WELCOME BACK
&
THANK YOU

<Cre8tive Devs. Software/>

Advance Java Crash Course

For TD Bank

MEET YOUR CRASH COURSE TEAM

TANGY F.

CEO

WILLIAM D.

DEVELOPER

TONY J.

T.A * DEVELOPER

Rapid Review

TRIVIA

Rapid Trivia

What was the original name of the Java language.

<Cre8tive Devs.
Software/>
MOBILE APPS•WEBSITES•AR•VIDEO GAMES•SECURITY

TODAY'S AGENDA

TRAINING DAY 3

1 Yesterday's Coding Exercise to Go Rapid Review

2 Multi Threading

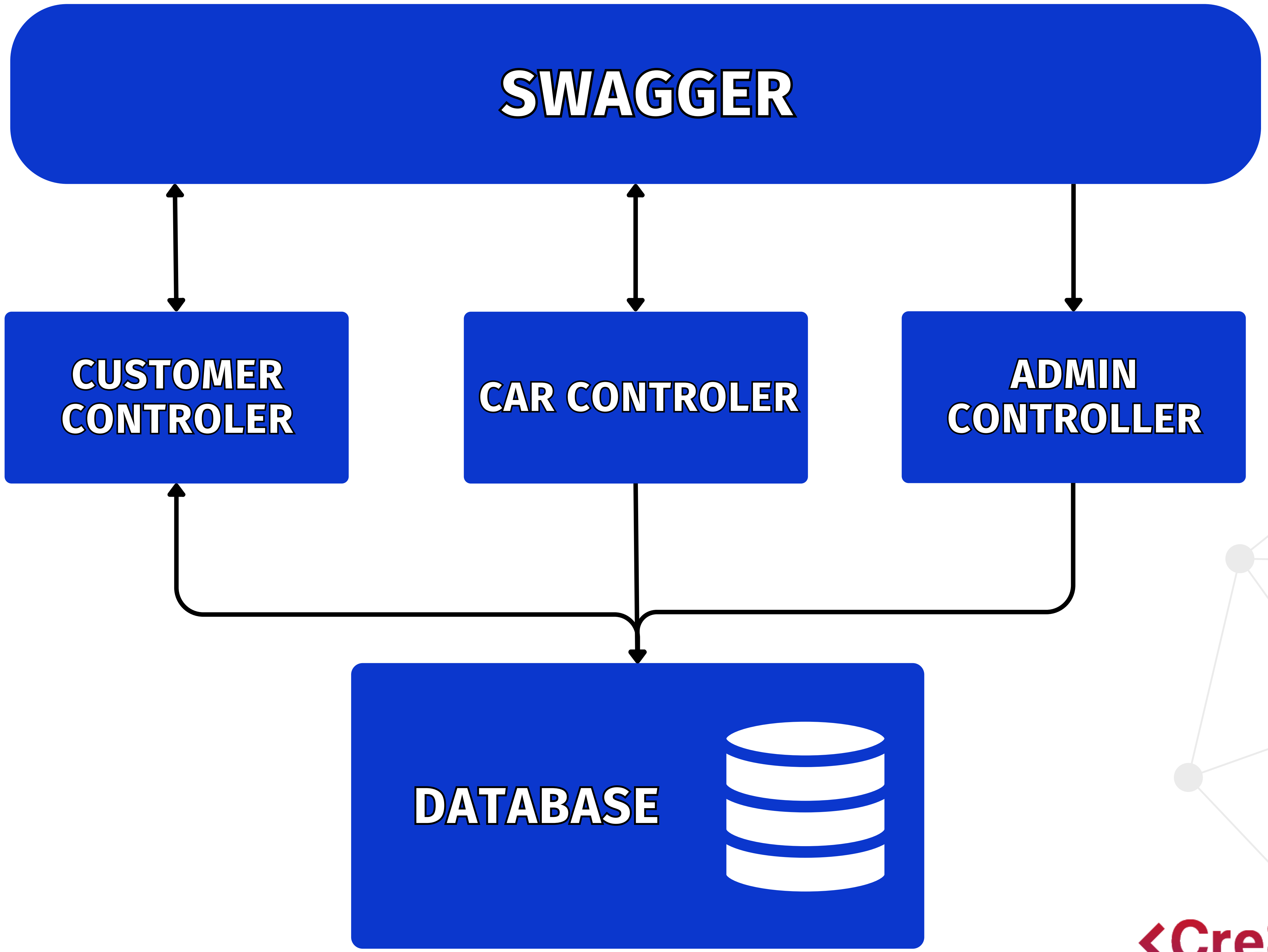
2 Circuit Breakers

3 Generics

4 Design Patterns. The Bridge Pattern

BITE SIZE
CAR RENTAL APP

WE ARE BUILDING



Rapid Review
Lesson 1

Rapid Review

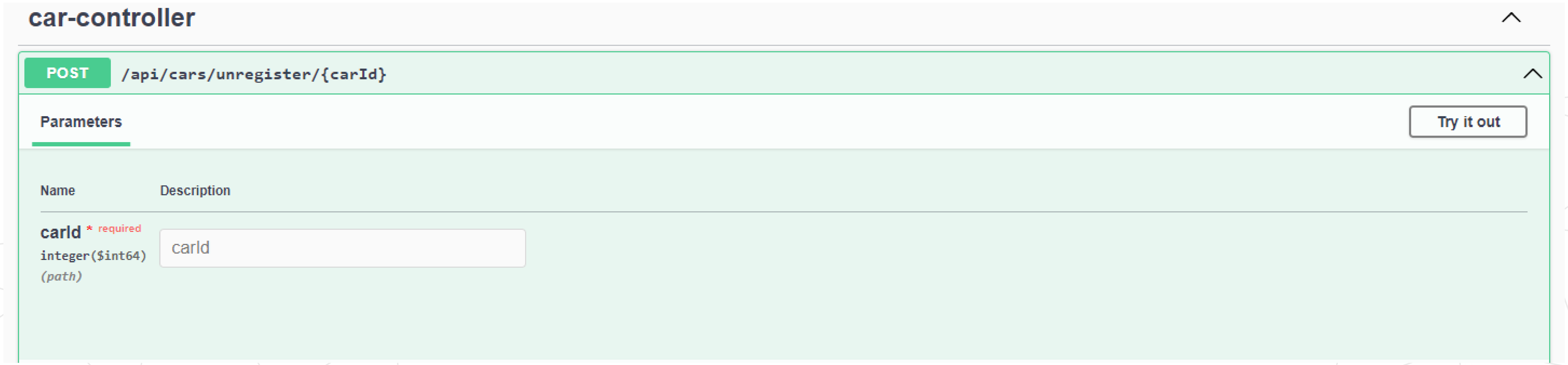
Adding commitment control to the unregister car API

Cre8tive Devs. Software/
MOBILE APPS•WEBSITES•AR•VIDEO GAMES•SECURITY

YERSTERDAYS CODING EXERCISE TO GO

In our **car-controler /api/cars/unregister by carId** the release of a car involves 2 tables with **executeUpdate().** Find the location where this happens and add commitment control to ensure they are not left out of sync if the second e**xecuteUpdate()** fails for any reason.

You will need to book a car first and then release it with the **unregister** API



