Spring framework 2.0

Roman Pichlík
http://sweb.cz/pichlik/

Nejdůležitejší slide

- http://springframework.org/
- http://www.springframework.org/docum

Historie

- 2002 - Rod Johnson
  - kniha Expert One-on-One J2EE Design and Development
  - popularita zdrojových kódů (30,000 řádků kódu)
- 2003 založení open source projektu Spring
  - Jürgen Höller, Yann Caroff
- 2004
  - Verze 1.0
- 2006
  - Verze 2.0

Základní myšlenky

- Zjednodušení komplexnosti J2EE (nejen) vývoje
  - proč dělat jednoduché věci složitě a ty složité ještě složitější?
  - neinvazivnost
  - zaměření na architekturu aplikace nikoliv na použitou technologii
  - jednoduchá testovatelnost
  - modulárnost

Myšlenky řečí technologie

- Inversion Of Control
  - Dependency injection
- Aspektově orientované programování (AOP)
- OOP přístup
  - programování rozhraním
  - Open-Closed princip
    - otevřený pro rozšíření
    - uzavřený pro modifikace
- POJO (Plain Old Java Objects) přístup
  - old != zastaralý ;-)

Myšlenky řečí technologie

- Pomoc při odstranění těsných programových vazeb jednotlivých POJO objektů a vrstev za pomoci návrhového vzoru Inversion of Control.
- Možnost volby implementace (EJB, POJO) business vrstvy pro aplikační architekturu a ne naopak (tedy aby architektura předepisovala implementaci).
- Řešení různých aplikačních domén bez nutnosti použití EJB, například transakční zpracování, podpora pro remoting business vrstvy formou webových služeb či RMI.
- Podpora implementace komponent pro přístup k datům, ať již formou přímého JDBC či ORM (object-relation mapping) technologií a nástrojů jako Hibernate, JPA, JDO
- Odstranění závislosti na roztroušených konfiguracích a pracného dohledávání jejich významu.
- Abstrakce vedoucí ke zjednodušenému používání dalších části J2EE, jako například JMS, JMX, JavaMail, JDBC, JCA nebo JNDI.
- Podpora tvorby a spouštění unit/integračních testů.
- Správa a konfigurační management business komponent.

Co je to vlastně Spring?

- Náhrada J2EE a především EJB?
- Sada modulů pro různé použití?
- Továrna na továrny?
- Webový framework?
- Sada pomocných API pro Hibernate?
- Přežitek v době EJB 3.0?
- Výstřelek zhýralých vývojářů?

Co je to vlastně Spring?

- Náhrada J2EE a především EJB? NE!
- Sada modulů pro různé použití? NE
- Továrna na továrny? NE
- Webový framework? NE
- Sada pomocných API pro Hibernate? NE
- Přežitek v době EJB 3.0? NE!
- Výstřelek zhýralých vývojářů? NE ;-)

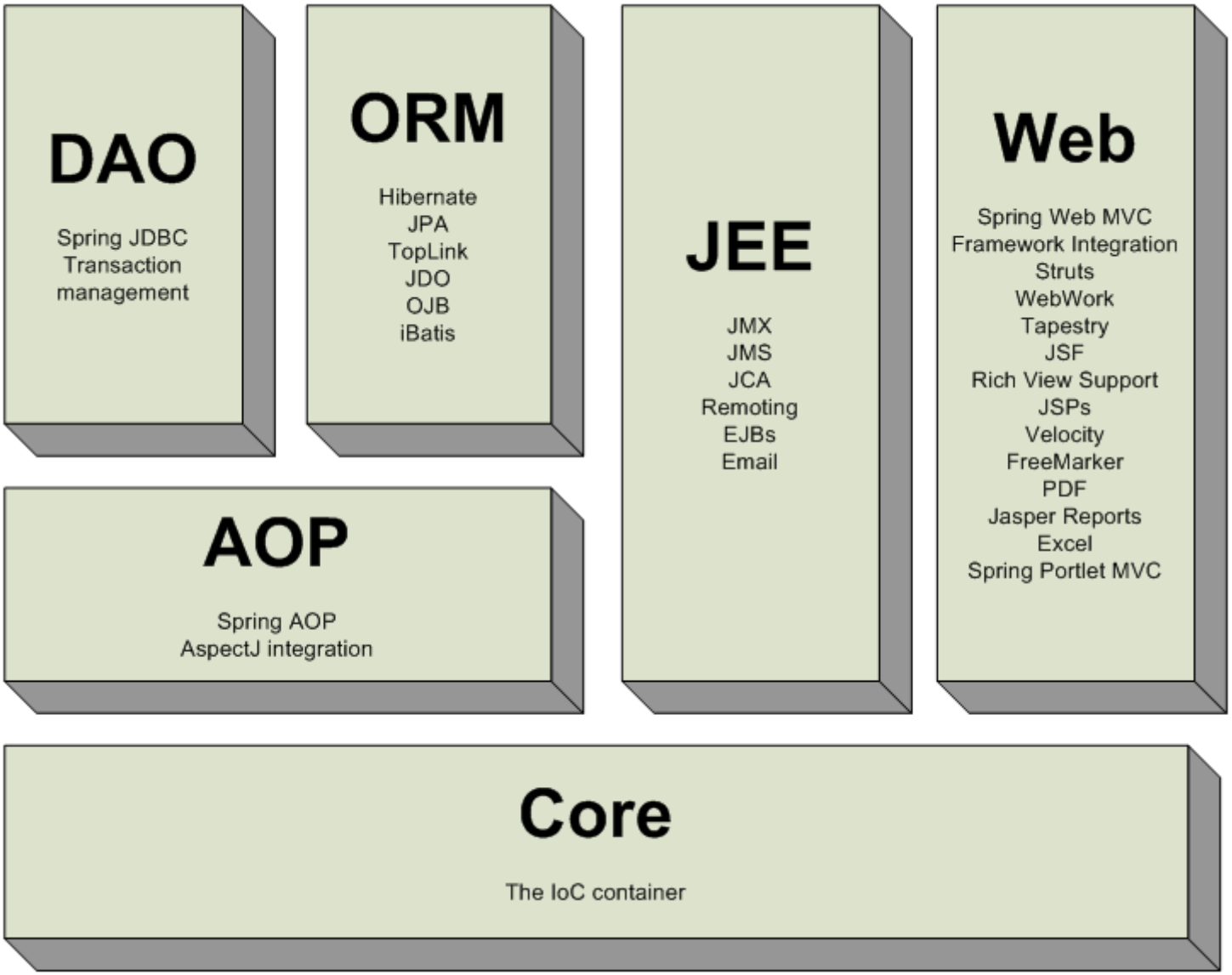
Co je to vlastně Spring?

