15",
    description:"Its awesome",

    display():void {
        console.log(this.id + " " + this.name);
        console.log(this.description);
    }
}

var product2:Product = {
    id:123,
    name:"iPhone4",
    description:"Its good",

    display():void {
        console.log(this.id + " " + this.name);
    }
}
product1.display();
product2.display();
```

==========

tsc 1_Product.ts

==========

1_Product.html:

```
<html>
    <script src="1_Product.js"></script>
</html>
```

==========

2_FunctionalInterfaces.ts:

```
interface Addition {
    (x:number, y:number):void
}
interface Subtraction {
    (x:number, y:number):number
}
var add: Addition;
var sub: Subtraction;
add=(x:number, y:number):void => {
    console.log("addition = ", x+y)
}
sub=(a:number, b:number):number => {
    return a-b;
}
add(5,8);
console.log(sub(20,7));
```

==========

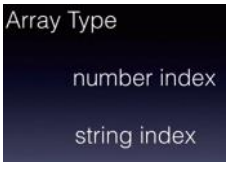
tsc 2_FunctionalInterfaces.ts

==========

2_FunctionalInterfaces.html:

```
<html>
    <script src="2_FunctionalInterfaces.js"></script>
</html>
```

==========



3_NumberIndex.ts:

```
interface Addition {
    (x:number, y:number):void
}
interface Subtraction {
    (x:number, y:number):number
}
var add: Addition;
var sub: Subtraction;
```

```
add=(x:number, y:number):void => {
    console.log("addition = ", x+y)
}
sub=(a:number, b:number):number => {
    return a-b;
}
add(5,8);
console.log(sub(20,7));
```

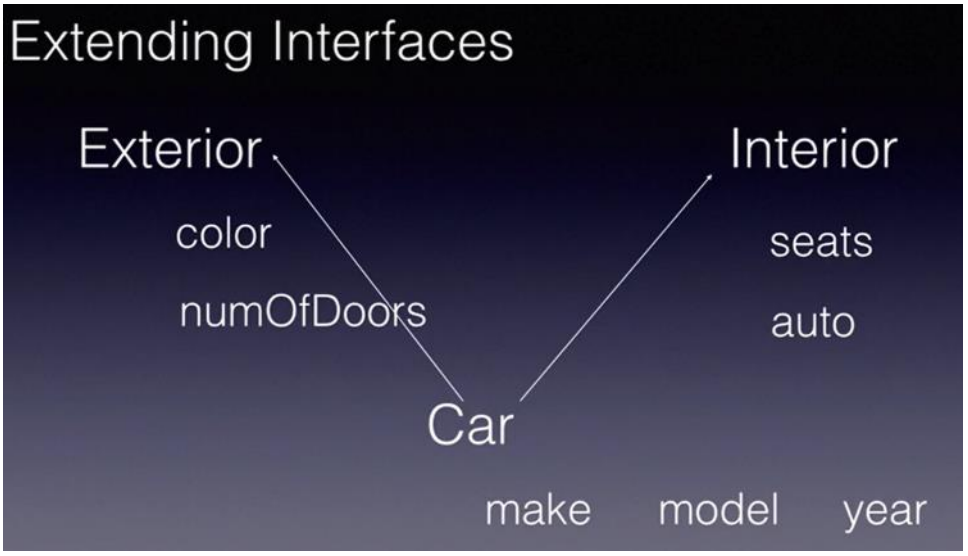
==========

tsc 3_NumberIndex.ts

==========

3_NumberIndex.html:

```
<html>
    <script src="3_NumberIndex.js"></script>
</html>
```



==========

4_ExtendingInterfaces.ts:

