Goals

- This exercise gives you a chance to demonstrate your learning ability, while also giving you a look of a technology very popular in our code base.

Instructions

Prerequisites

You must install the following tools to finish the exam

- Maven (https://maven.apache.org/index.html)
- Java JDK 8+

General guidelines

- This exam has 2 parts. It is recommended that you finish part 1 before jumping to part 2.
- This exam requires you to be somewhat familiar with Dependency Injection design pattern and specifically Dagger 2. Spend some time studying those concepts. You can check the References section at the end of this document for some starting materials.
- You don't need to read all materials, some of them are overlapped, you can select the form that you are most comfortable with.
- The implementation is not hard once you understand those concepts, so don't panic if this seems too complicated. Navigating through complex problems is an essential skill to be successful at MoMo.
- Don't give up. It's another highly rewarded skill at MoMo.
- You can submit your work even when it is not completed. This is only one stage of the whole interview process.

Part I

We are building a robot with 2 arms, 2 legs. Since a robot must be flexible with different types of arms and legs, we are defining `Arm` & `Leg` as Java interfaces. Here's the `Arm` interface

```
/**
 * An interface representing an arm of the robot.
 */
public interface Arm {

  /**
   * Give a punch.
```

```
   */
  void punch();
}
```

We can have an `Arm` implementation as follows:

```
public class FireArm implements Arm {

  @Inject
  FireArm() {}

  @Override
  public void punch() {
     System.out.println("Ejecting fire!");
  }
}
```

Similarly, we also define a `LaserArm` class. All of the above classes are contained in the `vn.com.momo.robot.arm` package. We also have another package for the `Leg` interface and implementations.
For Part I, given some provided code, you will need to write some code to create a `SimpleRobot` with a left `FireArm` and a right `LaserArm`.
The file you will need to modify is `SimpleRobotTest.java` (inside `robot/src/test/java/vn/com/momo/robot/`). This is an incomplete test, which provided test cases. Specifically, your task is:

- Finish the `LeftFireArmRightLaserArmModule` Dagger's Module, so that your test compiles properly.
- Test the class with maven test command
  `mvn test`
- Make sure that the test passes.

When your test passes, you're done with Part I.

Part II

In part II, you will create a `FullRobot` with both arms and legs. The file to implement is `Main.java` (inside `robot/src/main/java/vn/com/momo/robot/`). Specifically, your task is:

- Write a Module to create a `FullRobot` with
  - a left LaserArm
  - a right FireArm
  - a left PaperLeg
  - a right IronLeg

- Write a Dagger's Component to plug all components together.
- Finish the `main` function to create an instance of FullRobot and call `doIt()` function.
- Run the program with the following command
  `mvn compile exec:java -rf :robot`
- Your program should print the following lines
  Ejecting a laser beam!
  Ejecting fire!
  Kicking with a Paper Leg!!
  Kicking with an Iron Leg!

When you get to this point, you're done with the exam. Zip your submission and send back to us! You just need to send 1 file containing 2 parts.

References

- Dependency Injection concept (not related to Dagger 2), recommended if you are not familiar with the concept: https://www.journaldev.com/2394/java-dependency-injection-design-pattern-example-tutorial
- Dagger Homepage: https://dagger.dev/
- Dagger video: https://www.youtube.com/watch?v=oK_XtfXPkqw