Lists

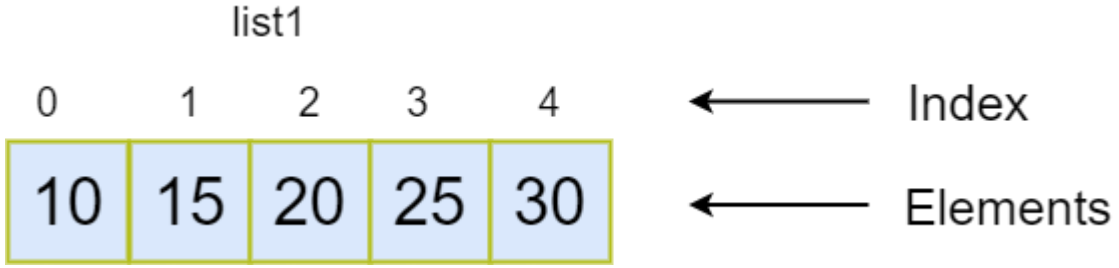
List is similar to an array, which is the ordered collection of the objects. The array is the most popular and commonly used collection in any other programming language.

Syntax:

```
var list1 = [10, 15, 20,25,25]
```

list is defined by storing all elements inside the square bracket ([]) and separated by commas (,).

Graphical Representation:



List1:

It is the list variable that refers to the list object.

Index:

Each element has its index number that tells the element position in the list. The index number is used to access the particular element from the list, such as list name[index]. The list indexing starts from 0 to length-1 where length denotes the numbers of the element present in the list.

For example:

The length of the above list is 5.

Elements:

The List elements refers to the actual value stored in the given list.

List Methods:

```
void main() {
  var lst = [1, 2, 3, 4, 5];
  var e1 = lst.first;
  var e2 = lst.last;
  var e3 = lst.elementAt(1);
  var len = lst.length;

  print('There are $len elements in the list');
  print('The first element is $e1');
  print('The last element is $e2');
  print('The second element is $e3');
}
```

Updating List

```
void main() {

    var myList = [0, 'one', 'two', 'three', 'four', 'five'];

    // replace the item at index '3'
    myList[3] = 3;
    print(myList);

    // replace the item at index '1'
    myList.replaceRange(1, 2, [1]);
    print(myList);
    // [0, 1, two, 3, four, five]


    // replace the items from index 2 to 4
    myList.replaceRange(2, 5, ['new 2', '3 and 4']);
    print(myList);

}
```

Sort Method

```
void main() {
var intList = [0, 5, 2, 3, 8, 17, 11];
intList.sort();
print(intList);

var tringList = ['vue', 'kotlin','dart', 'angular', 'flutter'];
tringList.sort();
print(tringList);

}
```

isEmpty & isNotEmpty Method

```
void main() {
  List vals = [];
  if (vals.isEmpty) {
    print('the list is empty');
  }

  vals.add(1);
  vals.add(2);
  vals.add(3);
  if (vals.isNotEmpty) {
    print('the list is not empty');
  }

  vals.clear();

  if (vals.isEmpty) {
    print('the list is empty');
  }
}
```

Reversed Method

```
void main() {
  var vals = [8, 4, 1, 2, 4, 5, 9];

  var reversed = List.of(vals.reversed);
  print(reversed);
}
```

Reversed List will be:
[9, 5, 4, 2, 1, 4, 8]

Add Methods

```
void main() {
  var vals = [1, 2, 3];

  vals.add(4);
  vals.addAll([5, 6, 7]);

  vals.insert(0, 0);
  vals.insertAll(4, [8, 9, 10]);

  print(vals);
}
```

Remove Methods

```
void main() {
  var vals = [1, 2, 3, 4, 5, 6];

  vals.remove(1);
  print(vals);

  vals.removeAt(vals.length - 1);
  print(vals);

  vals.removeLast();
  print(vals);

  vals.clear();
  print(vals);

}
```

Remove Methods

```
void main() {
   var vals2 = [-2, 1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

  vals2.removeWhere((e) => e < 0);
  print(vals2);

  vals2.removeRange(0, 5);
  print(vals2);

  vals2.retainWhere((e) => e > 7);
  print(vals2);

}
```

Map in Dart

A map is a collection of key/value pairs. The value is retrieved from a map with its associated key. Maps are also called dictionaries. A map literal consists of a pair of curly brackes, in which we specify the key/value pairs. The pairs are separated by comma. The key is separated from the value by colon.

Example

```
void main()
{
  var data = {'name': 'John Doe', 'occupation': 'gardener'};
  print(data);
  print(data.keys);
  print(data.values);

  var words = {1: 'sky', 2: 'falcon', 3: 'rock'};
  print(words);

}
```

Map Size

```
void main()
{
  var fruit = {1: 'Apple', 2: 'Orange'};

  print(fruit.length);
  print('There are ${fruit.length} elements in the map');
}
```

Result will be:

```
2
There are 2 elements in the map
```

isEmpty & isNotEmpty Method

```
void main() {
  var words = {
    1: 'sky',
    2: 'fly',
    3: 'ribbon',
    4: 'falcon',
    5: 'rock',
  };

  print(words.isEmpty);
  print(words.isNotEmpty);

  print('---------------');
  words.clear();

  print(words.isEmpty);
  print(words.isNotEmpty);
}
```

Add Method

```
void main() {
  var fruit = {1: 'Apple', 2: 'Orange'};

  fruit[3] = 'Banana';
  print(fruit);

  var val = fruit.putIfAbsent(3, () => 'Mango');
  print(fruit);
  print(val);

  var val2 = fruit.putIfAbsent(4, () => 'Cherry');
  print(fruit);
  print(val2);

}
```

Add Method

```
void main() {

   Map student = {'name':'Tom','age': 23};
   print('Map :${student}');

   student.addAll({'dept':'Civil','email':'tom@xyz.com'});
   print('Map after adding  key-values :${student}');

}
```

Remove Methods

```
void main() {
  var words = {
    1: 'sky',
    2: 'fly',
    3: 'ribbon',
    4: 'falcon',
    5: 'rock',
    6: 'ocean',
    7: 'cloud'
  };

  words.remove(1);
  print(words);

  words.removeWhere((key, value) => value.startsWith('f'));
  print(words);

  words.clear();
  print(words);
}
```

Merge Methods

```
void main() {
  var f1 = {1: 'Apple', 2: 'Orange'};
  var f2 = {3: 'Banana'};
  var f3 = {4: 'Mango'};

  var fruit = {}..addAll(f1)..addAll(f2)..addAll(f3);
  print(fruit);

  var fruit3 = {...f1, ...f2, ...f3};
  print(fruit3);

}
```

fromIterables Methods

```
void main() {
  var letters = ['I', 'II', 'V', 'X', 'L'];
  var numbers = [1, 2, 5, 10, 50];

  var data = Map.fromIterables(letters, numbers);
  print(data);
}
```

Result will be:

{I: 1, II: 2, V: 5, X: 10, L: 50}

containsKey & containsValue Methods

```
void main() {
  var myMap = {1: 'Apple', 2: 'Orange', 3: 'Banana'};
  print(myMap.containsKey(1));
  print(myMap.containsKey(3));

  print(myMap.containsValue('Apple'));
  print(myMap.containsValue('Cherry'));
}
```

Map Iteration Method

```
void main()
{
  var fruit = {1: 'Apple', 2: 'Banana', 3: 'Cherry', 4: 'Orange'};

  fruit.forEach((key, val) {
    print(' $key, $val');
  });

}
```