Spring Web Update

Rossen Stoyanchev

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

About The Speaker

- Senior Consultant at SpringSource
- Spring Web Team
- Consulting & Training
- “Rich Web Applications With Spring” Course Lead

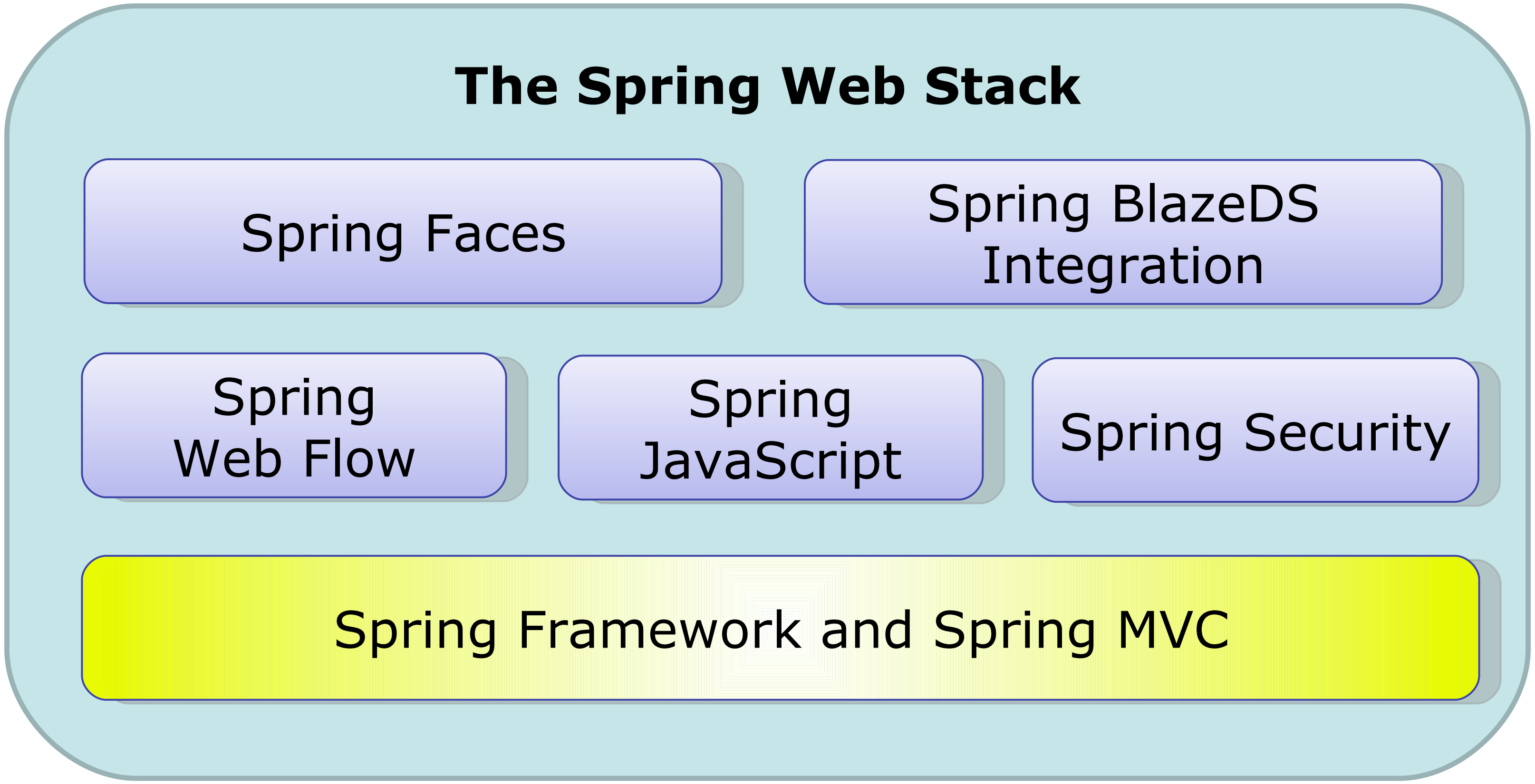
Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Topics

