

CS520 Web Programming

Spring – MVC Framework

Chengyu Sun
California State University, Los Angeles

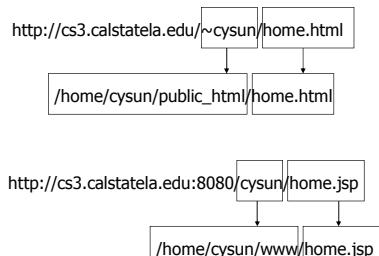
Roadmap

- ◆ Request Processing
- ◆ Transactions and hibernate support (Model)
- ◆ Controllers (Controller)
- ◆ Data input and display (View)

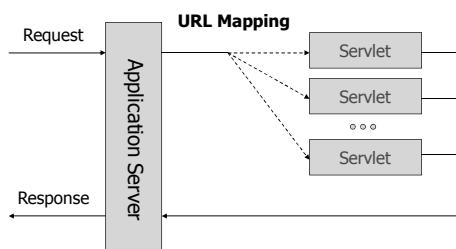
Understand Request Processing

- ◆ What happens when the server received a request like
`http://sun.calstatela.edu/csns/instructor/viewSubmissions.html?assignmentId=50001`

Direct Resource Mapping



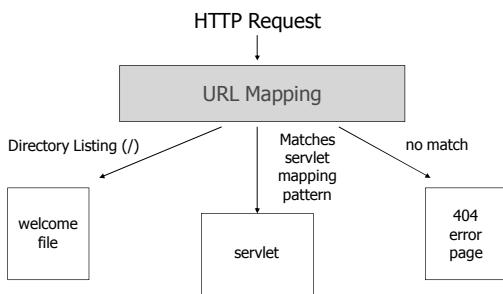
URL Mapping in a Java EE Application



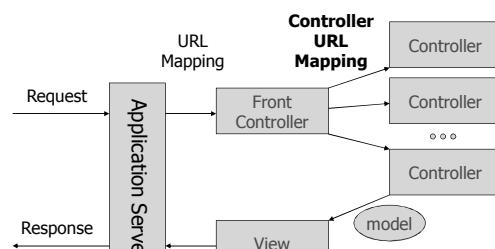
URL Mapping ...

- ◆ Configured in web.xml
 - <servlet> and <servlet-mapping>
 - <welcome-file-list>
 - <error-page>
- ◆ Specified in the Servlet Specification

... URL Mapping



Request Processing in an MVC Framework



Spring Configuration File(s)

- ❖ <name>-servlet.xml
 - <name> must be the same as the <servlet-name> of the DispatcherServlet specified in web.xml
- ❖ Can have additional bean configurations files
 - E.g. applicationContext.xml, csns-data.xml, csns-email.xml, csns-acegi.xml ...

More About Configuration Files (Welcome to Metadata Hell!)

- ❖ Under classpath (/WEB-INF/classes)
 - hibernate.cfg.xml
 - ehcache.xml
 - *.properties
- ❖ Under /WEB-INF
 - web.xml
 - Spring configuration files
 - server-config.wsdd
- ❖ Under /META-INF
 - context.xml

Configuration Files in CSNS ...

- ❖ All configuration files are under /conf
- ❖ init target in build.xml
 - Copy configuration files to the right places
 - Rename spring-* .xml to \${app.name} -* .xml
 - Insert some parameter values into the configuration files

... Configuration Files in CSNS

- ❖ Advantages
 - One folder (i.e. /conf) for all metadata files
 - One file (i.e. build.properties) for all configurable parameters
 - Reusable for other applications
- ❖ Disadvantages
 - Non-standard
 - "Resource out of sync" error in Eclipse

Controller URL Mapping ...

- Maps a URL pattern to a controller that will handle the request

BeanNameUrlHandlerMapping (default)

```
<bean name="/instructor/home.html"  
      class="csns.spring.controller.instructor.ViewSectionsController">
```

Controller URL Mapping ...

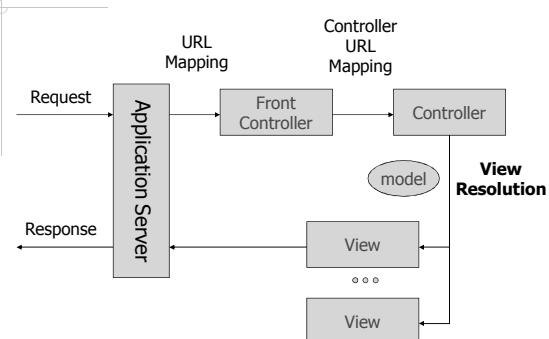
SimpleUrlHandlerMapping

```
<bean id="home" class="csns.spring.controller.HomeController" />  
  
<bean id="urlMapping"  
      class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">  
  <property name="mappings">  
    <props>  
      <prop key="/home.html">home</prop>  
    </props>  
  </property>  
</bean>
```

