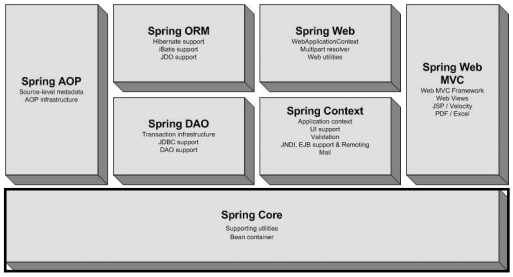
CS520 Web Programming
Spring – Bean Container

Chengyu Sun
California State University, Los Angeles

Background

- Developed by Rod Johnson
- Described in *Expert One-on-One: J2EE Design and Development (2002)*
- Addresses many problems of EJB
  - Overly complex
  - Invasive
  - Hard to test
  - Entity beans

Spring Framework



Example: Use of DAO (I)

CreateMailingListController.java:

```
MailingListDaoHibernateImpl mailingDao;
mailingDao = new MailingListDaoHibernateImpl();
mailingDao.save( mailingList );
```

SubscribeMailingListController.java:

```
MailingListDaoHibernateImpl mailingDao;
mailingDao = new MailingListDaoHibernateImpl();
mailingDao.save( mailingList );
```

UnsubscribeController.java:

ListMailingListMessagesController.java:

…

Potential Problem in the Example

- What if we decide to use JDBC instead of Hibernate, i.e. replace `MailingListDaoHibernateImpl` with `MailingListDaoJdbcImpl`

Example: Use of DAO (II)

CreateMailingListController.java:

```
MailingListDao mailingDao;
mailingDao = new MailingListDaoHibernateImpl();
mailingDao.save( mailingList );
```

SubscribeMailingListController.java:

```
MailingListDao mailingDao;
mailingDao = new MailingListDaoHibernateImpl();
mailingDao.save( mailingList );
```

UnsubscribeController.java:

ListMailingListMessagesController.java:

…

Example: Use of DAO (III)

```
MailingListDao mailingDao;
mailingDao.save( mailingList );
…

public void setMailingListDao( MailingListDao mailingListDao)
{
   this.mailingListDao = mailingListDao;
}
```

- No more dependency on a specific implementation of the DAO
- But who will call the setter?

Spring Bean Container

- Bean == POJO
- Container ??

Example: Hello World

- `Message` is a POJO managed by the Spring container
  - Created by the container
  - Property is set by the container

Bean Configuration File

```
<beans>

   <bean id="msgBean"
         class="cs520.spring.hello.Message">
      <property name="message" value="Hello World!" />
   </bean>

</beans>
```

- The string `"Hello World"` is injected to the bean `msgBean`

Dependency Injection

- Methods of injection
  - via Setters
  - via Constructors
- Objects that can be injected
  - Simple types: strings and numbers
  - Collection types: list, set, and maps
  - Other beans

Dependency Injection Example

- DjBean

Quick Summary of Bean Configuration

Bean	<bean>, "id", "class"
Simple type property	<property>, "name", "value"
Class type property	<property>, "name", "ref" (to another <bean>)
Collection type property	<list>/<set>/<map>/<props>, <value>/<ref>/<entry>/<prop>
Constructor arguments	<constructor-arg>, "index", same as other properties

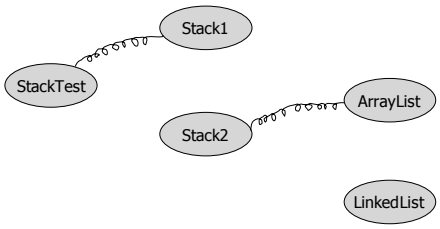
Some Bean Configuration Examples

```
<property name="foo">
   <set>
      <value>bar1</value>
      <ref bean="bar2" />
   </set>
</property>
```

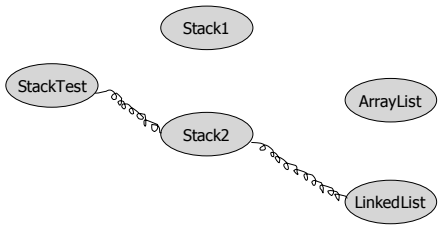
```
<property name="foo">
   <map>
      <entry key="key1">
         <value>bar1</value>
      </entry>
      <entry key="key2">
         <ref bean="bar2" />
      </entry>
   </map>
</property>
```

```
<property name="foo">
   <props>
      <prop key="key1">bar1</prop>
      <prop key="key2">bar2</prop>
   </props>
</property>
```

Wiring



Wiring



Auto Wiring

- <bean autowire="autowire type"/>
- <beans default-autowrire="autowire type">
- Auto wire types
  - byName
  - byType
  - constructor
  - autodetect

Inversion of Control (IoC)

- Another name for Dependency Injection
- Usually application objects are responsible for acquiring their own dependencies
  - E.g `Stack stack = new Stack1();`
- *Inversion of control* means the container injects the dependencies into the application objects

Why IoC is Good?

- Centralized dependency management
- Singleton objects improve performance
  - *Singleton* vs. *Prototype*

More Readings

- *Professional Java Development with the Spring Framework*
  - Chapter 1 and 2
- *Spring in Action*
  - Chapter 1.4 Understand Inversion of Control
- Spring Reference Manual for V2.0 - http://static.springframework.org/spring/docs/2.0.x/reference/index.html
  - Chapter 3