- komplexní technologie pro vývoj aplikací bez rozdílu zaměření
  - Web, Enterprise, Desktop…
- poskytovatel služeb
  - transakce, AOP, konfigurace, remoting...
- jednotné rozhraní mezi aplikací a deploy prostředím
  - aplikační server, webový kontejner
- zprostředkovatel mezi aplikací a technologiemi
  - Hibernate, JMS, JSF, JPA, JDBC, EJB...

Základní stavební kameny

- Spring rovná se modulárnost
  - použij co potřebuješ
  - malá závislost na knihovnách třetích stran
  - základní modul
    - CORE
  - volitelné
    - JEE
    - WEB
    - AOP
    - ORM
    - DAO

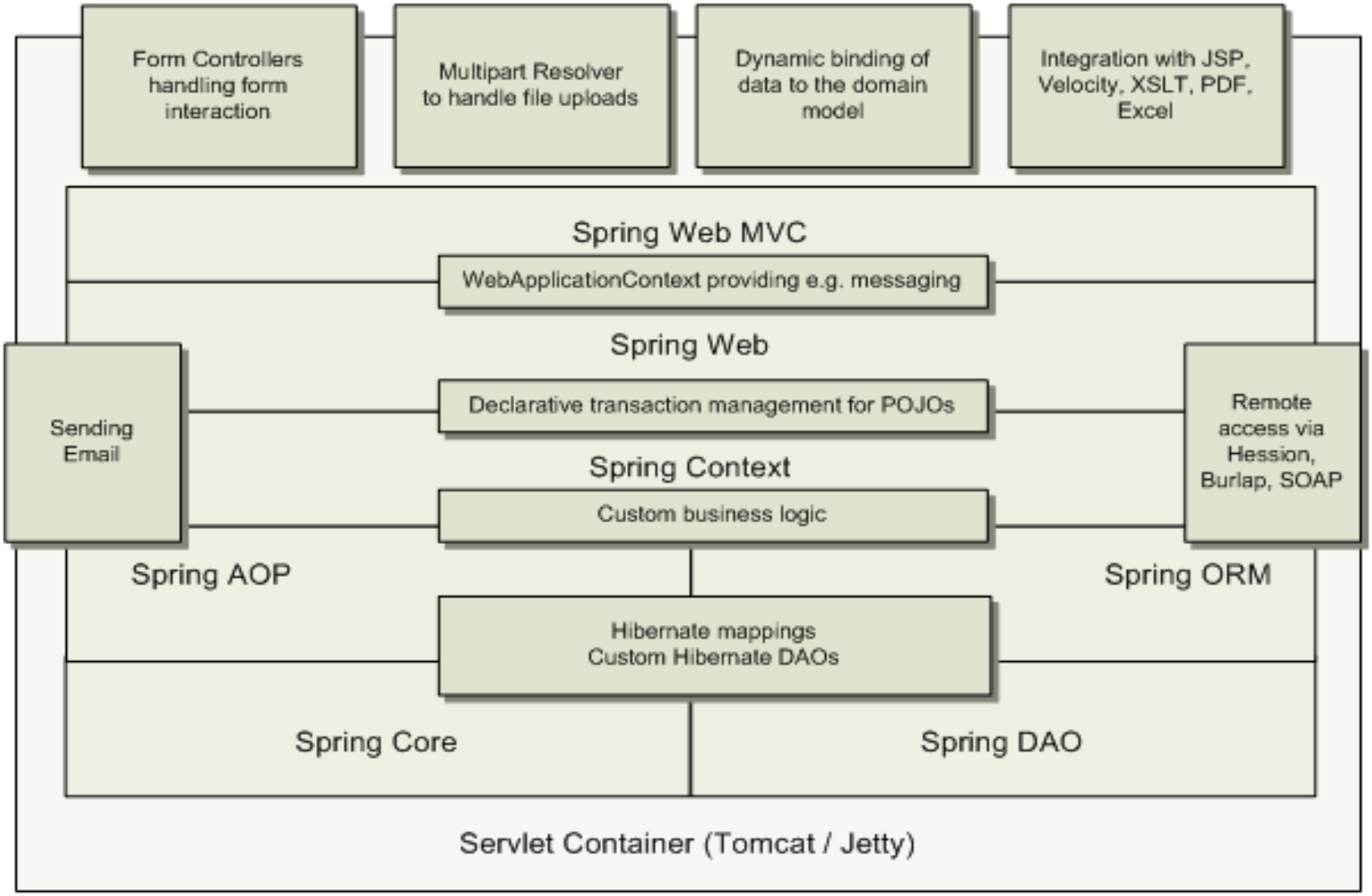
Moduly


DAO
Spring JDBC
Transaction
management
ORM
