# Dart – Data Structures

Girma B

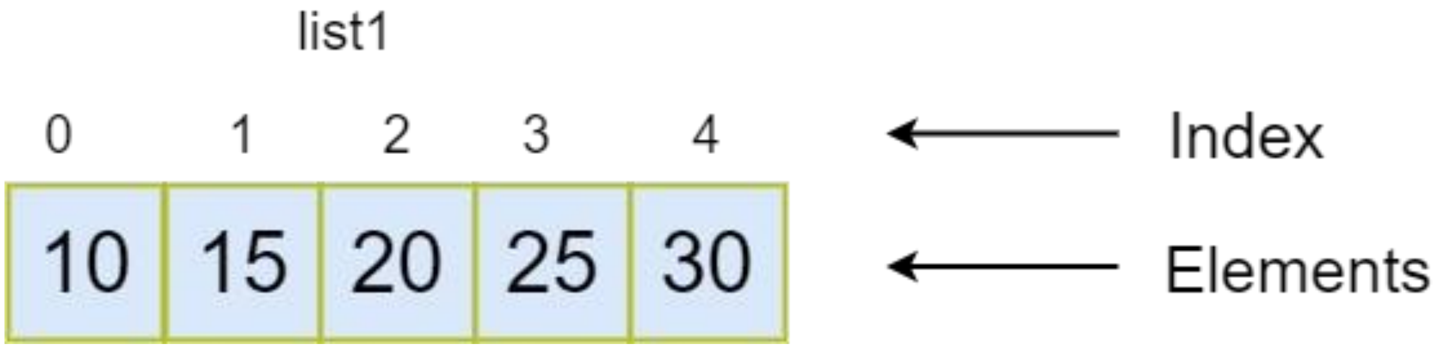
## List

- Dart 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. The Dart list looks like the JavaScript array literals.
- The syntax of declaring the list is given below.

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

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

# List



## Fixed Length List

- The fixed-length lists are defined with the specified length. We cannot change the size at runtime. The syntax is given below.
- Syntax - Create the list of fixed-size
- var list_name = new List(size)
  - The above syntax is used to create the list of the fixed size. We cannot add or delete an element at runtime. It will throw an exception if any try to modify its size.

## Fixed Length List

- Syntax - Initialize the fixed size list element

  list_name[index] = value;
- Let's understand the following example.
- Example -

```
void main() {
  var list1 = new List(5);
  list1[0] = 10;
  list1[1] = 11;
  list1[2] = 12;
  list1[3] = 13;
  list1[4] = 14;
  print(list1);
}
```

## Growable List

- The list is declared without specifying size is known as a Growable list. The size of the Growable list can be modified at the runtime.
- The syntax of the declaring Growable list is given below.
- Syntax - Declaring a List

```
// creates a list with values
var list_name = [val1, val2, val3]
Or
// creates a list of the size zero
var list_name = new List()
```

## Growable List

- Syntax - Initializing a List

  list_name[index] = value;
- Consider the following example -
- Example - 1

```
void main() {
  var list1 = [10,11,12,13,14,15];
  print(list1);
}
```

# Growable List

- In the following example, we are creating a list using the empty list or List() constructor. The add() method is used to add element dynamically in the given list.
- Example

```
void main() {
  var list1 = new List();
  list1.add(10);
  list1.add(11);
  list1.add(12);
  list1.add(13);
  print(list1);
}
```

## List Properties

| | |
|---|---|
| first | It returns the first element case. |
| isEmpty | It returns true if the list is empty. |
| isNotEmpty | It returns true if the list has at least one element. |
| length | It returns the length of the list. |
| last | It returns the last element of the list. |
| reversed | It returns a list in reverse order. |
| Single | It checks if the list has only one element and returns it. |

## Inserting Element into List

- Dart provides four methods which are used to insert the elements into the lists. These methods are given below.
  - add()
  - addAll()
  - insert()
  - insertAll()

## Removing Element from List

- The Dart provides following functions to remove the list elements.
  - remove()
  - removeAt()
  - removeLast()
  - removeRange()

Summary Table

| **Function** | **Description** |
|---|---|
| add(value) | Adds an element |
| remove(value) | Removes first occurrence |
| contains(value) | Checks if exists |
| sort() | Sorts the list |
| reversed | Returns reversed list |
| map(func) | Transforms elements |
| where(condition) | Filters elements |
| reduce(func) | Reduces elements |
| fold(init, func) | Aggregates with initial value |
| join(separator) | Converts to string |
| toList() | Copies list |
| toSet() | Removes duplicates |

## Set

- Set is the unordered collection of the different values of the same type. It has much functionality, which is the same as an array, but it is unordered.
- Set doesn't allow storing the duplicate values. The set must contain unique values.
- It plays an essential role when we want to store the distinct data of the same type into the single variable.
- Once we declare the type of the Set, then we can have an only value of the same type. The set cannot keep the order of the elements.