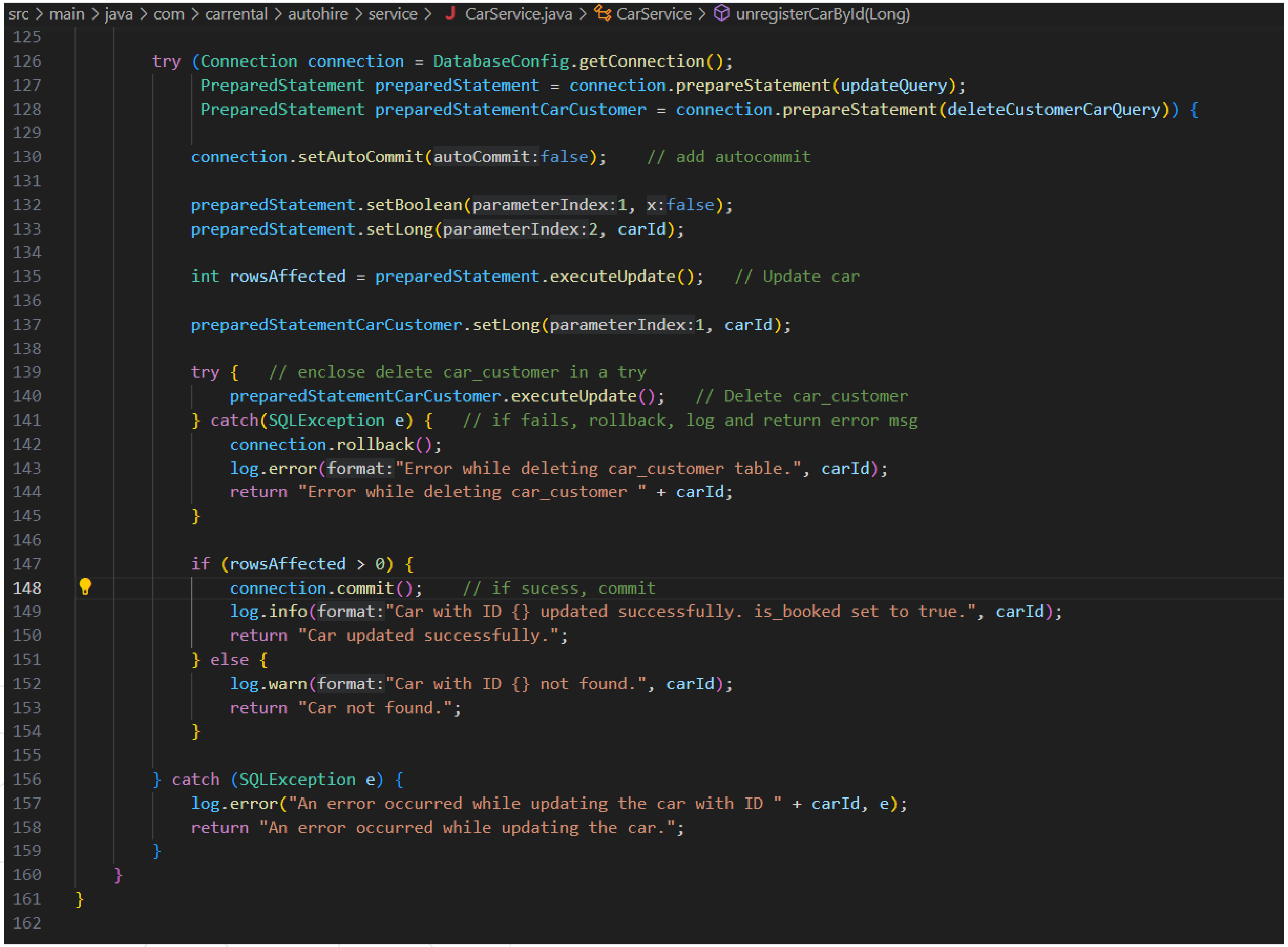
car-controller

POST `/api/cars/unregister/{carId}`

Parameters

Name	Description
carId * required integer($int64) *(path)*	carId

YESTERDAY'S CODING TO GO REVIEW



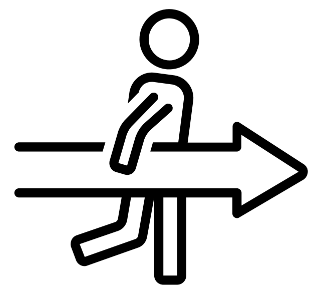
src > main > java > com > carrental > autohire > service > CarService.java > CarService > unregisterCarById(Long)

```
125
126             try (Connection connection = DatabaseConfig.getConnection();
127                  PreparedStatement preparedStatement = connection.prepareStatement(updateQuery);
128                  PreparedStatement preparedStatementCarCustomer = connection.prepareStatement(deleteCustomerCarQuery)) {
129
130                 connection.setAutoCommit(autoCommit:false);    // add autocommit
131
132                 preparedStatement.setBoolean(parameterIndex:1, x:false);
133                 preparedStatement.setLong(parameterIndex:2, carId);
134
135                 int rowsAffected = preparedStatement.executeUpdate();   // Update car
136
137                 preparedStatementCarCustomer.setLong(parameterIndex:1, carId);
138
139                 try {   // enclose delete car_customer in a try
140                     preparedStatementCarCustomer.executeUpdate();   // Delete car_customer
141                 } catch(SQLException e) {   // if fails, rollback, log and return error msg
142                     connection.rollback();
143                     log.error(format:"Error while deleting car_customer table.", carId);
144                     return "Error while deleting car_customer " + carId;
145                 }
146
147                 if (rowsAffected > 0) {
148                     connection.commit();    // if sucess, commit
149                     log.info(format:"Car with ID {} updated successfully. is_booked set to true.", carId);
150                     return "Car updated successfully.";
151                 } else {
152                     log.warn(format:"Car with ID {} not found.", carId);
153                     return "Car not found.";
154                 }
155
156             } catch (SQLException e) {
157                 log.error("An error occurred while updating the car with ID " + carId, e);
158                 return "An error occurred while updating the car.";
159             }
160         }
161     }
162
```

setAutoCommit to false #130

Enclose second executeUpdate() in a try. Include rollback() inside the catch and log and return errors #139

If success, commit #148



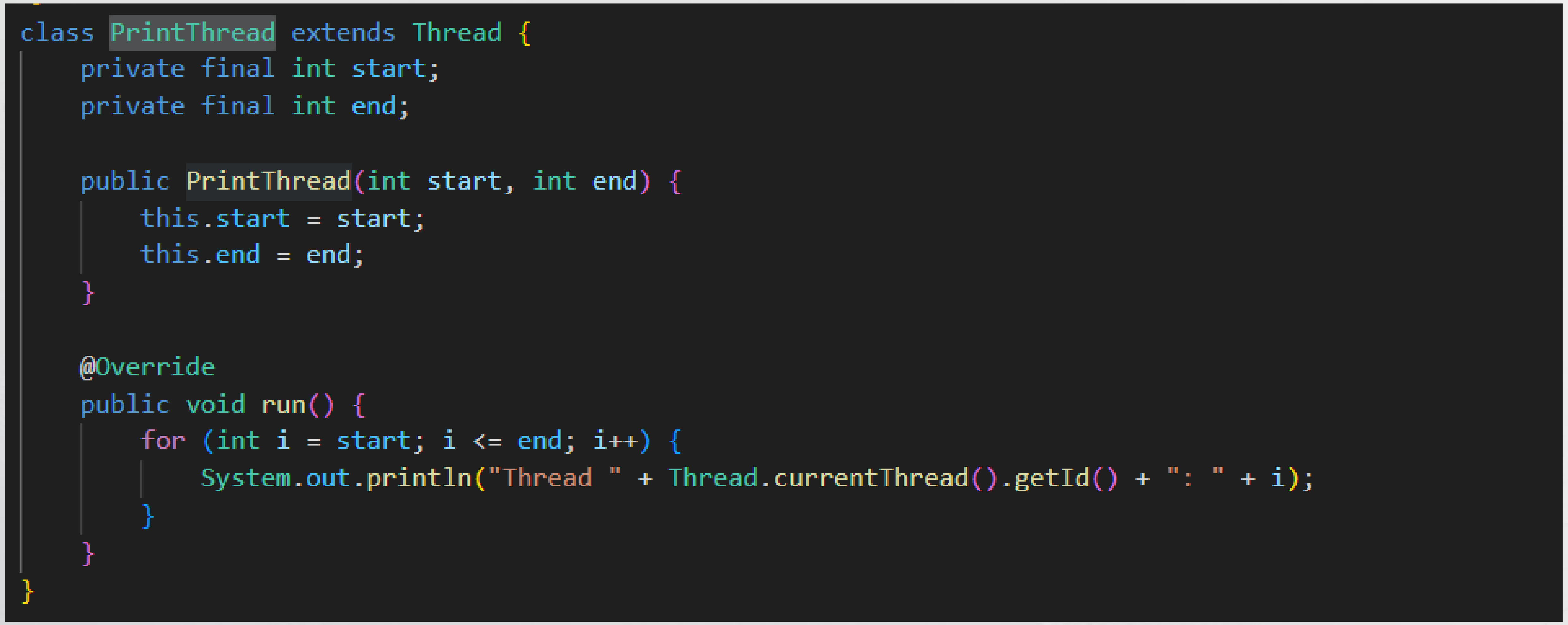
Let's see it running.

DAY 3
LESSON 2

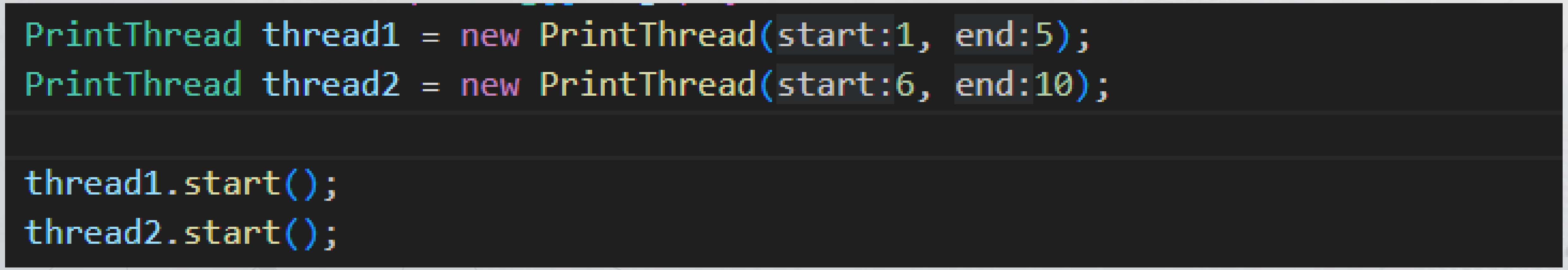
Multithreading

<Cre8tive Software/> Devs.
MOBILE APPS•WEBSITES•AR•VIDEO GAMES•SECURITY

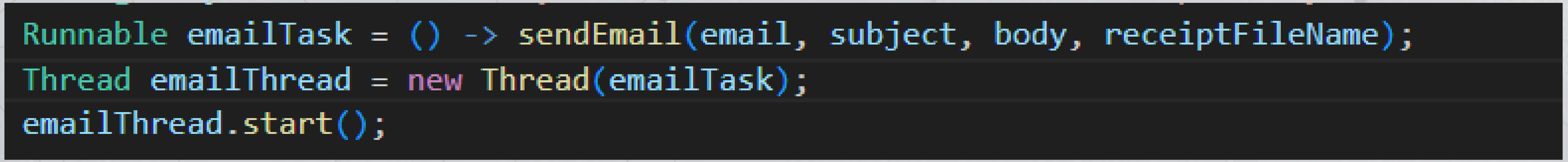
MULTITHREADING



```java
class PrintThread extends Thread {
    private final int start;
    private final int end;

    public PrintThread(int start, int end) {
        this.start = start;
        this.end = end;
    }

    @Override
    public void run() {
        for (int i = start; i <= end; i++) {
            System.out.println("Thread " + Thread.currentThread().getId() + ": " + i);
        }
    }
}
```



```java
PrintThread thread1 = new PrintThread(start:1, end:5);
PrintThread thread2 = new PrintThread(start:6, end:10);

thread1.start();
thread2.start();
```



