CS520 Web Programming
Spring – MVC Framework

Chengyu Sun
California State University, Los Angeles

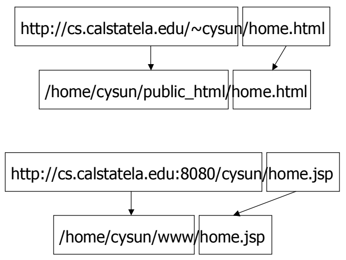
Roadmap

- **Web flow**
- Controllers and validation
- Transactions and hibernate support
- Bits and pieces
  - Tomcat server setup
  - Context, data source, and connection pooling
  - Creating WAR files
  - JSP pre-compilation
  - Displaytag

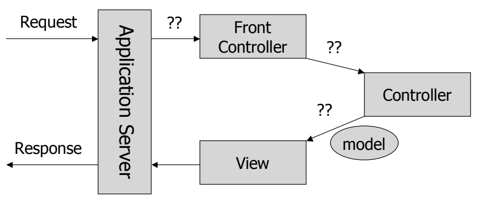
Understand Web Flow

- What happens when the server received a request like `http://cs.calstatela.edu:8080/evelyn/admin/index.html`

Direct Resource Mapping



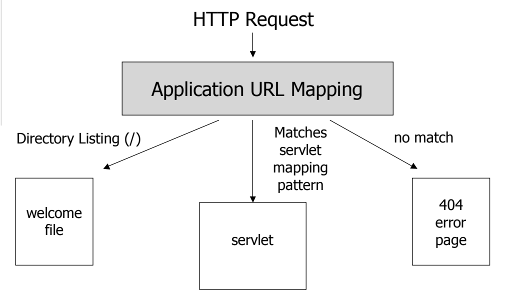
Web Flow of a MVC Framework



Application URL Mapping …

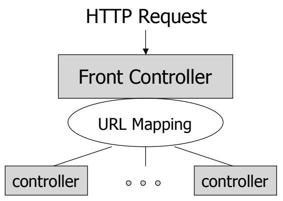
- web.xml
  - <servlet> and <servlet-mapping>
    - `"/servlet/*"`
    - `"*.do"`
  - <welcome-file-list>
  - <error-page>

… Application URL Mapping



Spring Front Controller

- org.springframework.web.servlet.DispatcherServlet



Spring Configuration File(s)

- <name>-servlet.xml
  - <name> must be the same as the <servlet-name> of the DispatcherServlet specified in web.xml
- Optional
  - evelyn-data.xml
  - evelyn-service.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
  - evelyn-servlet.xml, evelyn-data.xml, evelyn-service.xml
  - server-config.wsdd
- Under /META-INF
  - context.xml

URL Handler Mapping …

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

BeanNameUrlHandlerMapping (default)

```
<bean id="index" name="/index.html /public/*index.html"
      class="evelyn.spring.controller.IndexController" />
```

… URL Handler Mapping …

SimpleUrlHandlerMapping

```
<bean id="index" class="evelyn.spring.controller.IndexController" />

<bean id="urlMapping"
      class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
      <property name="mappings">
         <props>
            <prop key="/index.html">index</prop>
            <prop key="/public/*index.html">index</prop>
         </props>
      </property>
   </bean>
```

… URL Handler Mapping

- More than one URL handler
  - <property name="order" value="1"/>
- No mapping found
  - 404 error

ModelAndView

ModelAndView (
String viewName, $\rightarrow$ Resolve to a view
String modelName, $\rightarrow$ Attribute in Request scope
Object modelObject $\rightarrow$ Attribute in Request scope
)

Example:
ModelAndView in `MemeberController`

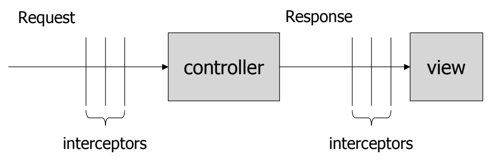
View Resolvers

- URL-based resolvers
  - `InternalResourceViewResolver` for JSP
  - `VelocityViewResolver` for Velocity
- View classes resolvers
  - `BeanNameViewResolver`
  - `XmlViewResolver`
  - `ResourceBundleViewResolver`
- Multiple resolvers
  - <property name="order" value="1"/>

Special Views

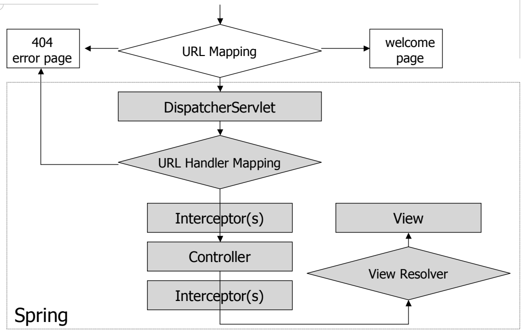
- Redirect
  - new ModelAndView("**redirect:**" + url)
- Forward
  - new ModelAndView("**forward:**" + url)

Interceptors



- Similar to Filters in J2EE specification
- `HandlerInterceptorAdapter`
- Example: AuthorizationInterceptor

Recap



Roadmap

- Web flow
- **Controllers and validation**
- Transactions and hibernate support
- Bits and pieces
  - Tomcat server setup
  - Context, data source, and connection pooling
  - Creating WAR files
  - JSP pre-compilation
  - Displaytag

