DART INHERITANCE

- Dart inheritance is defined as the process of deriving the properties and characteristics of another class.
- It provides the ability to create a new class from an existing class.
- It is the most essential concept of the oops(Object-Oriented programming approach).
- We can reuse the all the behavior and characteristics of the previous class in the new class.

Parent Class:

A class which is inherited by the other class is called superclass or parent class. It is also known as a base class.

Child Class:

A class which inherits properties from other class is called the child class. It is also known as the derived class or subclass.

```dart
class Father {
  var FatherTitle="Islam";
  FatherAsset(){
    print("House,Land");
  }
}
class Son extends Father {
  SonsAsset(){
    print(FatherTitle);
  }
}

void main() {
  var obj=new Son();
  obj.SonsAsset();
}
```

METHOD OVERRIDING

- When we declare the same method in the subclass, which is previously defined in the superclass is known as the method overriding.

- The subclass can define the same method by providing its own implementation, which is already exists in the superclass.

- The method in the superclass is called method overridden, and method in the subclass is called method overriding.

```
class Father {
  var FatherTitle="Islam";
  FatherAsset(){
    print("House,Land");
  }
}
class Son extends Father {
  FatherAsset(){
    print("House,Land,Gold");
  }
}

void main() {
  var obj=new Son();
  obj.FatherAsset();
}
```

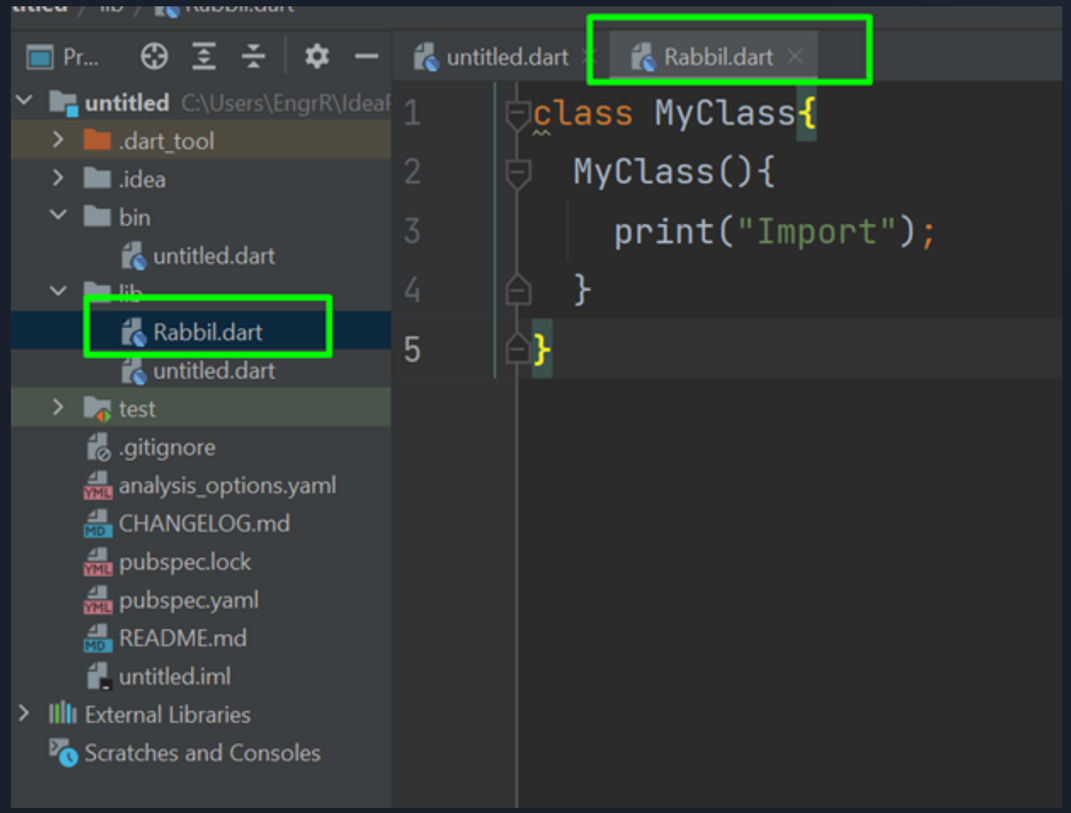
DART ABSTRACT CLASSES

- Abstract classes are the classes in Dart that has one or more abstract method.
- Abstraction is a part of the data encapsulation where the actual internal working of the function hides from the users.
- They interact only with external functionality.
- We can declare the abstract class by using the abstract keyword.
- There is a possibility that an abstract class may or may not have abstract methods.

```dart
abstract class Father {
  var FatherTitle="Islam";
  FatherAsset(){
    print("House,Land");
  }
}
class Son extends Father {
  FatherAsset(){
    print("House,Land,Gold");
  }
}

void main() {
  var obj=new Son();
  obj.FatherAsset();
}
```

DART IMPORT CODE FORM EXTERNAL FILES




```dart
import 'package:untitled/Rabbil.dart';
void main() {
 var obj= MyClass();
}
```

DART DEBUGGING

Learn with Rabbil

WHY DEBUGGING IS IMPORTANT

Learn with Rabbil

- To find out my mistake
- To test/check my code , that works well
- To understand complex program flow
- To works with complex action part by part
- To improve my code