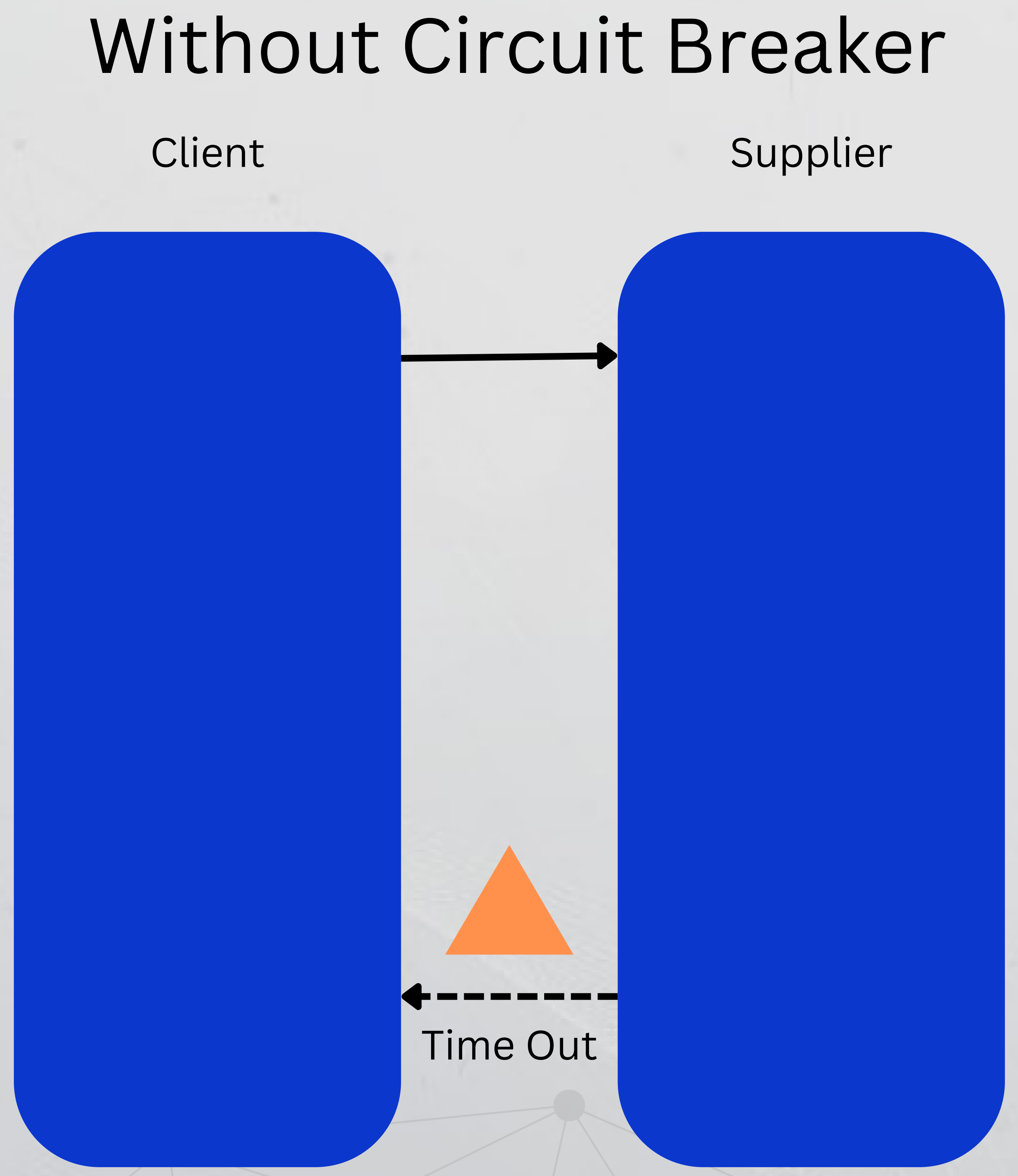
```java
Runnable emailTask = () -> sendEmail(email, subject, body, receiptFileName);
Thread emailThread = new Thread(emailTask);
emailThread.start();
```

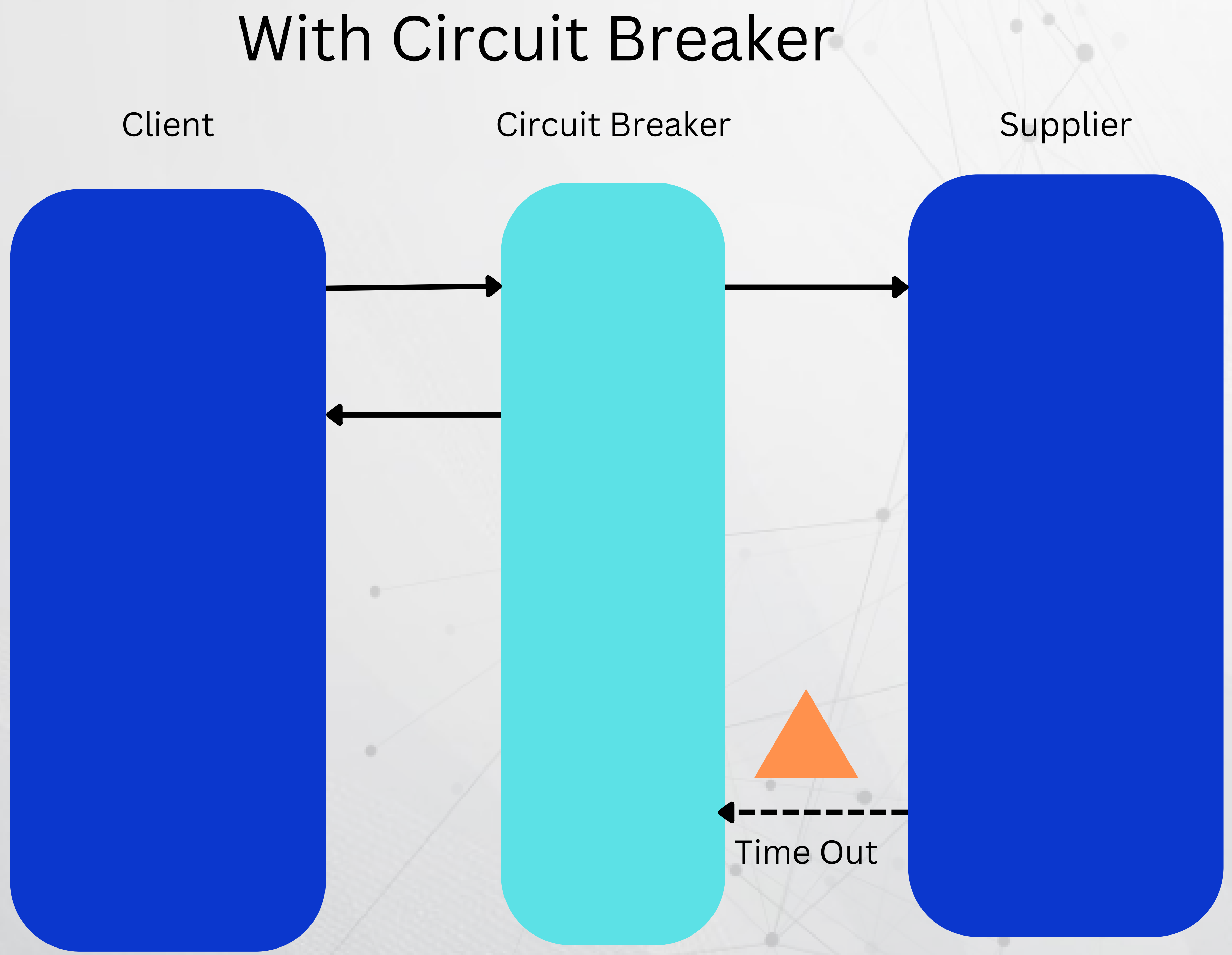
Let's see it running.