- **Spring Web**
- @MVC
- REST Support
- Spring EL
- Portfolio Changes
- Looking Forward

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

The Spring Web Stack



Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Spring Web Components

- Spring Framework and Spring MVC
  - The foundation for all other components
- Spring JavaScript
  - JavaScript and Ajax integration
- Spring Web Flow
  - Stateful web conversations
- Spring Security
  - Security framework

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

More Spring Web Components

- Spring Faces
  - Integration with JavaServer Faces™
- Spring BlazeDS Integration

  Integration with Adobe Flex™ clients

This focus of this presentation is ongoing changes Spring MVC in the Spring Framework

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Topics

- Spring Web
- **@MVC**
- REST Support
- Spring EL
- Portfolio Changes
- Looking Forward

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Review Of @MVC

- From Controller to @Controller
  - No base class
  - Multiple request-handling methods per class
  - Flexible method signatures
  - No XML for individual controllers
- Not just an alternative to XML
  - A more flexible programming model

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Default Request Mapping

```
@Controller
public class OwnerController {

  @RequestMapping("/owner/show")
  public void show(int id) {
    ...
  }
}
```

Method mapped to:

```
/owner/show
/owner/show.*
```

DefaultAnnotationHandlerMapping

(enabled by default)

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Splitting the Question: Which Controller vs. Which Method?

```
<bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
    <property class="mappings">
        <value>
            /owner/*=ownerController
        </value>
    </property>
</bean>
```

(1) A central place for deciding which controller

(2) Annotations for deciding which method?

```
@Controller
public class OwnerController {

    @RequestMapping
    public void show(int id) {
        ...
    }
}
```

Controller For Owner Resource

```
@Controller
public class OwnerController {

    @ModelAttribute
    public Owner loadOwner(@RequestParam("id") int id) { ... }

    @RequestMapping(method = RequestMethod.GET)
    public void show() { ... }

    @RequestMapping(method = RequestMethod.GET)
    public void form() { ... }

    @RequestMapping(method = RequestMethod.POST)
    public String form(Owner owner, BindingResult result) { ... }

}
```

Controller mapped to:
`/owner/*`

Annotations decide which method?

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Topics

- Spring Web
- @MVC
- **REST Support**
- Spring EL
- Portfolio Changes
- Looking Forward

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

REST Support In Spring MVC

- Why not “Spring REST” (or Spring WS)?
- Spring MVC well-suited for REST
  - Easy to process HTTP requests
  - Easy to render diverse content types
  - Familiar programming model
- @MVC brought further flexibility
  - Map by request method, request parameter

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

URI Templates (Server-Side)

```
@Controller
public class HotelController {

    GET /hotel/1/date/2009-03-26

    @RequestMapping(value = "/hotel/{hotel}/dates/{date}")
    public String show(@ParthVariable("hotel") long hotelId,
                       @ParthVariable Date date, Model model) {
        ...
    }

}
```

Extract path variables

URI Templates (Client-Side)

- <spring:url> tag
  - Supports template placeholders
  - Backwards-compatible with c:url

```
<spring:url value=”/hotel/{hotel}/dates/{date}” xmlEscape=”true”>
    <spring:param name=”hotel” value=”1”/>
    <spring:param name=”date” value=”2009-03-26”/>
</spring:url>
```

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Content Negotiation

- In REST the client decides acceptable representations of a resource
- Content type indicated in request header or file extension
- Server responds accordingly

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Different Representations

- JSON

```
GET http://host/context/app/hotels/1 accepts application/json
GET http://host/context/app/hotels/1.json
```

- XML

```
GET http://host/context/app/hotels/1 accepts application/xml
GET http://host/context/app/hotels/1.xml
```

- ATOM

```
GET http://host/context/app/hotels/1 accepts application/atom+xml
GET http://host/context/app/hotels/1.atom
```

