

Spring Framework 3.0

Key Themes and Features

Jürgen Höller
Principal Engineer
SpringSource, a division of VMware

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZÜRICH

-
- Quick Review: Spring 2.5
 - Spring 3.0 Themes and Features
 - Adopting Spring 3.0 in Practice

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Comprehensive support for annotation-based configuration**
 - **@Autowired**
 - with optional **@Qualifier** or custom qualifier
 - **@Transactional**
 - **@Service, @Repository, @Controller**
- **Common Java EE 5 annotations supported too**
 - **@PostConstruct, @PreDestroy**
 - **@PersistenceContext, @PersistenceUnit**
 - **@Resource, @EJB, @WebServiceRef**
 - **@TransactionAttribute**

Annotated Bean Component



@Service

```
public class RewardNetworkService  
    implements RewardNetwork {
```

@Autowired

```
public RewardNetworkService(AccountRepository ar) {  
    ...  
}
```

@Transactional

```
public RewardConfirmation rewardAccountFor(Dining d) {  
    ...  
}  
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

@MVC Controller Style



```
@Controller
```

```
public class MyController {
```

```
    private final MyService myService;
```

```
    @Autowired
```

```
    public MyController(MyService myService) {
```

```
        this.myService = myService;
```

```
    }
```

```
    @RequestMapping("/removeBook")
```

```
    public String removeBook(@RequestParam("book") String bookId) {
```

```
        this.myService.deleteBook(bookId);
```

```
        return "redirect:myBooks";
```

```
    }
```

```
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Powerful annotated component model**
 - stereotypes, factory methods, JSR-330 support
- **Spring Expression Language**
 - Unified EL++
- **Comprehensive REST support**
 - and other Spring @MVC additions
- **Support for Portlet 2.0**
 - action/event/resource request mappings
- **Declarative model validation**
 - integration with JSR-303 Bean Validation
- **Support for Java EE 6**
 - in particular for JSF 2.0 and JPA 2.0

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- More powerful options for custom annotations
 - combining meta-annotations e.g. on stereotype
 - automatically detected (no configuration necessary!)

```
@Service
```

```
@Scope("request")
```

```
@Transactional(rollbackFor=Exception.class)
```

```
@Retention(RetentionPolicy.RUNTIME)
```

```
public @interface MyService {}
```

```
@MyService
```

```
public class RewardsService {
```

```
    ...
```

```
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- Spring 3.0 includes the core functionality of the Spring JavaConfig project
 - configuration classes defining managed beans
 - common handling of annotated factory methods

@Bean @Primary @Lazy

```
public RewardsService rewardsService() {  
    RewardsServiceImpl service = new RewardsServiceImpl();  
    service.setDataSource(...);  
    return service;  
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

Standardized Annotations



@ManagedBean

```
public class RewardNetworkService  
    implements RewardNetwork {
```

@Inject

```
public RewardNetworkService(AccountRepository ar) {  
    ...  
}
```

@TransactionAttribute

```
public RewardConfirmation rewardAccountFor(Dining d) {  
    ...  
}  
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **@javax.inject.Inject** is part of JSR-330
 - "Dependency Injection for Java"
 - also defines @Qualifier semantics
 - and a Provider interface
- **@javax.annotation.ManagedBean** is part of JSR-250 v1.1
 - driven by the Java EE 6 specification
 - can be detected through classpath scanning
- **@javax.ejb.TransactionAttribute** is part of the EJB 3.0/3.1 specification
 - also supported for Spring beans

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

EL in Bean Definitions



```
<bean class="mycompany.RewardsTestDatabase">  
  <property name="databaseName"  
    value="#{systemProperties.databaseName}"/>  
  <property name="keyGenerator"  
    value="#{strategyBean.databaseKeyGenerator}"/>  
</bean>
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

@Repository

```
public class RewardsTestDatabase {
```

```
    @Value("#{systemProperties.databaseName}")
```

```
    public void setDatabaseName(String dbName) { ... }
```

```
    @Value("#{strategyBean.databaseKeyGenerator}")
```

```
    public void setKeyGenerator(KeyGenerator kg) { ... }
```

```
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Example showed access to EL attributes**
 - "systemProperties", "strategyBean"
 - implicit references in expressions
- **Implicit attributes to be exposed by default, depending on runtime context**
 - e.g. "systemProperties", "systemEnvironment"
 - global platform context
 - access to all Spring-defined beans by name
 - similar to managed beans in JSF expressions
 - extensible through Scope SPI
 - e.g. web scopes, or step scope in Spring Batch

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Spring MVC to provide first-class support for REST-style mappings**
 - extraction of URI template parameters
 - content negotiation in view resolver
- **Goal: native REST support within Spring MVC, for UI as well as non-UI usage**
 - in natural MVC style
 - preserving MVC strengths
- **Consists of several independent features**
 - path variable extraction for handler methods
 - content negotiation during view resolution
 - also: client-side RestTemplate