```
interface Addition {
    (x:number, y:number):void
}
interface Subtraction {
    (x:number, y:number):number
}
var add: Addition;
var sub: Subtraction;
add=(x:number, y:number):void => {
    console.log("addition = ", x+y)
}
sub=(a:number, b:number):number => {
    return a-b;
}
add(5,8);
console.log(sub(20,7));
```

==========

tsc 4_ExtendingInterfaces.ts

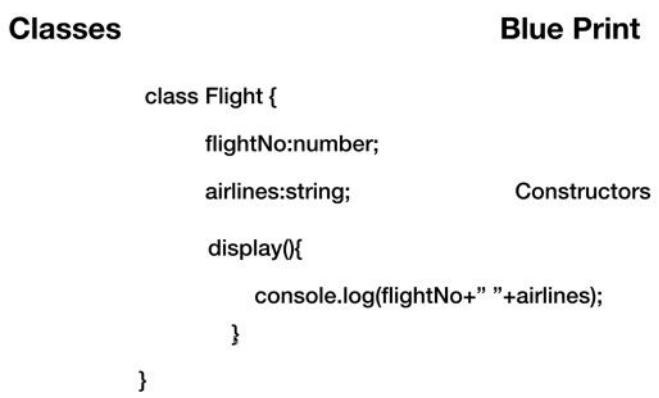
==========

4_ExtendingInterfaces.html:

```
<html>
    <script src="4_ExtendingInterfaces.js"></script>
</html>
```

TS 9 Classes

Wednesday, March 27, 2024 10:11 PM



1_Passenger.ts:

```
class Passenger{
    firstName:string;
    lastName:string;
    frequentFlyerNo:number;

    constructor(firstName:string,lastName:string,frequentFlyerNo:number){
        this.firstName = firstName;
        this.lastName = lastName;
        this.frequentFlyerNo = frequentFlyerNo;
    }
    display(){
        console.log(this.firstName+" "+this.lastName+" "+this.frequentFlyerNo);
    }

}
var passenger = new Passenger("John","Bailey",123);
passenger.display();
var passenger2 = new Passenger("Bob","Bailey",456);
passenger2.display();

for(let item in passenger){
    if(passenger[item] instanceof Function){
        continue;
    }else{
        console.log(item);
        console.log(passenger[item]);
    }
}
```

==========

tsc 1_Passenger.ts

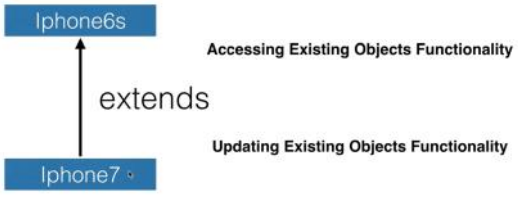
==========

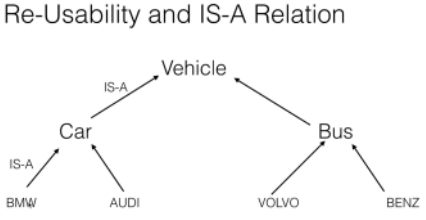
1_Passenger.html:

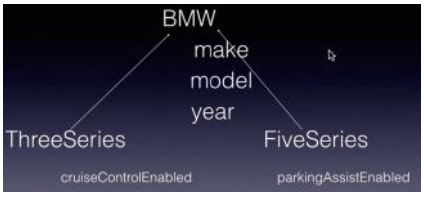
```
<html>
    <script src="1_Passenger.js"></script>
</html>
```

==========

Inheritance







==========

2_Flight.ts:

```
interface IFlight{
    flightNo:number;
    from:string;
    to:string;
    display():void;
}
class Flight implements IFlight{
    flightNo:number;
    from:string;
    to:string;
    constructor(flightNo:number,from:string,to:string){
        this.flightNo = flightNo;
        this.from = from;
        this.to = to;
    }
    display(){
        console.log(this.flightNo+" "+this.from+" "+this.to);
    }
}
var flight = new Flight(123,"Austin","Denver");
flight.display();
```

==========

tsc 2_Flight.ts

==========

2_Flight.html:

```
<html>
    <script src="2_Flight.js"></script>
</html>
```

==========

3_inheritance.ts:

```
class Passenger{
    firstName:string;
    lastName:string;
    frequentFlyerNo:number;

    constructor(firstName:string,lastName:string,frequentFlyerNo:number){
        this.firstName = firstName;
        this.lastName = lastName;
        this.frequentFlyerNo = frequentFlyerNo;
    }
    display(){
        console.log(this.firstName+" "+this.lastName+" "+this.frequentFlyerNo);
    }