Hibernate
JPA
TopLink
JDO
OJB
iBatis
JEE
JMX
JMS
JCA
Remoting
EJBs
Email
Web
Spring Web MVC
Framework Integration
Struts
WebWork
Tapestry
JSF
Rich View Support
JSPs
Velocity
FreeMarker
PDF
Jasper Reports
Excel
Spring Portlet MVC
AOP
Spring AOP
AspectJ integration
Core
The IoC container

Moduly - popis

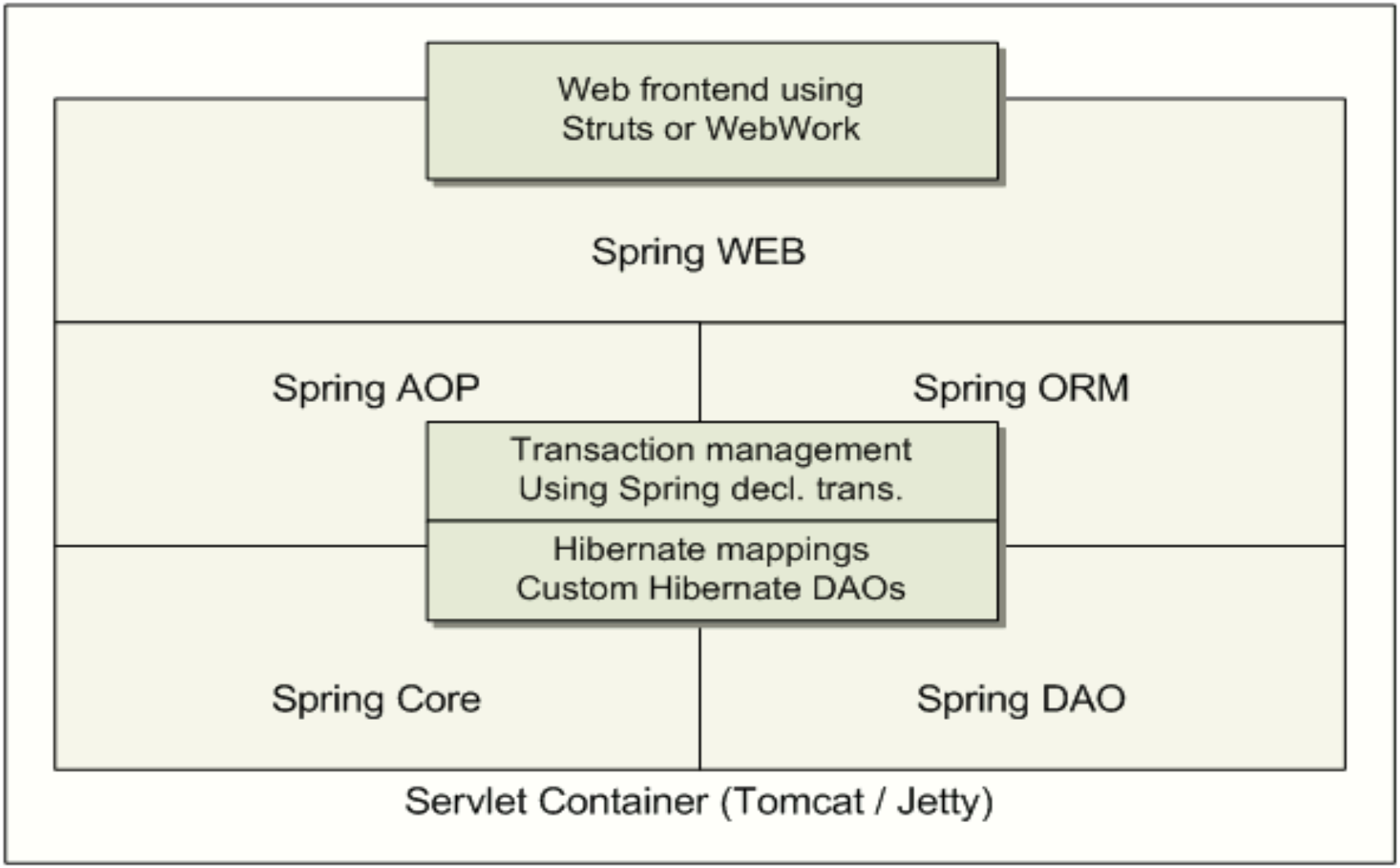
- CORE
  - IoC kontejner (register objektů, lifecycle management)
- AOP
  - AOP funkcionalita
- DAO
  - vrstva pro efektivní práci s JDBC
- JEE
  - integrace s J2EE (EJB, JMS, JCA)
- WEB
  - MVC a portlet implementace
  - Integrace s web frameworky (Struts, JSF, Tapestry...)
- ORM
  - vrstva pro práci s ORM technologiemi

