AOP in Spring

An Introduction to
Aspect-Oriented
Programming with the
Spring Framework

Introduction

- AOP stands for Aspect Oriented Programming
- AOP is one of the key components of the Spring framework

Topics in this session

- Looking at Programming Paradigms
- AOP Concepts
- Intro to Spring AOP

Looking at Programming Paradigms

- **Procedural Programming**
  a. you divide the application according to functionalities that you need to implement
  b. allows you to partition a program into entities called **procedures**.

Looking at Programming Paradigms

- OOP – goes a step further by obliging you to group related data items & their associated processing tasks into coherent entities – “classes”

Looking at Programming Paradigms

- In OOP, Applications are more reusable
  - Introduced the concept of "object" – initiated a new way to structure applications
- 1st version of OOP language appeared in 1967 with Simula 67
- Developers could visualize systems as groups of entities and the interaction between those entities, which allowed them to tackle larger, more complicated systems and develop them in less time than ever before.

Looking at Programming Paradigms

- Where OOP fails us:
  - Technical and functional concerns that are said to be cross-cutting are not easily dealt with using OOP

Looking at Programming Paradigms

- AOP – establishes a certain balance by allowing you to superimpose a new layer onto the data-driven composition of OOP
    - This layer corresponds to the “cross-cutting functionalities” that are difficult to integrate through OOP paradigm

Looking at Programming Paradigms

- What are cross-cutting concerns?
  - Generic functionality that is needed in many places in your application
  - Examples:
    - Logging and Tracing
    - Transaction Management
    - Security
    - Caching
    - Error Handling
    - Performance Monitoring
    - Custom Business Rules

Looking at Programming Paradigms

- An Example Requirement:
  - Perform a role-based security check before **every** application method.

