By: Eng. Elias Al-Showaiter

[Dart Programming Language] - Lecture [9]

[Exercises on advanced methods for lists and maps]

1. Introduction

In this lecture we will take a look at some of the most commonly used methods on lists and maps through examples, and to get there we will have to understand the concept of lambda functions.

Learning Objectives

1. Developing your skills to use maps and lists.

2. Examples

Lambda functions in Dart, also known as anonymous functions or inline functions, are functions that are created without a name. They are often used in situations where a function is needed temporarily or is passed as an argument to higher-order functions like forEach, map, or where. And their syntax looks like this:

```
(parameters)

{

 return value;

 };
```

Lambda functions will be showing in all the following examples.

Q1. Example 1:

Write a dart code that prints the element of a list.

A1: we can easily do this using a for loop. However, there is a built-in method that help us make this operation very fast. It is called forEach.

Let's take a look at how it works.

```
void main() {
  List<int> numbers = [1, 2, 3, 4, 5];
  numbers.forEach((number) {
    print(number);
  });
}
```

Now let's analyze this code as much as we can:

We have a list of integers called numbers containing the values 1,2,3,4,5. And from this list we called a function (behavior) called forEach. This function as we can see takes 1 input, but not as usual ***this input is not a variable***. This input is a function, and we need to ask ourselves a lot of questions.

1. Is this a function definition or a function call? This is a function definition.
2. Where is the name of the function? Lambda functions do not have a name.
3. How do I call this function if it does not have a name? they are called automatically.
4. Why is the number variable not defined and used directly? This variable is defined on the fly. At each iteration the variable number has a new value.
5. Can I add a new parameter? NO.
6. Why? Because you do not call the function, so you cannot control its value.

So this function will receive each element in the list and print it.

It is very important to note that the forEach function is void and does not return any value

Q2. Write a dart code that squares the element of a list.

As before we can use a normal for loop and do this task manually or do it using a built-in method called map, (there is no relation between this function and the data type Map).

```
void main() {
  List<int> numbers = [1, 2, 3, 4, 5];
  List<int> numbersSquered = numbers.map((number) {
   return number * number;
  }).toList();
  print(numbersSquered);}
```

In this code we defined a list of numbers and the task is to square each value.

To do this we need to loop through the list and apply something in each value in this case to square the number. There is a built in function called map that achieves this result.

This function similar to forEach has one parameter and it represents that elements of the list. The difference between map and forEach is that the map function is:

map: Transforms each element of a list and returns a new iteratable (which can be transformed into a list) containing the transformed elements.

forEach: Performs an action for each element of a list but does not return a new list.

So in this example the map function will loop through the list, and on each element it will take its value and transforms this value to its square. After it finishes the list, It returns an iteratable (a sequence of elements) to change this result into a list we use the .toList() method.

Q3. Extract the even number in this list [1,2,3,4,5,6] into a new list.

We can achieve this using the .where() method as follows:

```
void main() {
  List<int> numbers = [1, 2, 3, 4, 5, 6];
  List<int> evenNumbers = numbers.where((number) => number.isEven).toList();
  print(evenNumbers);
}
```

In this code we use the where function which loops through the list and returns the elements if the lambda function is true.

So we loop through each element in the list, and the lambda function checks if this value is even or not, if it is even then it returns its value, if not it will not return it. And similar to map the where method returns an iteratable, to change it into a list we use toList().

Q4. Calculate the discount of prices in this list [100.00 , 200.00 , 300.00] consider the discount 20%.

```
void main() {
  List<double> prices = [100.0, 200.0, 300.0];
  double discountRate = 0.20;
  List<double> discountedPrices = prices.map((price) {
    return price - (price * discountRate);
  }).toList();
  print(discountedPrices);  // Output: [80.0, 160.0, 240.0]
}
```

Q5. Write a dart code that filters inactive users from a list of users.

```
void main() {
  List<User> users = [
    User("Ali", true),
    User("Ahmed", false),
    User("Alia", false),
    User("Mona", true) ];
  List<User> activeUsers = users.where((user) {
    return user.isActive;
  }).toList();
 activeUsers.forEach((user)=>print(user.name));
}
class User {
  String name;
  bool isActive;
  User(this.name, this.isActive);
}
```

Q6. Write a code that prints the details of a list of books and determines if a book is a classic or not.

```
class Book {
  String title;
  String author;
  int publicationYear;
  double rating;
  Book(this.title, this.author, this.publicationYear, this.rating);
  void printSummary() {
    print(
        'Title: $title, Author: $author, Year: $publicationYear, Rating:
${rating.toStringAsFixed(1)}');
  }
  bool isClassic() {
    return DateTime.now().year - publicationYear > 50;
  }
}
void main() {
  List<Book> books = [
    Book('Crime and Punishment', 'Dostoevsky', 1866, 4.6),
    Book('The one', 'Talal Quassem', 2012, 4.8),
    Book('Dracula', 'Stoker', 1897, 4.4)];
  books.forEach((book) {
    book.printSummary();
    if (book.isClassic()) {
      print('${book.title} is a classic.');
    }
```