Příklady aplikací postavených na Springu



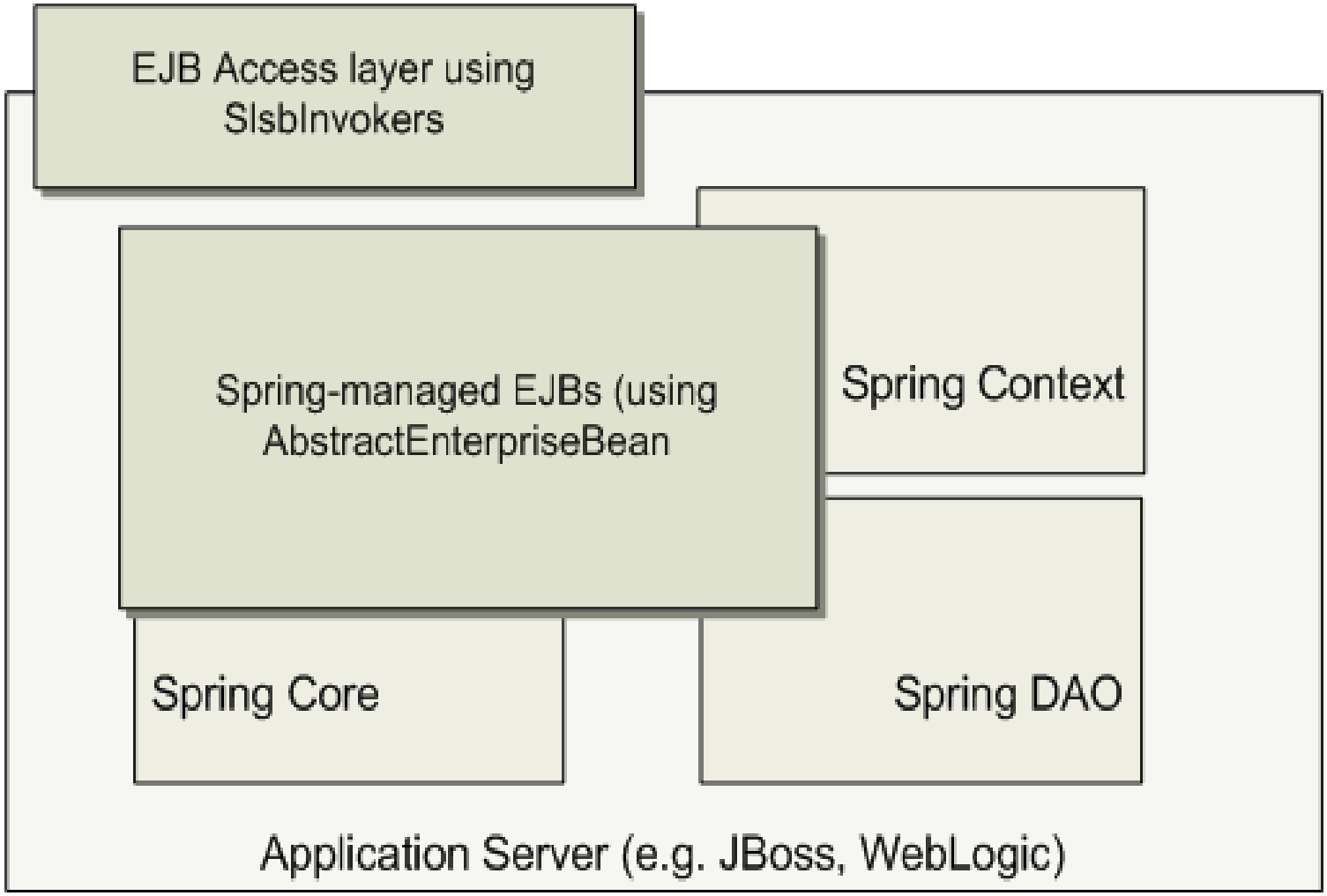
Typical full-fledged Spring web application

