Dart Lists

- In Dart, lists are ordered collections of objects that can store multiple values of any type. Dart provides the `List` class to work with lists, and it supports various operations such as adding, removing, and manipulating items.
-

Creating a List

You can create a list using literals or by using the `List` constructor.

- **List of integers**
- List<int> numbers = [1, 2, 3, 4, 5];
- **List of strings**
- List<String> fruits = ['Apple', 'Banana', 'Mango'];

- **Using the List Constructor**
- List<int> emptyList = List.empty();
- **List of fixed length, all elements initialized to null**
- List<int?> fixedLengthList = List.filled(3, null);
- **Growable list of strings**
- List<String> growableList = List.empty(growable: true);

Adding Elements to a List

- Dart provides several ways to add elements to a list.
- **Using `add()`, `insert()`, `addAll()`**
  - List<String> fruits = ['Apple', 'Banana’];
- Add a single element
  - fruits.add('Mango’);
- Insert element at a specific index
  - fruits.insert(1, 'Orange'); // Inserts 'Orange' at index 1
- Add multiple elements at once
  - fruits.addAll(['Grapes', 'Pineapple']);
  - print(fruits);

Removing Elements

- You can remove elements by value or index.
- List<String> fruits = ['Apple', 'Orange', 'Banana', 'Mango’];
- Remove by value
  - fruits.remove('Banana’);
- Remove by index
  - fruits.removeAt(0); // Removes 'Apple’
- Remove the last element
  - fruits.removeLast();
  - print(fruits);

Accessing List Elements

- You can access list elements using their index. Dart uses zero-based indexing.
- List<String> fruits = ['Apple', 'Banana', 'Mango’];
- Access element by index
  - String firstFruit = fruits[0]; // 'Apple’
- Modify an element by index
  - fruits[1] = 'Orange'; // Changes 'Banana' to 'Orange’
- List Properties and Methods
  - **Length**: Get the number of elements in the list.
  - **isEmpty** and **isNotEmpty**: Check if the list is empty or not.

- List<int> numbers = [1, 2, 3];
- print(numbers.length); // Output: 3
- print(numbers.isEmpty); // Output: false
- print(numbers.isNotEmpty); // Output: true

Iterating Over a List

- You can use loops or higher-order methods like `forEach`.
- Using `for` Loop and `forEach()`

```
List<String> fruits = ['Apple', 'Banana', 'Mango'];

// Using a for loop
for (int i = 0; i < fruits.length; i++) {
  print(fruits[i]);
}

// Using forEach
fruits.forEach((fruit) => print(fruit));
```

LIST OPERATIONS

```
List<int> numbers = [1, 2, 3, 4, 5];

// map: Transform each element
List<int> squaredNumbers = numbers.map((n) => n * n).toList();
print(squaredNumbers);  // Output: [1, 4, 9, 16, 25]

// where: Filter elements
List<int> evenNumbers = numbers.where((n) => n.isEven).toList();
print(evenNumbers);  // Output: [2, 4]

// reduce: Combine elements
int sum = numbers.reduce((a, b) => a + b);
print(sum);  // Output: 15
```

Common List Methods

- sort(): Sorts the list in-place.
- contains(): Checks if the list contains an element.
- join(): Combines elements into a string with a separator.
- indexOf(): Returns the index of the first occurrence of an element.

```
List<String> fruits = ['Apple', 'Banana', 'Mango'];

fruits.sort();
print(fruits);  // Output: [Apple, Banana, Mango]

print(fruits.contains('Banana'));  // Output: true

String joinedFruits = fruits.join(', ');
print(joinedFruits);  // Output: Apple, Banana, Mango

int index = fruits.indexOf('Mango');
print(index);  // Output: 2
```

Displaying Data in a ListView

```
import 'package:flutter/material.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text('Fruits List')),
        body: FruitList(),  // Custom widget to display the list
      ),
    );
  }
}
```

```
class FruitList extends StatelessWidget {
  // List of fruits (data)
  final List<String> fruits = ['Apple', 'Banana', 'Orange',
'Mango', 'Grapes'];

  @override
  Widget build(BuildContext context) {
    return ListView.builder(
      itemCount: fruits.length,  // Number of items in the lis
      itemBuilder: (context, index) {
        return ListTile(
          title: Text(fruits[index]),  // Display each fruit
        );
      },
    );
  }
}
```

Displaying a List in a GridView

```
class GridList extends StatelessWidget {
  final List<String> items = ['1', '2', '3', '4', '5', '6'];

  @override
  Widget build(BuildContext context) {
    return GridView.builder(
      gridDelegate:
SliverGridDelegateWithFixedCrossAxisCount(
        crossAxisCount: 2,  // Two items per row
      ),
      itemCount: items.length,
      itemBuilder: (context, index) {
        return Card(
          child: Center(
            child: Text(items[index], style: TextStyle(fontSize:
24)),
          ),
        );
      },
    );
  }
}
```