Grails Application Development

Part 7 - Services

Ardhika

Objectives

- To learn how a service layer can be designed and built in Grails

Session Plan

- Services
  - Why?
  - What is?
- Creating a Service
- Injecting a Service
- Transactions

Services - Why do we need?

- Service Layer - Part of JEE application Stack
- Abstracted business Layer with centralized business logic
- logic involves two or more domain classes - that wont fit in a controller
- Help avoid controller bloating with logic/code
    - Lean controllers - easy to maintain
- An application level API
- Can be injected where ever we want

What is a service?

- So far we have been creating circles with an owner
- Cant we make owner automatically become a member of the circle that he/she created?
- Consider putting this code in the “save” action of the controller
- You need to first save the circle and then add the membership - 2 pieces of logic & out of place for create circle action

Creating a service

- New -> Service menu or
- grails create-service command with a name (with package)
- grails create-service com.ardhika.learn.Circle
- name is given without Service suffix (even while using the menu of GGTS)
- A class by name CircleService.groovy will be created with a default method

```
package com.ardhika.learn

class CircleService {

    def serviceMethod() {

    }
}
```

Creating a Service

- Rename the method and write this code

```
boolean createCircle(Circle circle) {
if(!circle.save()) return false
if(!Membership.subscribe(circle.owner, circle))
        return false
return true
}
```

- Here owner of the circle is added as a member using the subscribe method of the Membership domain class
- You can also use the circle.addToMembers method

Consuming a Service

- Done with Spring Dependency Injection
- Declare a variable with name circleService
- Note the came casing
- variable name is the same as Service class name
- Top of the CircleController class add this

  `CircleService circleService`
- Spring will infer this and inject an object of type CircleService
- No need to create the object with `new CircleService()`
- Time to rewrite the create action of CircleController

Consuming a Service

- New code for create action

```
def save() {
    def circleInstance = new Circle(params)
   //if (!circleInstance.save(flush: true)) {
if(!circleService.createCircle(circleInstance)) {
        render(view: "create",
               model: [circleInstance: circleInstance])
        return
    }

    flash.message = message(code:
                            'default.created.message',
                args: [message(code: 'circle.label',
                    default: 'Circle'), circleInstance.id])
    redirect(action: "show", id: circleInstance.id)
}
```

Transaction

- By default grails service methods are transactional
- if you want to switch it off use

```
class CircleService {
  static transactional = false

  //service methods
}
```

- Transactions work only with dependency injection
- If an exception is thrown in the service method due to validation or any other error transaction will be rolled back

Tasks

- Add methods in CircleService for
  - List of unsubscribed circle for a user
  - Any other suggestions?

Thank You!

Ardhika

Bala Sundarasamy
bala@ardhika.com