Příklady aplikací postavených na Springu



Spring middle-tier using a third-party web framework

Příklady aplikací postavených na Springu

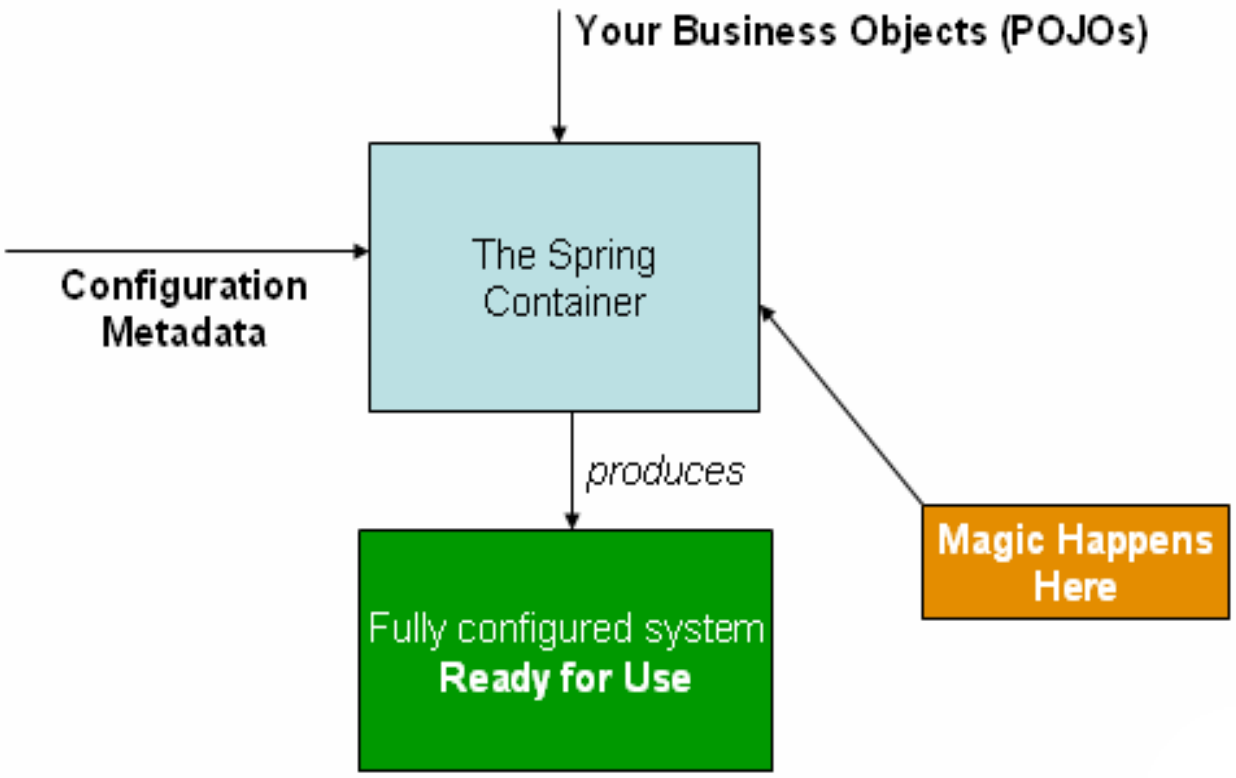


EJBs - Wrapping existing POJOs

IoC kontejner

- Implementace návrhového vzoru IoC
  - odstranění těsných vazeb mezi komponentami
  - odstranění lookup kódu např. volání JNDI
- Řízení životního cyklu managovaných objektů
  - Hollywod princip („Nevolejte mi, ozvu se Vám!“)
  - podobnost s EJB kontejnerem
- Konfigurace skrze metadata
  - XML
  - Anotace
  - Properties

IoC kontejner



The Spring IoC container

IoC kontejner

- Kontejner se startuje programově
  - připravená startovací API pro různá prostředí
    - WEB
      - Servlet, Servlet Context Listener
    - Junit
      - Speciální TestCase
  - z ruky

```
XmlBeanFactory bf = new XmlBeanFactory("applicationContext.xml");
MyBusinessObject mbo = (MyBusinessObject)  bf.getBean("exampleBusinessObject");
```

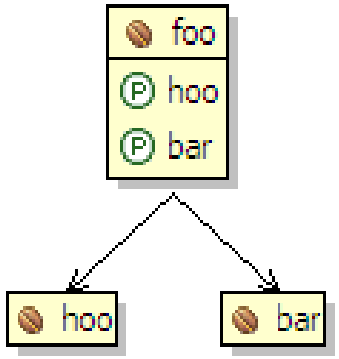
IoC kontejner - objekty

- Managované objekty
  - Spring je nazývá beany
    - Neplést s EJB beanami
  - nepotřebují žádné speciální rozhraní
  - nepotřebují žádný marker
  - prostě POJO
- Dependency injection realizace
  - settery
  - konstruktor

