Regular Expression Type

- Regular expression is used to verify the format of input value.
- It uses a pattern with meta characters and quantifiers to verify whether the input value is matching with the pattern defined.
- TypeScript don’t have any pre-defined data type for regular expression. We have to use “any” as type.
- Pattern is a combination of meta characters and quantifiers enclosed in “/ /”.
- TypeScript can verify the input value by using “match()”.

Meta Character	Description
?	Zero or One occurrence
+	One or More occurrences
*	Zero or more occurrences
\	Defines an entity
w	It is an entity that specifies alpha numeric with underscore.

d	It is an entity for numeric.
s	Blank spaces
[A-Z]	Only uppercase letters
[a-z]	Only lowercase letters
[a-zA-Z] or [a-Z]	Both upper and lower case
[0-9]	Only numeric
[a-zA-Z0-9]	Alpha numeric
[a,d,s]	Only specified chars
[^a,d,s]	Exclude specified chars
[a-mA-M4-9]	Chars in specified range
\^	Starts with
$	Ends with
\+ \- *	Special characters individually
[!@#$%&]	Set of special characters
(?=.*[A-Z])	At least one upper case letter
(?=.*[0-9])	At least one numeric value.
(?=.*[!@#$%&])	At least one special character

Quantifier	**Description**

{n}	Exactly n-number of chars {10}
{n, m}	Minimum-n and Maximum-m {4,10}
{n, }	Minimum-n and Maximum-any {4, }

Ex:

```
let mobile:string = "+919876543210";
let regExp:any = /\+91[0-9]{10}/;
if(mobile.match(regExp)){
   console.log(`${mobile} Verified..`);
} else {
   console.log(`Invalid Mobile`);
}
```

Ex:

```
let password:string = "john123A";
let regExp:any = /(?=.*[A-Z])\w{4,10}/;
if(password.match(regExp)) {
   console.log(`Password is Strong`);
} else {
   if(password.length<4) {
      console.log(`Poor Password`);
   } else {
```

```
        console.log(`Weak Password`);
    }
}
```

Date Type

- There is no specific data type to defined date.
- You can use “Date()” method from Date interface that allocates memory for date type values, which you can access with reference of any variable whose type is “any”.

Syntax:

let ref:any = new Date(); → Stores the current system date

let ref:any = new Date(“dateValue”);

- Date value is defined in “YY-MM-DD” format.

Ex:

```
let Name:string = "Samsung TV";
let Mfd:any = new Date();
console.log(`Name=${Name}\nManufactured=${Mfd}`);
```

Ex:

```
let Name:string = "Samsung TV";
let Mfd:any = new Date("2020-02-20");
console.log(`Name=${Name}\nManufactured=${Mfd}`);
```

- To Get the date values form any date reference we can use the following functions, which return always a number.

Function	Description
getHours()	It gets the hour number in 24 hr format. (0-23)
getMinutes()	It gets the minutes number. (0-59)
getSeconds()	It gets the seconds number. (0-59)
getMilliseconds()	It gets the milliseconds number. (0-999)
getYear()	It returns the year number as per Y2K [Obsolete]

getFullYear()	It returns the year number in four digits.
getMonth()	It returns the month number. (0-11) 0-January
getDate()	It returns the day as number. (1-31)
getDay()	It returns the weekday number. (0-6) 0-Sunday
getTime()	It returns the time.
toDateString()	It converts the date portion into string.
toLocaleDateString()	Returns date in locale format.
toString()	Returns in string format.
toTimeString()	Returns time in string format.

Ex:

```
let Name:string = "Samsung TV";
let Mfd:any = new Date("2020-04-10");
```

```
let months:string[] =
["Jan","February","March","April","May","Jun
e"];
let days:string[] =
["Sun","Mon","Tue","Wednesday","Thu","Fri"
,"Sat"];

console.log(`Name=${Name}`);
console.log(
   `\nManufactured Month:
${months[Mfd.getMonth()]}\n
    Manufactured Day  : ${Mfd.getDate()}\n
    Manufactured Year : ${Mfd.getFullYear()}\n
    Weekday           : ${days[Mfd.getDay()]}\n
    Short Date        :
${Mfd.toLocaleDateString()}
   `
)
```

- You set value into date by using date functions

Function	Description
setDate()	Sets the day of the

	month.
setFullYear()	Set the year of a date.
setHours()	Sets the hour of a date.
setMinutes()	
setSeconds()	
setMilliseconds()	
setTime()	
setMonth()	

The limit for setting a value is “Jan-1, 1970” up to “9999”

EX:

```
let Name:string = "Samsung TV";
let Mfd:any = new Date("2020-04-10");
let months:string[] =
["Jan","February","March","April","May","Jun
e"];
let days:string[] =
["Sun","Mon","Tue","Wednesday","Thu","Fri"
,"Sat"];

Mfd.setMonth(1);
Mfd.setFullYear(2019);
```

```
    console.log(`Name=${Name}`);
    console.log(
      `\nManufactured Month:
    ${months[Mfd.getMonth()]}\n
        Manufactured Day  : ${Mfd.getDate()}\n
        Manufactured Year : ${Mfd.getFullYear()}\n
        Weekday           : ${days[Mfd.getDay()]}\n
        Short Date        :
    ${Mfd.toLocaleDateString()}
      `

    )
```

Ex:

```
let product:any = {
   Name: "TV",
   Mfd: new Date("2020-02-21")
}
let months:string[] =
["Jan","February","March","April","May","June"];
let days:string[] =
["Sun","Mon","Tue","Wednesday","Thu","Fri","Sa
t"];
```

```
product.Mfd.setMonth(1);
product.Mfd.setFullYear(2019);


console.log(`Name=${product.Name}`);
console.log(
   `\nManufactured Month:
${months[product.Mfd.getMonth()]}\n
    Manufactured Day  :
${product.Mfd.getDate()}\n
    Manufactured Year :
${product.Mfd.getFullYear()}\n
    Weekday           :
${days[product.Mfd.getDay()]}\n
    Short Date        :
${product.Mfd.toLocaleDateString()}
   `
)
```