AOP Service

Summary

The AOP service implements and supports the Aspect Oriented Programming(AOP). The execution environment AOP service uses Spring AOP. This chapter examines the overview of AOP and AOP support of Spring.

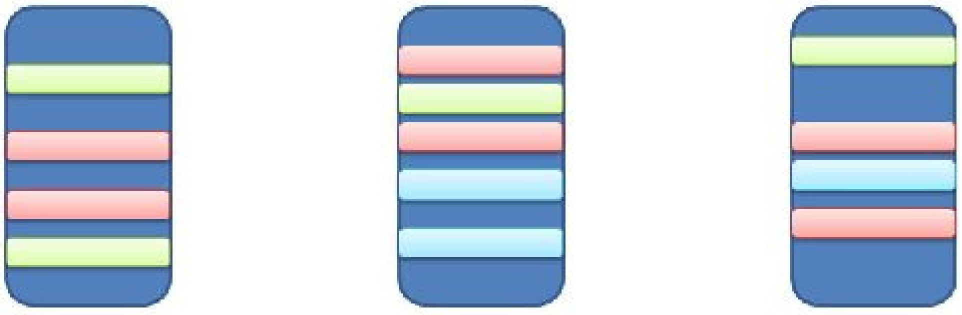
Description

AOP Overview

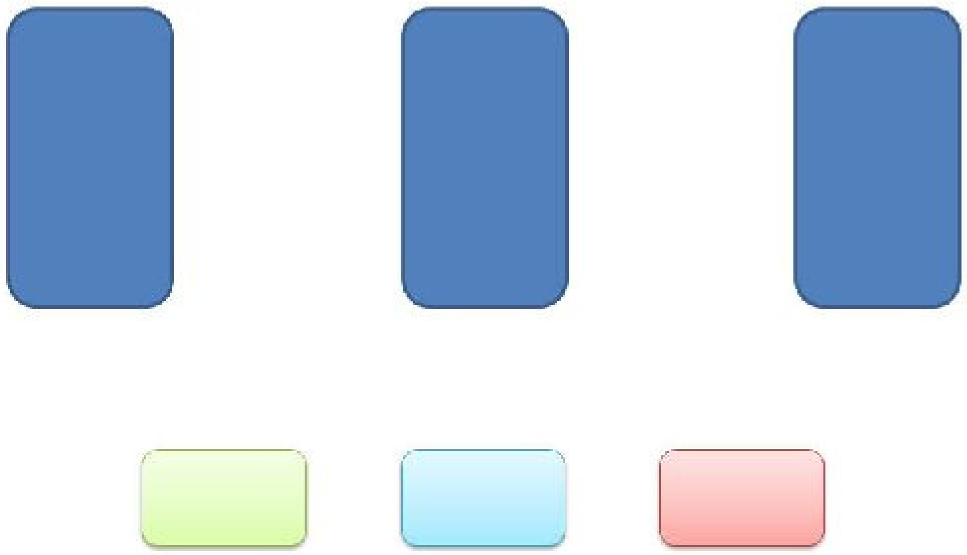
Individual programming language has Separation of Concerns paradigm to develop program. For example, procedural programming understands the running of continued functions without status value and defines the module in the unit of main separation. Object-oriented programming regards it as the group of interacting object rather than execution of series of functions, and has class as the main unit. Despite many advantages, object-oriented programming is distributed in many objects so that common concern exists in duplicate, which complicates the program and makes it difficult to change codes.
Aspect-oriented programming (AOP) is the method of supplementing this problem of object-oriented programming and is the programming styles that improves program modulation by separating the core concern. AOP separates the object into core concern and cross-cutting concern and defines the cross-cutting concern with the module called Aspect, and provides the method of processing it in connection with core concern.

- Aspect means the common program area that is applied over the core concern of program. That is, logging, authentication, permission check and transaction are the element that is applied in common when implementing business function. For example, logging, authentication, permission check and transaction are the element applied in common when implementing business function, and can be defined with one aspect.
- Core concern is the area where the core value and purpose of creating the program are exposed, and corresponds to ordinary core business function.
- Cross-cutting concern is the area of program that affects the core concern and corresponds to the system common processing area such as logging, transaction and authentication processing.

Following figure shows the case when the core concern and cross-cutting concern are integrated into one code in object-oriented programming development and developed.



When AOP is applied to the object-oriented programming code, the cross-cutting concern distributed to each code as shown in the following figure is defined separated into aspect. AOP links the aspect separated using the way of Weaving to the core concern.



AOP Main Concept

Aspect oriented programming provides the method of separating the cross-cutting concern and combining it with core concern. New concepts are included as shown below.