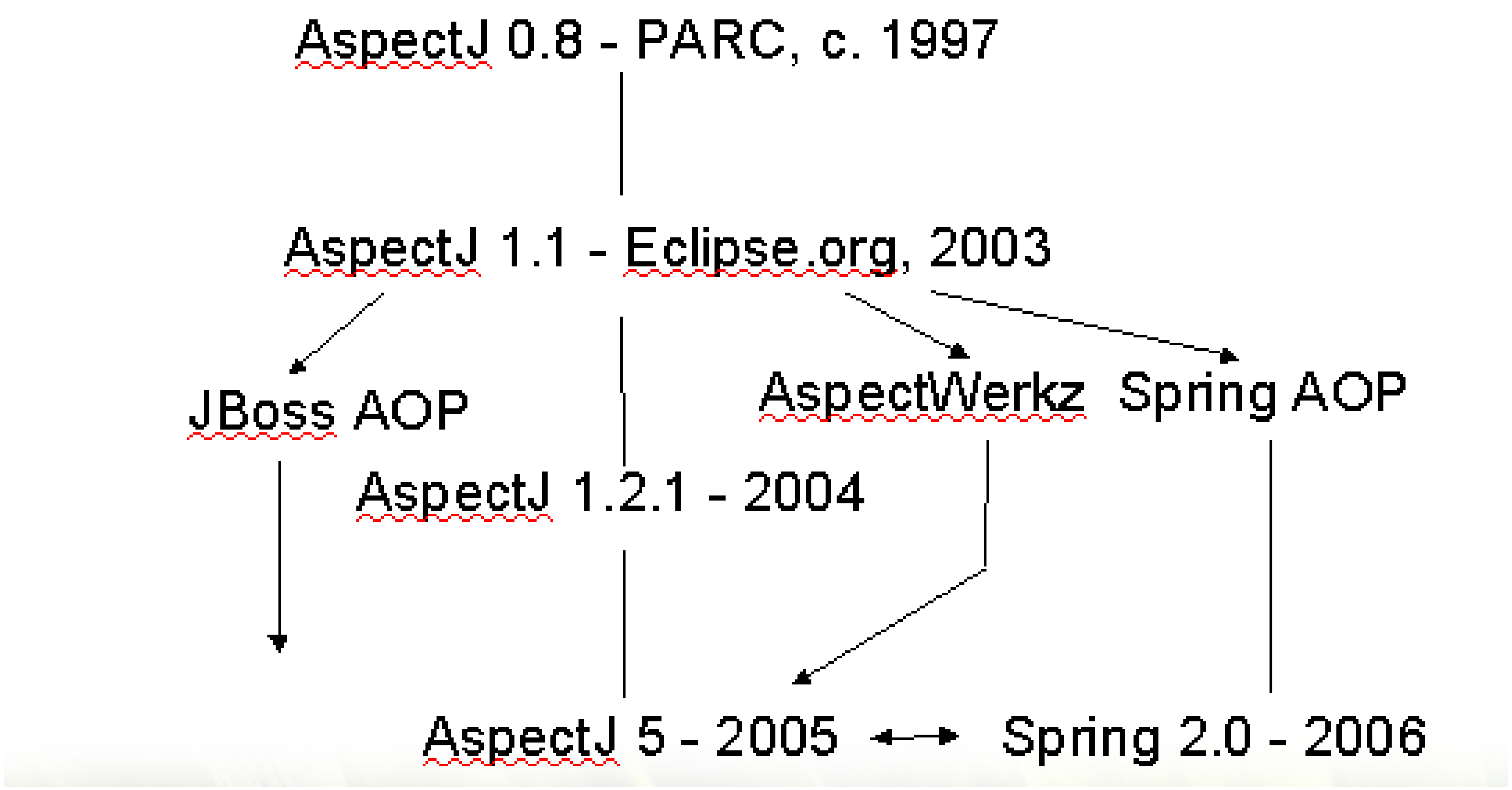
AOP

- Paradigm for separating cross-cutting concerns in well-defined software entities called “aspects”.
- Does not replace OOP.
- Complements OOP by modularizing crosscutting concerns.
- Still a matter of writing classes w/ fields & methods

Leading AOP Technologies

- AspectJ
  - Original AOP technology (first version in 1995)
  - Offers a full-blown Aspect Oriented Programming language
    - Uses byte code modification for aspect weaving
- Spring AOP
  - Java-based AOP framework
    - Uses dynamic proxies for aspect weaving
  - Focuses on using AOP to solve enterprise problems
  - **The focus of this session**

AOP family tree (industry)



AOP Concepts

- Join Point
  - A point in the execution of a program such as a method call or field assignment
- Pointcut
  - A collection of joinpoints that you use to define when advice should be executed
- Advice (the code you want to run)
  - Code to be executed at a Join Point that has been selected by a Pointcut
- Aspect
  - A module that encapsulates pointcuts and advice

Example

– A typical joinpoint is a method invocation.

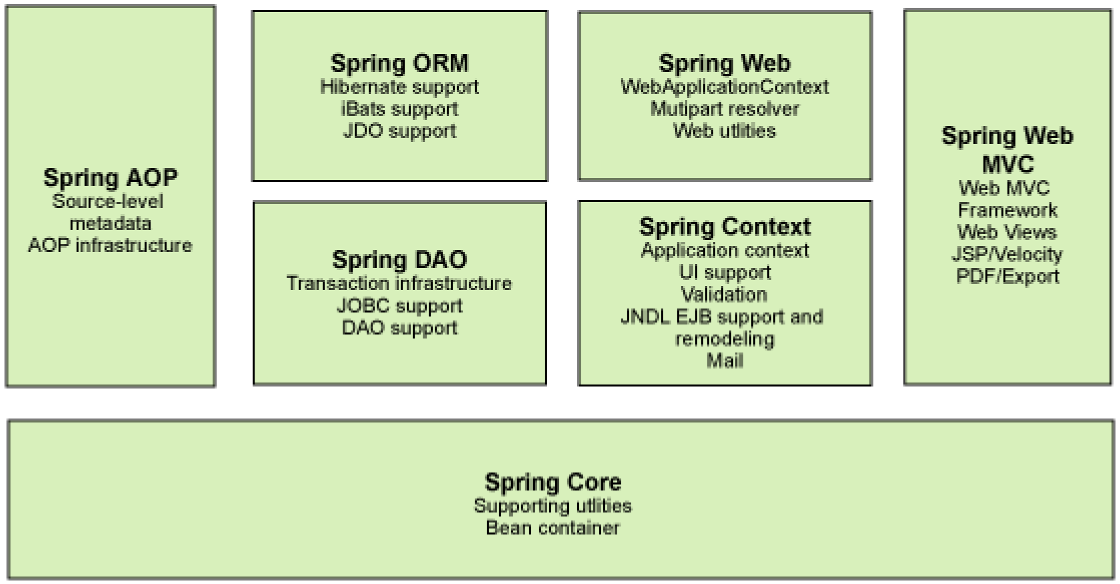
– A typical pointcut is a collection of all method invocations in a particular class