... Controller URL Mapping

- More than one URL handler
 - <property name="order" value="0"/>
- No mapping found
 - 404 error

Request Processing in an MVC Framework



Model and View Examples

- AddInstructorController
- ViewSubmissionsController
- TakeSurveyController

ModelAndView

```
ModelAndView(  
  String viewName,  
  String modelName,  
  Object modelObject  
)
```

```
ModelAndView( String viewName )  
  .addObject( String modelName, String modelObject )  
  .addObject( String modelName, String modelObject )  
  ...
```

View Resolvers ...

- <http://static.springsource.org/spring/docs/2.5.x/api/org/springframework/web/servlet/ViewResolver.html>

... View Resolvers

- ◆ InternalResourceViewResolver for JSP
- ◆ Support for non-JSP view technologies
 - Velocity, FreeMarker, JasperReports, XSLT
- ◆ Views generated by Java classes
 - BeanNameViewResolver
 - XmlViewResolver
 - ResourceBundleViewResolver
- ◆ Multiple view resolvers
 - <property name="order" value="0"/>

InternalResourceViewResolver Example

```
<bean id="viewResolver"
      class="org.springframework.web.servlet.view.InternalResourceViewResolver">
    <property name="prefix" value="/WEB-INF/jsp/" />
    <property name="suffix" value=".jsp" />
</bean>
```

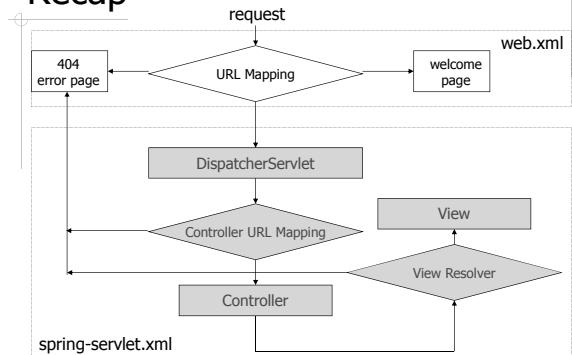


Prefix + ViewName + Suffix = JSP File

Special Views

- ◆ Redirect
 - new ModelAndView("redirect:" + url)
 - E.g. LogoutController
- ◆ null
 - E.g. DownloadController

Recap



Database Access

- ◆ DAO interfaces
 - E.g. csns.model.dao
- ◆ DAO implementations
 - E.g. csns.model.dao.hibernate
 - Hibernate DAO implementation classes inherit from HibernateDaoSupport
- ◆ HibernateTemplate
 - <http://static.springsource.org/spring/docs/2.5.x/api/org/springframework/orm/hibernate3/HibernateTemplate.html>

Programmatic vs. Declarative Transaction Management

Programmatic:

```
void saveUser( User u )  
{  
    transaction.start();  
    ...  
    transaction.end();  
}
```

Declarative:

```
<transaction>  
    <method>  
        saveUser  
    </method>  
</transaction>
```

Hibernate Support in Spring

Without Spring

```
Transaction tx = null;  
try  
{  
    tx = s.beginTransaction();  
    s.saveOrUpdate( e );  
    tx.commit();  
}  
catch( Exception e )  
{  
    if( tx != null ) tx.rollback();  
    e.printStackTrace();  
}
```

With Spring

```
getHibernateTemplate()  
.saveOrUpdate( user );
```

Spring Transaction Managers

- ◆ JDBC
- ◆ Hibernate
 - V3
 - Before V3
- ◆ JTA
- ◆ Object-Relational Bridge (ORB)

Configure Hibernate Transaction Manager

- ◆ dataSource
 - Database connection information
 - DBCP - <http://jakarta.apache.org/commons/dbcp/>
- ◆ sessionFactory
 - Hibernate information
- ◆ transactionManager
- ◆ transactionAttributeSource

See /conf/spring-data.xml

Transaction Attributes

- ◆ Isolation levels
- ◆ Read-only hints
- ◆ Transaction timeout period
- ◆ Method name patterns
- ◆ *Propagation behaviors*

Propagation Behaviors

- ◆ Determines whether the method should be run in a transaction, and if so, whether it should run within an existing transaction, a new transaction, or a nested transaction within an existing transaction.