IoC kontejner - metadata

- Popisují vztahy mezi jednotlivými objekty
- Slouží kontejneru k instancování a sestavení objektů podle jejichvazeb
- Mohou mít různou podobu
  - XML
  - Anotace
  - Properties
  - Jakarta Commons attributes

IoC kontejner - overview



POJO

```
public class Foo {
    private Hoo hoo;
    private Bar bar;

    public void setBar(Bar bar)
    public void setHoo(Hoo hoo)

}
```

Metadata

```
<?xml version="1.0" encoding="UTF-8"?>
<beans>
 <bean name="foo" class="cz.sweb.pichlik.samples.ioc.standard.Foo">
    <property name="hoo" ref="hoo"/>
    <property name="bar" ref="bar"/>
 </bean>

  <bean name="hoo" class="cz.sweb.pichlik.samples.ioc.standard.Hoo" />
  <bean name="bar" class="cz.sweb.pichlik.samples.ioc.standard.Bar" />
</beans>
```

Magie

```
String path = FooTest.class.getResource("applicationContext.xml").getFile();
ApplicationContext context =  new FileSystemXmlApplicationContext(path);
```

Použití

```
Foo foo = (Foo) context.getBean("foo");
```

Spring IoC vs EJB 3.0

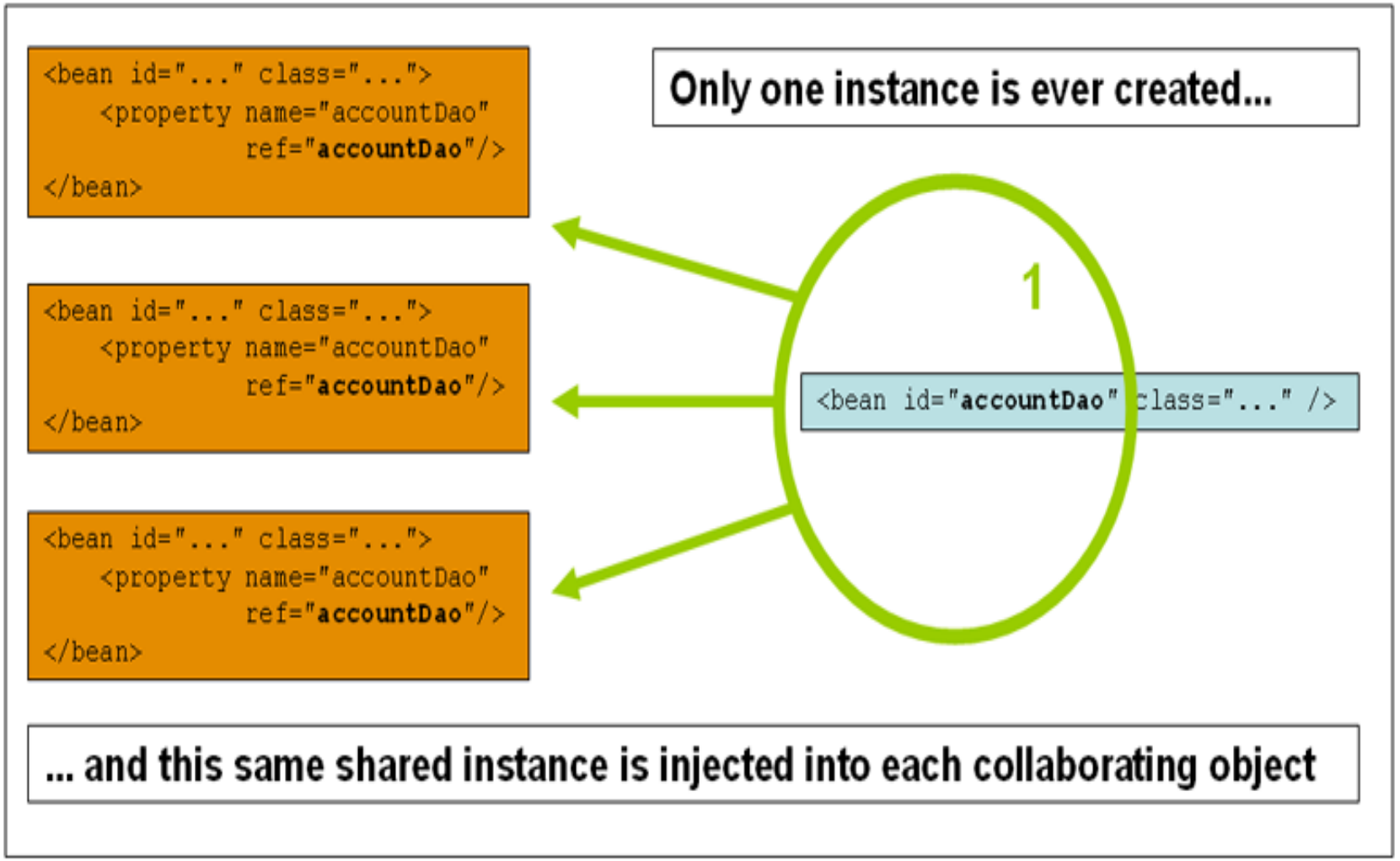
	Spring IoC	**EJB 3.0 IoC**
definice závislosti	XML descriptor, anotace, properties, konfigurační java kód	anotace, XML descriptor
aplikovatelnost	bez omezení	Servlet, listener classes, web services end-point, JAX-RPC handlers, DataSource, JMS, Mail, EJB, Environment entries, EntityManager, UserTransaction, EJB Beans, interceptors, web services end-point, DataSource, JMS, Mail, Environment entries, EntityManager, EJB Context, UserTransaction, TimerService
nepřímé závislosti	ano	ne
způsob nastavení závislosti	setter, anotace proměnné, konstruktor	setter, anotace proměnné
extension pointy	AOP, BeanPostProcessor, Factory, lifecycle rozhrani, typová konverze	interceptory, lifecycle rozhrani
autowiring	typem, jménem	ne