Controller Class Hierarchy

- org.springframework.web.servlet.mvc
  - Controller

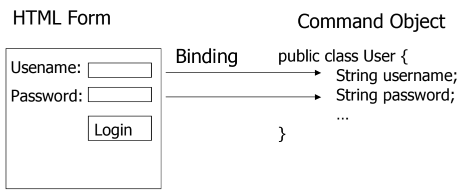
Select A Controller

Controller (interface) AbstractController	For simple controllers that do not use request parameters
BaseCommandController AbstractCommandController	Use and validate request parameters
AbstractFormController SimpleFormController	Handles form input
AbstractWizardFormController	Handles multi-page form input

Simple Controller Examples

- LogoutController
- IndexController
- AdminController
- MemberController

Command Object



```
public class User {
        String username;
        String password;
        …
}
```

- Any class can be used as a command class in Spring
  - vs. ActionForm in Struts

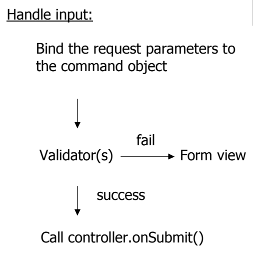
Simple Form Handling

- The controller needs to handle two cases:
  - Display form
  - Process input data

Handle first request:	Handle input:
Create a command object and expose the object as a page scope variable	Bind the request parameters to the command object
↓	↓
Display the form view	Call controller.onSubmit()

Validation

- org.springframework .validation
  - Validator
  - Errors
- Examples:
  - LoginValidator
  - ItemValidator


Handle input:
Bind the request parameters to the command object
Validator(s)
fail
Form view
success
Call controller.onSubmit()

messages.properties

- <code,string> pairs
- A single place for output messages
  - Easy to change
  - I18N
- Need to declare a messageSource bean in Spring configuration file
- Can be used by <fmt> tags in JSTL

Display Errors

- Spring Tag Library
  - http://static.springframework.org/spring/docs/1.2.5/taglib/
- <spring:hasBindErrors>
- A few other useful tags

Limitation of Spring Validation

- Server-side only
- Takes lots of coding for anything other than empty/white spaces
- Vs. Common-validator support in Struts
- Current efforts
  - Spring + Struts
  - Porting common-validator
  - Spring validation plugin

A More Complex Example

- ItemController
  - Handles three cases
    - Add a new item
    - Edit items that belong to the user
    - Edit items that do not belong to the user
  - Custom command class
  - Override `formBackingOject()`

Roadmap

- Web flow
- Controllers and validation
- **Transactions and hibernate support**
- Bits and pieces
  - Tomcat server setup
  - Context, data source, and connection pooling
  - Creating WAR files
  - JSP pre-compilation
  - Displaytag

Programmatic vs. Declarative Transaction Management

Programmatic:

```
void saveItem( Item I )
{
   transaction.start();
   …
   transaction.end();
}
```

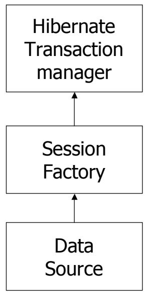
Declarative:

```
<transaction>
   <method>
      saveItem
   </method>
</transaction>
```

Spring Transaction Managers

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

Configure Hibernate Transaction Manager



Adding Transaction Aspect

- Transaction advisor + auto proxy
- TransactionProxyFactoryBean
  - target
  - transactionManager
  - transactionAttributeSource

Transaction Attributes

- Propagation behaviors
- Isolation levels
- Read-only hints
- Transaction timeout period

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.

Other Hibernate Support

- HibernateTemplate
- OpenSessionInViewFilter and OpenSessionInViewInterceptor

Roadmap

- Web flow
- Controllers and validation
- Transactions and hibernate support
- **Bits and pieces**
  - Tomcat server setup
  - Context, data source, and connection pooling
  - Creating WAR files
  - JSP pre-compilation
  - Displaytag

Tomcat Server Setup

- Unzip the Tomcat binary release package to a directory
- Copy the JDBC driver of your database to common/lib
- [Optional] Edit conf/server.xml to change the default port number

Context, Data Source, and Connection Pooling

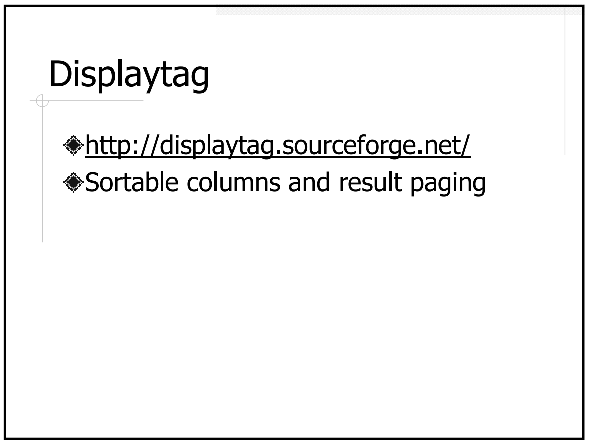
- Connection pooling
- Configure Tomcat and DBCP
  - http://tomcat.apache.org/tomcat-5.0-doc/jndi-datasource-examples-howto.html

Creating WAR Files

- Understand the directory structure
- Use the war ANT task

JSP Pre-compilation

- Usually a JSP is converted to a servlet and then compiled into byte code *when the JSP is request for the first time*.
- JSP pre-compilation
  - Eliminate the "first request overhead"
  - Speed up development
- Tomcat provides JSP pre-compiler which can be used as an ANT task



Displaytag

- http://displaytag.sourceforge.net/
- Sortable columns and result paging