**Defining and Using Functions**

**function writeValue(val: string | null) {**

**console.log('Value: ${val ?? "Fallback value"}')**

**}**

**writeValue("London");**

**writeValue(null);**

**Defining Optional Function Parameters**

**function writeValue(val?: string) {**

console.log('Value: ${val ?? "Fallback value"}')

}

writeValue("London");

**writeValue();**

**Defining Default Parameter Values**

**function writeValue(val: string = "default value") {**

**console.log('Value: ${val}')**

**}**

writeValue("London");

writeValue();

output :

Value: London

Value: default value

**Defining Rest Parameters**

**function writeValue(val: string, ...extraInfo: string[]) {**

**console.log('Value: ${val}, Extras: ${extraInfo}')**

}

**writeValue("London", "Raining", "Cold");**

**writeValue("Paris", "Sunny");**

**writeValue("New York");**

**O/p:** Value: London, Extras: Raining,Cold

Value: Paris, Extras: Sunny

Value: New York, Extras:

**Defining Functions That Return Results**

**function composeString(val: string) : string {**

**return 'Composed string: ${val}';**

**}**

**function writeValue(val?: string) {**

**console.log(composeString(val ?? "Fallback value"));**

**}**

**writeValue("London");**

**writeValue();**

**O/p:**

Composed string: London

Composed string: Fallback value

**Using Functions as Arguments to other Functions**

**function getUKCapital() : string {**

**return "London";**

**}**

**function writeCity(f: () => string) {**

**console.log('City: ${f()}')**

**}**

**writeCity(getUKCapital);**

O/p:

City: London

**Defining Functions Using the Arrow Syntax**

function getUKCapital() : string {

return "London";

}

function writeCity(f: () => string) {

console.log('City: ${f()}')

}

writeCity(getUKCapital);

**writeCity(() => "Paris");**

**output:**

City: London

City: Paris

**Enumerating the Contents of an Array**

let myArray: (number | string | boolean)[] = [100, "Adam", true];

**for (let i = 0; i < myArray.length; i++) {**

**console.log("Index " + i + ": " + myArray[i]);**

**}**

**console.log("---");**

**myArray.forEach((value, index) => console.log("Index " + index + ": " + value));**

**O/p:**

Index 0: 100

Index 1: Adam

Index 2: true

---

Index 0: 100

Index 1: Adam

Index 2: true

**Using the Spread Operator**

The spread operator is used to expand an array so that its contents can be used as function arguments or combined with other arrays. In [Listing 4-17](https://cdn2.percipio.com/1713968899.ca7a1a7264ea9edbf63e27ceb56b49732f5d02bc/eod/books/159581/OEBPS/section-38.xhtml#ch04PC33), I used the spread operator to expand an array so that its items can be combined into another array.

let myArray: (number | string | boolean)[] = [100, "Adam", true];

**let otherArray = [...myArray, 200, "Bob", false];**

**// for (let i = 0; i < myArray.length; i++) {**

**// console.log("Index " + i + ": " + myArray[i]);**

**// }**

**// console.log("---");**

**otherArray.forEach((value, index) => console.log("Index " + index + ": " + value));**

let otherArray = [**...myArray**, 200, "Bob", false];

o/p:

Index 0: 100

Index 1: Adam

Index 2: true

Index 3: 200

Index 4: Bob

Index 5: false

| **Method** | **Description** |
| --- | --- |
| concat(otherArray) | This method returns a new array that concatenates the array on which it has been called with the array specified as the argument. Multiple arrays can be specified. |
| join(separator) | This method joins all the elements in the array to form a string. The argument specifies the character used to delimit the items. |
| pop() | This method removes and returns the last item in the array. |
| shift() | This method removes and returns the first element in the array. |
| push(item) | This method appends the specified item to the end of the array. |
| unshift(item) | This method inserts a new item at the start of the array. |
| reverse() | This method returns a new array that contains the items in reverse order. |
| slice(start,end) | This method returns a section of the array. |
| sort() | This method sorts the array. An optional comparison function can be used to perform custom comparisons. |
| splice(index, count) | This method removes count items from the array, starting at the specified index. The removed items are returned as the result of the method. |
| unshift(item) | This method inserts a new item at the start of the array. |
| every(test) | This method calls the test function for each item in the array and returns true if the function returns true for all of them and false otherwise. |
| some(test) | This method returns true if calling the test function for each item in the array returns true at least once. |
| filter(test) | This method returns a new array containing the items for which the test function returns true. |
| find(test) | This method returns the first item in the array for which the test function returns true. |
| findIndex(test) | This method returns the index of the first item in the array for which the test function returns true. |
| foreach(callback) | This method invokes the callback function for each item in the array, as described in the previous section. |
| includes(value) | This method returns true if the array contains the specified value. |
| map(callback) | This method returns a new array containing the result of invoking the callback function for every item in the array. |
| reduce(callback) | This method returns the accumulated value produced by invoking the callback function for every item in the array. |

**Using the Built-in Array Methods**