The Need for Propagation Behaviors

```
Class FoobarDao {  
    void saveFoo( Foo foo ) { save(foo); }  
    void saveBar( Bar bar ) { save(bar); }  
    void saveFoobar( Foobar foobar )  
    {  
        saveFoo( foobar.getFoo() );  
        saveBar( foobar.getBar() );  
    }  
}
```

Propagation Behaviors in Spring

Propagation Behaviors	Run in ...
PROPAGATION_MANDATORY	Existing transaction
PROPAGATION_NESTED	Nested transaction
PROPAGATION_NEVER	No existing transaction
PROPAGATION_NOT_SUPPORTED	Suspended during existing transaction
PROPAGATION_REQUIRED	Existing or new transaction
PROPAGATION_REQUIRES_NEW	New transaction
PROPAGATION_SUPPORTS	May run in existing transaction

Add Transaction Support for DAO Classes

- TransactionProxyFactoryBean
- Bean definition *inheritance*

```
class FooDao {  
    void saveFoo( Foo foo )  
    {  
        save(foo);  
    }  
}  
  
class FooDaoProxy {  
    void saveFoo( Foo foo )  
    {  
        transaction.begin();  
        save(foo);  
        transaction.end();  
    }  
}
```

Examples: Course Management

- List courses
- Add a course
- Edit a course

Controller Interface

- org.springframework.web.servlet.mvc.Controller -
<http://static.springsource.org/spring/docs/2.5.x/api/org/springframework/web/servlet/mvc/Controller.html>

```
ModelAndView handleRequest (   
    HttpServletRequest request,  
    HttpServletResponse response )
```

Select A Controller

ParameterizableViewController	Simply display a view, i.e. the request does not need to be processed
Controller (interface) AbstractController	do not use request parameters or use only simple parameters
BaseCommandController AbstractCommandController	request parameters can be mapped to an object; can use <i>validators</i> to validate request parameters
AbstractFormController SimpleFormController	Handles form input
AbstractWizardFormController	Handles multi-page form input

ParameterizableViewController

- ◆ CSNS Examples
 - login

AbstractController

- ◆ CSNS Examples
 - ViewSubmissionController
 - DeleteAssignmentController
- ◆ Example 1: List all courses

Displaytag

- ◆ <http://displaytag.sourceforge.net/>
- ◆ Sortable columns and result paging
- ◆ Use displaytag, *not* displaytag-el

Displaytag Examples

- ◆ CSNS Examples
 - courses.jsp
 - instructor/viewSubmissions.jsp
 - forum/viewTopic.jsp

<display:table>

- ◆ name
 - Name of the collection to be displayed
 - Like items in <c:forEach>
- ◆ uid
 - Name of the object in each iteration
 - Like var in <c:forEach>
- ◆ requestURI
 - The URL used to request the page
 - E.g. viewSubmission.html, not instructor/viewSubmission.jsp

<display:column>

- ◆ property
- ◆ sortable
- ◆ title
- ◆ sortProperty

Customize Displaytag Tables

◆ Using CSS

- class and style attributes for <display:table> and <display:column>

◆ Using properties

- displaytag.properties for the whole application
- <display:setProperty> for a particular table
- <http://displaytag.sourceforge.net/11/configuration.html>

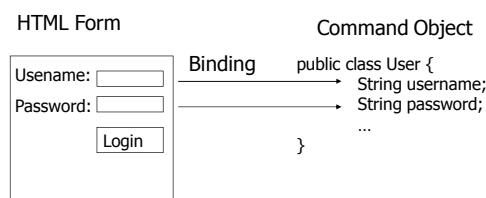
SimpleFormController

◆ CSNS Examples

- EditAssignmentController
- CreateTopicController

◆ Example 2: Add a course

Command Object



◆ Any class can be used as a command class in Spring

- vs. ActionForm in Struts 1

Simple Form Handling

◆ The controller needs to handle two cases:

- Display form
- Process input data

Handle first request:

Create a command object
and expose the object as
a page scope variable
↓
Display the *form view*

Handle input:

Bind the request parameters to
the command object
↓
Call controller.onSubmit()
↓
Display the *success view*

Spring's form Tag Library

◆ Documentation -

<http://static.springsource.org/spring/docs/2.5.x/reference/view.html#view-jsp-formtaglib>

◆ Tag reference -

<http://static.springsource.org/spring/docs/2.5.x/reference/spring-form.tld.html>

◆ Example

- forum/createTopic.jsp

formBackingObject()

◆ Override default command object creation

◆ Example 3: Edit a course

AbstractWizardFormController

◆ CSNS Examples

- ImportRosterController

Summary

◆ Request mapping and view resolution

◆ Controller

- AbstractController and SimpleFormController
- Command object
- /conf/spring-servlet.xml

◆ Model

- Dao and DaoImpl
- /conf/spring-data.xml

◆ View

- Form view and success view
- <spring:form>