DAY 3
LESSON 3

Circuit Brakers

CIRCUIT BREAKERS



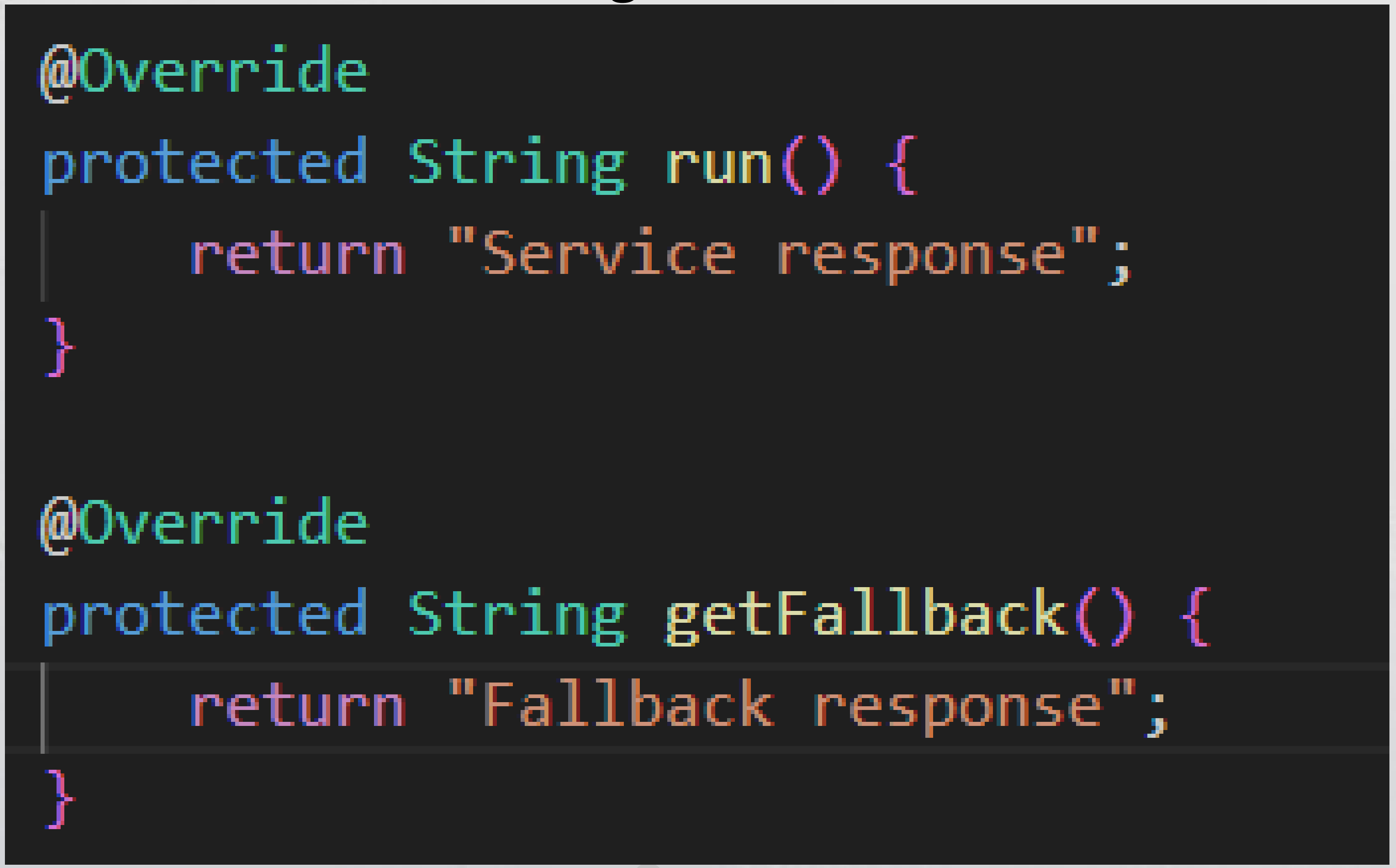


Let's see it running.

***The Circuit Breaker is on the server layer**

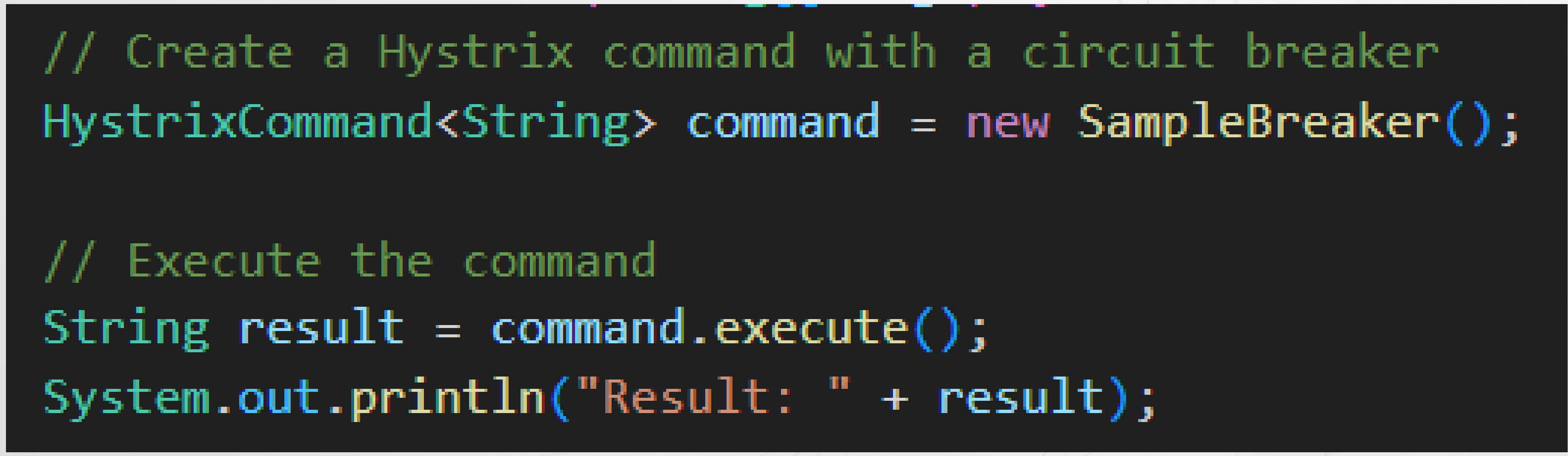
CIRCUIT BREAKERS

Create a class that extends **HystrixCommand** and then ***override Run()*** And ***getFallBack()***



```java
@Override
protected String run() {
    return "Service response";
}

@Override
protected String getFallback() {
    return "Fallback response";
}
```

The consumer class instantiate the **HystrixCommand** subclass and then runs the execute method.



```java
// Create a Hystrix command with a circuit breaker
HystrixCommand<String> command = new SampleBreaker();

// Execute the command
String result = command.execute();
System.out.println("Result: " + result);
```

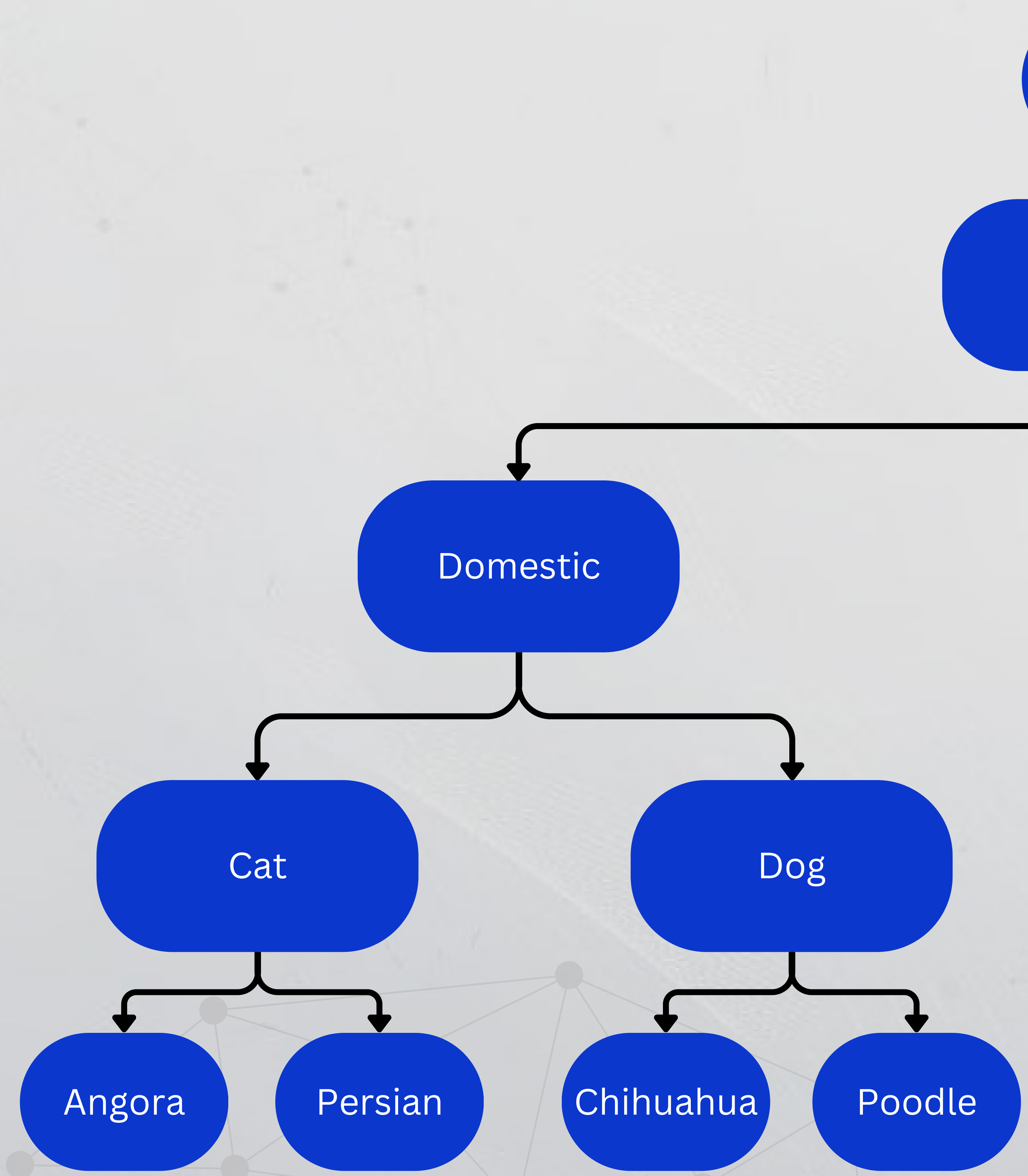
Let's see it running.

DAY 3
LESSON 4

Generic Types

<Cre8tive Software/>
Devs.
MOBILE APPS•WEBSITES•AR•VIDEO GAMES•SECURITY

GENERICS



Let's see the code.