Chudokrevné a plnokrevné IoC, EJB vs. Spring

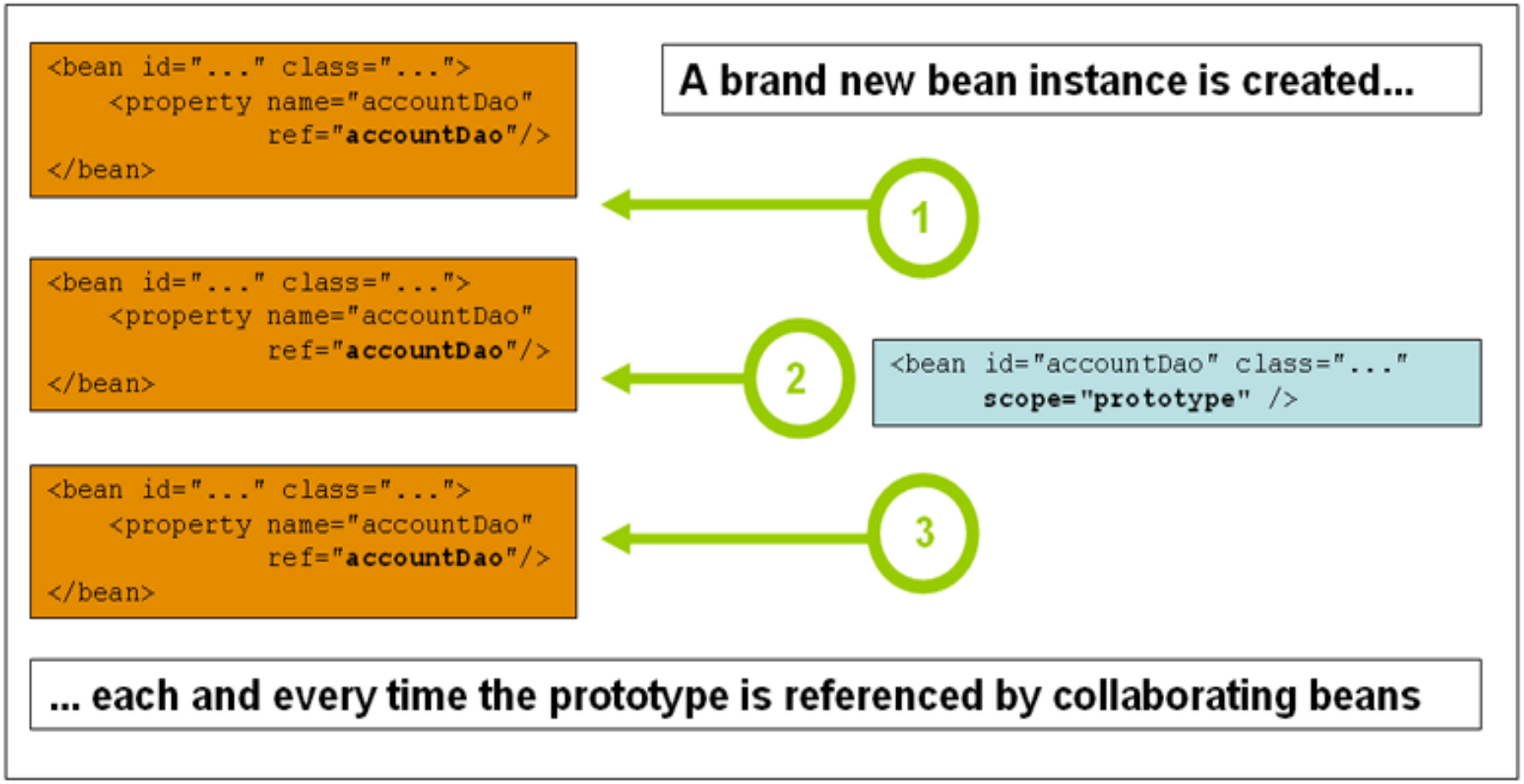
Bean scopes

- Každý managovaný objekt (beana) má takazvaný scope
- Scope definuje životnost objektu z hlediska konverze kontejneru s klientem
  - singleton
  - prototype
  - web scopes (request, session, global session)
  - custom