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

New Spring MVC Views

- AbstractAtomFeedView
- AbstractRssFeedView
- MarshallingView
  - `org.springframework.oxm`
- JacksonJsonView

New Spring module (originally in Spring WS)

Part of Spring JS

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Selecting A View

- Controllers are agnostic to the view technology (typically)

```
@RequestMapping(value = ”/hotel/{hotel}”)
public String show(@ParthVariable(”hotel”) long id, Model model) {
    ...
    return ”hotel/show”;        <-------- Logical view name
}
```

Alternating View Types

- Requires a ViewResolver chain
  - BeanNameViewResolver (order=0)
  - InternalResourceViewResolver (order=1)
- One or more custom view beans
  - HotelsAtomView (id="hotel/show.atom")
- A HandlerInterceptor to detect the content type and modify the view name
  - "hotels/show" => "hotel/show.atom"

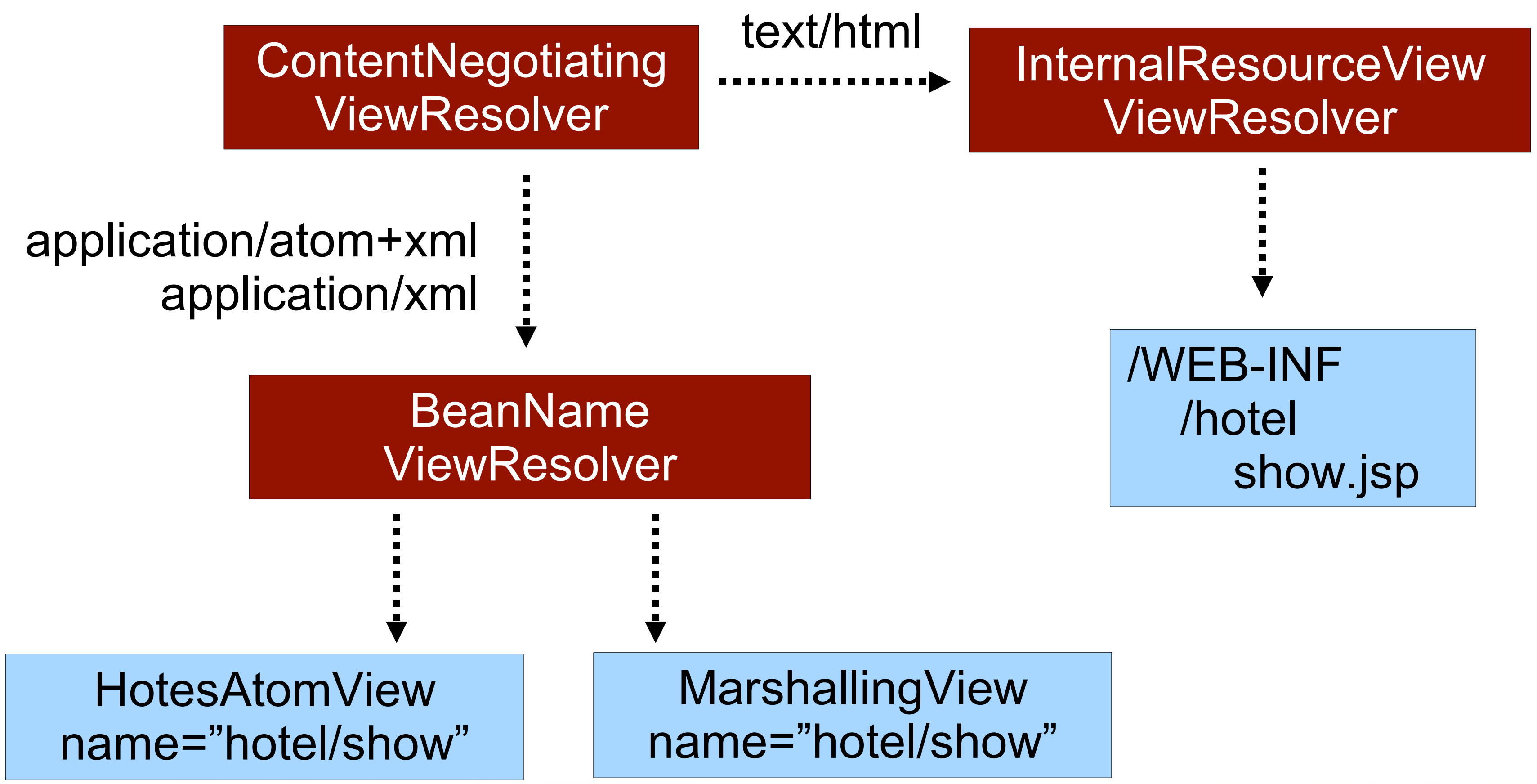
Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