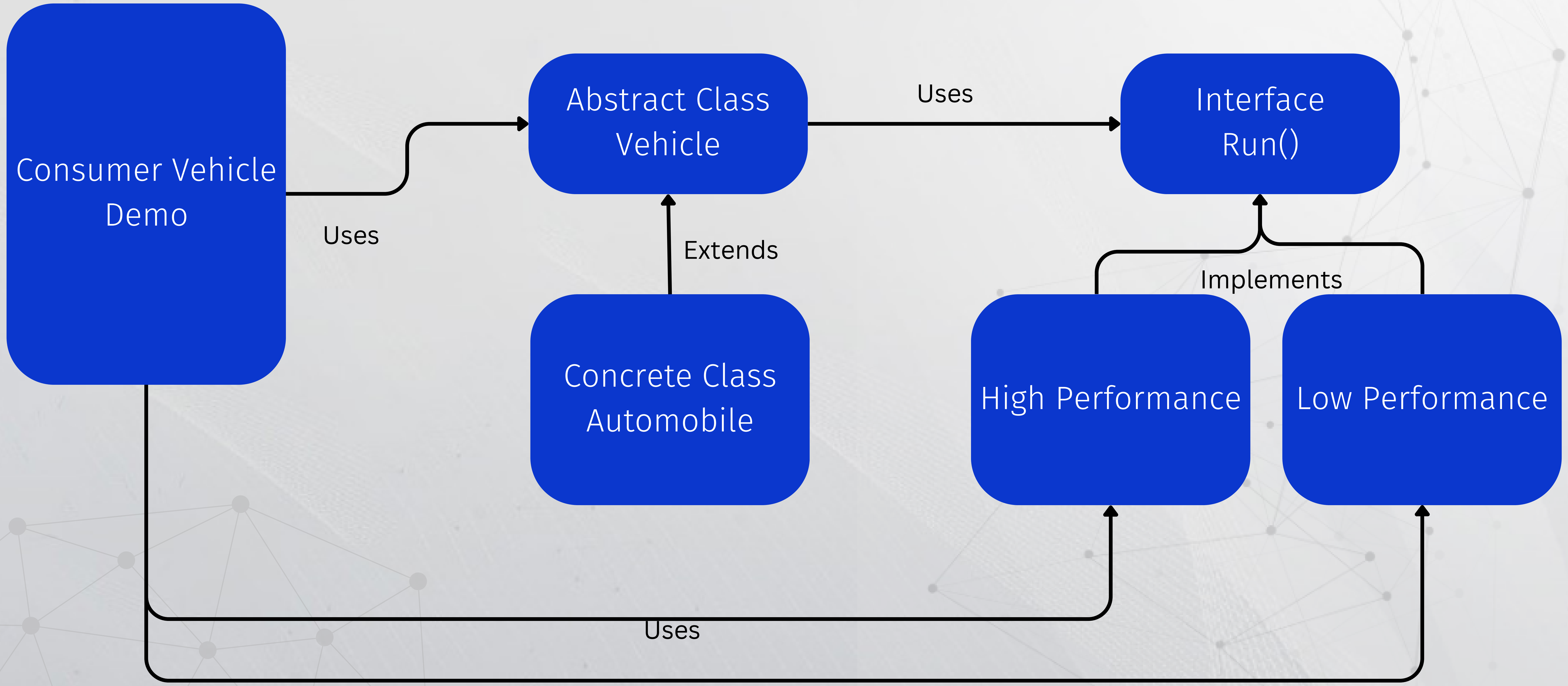
```
List<Domestic> domesticList = new ArrayList<>(); // A list of dogs and Cats
List<Wild> wildList = new ArrayList<>(); // A list of Lions and Zebras
List<Cat> catList = new ArrayList<>(); // A list of cat onlys but with Angoras and Persians
```

