Groovy and Grails Course

Exercise 2: Metaprogramming

Overview: This exercise is designed to get you familiar with the capabilities offered by Groovy's meta programming APIs and used by Grails.

In this exercise you will learn how to use methodMissing to implement Grails style dynamic finders that translate static method calls like Book.findByTitle("Groovy in Action") into the equivalent SQL.

Instructions

Take a look at the Groovy script in the Exercise 2/code directory. The top part of the script defines a class called Book and creates an in-memory database, some tables and data to operate on. The interesting part starts on line 28:

```
Book.metaClass.static.methodMissing = { String name, args ->
```

This is where we register the metaprogramming hook methodMissing in the static context of the class. Every time a method fails to dispatch Groovy will delegate control to this method.

As we learned in the metaprogramming paterns part of the module the basic meta-programming pattern is intercept, cache invoke. The intercept part we can see on line 30:

```
     if(name.startsWith("findBy") && args) {
```

Here we intercept all calls that start with "findBy". The goal is to then dynamically create a new method that executes a SQL query. This is the act of "caching" the new behaviour, and can be seen in 2 parts. First we create a new closure that encapsulates the new behaviour on line 33:

```
          def newMethod = { Object[] varArgs ->
```

Then on line 38 we register this new method on the class:

```
Book.metaClass."$name" = newMethod // cache
```

Your job is to complete populate the closure with the functionality to dynamically create a SQL SELECT statement from the method signature.

Good luck!