AIM:What Does Spring Provide, in Terms of AOP?

"The aim is not to provide the most complete AOP implementation (although Spring AOP is quite capable); it is rather to provide a close integration between AOP implementation and Spring IoC to help solve common problems in enterprise applications."
The Spring Framework Reference Documentation

AOP

- **The seven modules of the Spring framework**



Intro to Spring AOP

- Spring's support for AOP comes in four flavors:
  - ■ Classic Spring proxy-based AOP (available in all versions of Spring)
  - ■ @AspectJ annotation-driven aspects (only available in Spring 2.0)
  - ■ Pure-POJO aspects (only available in Spring 2.0)
  - ■ Injected AspectJ aspects (available in all versions of Spring)

Intro to Spring AOP

- ***Spring advice is written in Java***

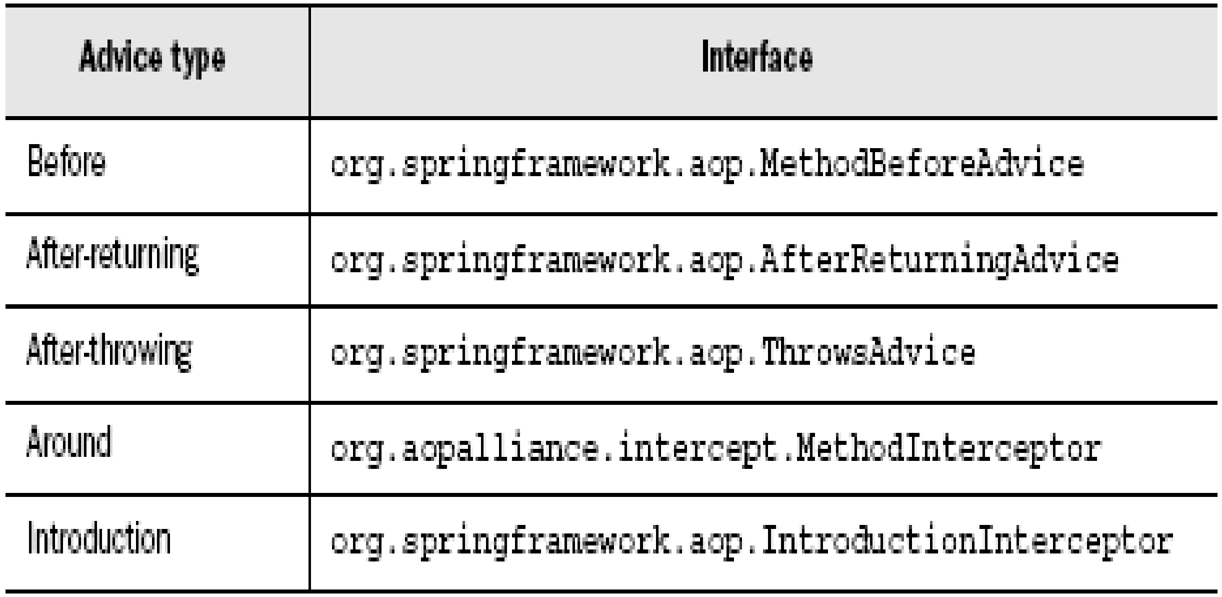
  ■ All of the advice you create within Spring will be written in a standard Java class. That way, you will get the benefit of developing your aspects in the same integrated development environment (IDE) you would use for your normal Java development.