DAY 3
LESSON 5

Design Patterns
The Bridge Pattern

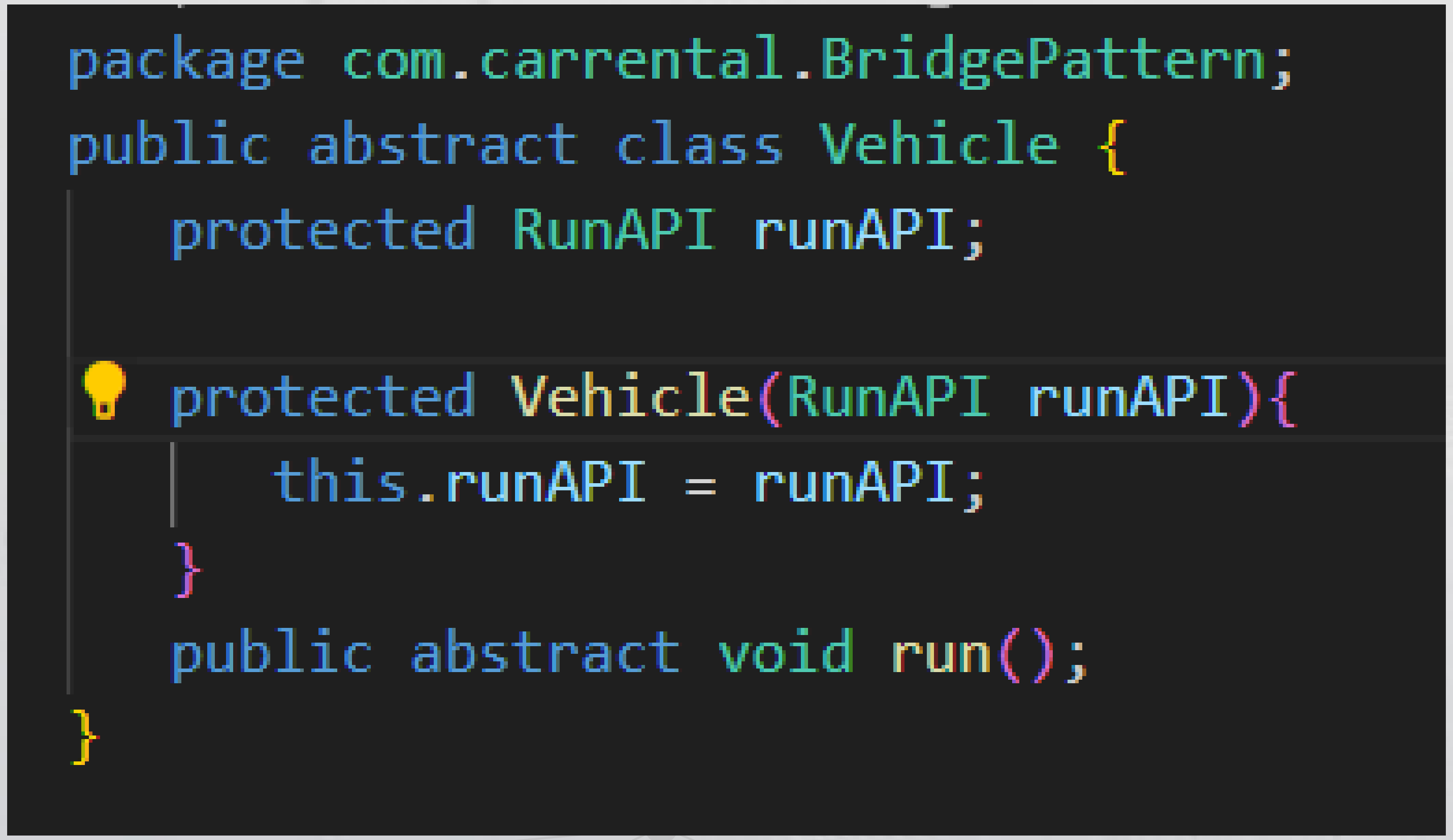
UNDERSTANDING DESIGN PATTERNS

Bridge Pattern

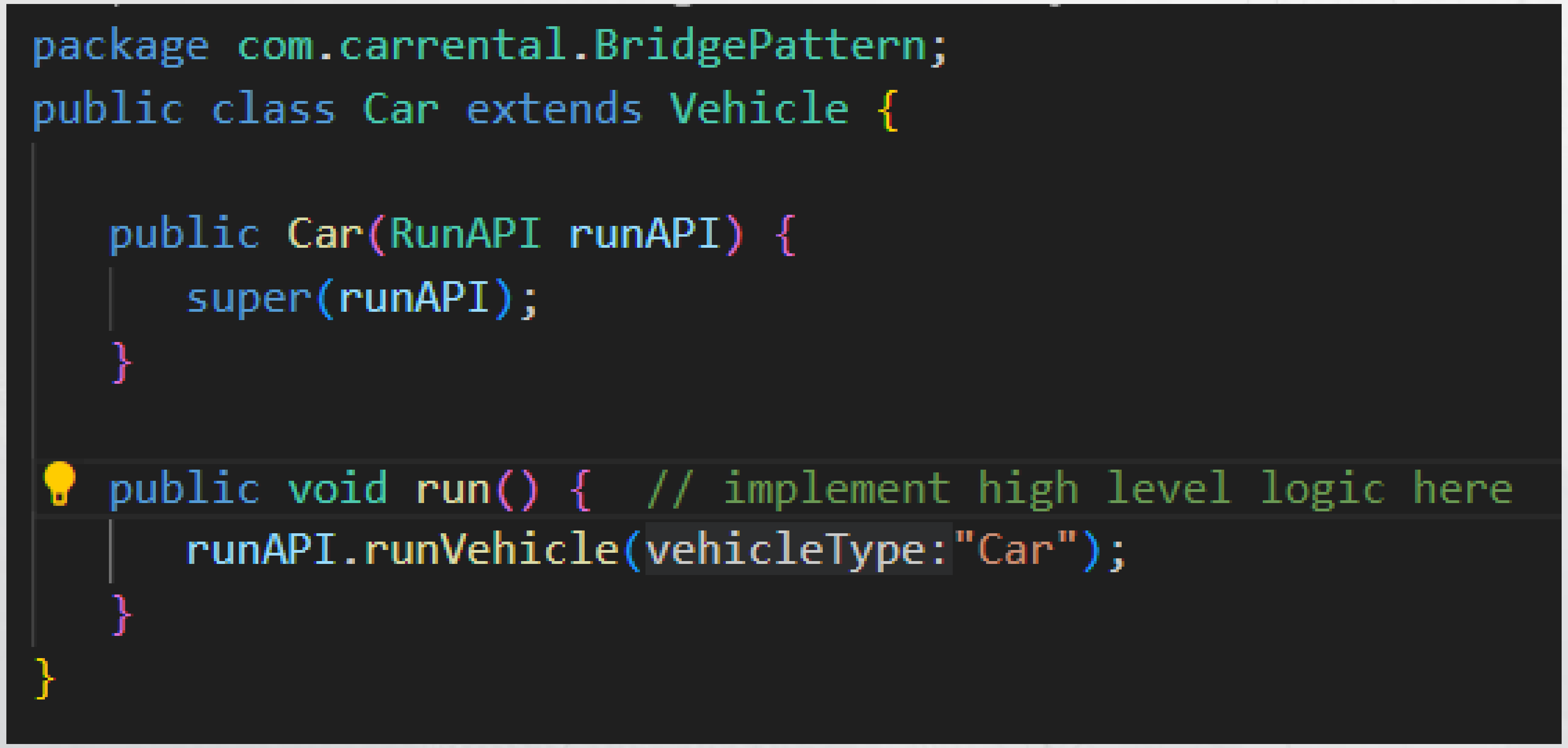


UNDERSTANDING DESIGN PATTERNS

Bridge Pattern High Level Abstraction



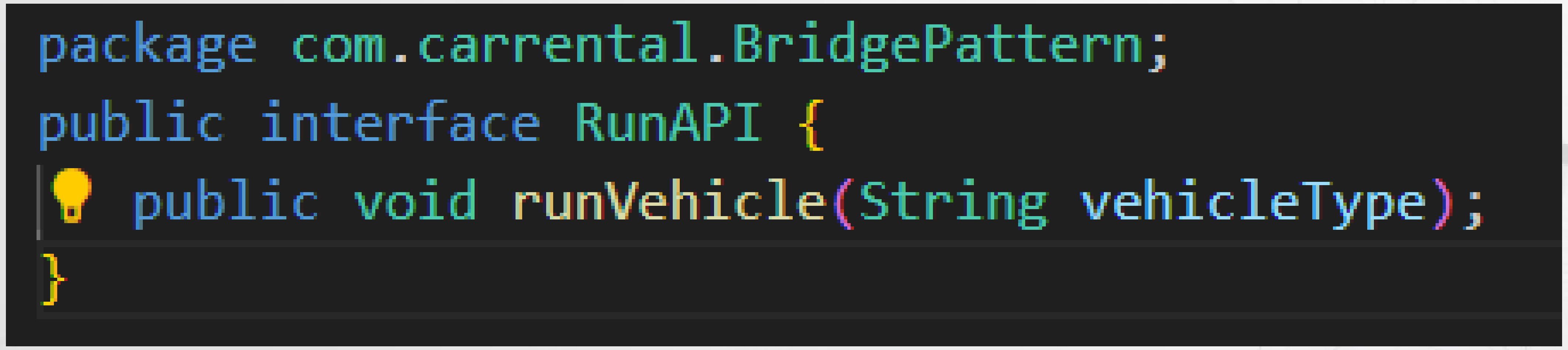
```java
package com.carrental.BridgePattern;
public abstract class Vehicle {
    protected RunAPI runAPI;

    protected Vehicle(RunAPI runAPI){
        this.runAPI = runAPI;
    }
    public abstract void run();
}
```



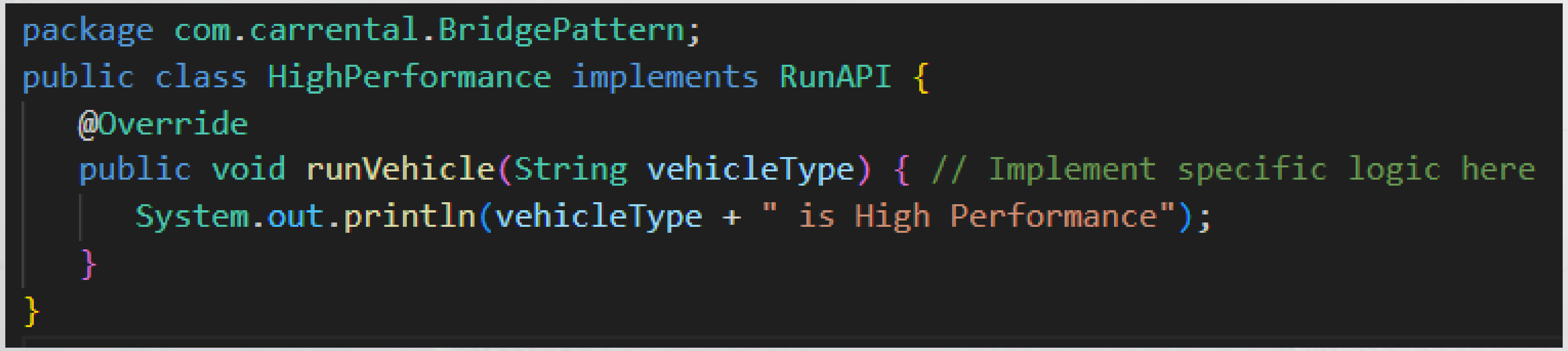
```java
package com.carrental.BridgePattern;
public class Car extends Vehicle {

    public Car(RunAPI runAPI) {
        super(runAPI);
    }

    public void run() {  // implement high level logic here
        runAPI.runVehicle(vehicleType:"Car");
    }
}
```

UNDERSTANDING DESIGN PATTERNS

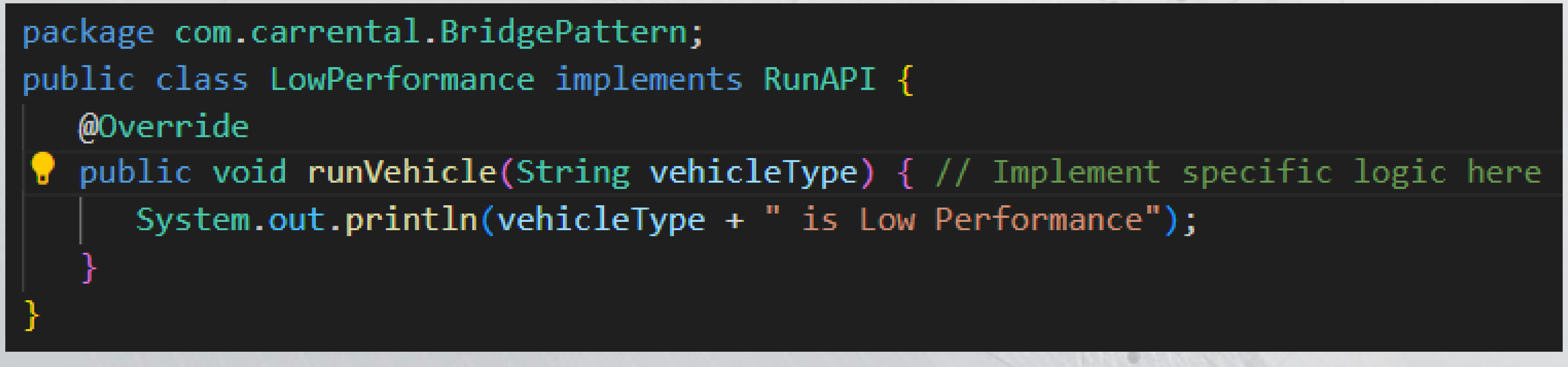
Bridge Pattern Low Level Implementation



```java
package com.carrental.BridgePattern;
public interface RunAPI {
    public void runVehicle(String vehicleType);
}
```



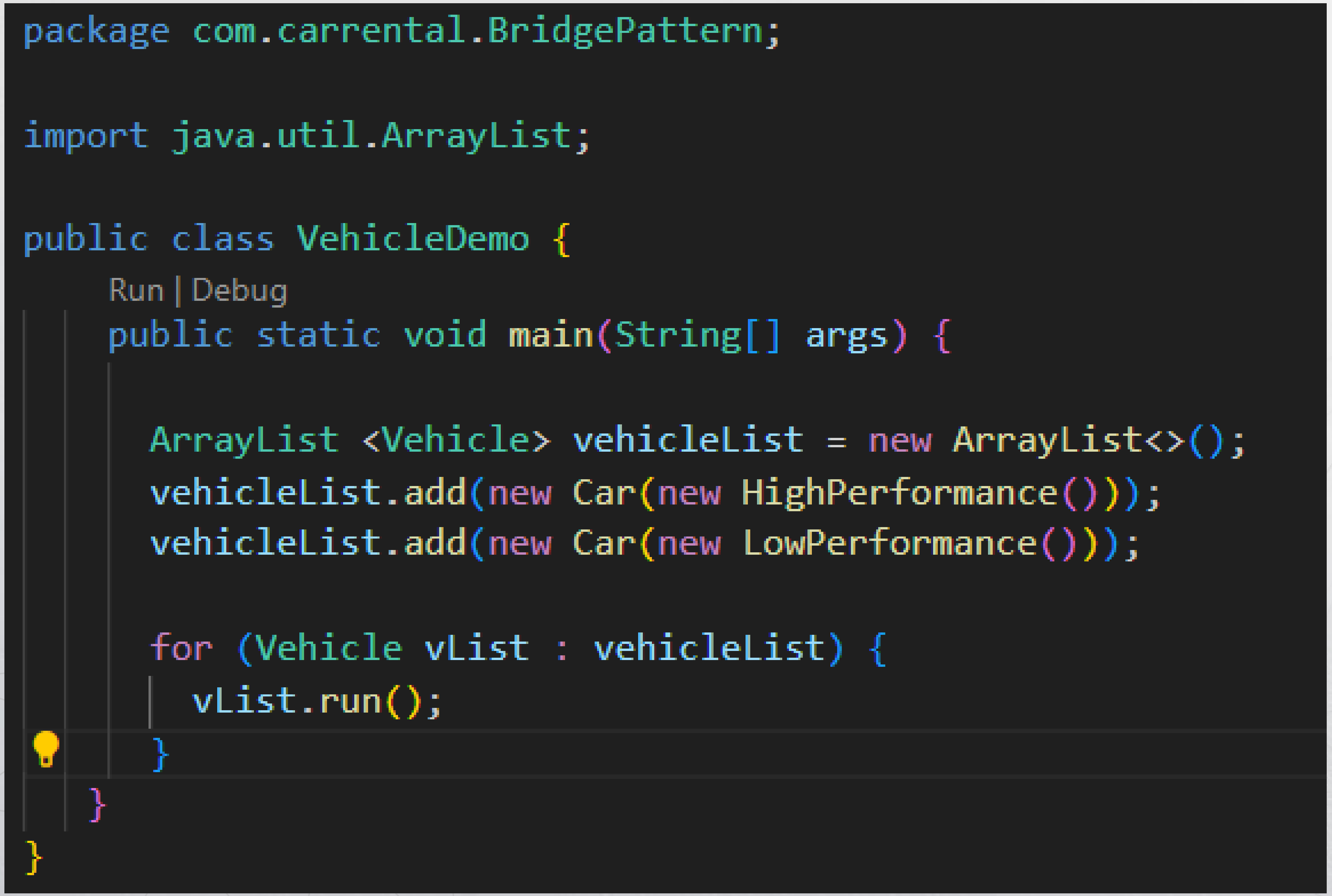
```java
package com.carrental.BridgePattern;
public class HighPerformance implements RunAPI {
    @Override
    public void runVehicle(String vehicleType) { // Implement specific logic here
        System.out.println(vehicleType + " is High Performance");
    }
}
```



```java
package com.carrental.BridgePattern;
public class LowPerformance implements RunAPI {
    @Override
    public void runVehicle(String vehicleType) { // Implement specific logic here
        System.out.println(vehicleType + " is Low Performance");
    }
}
```

