## Iterating an Object Keys with For-In

# WE'LL COVER THE FOLLOWING Iterating an object's keys with for-in Iterating an object's keys with for of Iterating with forEach

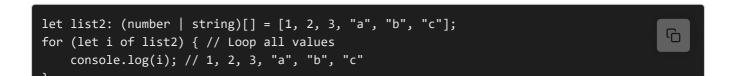
# Iterating an object's keys with for-in #

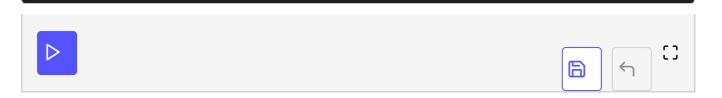
The first way is to iterate the key of an object. Looping the key is the role of the for-in loop. You can use for-in on a normal array. The result is a list of indices which would be the sequential order of each element. On an object, you will iterate all members' names.

```
let list1: (number | string)[] = [1, 2, 3, "a", "b", "c"];
for (let i in list1) { // Loop all indexes, not values
        console.log(i); // Print: 0, 1, 2, 3, 4, 5
}
```

# Iterating an object's keys with for of #

The second way to iterate a collection is on the value of an object which is done using <code>for-of</code>. Using it on an array will provide the values in the array. Using it on an object will not work; <code>for-of</code> is more restrictive because it must implement <code>Symbol.iterator</code>. The difference is that you cannot use this iteration mechanism on a literal object like you could with <code>for-in</code>.





# Iterating with for Each #

There is a short-hand for a structure using Symbol.iterator which is to use the function <code>foreach</code>. The parameter is the element inside the iterable structure. If you have an array of numbers, each result will be a number; if it's an array of objects, it will be an object. <code>foreach</code> also has a second parameter which is the index of the element in the array. The third parameter is the array itself. The last parameter is rarely used.