The Content Negotiating ViewResolver

- Does not resolve views itself
- At startup
  - Picks up other ViewResolvers from the web application context
- At runtime
  - Detects client requested content type
  - Queries each ViewResolver
  - Searches for a matching view by content type

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Content Negotiating View Resolver Example


ContentNegotiating ViewResolver
text/html
InternalResourceView ViewResolver
application/atom+xml
application/xml
BeanName ViewResolver
/WEB-INF
/hotel
show.jsp
HotesAtomView name=”hotel/show”
MarshallingView name=”hotel/show”

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

REST Uniform Interface

- Many resources (URLs)
- Limited set of methods (GET, PUT, POST, DELETE)
- Increases the value of an application
- HTML supports only GET and POST

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Hidden HTTP Method Filter

- Spring form tag allows all request methods

```
<form:form method="delete">
  <input type="submit" value="Delete"/>
</form:form>
```

Submits POST, actual DELETE is in a hidden parameter.

- Request can be mapped by "real" method

```
@RequestMapping(value = "/hotel/{hotel}", method = DELETE)
public String delete(@ParthVariable("hotel") long id) { ... }
```

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

REST Template

- Methods for REST full communication

– DELETE	delete
– GET	getForObject
– HEAD	headForHeaders
– OPTIONS	optionsForAllow
– POST	postForLocation
– PUT	put
– ANY	execute

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

REST Template Details

- Methods work with URI templates
- Conversion of input & output objects via HttpMessageConverter

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

ETag Support

- In REST GET is Cacheable
- Server returns ETag header value
  - `ETag: "c5de2d"`
- Value is sent on subsequent retrieval
  - `If-None-Match: "c5de2d"`
- Server returns 304 (Not Modified)

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

ShallowETagHeaderFilter

- Computes ETag header value based on MD5 of rendered view
- Saves bandwidth only

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Other @MVC Refinements

- @RequestHeader
  - Access to request headers
- @CookieValue
  - HTTP cookie access

```
@RequestMapping
public String delete(@RequestHeader(”region”) long regionId,
            @CookieValue(”language”) String languageId) {
   …
}
```

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Demo

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Topics

- Spring Web
- @MVC
- REST Support
- **Spring EL**
- Portfolio Changes
- Looking Forward

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Spring EL

- EL implementation included in Spring 3.0
  - package org.springframework.expression
  - next-generation expression engine inspired by Spring Web Flow 2.0's expression support
- Compatible with Unified EL syntax and more powerful
  - navigating bean properties, collections, maps, custom
  - method invocations
  - construction of value objects

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

EL In Bean Definitions

```
<bean class="mycompany.RewardsTestDatabase">

   <property name="databaseName"
         value=""#{systemProperties.databaseName}"/>

   <property name="keyGenerator"
         value=""#{strategyBean.databaseKeyGenerator}"/>

</bean>
```

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

EL In Component Annotations

```
@Repository
public class RewardsTestDatabase {

    @Value(“#{systemProperties.databaseName}”)
    public void setDatabaseName(String dbName) { … }

    @Value(“#{strategyBean.databaseKeyGenerator}”)
    public void setKeyGenerator(KeyGenerator kg) { … }
}
```