UNDERSTANDING DESIGN PATTERNS

Bridge Pattern Consumer Class.



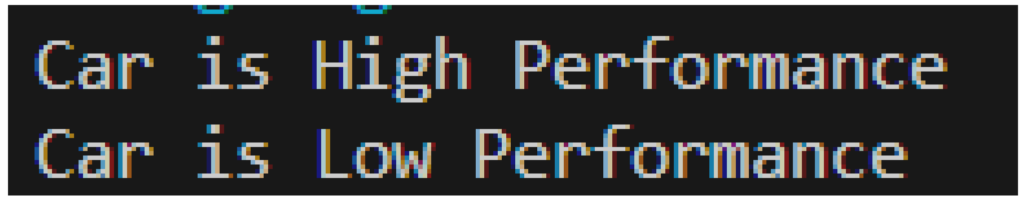
```java
package com.carrental.BridgePattern;

import java.util.ArrayList;

public class VehicleDemo {
    Run | Debug
    public static void main(String[] args) {

        ArrayList <Vehicle> vehicleList = new ArrayList<>();
        vehicleList.add(new Car(new HighPerformance()));
        vehicleList.add(new Car(new LowPerformance()));

        for (Vehicle vList : vehicleList) {
            vList.run();
        }
    }
}
```

Let's see it running.

CODING TRIVA QUESTION

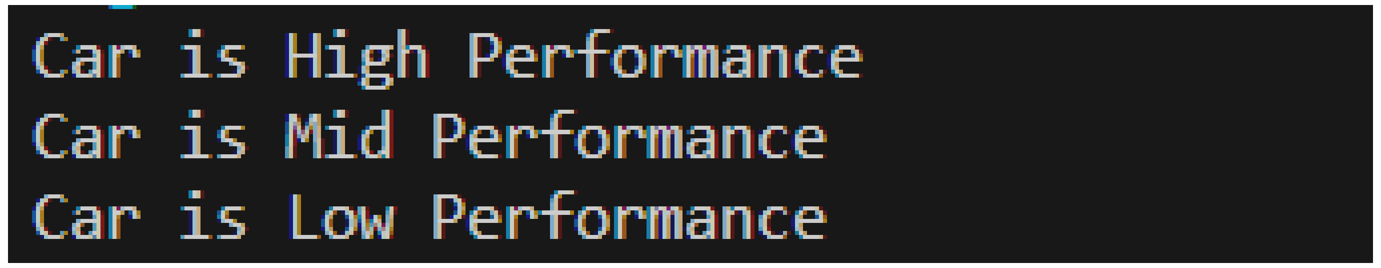
Add a new Mid Performance vehicle In our VehicleDemo class

How it started



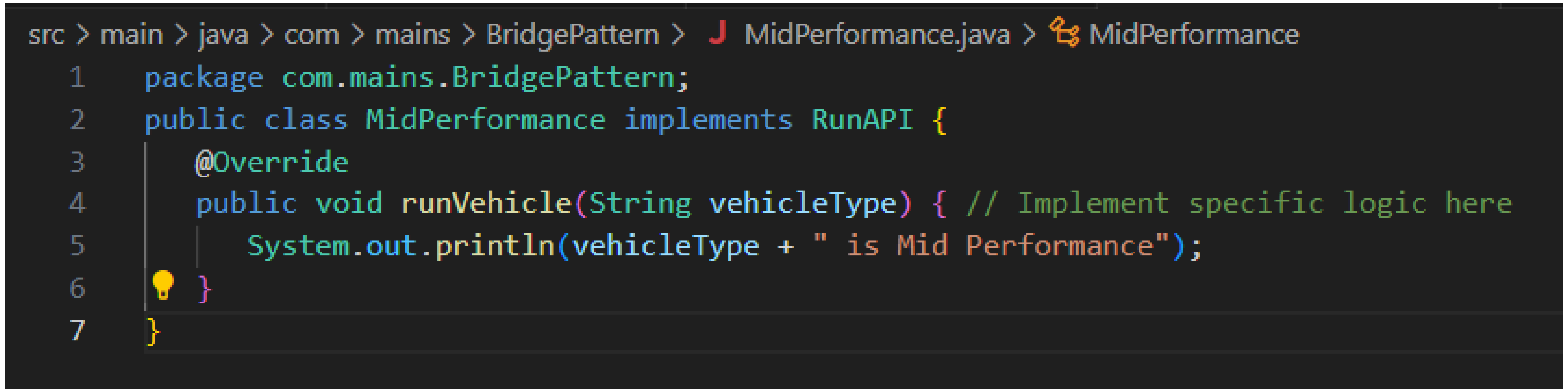
```
Car is High Performance
Car is Low Performance
```

After adding Midperformance



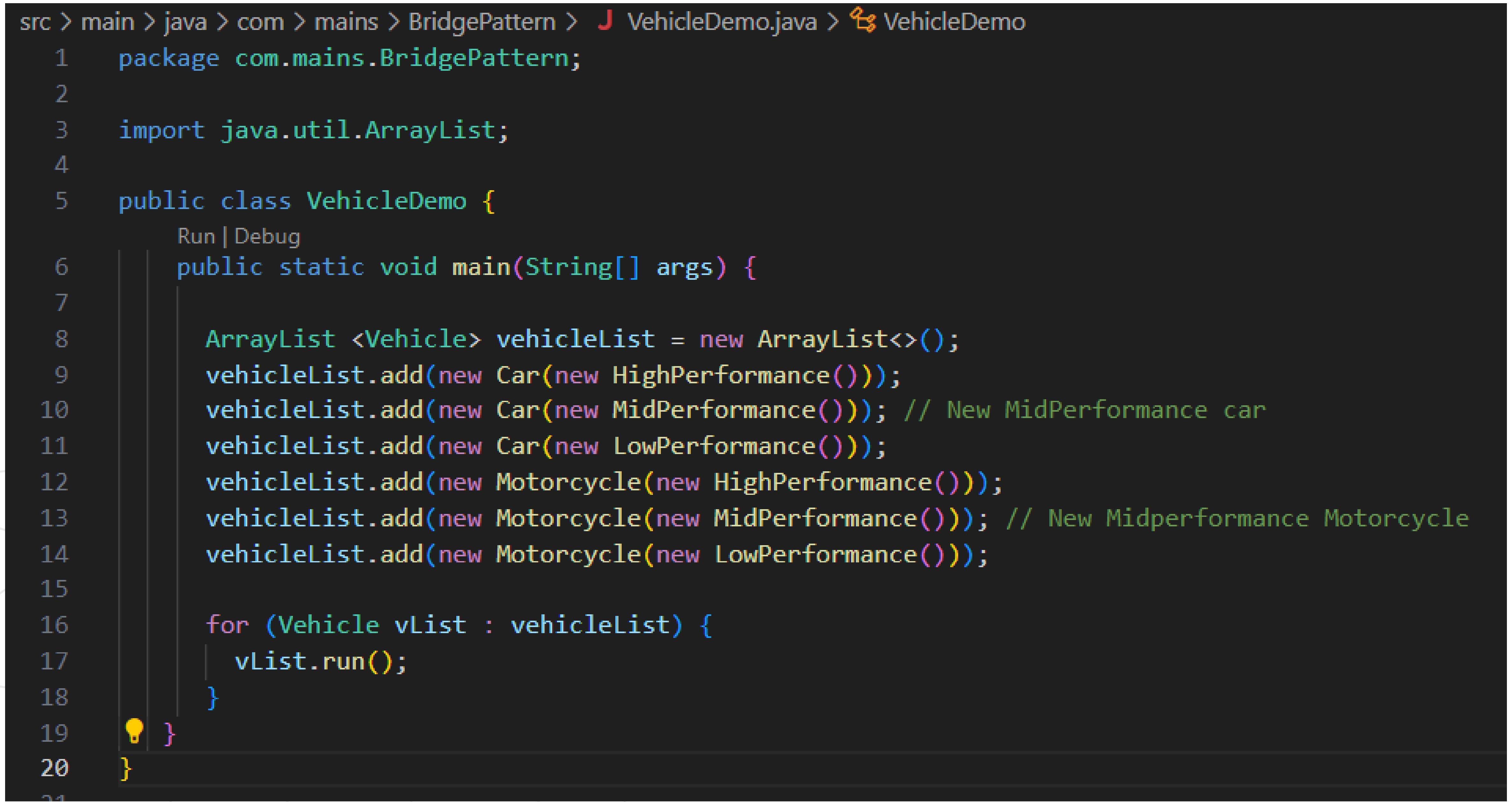
```
Car is High Performance
Car is Mid Performance
Car is Low Performance
```