}
var passenger = new Passenger("John","Bailey",123);
passenger.display();
var passenger2 = new Passenger("Bob","Bailey",456);
passenger2.display();

for(let item in passenger){
    if(passenger[item] instanceof Function){
        continue;
    }else{
        console.log(item);
        console.log(passenger[item]);
    }
}
```

==========

tsc 3_inheritance.ts

==========

3_inheritance.html:

```
<html>
    <script src="3_inheritance.js"></script>
</html>
```

==========

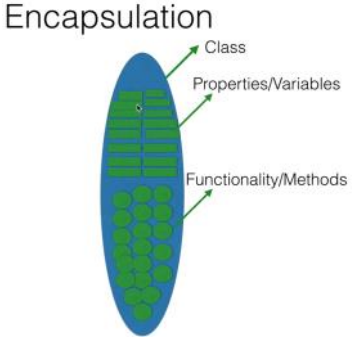
4_access.ts:

```
class Student{
    private _name:string;

    display(){
        console.log(this._name);
    }

    get name():string{
     return this._name;
    }

    set name(name:string){
        this._name = name;
    }
```



```
 }

 var student = new Student();
 student.name = "Bob";
 console.log(student.name);

==================
```

tsc --target es5 4_access.ts

```
================
```

4_access.html:

```
<html>
    <script src="4_access.js"></script>
</html>
==================
```

5_static.ts:

```
class Check{
    static bankName:string="Bank Of America";
    customerName:string;
    accNo:number;
    routingNo:number;
    display(){
        Check.bankName = "BOA";
        console.log(Check.bankName);
    }
}
var check = new Check();
Check.bankName = "BOA";
check.display();
var check2 = new Check();
check2.accNo;
console.log(Check.bankName);

==================
```

tsc 5_static.ts

```
================
```

5_static.html:

```
<html>
    <script src="5_static.js"></script>
</html>
==================
```

6_calcultor.ts:

```
class Check{
    static bankName:string="Bank Of America";
    customerName:string;
    accNo:number;
    routingNo:number;
    display(){
        Check.bankName = "BOA";
        console.log(Check.bankName);
    }
}
var check = new Check();
Check.bankName = "BOA";
check.display();
var check2 = new Check();
check2.accNo;
console.log(Check.bankName);

==================
```

tsc 6_calculator.ts

```
================
```

6_calculator.html:

```
<html>
    <script src="6_calculator.js"></script>
</html>
==================
```

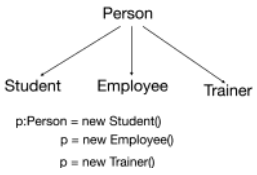
TS 10 Polymorphism

Thursday, March 28, 2024 4:32 PM

Polymorphism

Poly Multi

Morphic Forms



1_poly.ts:

```
class Employee{
    public firstName:string;
    public lastName:string;
    public designation:string;
    public print():void{
        console.log("Employee Details");
    }
}
class Manager extends Employee{
    constructor(firstName:string,lastName:string,designation:string){
        super();
        this.firstName = firstName;
        this.lastName = lastName;
        this.designation = designation;
    }
    public print():void{
        super.print();
        console.log(`${this.firstName} ${this.lastName} - ${this.designation}`);
    }
}

class Lead extends Employee{
    constructor(firstName:string,lastName:string,designation:string){
        super();
        this.firstName = firstName;
        this.lastName = lastName;
        this.designation = designation;
    }
    public print():void{
        super.print();
        console.log(`${this.firstName} ${this.lastName} - ${this.designation}`);
    }
}
class Developer extends Employee{
    constructor(firstName:string,lastName:string,designation:string){
        super();
        this.firstName = firstName;
        this.lastName = lastName;
        this.designation = designation;
    }
    public print():void{
        super.print();
        console.log(`${this.firstName} ${this.lastName} - ${this.designation}`);
    }
}
let employees:Employee[] = new Array(new Manager("John","Ferguson","Manager"),new Lead("Doug","Bailey","Lead"),new Developer("Raja Ram","Seetha","developer"));
for(var employee of employees){
    employee.print();
}