## Set

- Dart provides the two methods to declare/initialize an empty set.
- The set can be declared by using the {} curly braces proceeded by a type argument, or declare the variable type Set with curly braces {}. The syntax of declaring set is given below.
- Syntax -

```
var setName = <type>{};
Or
Set<type> setname = {};
```

## Set

```
void main(){
  print("Initializing the Set");
  var names = <String>{"Rima","Mohan","Babu","Ram"};
  print(names);
}
```

## Set - Add Elements

Set_name.add(<value>);

Or

Set_name.addAll(val1,val2....valN)

## Set - Add Elements

```
// Declaring empty set
var emp = <String>{};
emp.add("Jay");
print(emp);

// Adding multiple elements
emp.addAll(names);
print(emp);
```

# Accessing Elements

- Dart provides the elementAt() method, which is used to access the item by passing its specified index position.
- The set indexing starts from the 0 and goes up to size - 1, where size is the number of the element exist in the Set.
- It will throw an error if we enter the bigger index number than its size. The syntax is given below.
- Syntax:
  - Set_name.elementAt(index)

## Other methods

- set_name.contains(value);
- set_name.remove(value);
- set_name.clear();

- x.union(y)
- x.intersection(y)
- y.difference(z)

## Map

- Map is an object that stores data in the form of a key-value pair. Each value is associated with its key, and it is used to access its corresponding value. Both keys and values can be any type.
- In Dart Map, each key must be unique, but the same value can occur multiple times. The Map representation is quite similar to Python Dictionary.
- The Map can be declared by using curly braces {} ,and each key-value pair is separated by the commas(,).
- The value of the key can be accessed by using a square bracket([]).

## Map

```
void main() {
  var student = {'name':'Raju','age':'21'};
  print(student);
}
```

- student['course'] = 'B.tech';

## Map

```
void main() {
  var student = new Map();
  student['name'] = 'Raju';
  student['age'] = 21;
  student['course'] = 'B.tech';
  student['Branch'] = 'Computer Science';
  print(student);
}
```

## Map - Properties

| | |
|---|---|
| Keys | It is used to get all keys as an iterable object. |
| values | It is used to get all values as an iterable object. |
| Length | It returns the length of the Map object. |
| isEmpty | If the Map object contains no value, it returns true. |
| isNotEmpty | If the Map object contains at least one value, it returns true. |

## Map - Methods

- student.addAll({'dept':'Civil','email':'raj@abc.in'});
- student.clear();
- student.remove('age');

# Map - Methods

| Method | Description |
|---|---|
| **map[key] = value** | Add or update value |
| **putIfAbsent(key, () => value)** | Add only if key doesn’t exist |
| **addAll({key: value})** | Add multiple key-value pairs |
| **remove(key)** | Remove key-value pair |
| **removeWhere((key, value) => condition)** | Remove conditionally |
| **containsKey(key)** | Check if key exists |
| **containsValue(value)** | Check if value exists |
| **forEach((key, value) => …)** | Loop through the map |
| **keys / values** | Get all keys/values |
| **map((key, value) => MapEntry(newKey, newValue))** | Modify map entries |
| **where((key, value) => condition)** | Filter entries |
| **entries.toList()** | Convert map to a list |
| **jsonEncode(map)** | Convert map to JSON |
| **jsonDecode(jsonString)** | Convert JSON to map |

## Map - Methods

## Symbol

- Symbol object is used to specify an operator or identifier declared in a Dart programming language.
- Generally, we do not need to use symbols while Dart programming, but they are helpful for APIs. It usually refers to identifiers by name, because identifier names can vary but not identifier symbols.
- Dart symbols are dynamic string name that is used to derive the metadata from a library.
- It mainly accumulates the connection between human-readable strings that are enhanced to be used by computers.

## Symbol

- Symbols have a term which is called Reflection; it is a technique that used to the metadata of at run-time, for example - the number of methods used in class, a number of constructors in a class, or numbers of arguments in a function.
- The dart:mirrors library has all of the reflection related classes. It can be used with the command-line applications as well as web applications.

## Thank you