Dart Building Blocks Part 2

Key Points	Notes

1. **Lists**
   a. In Dart an **array** is called a list or **ordered group of objects**. There are two (2) types of list.

   b. A fixed list is a list with pre-determined size.

```
//Syntax A
List<int> numbers = [2]; //list with a value of 2

numbers.add(1);
numbers[1] = 2;

print(numbers); //print the whole list

for(int number in numbers){ //print each value in the list
  print(number);
}


//Syntax B
var numbers = [2]; //list with a value of 2

numbers.add(1);
numbers[1] = 2;

print(numbers); //print the whole list

for(var number in numbers){ //print each value in the list
  print(number);
}
```

Summary

Key Points	Notes

c. Growable or Dynamic list is a list with expandable size or **NO** pre-determined size.

```
//Syntax A
List<int> numbers = []; //integer values only

numbers.add(3); //insert new value at the end of the list
numbers.insert(1,4); //insert at position 1, value 4

print(numbers); //print whole list

for(int number in numbers){ //print each value in the list
  print(number);
}

//Syntax B
var numbers = [1,2]; //value with any data types

numbers.add(3); //insert new value at the end of the list
numbers.insert(1,4); //insert at position 1, value 4

print(numbers); //print the whole list

for(var number in numbers){ //print each value in the list
  print(number);
}
```

d. Commonly used list properties.

Property	Description
numbers.first	Returns first element in the list.
numbers.isEmpty	If list is empty returns true.
numbers.isNotEmpty	If list not empty returns true.
numbers.length	Returns size of the list.
numbers.last	Returns the last value in the list.
numbers.reversed	Returns values in reverse order.
numbers.single	If there is only one value and prints it.

Summary

Key Points	Notes

2. Sets

a. A set is a **unique collection of unordered objects** or object that can only occur once.

```
//Syntax A
var exotics = {'rambutan', 'durian', 'mangosteen'};

Set<String> fruits = {};

fruits.add('apple');
fruits.add('orange');
fruits.add('apple');
fruits.addAll(exotics);

print(fruits); //print unique values only
print(fruits.length) //print size of the set
for(String fruit in fruits){ //print each value in the set
  print(fruit);
}
```

```
//Syntax B
var exotics = {'rambutan', 'durian', 'mangosteen'};

var fruits = <String>{};
//var fruits = {}; will create a map not a set

fruits.add('orange');
fruits.add('pear');
fruits.addAll(exotics);

print(fruits); //print the whole list
print(fruit.length) //print the size of the set
for(var fruit in fruits){ //print each value in the set
  print(fruit);
}
```

Summary

Key Points	Notes

b. A set can also contains constant value at compile time.

```
final exotics = const { //final variable with const values
‘rambutan’,
‘durian’,
‘mangosteen’};

exotics.add(‘starfruit’); //error, can’t add to const set
print(exotics);
```

3. Maps

a. A map is a key-pair value object storage or an object that contains keys and values.

b. The keys are unique but the values can contain duplicate values.

```
//Syntax A

Map<int, String> fruits = {
 101: ‘apple’,
 102: ‘orange’,
 103: ‘pear’,
};

fruits[104] = ‘watermelon’;

print(fruits); //print maps
fruits.forEach((code,fruit) => print(‘$code is ${fruit}));
```

Summary

Key Points

Notes

```
//Syntax B
var fruits = {};
//var fruits = {101: ‘apple’, 102: ‘orange’, 103: ‘pear’};

fruits[101] = ‘apple’;
fruits[102] = ‘orange’;
fruits[103] = ‘pear’;
fruits[104] = ‘watermelon’;

print(fruits); //print maps
fruits.forEach((code,fruit) => print(‘$code is
${fruit}’));
```

c. A map can also contains constant key-value pair at compile time.

```
final fruits = const {
 101: ‘apple’,
 102: ‘orange’,
 103: ‘pear’
};

fruits[104] = ‘watermelon’; //error, can’t add to the map

print(fruits); //print maps
fruits.forEach((code,fruit) => print(‘$code is
${fruit}’));
```

d. Commonly used map properties.

Property	Description
`fruits.keys`	Returns all keys in the map.
`fruits.values`	Returns all values in the map.
`fruits.isNotEmpty`	If map not empty returns true.
`fruits.length`	Returns size of the map.
`fruits.isEmpty`	If map is empty returns true.

Summary

Key Points

Notes

e. Commonly used map functions.

Function	Description
`fruits.addAll()`	Add additional key-value pairs in the map.
`fruits.clear()`	Removes all values in the map.
`fruits.remove()`	Removes specific key and its value.
`fruits.forEach()`	Applies specific function on each key-value pair.

4. Functions

a. A function allows developer to implement modular approach, which each task is coded into a reusable function and called when it is needed.

b. A standard function in dart must consist the following elements:

```
Data_Type Function_Name (parameters){
   //Operation on the given parameter;
   //Optional Return of the operation's result;
}
```

c. **Optional Parameters (Parameter with [..])**

```
String fullName(String first, String last, [String?
middle]) {
  return "$first ${middle == null ? "" : "$middle "}
$last";
}

print(fullName("Ali","Abu"));
```

d. **Optional Named Arguments (Parameter with {...})**

```
bool threshold({int? value, int? min, int? max}) {
  return min! <= value! && value <= max!;
}

print(threshold(min:1, max:10, value:11));
```

Summary

Key Points	Notes

e. **Default Values (Parameter with =)**

```
bool threshold({required int value, int min = 0, int max = 10}) {
  return min <= value && value <= max;
}

print(threshold(value:5));
```

f. **First-Class Functions (Function as parameter)**

```
int applyTo(int value, int Function(int) op) {
  return op(value);
}
```

g. **Anonymous Functions and Closures (No function name)**

```
var multiply = (int a, int b) {
  return a * b;
};
```

h. **Lambda or Arrow Functions (Function with =>).**

```
var multiply = (int a, int b) => a * b;
```

5. References

f. A tour of the Dart Language. Retrieved on October 2nd, 2020 from https://dart.dev/guides/language/language-tour

g. Howard, J. (2019). Dart Basics. Retrieved from https://www.raywenderlich.com/4482551-dart-basics#toc-anchor-014

h. Dart Programming Tutorial. Retrieved on October 2nd, 2020 from https://www.tutorialspoint.com/dart_programming/index.htm

Summary