==================
```

tsc 1_poly.ts

```
================
```

1_poly.html:

```
<html>
    <script src="1_poly.js"></script>
</html>
==================
```

TS 11 Typecasting

Thursday, March 28, 2024 5:08 PM

1_numericcasting.ts:

```
let a = prompt("Enter a number");
if (a != null) {
    let x:number = parseFloat(a);
    console.log(x+3);
}

var courses = ["Angular", "React", "Express"];
console.log(courses);
var s2:string = courses.toString();
console.log(s2);
console.log(s2.length);
console.log(s2.charAt(0));
console.log(s2.charAt(7));
console.log(s2.charAt(8));
console.log(s2.charAt(9));
console.log(s2.indexOf('n'));
console.log(s2.lastIndexOf('a'));
var mybool = false;
var y = mybool.toString();
console.log(y);
s2 = y;
console.log(s2.length);
console.log(s2.charAt(0));
console.log(s2.indexOf('n'));
console.log(s2.lastIndexOf('a'));
```

==========

tsc 1_numericcasting.ts

==========

1_numericcasting.html:

```
<html>
    <script src="1_numericcasting.js"></script>
</html>
```

==========

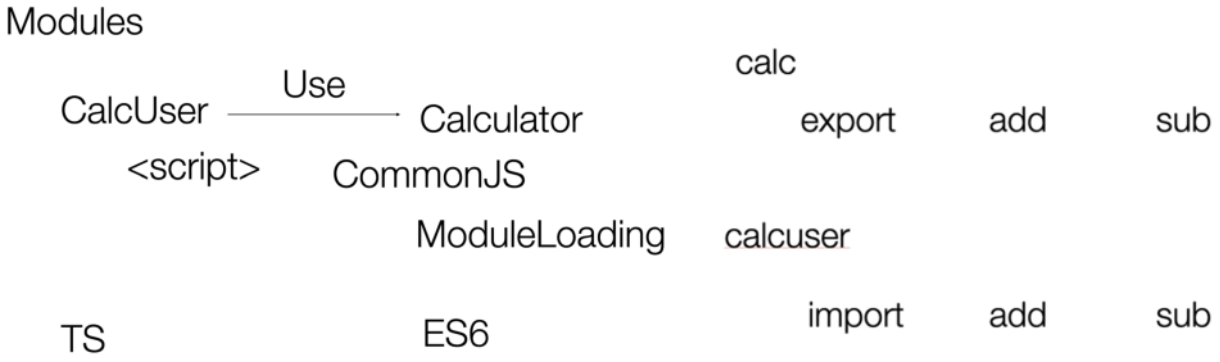
2_objectcasting.ts:

```
interface Employee{
    id:number;
}
let e1:Employee;
let e2 = {id:123,name:"John"};
e1=e2;
// e2=e1; // gives an error!
```

==========

TS 12 Modules

Sunday, March 31, 2024 1:58 PM



1_calc.ts:

```
function add(x:number, y:number):number { return x+y; }

function sub(x:number, y:number):number { return x-y; }

export {add, sub}

==================
```

1_calcUser.ts:

```
import {add, sub} from './1_calc';
console.log(add(2,3));
console.log(sub(3,1));

==================

tsc 1_calc.ts 1_calcUser.ts
node 1_calcUser

==================
```

1_calcUserVer2.ts:

```
import {add as A, sub as S} from './1_calc';
console.log(A(2,3));
console.log(S(3,1));

==================

tsc 1_calc.ts 1_calcUserVer2.ts
```

```
node 1_calcUserVer2
```

```
==================
```

2_calc.ts:

```
export default class Calculator{
    add(x:number,y:number):number{
        return x+y;
    }

    sub(x:number,y:number):number{
        return x-y;
    }
}

export  class Calculator1{
    add(x:number,y:number):number{
            return x+y;
    }

    sub(x:number,y:number):number{
            return x-y;
    }
}
```

```
==================
```

2_calcUser.ts:

```
import Calculator, {Calculator1} from './2_calc';
var calculator = new Calculator();
console.log(calculator.add(2,3));
```