Singleton scope



Prototype scope



Web + custom scope

- request, session, thread scope

```
<bean name="shoppingCart" class="com.acme.ShoppingCart" scope="session">
    <aop:scoped-proxy/>
</bean>

<bean name="orderCollector" class="com.acme.OrderCollector">
    <property name="shoppingCart" ref="shoppingCart"/>
</bean>
```

singleton

```
public class OrderCollector {
    private ShoppingCart shoppingCart;

    public void processOrder() {
        List<Item> items = shoppingCart.getItems();
        ...
    }
}
```

proxy objekt

Všechny volání jdou na proxy, která získá objekt ze scope a deleguje volání

Metadata - anotace

- Projekt Spring Annotation

```
@Bean(name="foo")
public class Foo {

   @Property(bean="hoo")
   private Hoo hoo;

   @Property(bean="bar")
   private Bar bar;

}
```

Metadata - anotace

- Subprojekt Spring configuration

```
@Configuration
public class
ProgrammaticConfiguration {

    @Bean
    public Hoo hoo() {
        return new Hoo();
    }

    @Bean
    public Bar bar() {
        return new Bar();
    }

    @Bean(scope = Scope.PROTOTYPE)
    public Foo foo() {
        Foo foo = new Foo();
        foo.setBar(bar());
        foo.setHoo(hoo());
        return foo;
    }
}
```

Metadata - XML

- XML schema
  - lepší podpora v nástrojích
    - code completion
    - Validace
  - vhodnější pro rozšiřování
    - nové XML tagy
- zpětná kompatibilita
  - DTD je stále podporované

Metadata - XML

- Možnost rozšíření o nové tagy
  - Schema
  - Java code
  - Registrace
- Standardní beany mají vlastní tagy

Spring 1.x

```
<bean id="simple"
      class="org.springframework.ejb.access.LocalStatelessSessionProxyFactoryBean">
  <property name="jndiName" value="ejb/RentalServiceBean"/>
  <property name="businessInterface" value="com.foo.service.RentalService"/>
</bean>
```

Spring 2.x

```
<jee:local-slsb id="simpleSlsb" jndi-name="ejb/RentalServiceBean"
    business-interface="com.foo.service.RentalService"/>
```

Spring AOP

„V každé aplikaci máme kousky kódu, které se nám prolínají všemi vrstvami naší aplikace, ale do žádné nepatří konkrétně. Může se jednat o kus kódu pro logování-debug, bezpečnost, auditování, synchronizaci atd. Těmto kouskům kódu můžeme říkat aspekty. To co nám AOP nabízí je že můžeme tyto aspekty prolínat stávajícím kódem aniž bychom tento kód museli modifikovat.“

- vlastní řešení
  - řeší 80% AOP požadavků
- AOP se používá na klíčové vlastnosti jako transakce

Spring AOP

- Integrace s AspectJ
  - pointcat EL
  - AspectJ definice
    - možné sdílení aspektů
- Kombinace Spring AOP a AspectJ nezávisle na sobě

Další novinky

- Integrace JPA (součást Martinovi prezentace)
- Transaction demarcation via anotace
- Portlet MVC
- Podpora dynamických jazyků
  - JRuby, Groovy, BeanShell
- Java 5
  - anotace
  - generické typy
- a další...

Největší mýty o Springu

- Spring se nehodí pro Enterprise aplikace
- Spring nepoužívají velké firmy
- Spring nejde škálovat

Největší mýty o Springu

- Spring se nehodí pro Enterprise aplikace
  - deset velkých bankovních institucí používá Spring
  - pojišťovny (Evropa, US)
  - European Patent Office
  - French Online Taxation System
  - US, Canadian and Australian Government Agencies

Největší mýty o Springu

- Spring nepoužívají velké firmy
  - Oracle
  - BEA
  - eBay
  - CERN
- 1,1M stažení

Největší mýty o Springu

- Spring nejde škálovat
  - kompletní řešení
    - Open Terracotta http://www.terracotta.org/
    - Tangosol Coherence Data Grid http://www.tangosol.com/
- Únorové CZJUG setkání
  - Jonas Boner (Terracota)
    - Using AOP to Cluster the JVM
    - http://java.cz/detail.do?articleId=2624

Spring vs EJB 3.0

- Proč používat Spring v době EJB 3.0.
  - IoC kdekoliv
  - transparentní transakce
  - snadná testovatelnost
  - stále se vyvíjí
  - nezávislost na AS
  - open source se vším všudy
  - nevyžaduje Javu 5
  - built in AOP
    - proxy based weaving

Zajímavé subprojekty

- Spring Web Flow
- Spring Security (Acegi Security)
- Spring Rich Client
- Spring IDE for Eclipse
- Spring BeanDoc
- Spring OSGi
- Spring JavaConfig
- Spring .NET

Q&A

- čas na Vaše otázky
  - šetřete mě ;-)

???

The end

- Děkuji za pozornost!