Interface

Workshop #8

Flutter Developer Bootcamp

# **Purpose**

This workshop demonstrates how to use interfaces in Dart to enforce a class to implement specific methods. It showcases how a class (Mobile) can implement multiple interfaces (Message and Call), each defining certain behaviors that the class must adhere to.

**Problem**

In the given workshop demonstrates the implementation of interfaces using abstract classes. The Mobile class implements both the Message and Call interfaces, which require methods like text, number, call, and receiveCall. In the main function, an instance of Mobile is created, showcasing how Dart supports multiple interface implementations in a structured and concise manner. This approach emphasizes code reusability and adherence to object-oriented principles like abstraction and encapsulation. You need to implement abstract class for Message and Call and print the message ‘the mobile must do these functions’.

**How to Solve**

1. Checkout the workshop from Git Repo:

git clone -b <user-branch> <repo-URL>

1. Open the root folder inside VS Code
2. Open the root folder in terminal
3. Run the command dart run filename.dart
4. Abstract Classes (Message and Call):
5. Define methods without implementation, acting as interfaces.

* Message interface: text() and number().
* Call interface: call() and receiveCall().

1. Concrete Class (Mobile):

* Implements both Message and Call interfaces.
* Constructor prints The mobile must do these functions.
* Overrides methods from interfaces with concrete implementations:
* text() sends a text.
* number() stores a number.
* call() makes a call.
* receiveCall() receives a call.

1. Main Function:

* Creates an instance of Mobile.
* Demonstrates calling interface methods (text(), number(), call(), receiveCall()) to showcase functionality.

1. Go To File: <specific-file--method> à <method-name>, implement your logic.

**You will Achieve**

When you complete this workshop you will learn the following:

**Methods and Functions Used:**

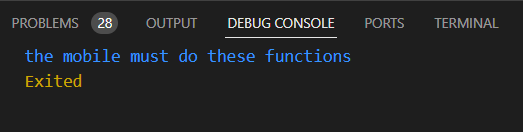
* **Abstract Classes (Message and Call):**
* Define method signatures (text(), number() in Message; call(), receiveCall() in Call) without implementation.
* Act as interfaces that specify what methods must be implemented by any class that implements them.
* **Concrete Class (Mobile):**
* Constructor (Mobile()):
* Initializes and prints a message indicating that the mobile must perform specific functions.
* Method Implementations:
* text():
* Sends a text message.
* number():
* Stores a phone number.
* call():
* Makes a phone call.
* receiveCall():
* Receives an incoming phone call.
* **Main Function:**
* Instantiation:
* Creates an instance of the Mobile class.
* Method Calls:
* Demonstrates calling each implemented method (text(), number(), call(), receiveCall()) of the Mobile instance.
* Shows how Dart supports multiple interface implementations in a structured manner.

# **Screenshots**

## **Before implementation (without creating abstract class)**



## **After implementation (with creating abstract class)**



# **How to submit your workshop**

Push your project back to the same git branch using command:

<command name>

# **Happy Coding!**