Intro to Spring AOP

Types of Advice

– before advice, which executes before joinpoint
– after advice, which executes after joinpoint
– around advice, which executes around joinpoint

In Spring AOP, there are five types of advice, each defined by an interface:



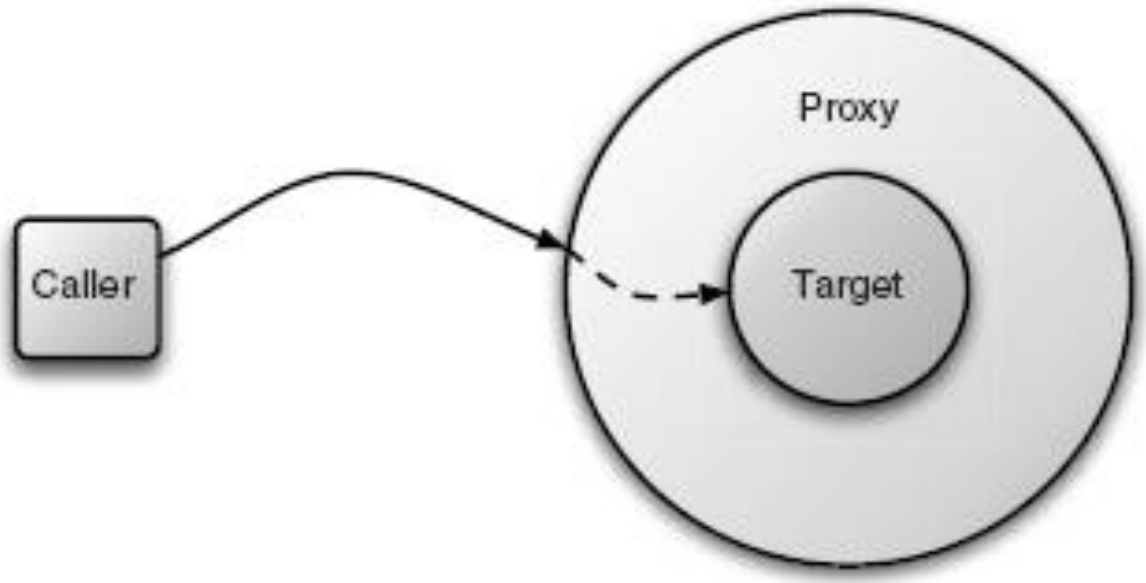
Advice type	Interface
Before	org.springframework.aop.MethodBeforeAdvice
After-returning	org.springframework.aop.AfterReturningAdvice
After-throwing	org.springframework.aop.ThrowsAdvice
Around	org.aopalliance.intercept.MethodInterceptor
Introduction	org.springframework.aop.IntroductionInterceptor

Intro to Spring AOP

- Spring AOP is based on proxies
    - When you want to create an advised instance of a class, you must use the *ProxyFactory* class to create a proxy of an instance of that class, first providing the *ProxyFactory* with all the aspects that you want to be woven into the proxy
    - You typically use *ProxyFactoryBean* class to provide declarative proxy creation

Intro to Spring AOP

In Spring, aspects are woven into Spring-managed beans at runtime by wrapping them with a proxy class. Spring aspects are implemented as proxies that wrap the target object. The proxy handles method calls, performs additional aspect logic, and then invokes the target method. Spring does not create a proxied object until that proxied bean is needed by the application.



Intro to Spring AOP

- **This presentation will show you how to use the following AOP Concepts as they are implemented in the Spring Framework**
- **Advice:** How to declare before, afterReturning and afterThrowing advice as beans.
- **Pointcuts:** How to declare static pointcut logic to tie everything together in the XML Spring Bean Configuration files.
- **Advisors:** The way to associate pointcut definitions with advice beans.

Where to Get More Information

- Foundations of AOP for J2EE Development (Apress)
- Manning Spring in Action 2nd.Edition
- http://springframework.org