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

EL Context Attributes

- Example showed access to EL variables
  - "systemProperties", "strategyBean"

- Implicit variables 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. for step scope in Spring Batch 2.0

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Web Context Attributes

- Implicit web context variables to be exposed by default as well
  - "contextProperties": web.xml init-params
  - "contextAttributes": ServletContext attributes
  - "request": current Servlet/PortletRequest
  - "session": current Http/PortletSession

- Exposure of all implicit JSF variables when running within a JSF request context
  - "param", "initParam", "facesContext", etc
  - full compatibility with JSF managed bean facility

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Topics

- Spring Web
- @MVC
- REST Support
- Spring EL
- **Portfolio Changes**
- Looking Forward

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

New Project Layout

- Framework modules revised
  - now managed in Maven style
  - one source tree per module jar
    - spring-beans.jar, spring-aop.jar, etc
  - no spring.jar anymore!
- Built with new Spring build system as known from Spring Web Flow 2.0
  - consistent deployment procedure
  - consistent dependency management
  - consistent generation of OSGi manifests

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Portfolio Rearrangements

- Spring 3.0 will include a revised version of the Object/XML Mapping (OXM) module
  - known from Spring Web Services
  - JAXB2, JiBX, Castor, XMLBean, XStream

- Spring 3.0 will include the core functionality of Spring JavaConfig
  - configuration classes defining managed beans

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Pruning & Deprecation in 3.0

- Some pruning planned
  - Commons Attributes support
  - traditional TopLink API support
    - in favor of JPA (EclipseLink)
  - subclass-style Struts 1.x support
- Some deprecation planned
  - traditional MVC controller class hierarchy
    - superseded by annotated controller style
  - traditional JUnit 3.8 test class hierarchy
    - superseded by test context framework
  - several outdated helper classes

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Spring 2.5 Mission Continued

- Spring 3 continues Spring 2.5's mission
  - fully embracing Java 5 in the core Spring programming and configuration model
  - now with even the core framework requiring Java 5
    - all framework classes using Java 5 language syntax
- Backwards compatibility with Spring 2.5
  - 100% compatibility of programming model
  - 95% compatibility of extension points
  - all previously deprecated API to be removed
    - Make sure you're not using outdated Spring 1.2 / 2.0 API anymore!

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Topics

- Spring Web
- @MVC
- REST Support
- Spring EL
- Portfolio Changes
- **Looking Forward**

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Model Validation

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

In view:

```
  <form:input path="transactionDate">
```

- Common Validation System SPI used by MVC and Web Flow
- **Hibernate Validator** annotations supported
- **JSR 303 (Bean Validation)** to be supported as well
- Same metadata can be used for persisting, rendering, etc

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Conversation Management

- Key problem: **isolating concurrent windows** in same browser
  - shared HTTP session
  - several independent conversations going on
    - keeping independent state
- Generalized solution: conversation scope with shorter lifetime than session
  - scope="conversation"
    - on-demand scoping of conversational Spring beans
    - MyFaces Orchestra style
  - Spring Web Flow 3 provides more sophisticated flow navigation management on top

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Binding & Type Conversion

- Revised binding & type conversion infrastructure
- Includes the capabilities of Web Flow's binding
  - Stateless type converter objects
  - EL integration
- Spring MVC and Web Flow will share this infrastructure

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Resources

SpringSource Blogs:

- Spring Framework 3.0 M2 released
- Building Spring 3
- REST in Spring 3: @MVC
- Adding an Atom view using Spring's REST support
- REST Template (to be posted today)
- http://blog.springsource.com

Check out from trunk:

- Petclinic Sample (Eclipse-enabled with WTP settings)
- Unit tests

Track changes:

- https://fisheye.springframework.org/

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Questions and Comments

Copyright 2008 SpringSource. Copying, publishing or distributing without express written permission is prohibited.