CODING TRIVA ANSWER



src > main > java > com > mains > BridgePattern > MidPerformance.java > MidPerformance

```java
package com.mains.BridgePattern;
public class MidPerformance implements RunAPI {
    @Override
    public void runVehicle(String vehicleType) { // Implement specific logic here
        System.out.println(vehicleType + " is Mid Performance");
    }
}
```



src > main > java > com > mains > BridgePattern > VehicleDemo.java > VehicleDemo

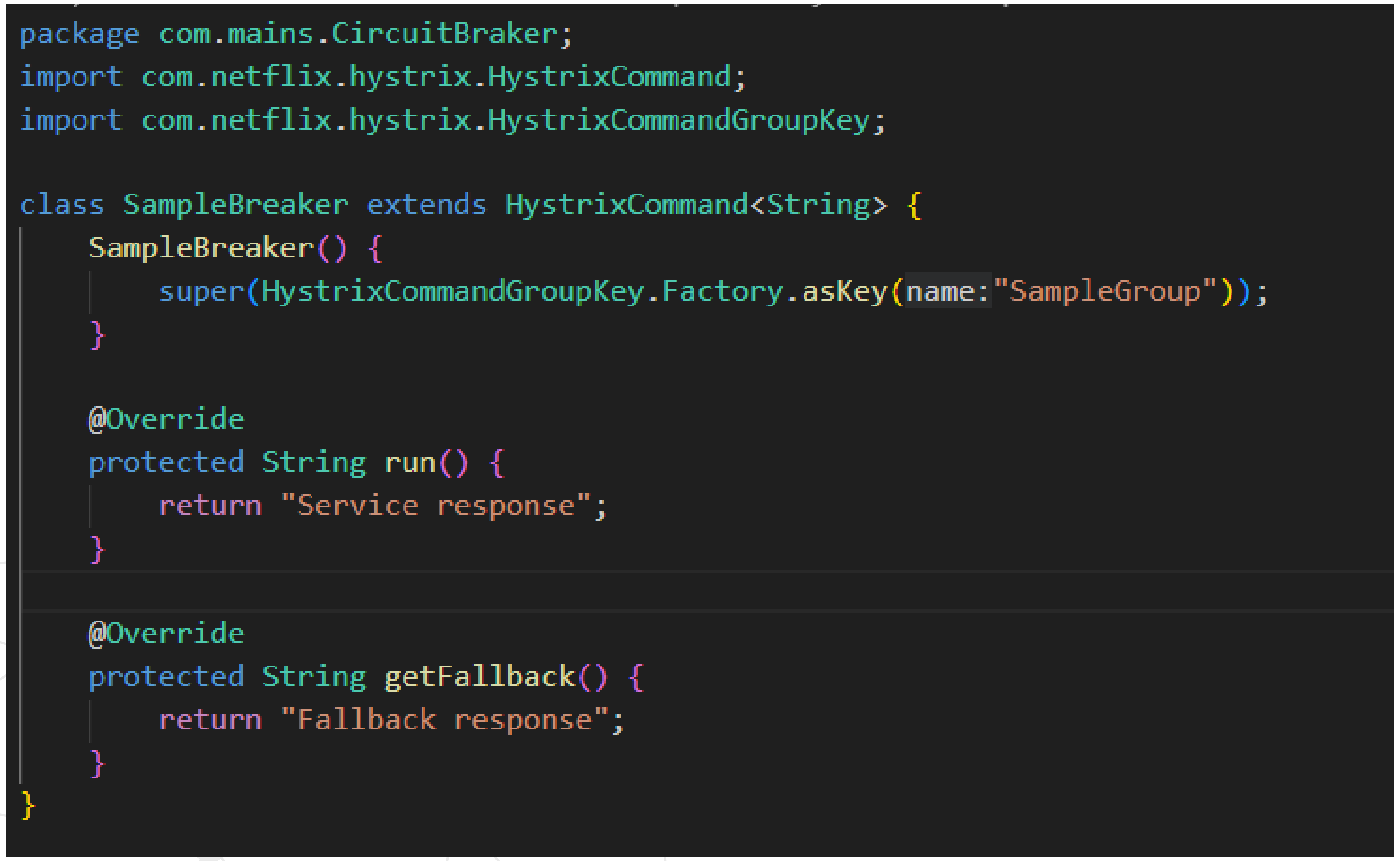
```java
package com.mains.BridgePattern;

import java.util.ArrayList;

public class VehicleDemo {
    Run | Debug
    public static void main(String[] args) {

        ArrayList <Vehicle> vehicleList = new ArrayList<>();
        vehicleList.add(new Car(new HighPerformance()));
        vehicleList.add(new Car(new MidPerformance())); // New MidPerformance car
        vehicleList.add(new Car(new LowPerformance()));
        vehicleList.add(new Motorcycle(new HighPerformance()));
        vehicleList.add(new Motorcycle(new MidPerformance())); // New Midperformance Motorcycle
        vehicleList.add(new Motorcycle(new LowPerformance()));

        for (Vehicle vList : vehicleList) {
            vList.run();
        }
    }
}
```

Let's Take a Look.

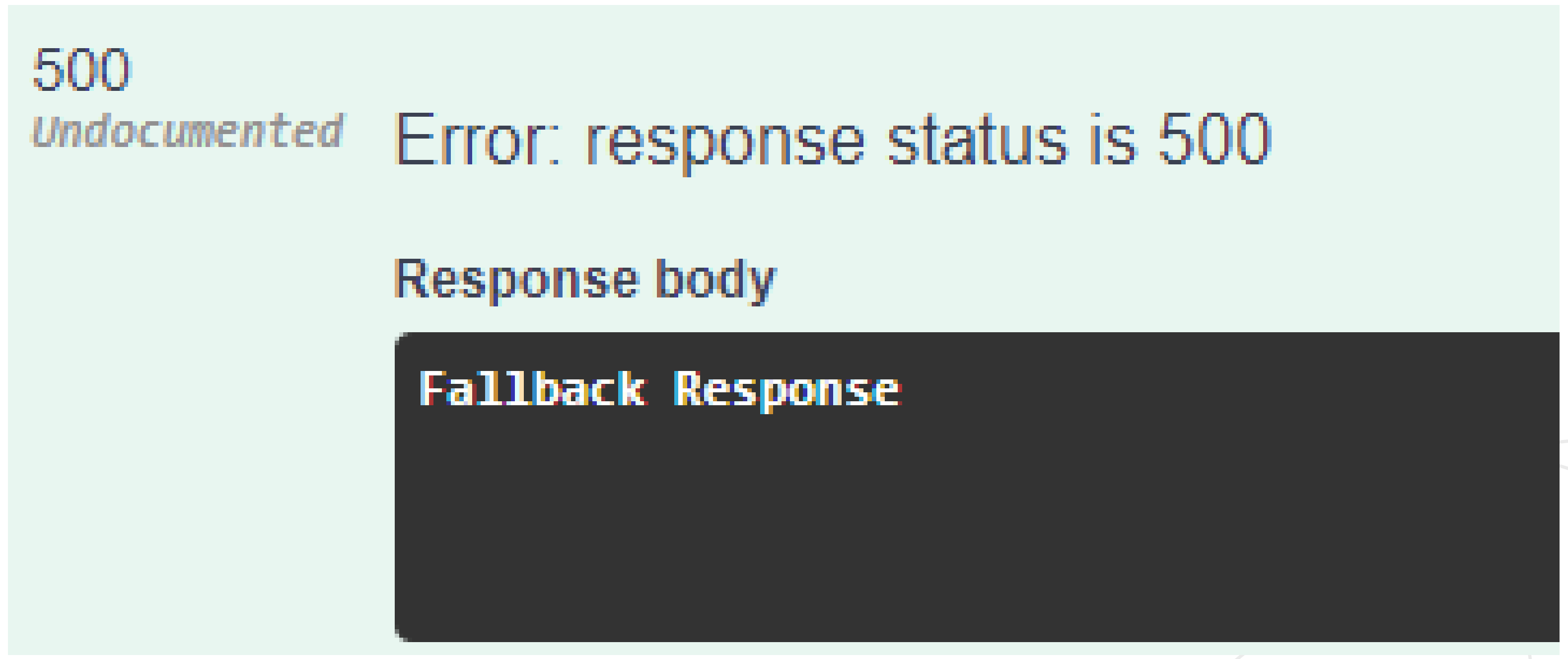
CODING EXERCISE TO GO

Create an UnregisterCircuitBreaker class that extends HystrixCommand<ResponseEntity<String>>. Use this class to add a circuit breaker to **unregister** API. You can use SampleBreaker bellow left as guide.

Use breakpoints to test and you should receive a return like the one bellow



```java
package com.mains.CircuitBraker;
import com.netflix.hystrix.HystrixCommand;
import com.netflix.hystrix.HystrixCommandGroupKey;

class SampleBreaker extends HystrixCommand<String> {
    SampleBreaker() {
        super(HystrixCommandGroupKey.Factory.asKey(name:"SampleGroup"));
    }

    @Override
    protected String run() {
        return "Service response";
    }

    @Override
    protected String getFallback() {
        return "Fallback response";
    }
}
```



500
Undocumented

Error: response status is 500

Response body

```
Fallback Response
```

THANK YOU

<Cre8tive Devs. Software/>

Crash Course

We will see you Monday