```
==================
```

```
tsc 2_calc.ts 2_calcUser.ts
node 2_calcUser
```

```
==================
```

TS 13 More Types

Sunday, March 31, 2024 7:52 PM

1_maptype.ts:

```
//const studentScores = new Map();
//studentScores.set('John', 90);
//studentScores.set('Bob', 80);
//studentScores.set('Ahmed', 90);

let studentScores = new Map([["john",90],["bob",80],["ahmed",90]]);

studentScores.set('Raja Ram', 100);

console.log(studentScores.get('John'));
console.log(studentScores.get('Bob'));
console.log(studentScores.get('Raja Ram'));

console.log(studentScores);
console.log(studentScores.size)

studentScores.delete("Bob")
console.log(studentScores.has("Bob"))

console.log(studentScores)

console.log(studentScores.keys());

for (let key of Array.from(studentScores.keys())) {
     console.log(key);
     console.log(studentScores.get(key));
}

console.log(studentScores.values());

console.log(studentScores.entries())

studentScores.clear()
console.log(studentScores)
```

==========

tsc -target ES6 1_maptype.ts

==========

1_maptype.html:

```
<html>
    <script src="1_maptype.js"></script>
</html>
```

==========

```
PS C:\TS_Tutorial\12_MapandSet> node 1_maptype
90
80
100

Map(4) { 'John' => 90, 'Bob' => 80, 'Ahmed' => 90, 'Raja Ram' => 100 }

4

false
Map(3) { 'John' => 90, 'Ahmed' => 90, 'Raja Ram' => 100 }

[Map Iterator] { 'John', 'Ahmed', 'Raja Ram' }

John
90
Ahmed
90
Raja Ram
100

[Map Iterator] { 90, 90, 100 }

[Map Entries] { [ 'John', 90 ], [ 'Ahmed', 90 ], [ 'Raja Ram', 100 ] }

Map(0) {}
```

2_settype.ts:

```
let courses = new Set()

courses.add("Angular Crash Course")
courses.add("React")
courses.add("Node")
courses.add("Serverless")
courses.add("React")

console.log(courses.size)

console.log(courses.values())

courses.forEach ( function(course) {
    console.log(course)
})
```

==========

tsc -target ES6 2_settype.ts

==========

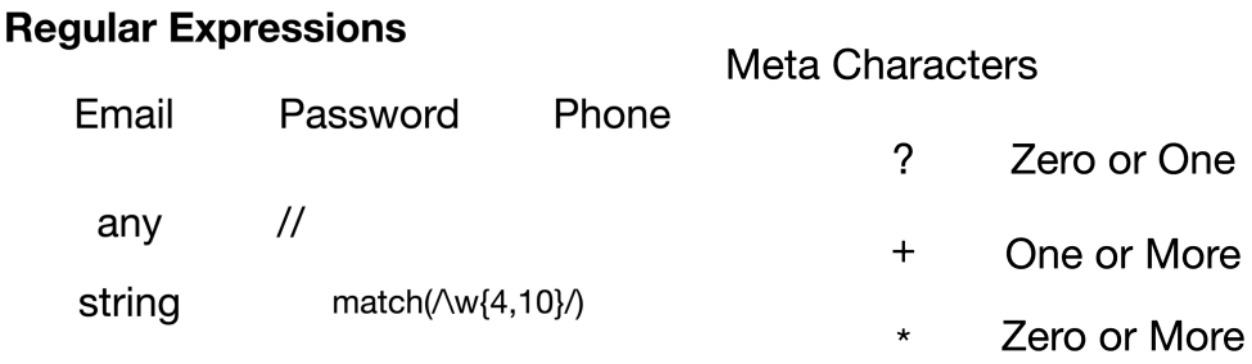
2_settype.html:

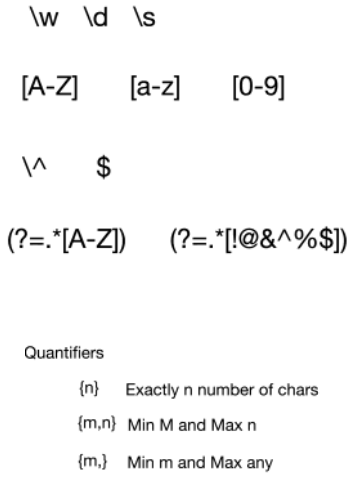
```
PS C:\TS_Tutorial\12_MapandSet> node 2_settype.ts
4