Aspect

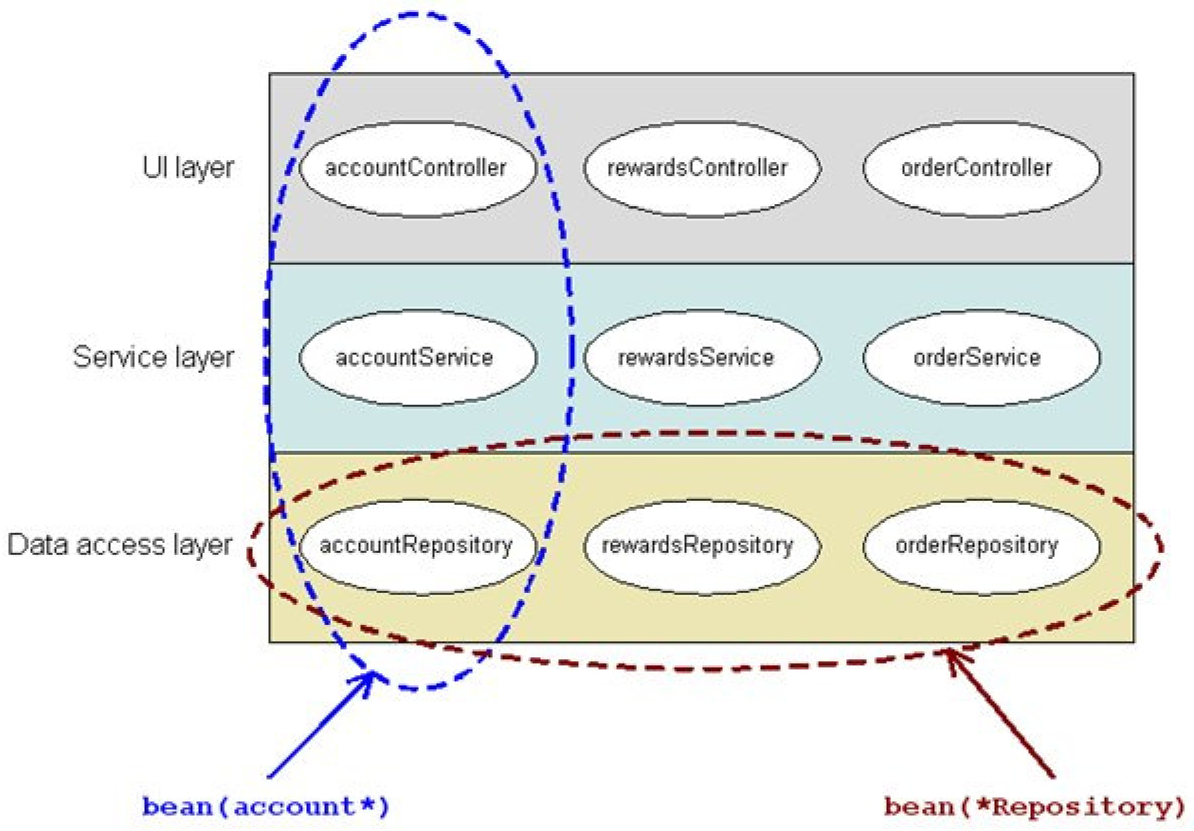
Aspect is the function of cross-cutting concern to implement.

Join point

Joint point refers to the specific point of application that can be executed by inserting aspect.

Pointcut

Pointcut means the grouping of the joint point. Pointcut defines the rule using the pattern matching to determine which joint point will be used. Following figure shows the example of selecting the bin vertically and horizontally using the bean() point cut included in Spring 2.5.



Advice

Advice is the actual implement of aspect and is the code that can operate with insertion to joint point. Advice is divided into before advice, after advice, around advice type depending on the time of operation in combination with the point of combination.

- Before advice: advice performed before joinpoint
- After returning advice: advice operating after joinpoint successfully returns
- After throwing advice: advice performed when an exception occurs and joinpoint escapes
- After (finally) advice: advice performed regardless of method of escaping join point (normal or exceptional return)
- Around advice: advice performed before and after joinpoint

Weaving

Weaving is the process of creating new proxy object by applying aspect to target object. Weaving type is separated into follows:

- Weaving at the time of compile: cross-cutting concern codes created in the form of aspect between core concern modules through separate compiler are inserted and final binary is created with application of aspect. (ex. AspectJ, …)
- Weaving at the time of class loading: when JVM loads the class using separate Agent, it changes binary information of relevant class. That is, Agent supports APO by providing binary codes inserted with cross-cutting concern codes. (ex. AspectWerkz, …)
- Runtime weaving: the type of supporting AOP using proxy without changing source code or binary file. While it approaches the objects implementing core concern through proxy, proxy performs the cross-cutting concern right before and after performing the core concern. Accordingly, it is restricted in that AOP can be applied only when calling method in case of proxy-based runtime weaving. (ex. Spring AOP, …)

Introduction

Introduction adds new method or property. Spring AOP can add new interface to the target object of receiving advice.

AOP Proxy

AOP Proxy is the object to be created after the advice is applied to Target Object.

Target Object

Target object is the object to receive Advice. Since Spring AOP uses runtime proxy, the target object always becomes the proxy object.

AOP Support of Spring

Spring support proxy based runtime weaving type. Spring provides the following 3 types for AOP implementation. Among these,we'll examine the AOP type that uses @AspectJannotation and XML schema in details.

- AOP Implementation using @AspectJ Annotation
- AOP implementation using XML Schema
- AOP implementation using spring API

Execution Environment AOP Guideline

* Execution Environment AOP Guideline

Reference

- Spring 2.5 Reference Documentation
- Spring bean() Pointcut