```
http://rewarddining.com/rewards/12345
```

```
@RequestMapping(value = "/rewards/{id}", method = GET)
public Reward reward(@PathVariable("id") long id) {
    return this.rewardsAdminService.findReward(id);
}
```

- More options for handler method parameters
 - in addition to `@RequestParam` and `@PathVariable`
 - `@RequestHeader`: access to request headers
 - `@CookieValue`: HTTP cookie access
 - supported for Servlet MVC and Portlet MVC

```
@RequestMapping("/show")
```

```
public Reward show(@RequestHeader("region") long regionId,  
                  @CookieValue("language") String langId) {
```

```
    ...
```

```
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Portlet 2.0: major new capabilities**
 - explicit action name concept for dispatching
 - resource requests for servlet-style serving
 - events for inter-portlet communication
 - portlet filters analogous to servlet filters
- **Spring's Portlet MVC 3.0 to support explicit mapping annotations**
 - @ActionMapping, @RenderMapping, @ResourceMapping, @EventMapping
 - specializations of Spring's @RequestMapping
 - supporting action names, window states, etc

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

Spring Portlet MVC 3.0



```
@Controller
@RequestMapping("EDIT")
public class MyPortletController {
    ...

    @RequestMapping("delete")
    public void removeBook(@RequestParam("book") String bookId) {
        this.myService.deleteBook(bookId);
    }

    @RequestMapping("BookUpdate")
    public void updateBook(BookUpdateEvent bookUpdate) {
        // extract book entity data from event payload object
        this.myService.updateBook(...);
    }
}
```

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

```
public class Reward {  
    @NotNull  
    @Past  
    private Date transactionDate;  
}
```

```
@RequestMapping("/rewards/new")
```

```
public void newReward(@Valid Reward reward) { ... }
```

- JSR-303 "Bean Validation" as the common ground
- Spring 3.0 fully supports JSR-303 for MVC data binding
- Same metadata can be used for persisting, rendering, etc

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- Spring 3.0 introduces a major overhaul of the scheduling package
 - TaskScheduler interface with Trigger abstraction
 - XML scheduling namespace with cron support
 - `@Async` annotation for asynchronous user methods
 - `@Scheduled` annotation for cron-triggered methods

```
@Scheduled(cron = "0 0 12 * * ?")  
public void performTempFileCleanup() {  
    ...  
}
```

- **Spring 3.0 includes a revised version of the Object/XML Mapping (OXM) module**
 - known from Spring Web Services
 - supported for REST-style payload conversion
 - also useful e.g. for SQL XML access
- **Spring 3.0 features a revised binding and type conversion infrastructure**
 - including the capabilities of Spring Web Flow's binding
 - stateless Java 5+ type converters and formatters
 - superseding standard JDK PropertyEditors
 - annotation-driven number/date formatting

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Java EE 6 API support in Spring 3.0**
 - integration with **JPA 2.0**
 - support for query builder, shared cache setup, etc
 - integration with **JSF 2.0**
 - full compatibility as managed bean facility
 - **JSR-303 Bean Validation** integration
 - through Hibernate Validator 4.0 (the JSR-303 RI)
 - **JSR-330**: common dependency injection annotations
 - natively supported by Spring itself
- **Spring 3.0 can run on a Java EE 6 server**
 - or be combined with individual JPA & JSF providers

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Spring 3.0 should be a drop-in replacement**
 - module jars got renamed but basically still represent the same modules
- **No need to change component code**
 - incremental adoption of Spring 3.0 features
 - incremental migration to Spring 3.0 style
- **However, we learned that many people kept using Spring 2.5 like they used Spring 2.0...**
 - XML bean definitions only
 - consider adopting Spring 2.5/3.0's annotation-based bean configuration when moving to 3.0

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Consider adopting Spring's latest component features right from the start**
 - minimal XML setup through the use of annotations
 - classpath scanning instead of bean registration
 - dynamic configuration through expression language
- **Consider adopting standard annotations**
 - JSR-330's `@Inject` in your service classes
 - JSR-303 constraints in your entity classes
- **Consider stereotypes for your architecture**
 - specific component roles with specific scopes, transaction characteristics etc
 - custom stereotypes as shortcuts and for readability

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH

- **Spring 3.0 embraces REST and EL**
 - full-scale REST support in Spring MVC
 - broad Unified EL++ support in the core
- **Spring 3.0 significantly extends its annotated component model**
 - JSR-330 dependency injection annotations
 - validation, scheduling, formatting
- **Spring 3.0 integrates with Java EE 6 APIs such as JSF 2.0, JPA 2.0, and JSR-303**
 - and supports running on Java EE 6 platforms such as GlassFish v3

JAZOON

THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
1-3 JUNE 2010, ZURICH