[Set Iterator] {
  'Angular Crash Course',
  'React',
  'Node',
  'Serverless'
}

Angular Crash Course
React
Node
Serverless
```

```
<html>
    <script src="2_aettype.js"></script>
</html>
==================
```





\w alpha-numeric occurrences
\d digits
\s spaces
[A-Z] all capital letters
[a-z] all small letters
[0-9] digits
\^ starts with
$ ends with

?=. At least one

(?=.*[A-Z]) at least one occurrence of capital letter

(?=.*[!@&^%$]) at least one occurrence of special characters listed in the square brackets

3_cellvalidation.ts:

```
let cell1:string = "123456789";
let cell2:string = "1234567890";
let cell3:string = "6++476635312";
let i:number=0;

function checkCell(cell:string):void{

    let re:any = /[0-9]{10}/;  //it means any 10 digits
    if (cell.match(re)) {
        console.log("Cell number is valid")
    } else {
        console.log("invalid cell number " + cell)
    }
    i++;

    console.log('---- ' + i + ' ----')
}

checkCell(cell1);
checkCell(cell2);
checkCell(cell3);
```

```
==================
```

```
tsc  3_cellvalidation.ts
```

```
================
```

3_cellvalidation.html:

```
<html>
    <script src="3_cellvalidation.js"></script>
</html>
==================
```

4_passwordvalidation.ts:

```
var pwd1 = "te4";
var pwd2 = "test1234";
var pwd3 = "+++++234";
var pwd4 = "Test123456";
var pwd5 = "Test1234";
var pwd6 = "tEsT1234";
var pwd7 = "tE_sT234";
var pwd8 = "tE$T1234";
var pwd9 = "tE%%T1234";
let i:number=0;
function checkPwd(pwd:string):void{

    let re:any = /(?=.*[A-Z])\w{4,10}/;;
    //it means
    //(a) Maximum of 10 characters
    //(b) Minimum of 4 characters
    //(c) At least one capital letter
    //(d) Underscore or special characters are permitted
    if (pwd.match(re)) {
        console.log("Password  is valid")
    } else {
        console.log("invalid password " + pwd)
    }
    i++;

    console.log('---- ' + i + ' ----')
}

checkPwd(pwd1);
checkPwd(pwd2);
checkPwd(pwd3);
checkPwd(pwd4);
checkPwd(pwd5);
checkPwd(pwd6);
checkPwd(pwd7);
checkPwd(pwd8);
checkPwd(pwd9);

==================
```

tsc 4_passwordvalidation.ts

```
================
```

4_passwordvalidation.html:

```
<html>
    <script src="4_passwordvalidation.js"></script>
</html>
==================
```

5_datetype.ts:

```
let eDate:any = new Date();
console.log(eDate);
console.log(eDate.getDay());  // gives the index of the day, for example 0 =
Sunday
console.log(eDate.getDate());
console.log(eDate.getMinutes());
console.log(eDate.toTimeString());
eDate = new Date("2024-03-23 2:05 PM");
console.log(eDate);
console.log(eDate.getDay());  // gives the index of the day, for example 5 =
Friday
console.log(eDate.getDate());
console.log(eDate.getHours());
console.log(eDate.getMinutes());
console.log(eDate.toTimeString());

eDate = new Date("2024-04-02 2:05 AM");
console.log(eDate);
```

```
console.log(eDate.getDay());  // gives the index of the day, for example 1 =
Monday
console.log(eDate.getDate());
console.log(eDate.getHours());
console.log(eDate.getMinutes());
console.log(eDate.toTimeString());
eDate.setDate(20);
console.log(eDate);
// 0 = Sun
// 1 = Mon
// 2 = Tues
// 3 = Wed
// 4 = Thu
// 5 = Fri
// 6 = Sat
```

==========

tsc 5_datetype.ts

==========

5_datetype.html:

```
<html>
    <script src="5_datetype.js"></script>
</html>
```

==========