TypeScript - Arrow Functions

Fat arrow notations are used for anonymous functions i.e for function expressions. They are also called lambda functions in other languages.

Syntax:

(param1, param2, ..., paramN) => expression

Using fat arrow (=>) we drop the need to use the 'function' keyword. Parameters are passed in the angular brackets <>, and the function expression is enclosed within the curly brackets {}.

Example: Fat Arrow Function

 Copy

let sum = (x: number, y: number): number => {

return x + y;

}

sum(10, 20); //returns 30

In the above example, sum is an arrow function. (x:number, y:number) denotes the parameter types, :number specifies the return type. The fat arrow => separates the function parameters and the function body. The right side of => can contain one or more code statements.

The above arrow function sum will be converted into the following JavaScript code.

var sum = function (x, y) {

return x + y;

}

The following is an arrow function without parameters.

Example: Parameterless Arrow Function

 Copy

let Print = () => console.log("Hello TypeScript");

Print(); //Output: Hello TypeScript

Furthermore, if the function body consists of only one statement then no need for the curly brackets and the return keyword, as shown below.

let sum = (x: number, y: number) => x + y;

sum(3, 4); //returns 7

A class can include an arrow function as a property, as shown below.

Example: Arrow Function in Class

 Copy

class Employee {

empCode: number;

empName: string;

constructor(code: number, name: string) {

this.empName = name;

this.empCode = code;

}

display = () => console.log(this.empCode +' ' + this.empName)

}

let emp = new Employee(1, 'Ram');

emp.display();