分享经验

永久域名 http://jdh.javaeye.com



jxh118

浏览: 33500 次

性别: 💣

来自: 广州



详细资料

留言簿

搜索本博客

最近访客 客 <u>>>更多访</u>





pn2008

rendajuan





taizi982233 jiming

博客分类

2008-09-08

spring执行定时任务 | XFire与Spring结合

CXF简单例子

service框架用apache CXF, 服务器tomcat, 还有spring框架

OpenSource中关于CXF的简介: Apache CXF一个开源的Service框架,它实现了JCP与Web Service中一些重要标准。CXF简化了构造,集成,面向服务架构(SOA)业务组件与技术的灵活复用。在CXF中,Service使用WSDL标准定义并能够使用各种不同的消息格式(或binding)和网络协议(transports)包括SOAP、XML(通过HTTP或JMS)进行访问。CXF同样支持多种model 如: JAX-WS,JBI,SCA和CORBA service。CXF设计成可灵活部署到各种容器中包括Spring-based,JBI,SCA,Servlet和J2EE容器。

用CXF构建webservice程序的大致流程如下:

- 1 配置web.xml文件,加载CXF
- 2编写java接口,和接口的实现
- 3 在spring的applicationContext.xml中配置java文件
- 4 创建beans.xml文件,里面配置接口的实现类
- 5 运行tomca服务器即可生成wsdl文件
- 6 编写测试代码:编写java测试文件和client-beans.xml文件

7编写运行客户端文件所需的服务器,可以单独编写,也可以在随tomcat启动

示例代码。首先下载cxf,然后解压,将lib下的所有jar包拷到工程下面

- 一最简单的helloword
- 1 web.xml

Xml代码

- 1. <web-app>
- 2. <context-param>

```
3.
                                                   <param-name>contextConfigLocation
■ 全部博客 (55)
                                     4.
                                                   <param-value>WEB-INF/beans.xml</param-value>
                                     5.
                                              </context-param>

    Seam (1)

                                     6.
JSF (11)
                                     7.
                                              tener>
                                                   <listener-class>
                                     8.

    Struts2 (1)

                                     9.
                                                       org.springframework.web.context.ContextLoaderListener
                                    10.
                                                   </listener-class>
Richfaces (5)
                                    11.
                                              </listener>

    Hibernate (4)

                                    12.
                                              <servlet>
                                    13.

    Spring (10)

                                                   <servlet-name>CXFServlet</servlet-name>
                                    14.
WebServices (7)
                                   15.
                                                   <display-name>CXF Servlet</display-name>
                                                   <servlet-class>
                                    16.

    XML (3)

                                    17.
                                                       org.apache.cxf.transport.servlet.CXFServlet

    Ajax (3)

                                    18.
                                                   </servlet-class>
                                                   <load-on-startup>1</load-on-startup>
                                    19.
Compass (1)
                                    20.
                                              </servlet>
                                    21.

    JMS (3)

                                    22.
                                              <servlet-mapping>
                                    23.
                                                   <servlet-name>CXFServlet</servlet-name>
 我的留言簿
              >>更多留言
                                    24.
                                                   <url-pattern>/*</url-pattern>
■ 交个朋友,我做搜索的
                                    25.
                                              </servlet-mapping>
  www.geshuai.com
                                    26.
                                    27.
                                          </web-app>
  www.lucene.c ...
                                 2 接口HelloWord.java和接口的实现HelloWordImpl.java
  -- by sonx2
                                   Java代码
 其他分类
                                          package com.demo;
                                     2.
                                          import javax.jws.WebService;
                                     3.
 我的论坛主题贴 (55)
                                     4.
```

- 我的收藏 (52)
- 我的所有论坛贴 (8)
- 我的精华良好贴 (0)

最近加入圈子

- EXT
- JBPM @net
- Ubuntu For Fun
- GlassFish

```
/** *//**
 5.
      * service interface
 6.
 7.
      * @author
 8.
9.
10.
      @WebService
11.
      public interface HelloWord ...{
12.
           String sayHi(String text);
13.
      }
14.
15.
```

```
struts2
存档
2008-09 (7)
2008-08 (7)
2008-07 (4)
■ 更多存档...
 最新评论
■ <u>JSF实现验</u>证码
  谢谢,很好用!
  -- by pangbuddy
■ <u>a4j:ajaxListener 实现aj ...</u>
  jxh118高手,不知道你是否可以
  做一个使用rich:ajaxValidator ...
  -- by <u>lizhiquo6</u>
■ RichFaces文件上传
  FacesUtil对象找不到啊
  -- by xiguang
■ 动态自增表格rich:dataTab ...
  可以详细介绍一
  下, a4i:commandLink在richface
  richta ...
  -- by wyyacyy
■ JSF如何实现复杂表格的呈 ...
  这个实现方法没有使
  用datatable,如果要求排序的和
  分页的, 您如何实现呢?
  -- by rsic
 评论排行榜
```

```
16.
17.
18.
19.
      package com.demo;
20.
      import javax.jws.WebService;
21.
22.
      /** *//**
23.
      * implementation
24.
25.
      * @author
26.
27.
      */
28.
      @WebService
29.
      public class HelloWordImpl implements HelloWord ...{
30.
31.
          public String sayHi(String text) ...{
32.
               return "Hi" + text;
33.
34.
          }
35.
     }
```

3 beans.xml

Xml代码

```
1.
      <?xml version="1.0" encoding="UTF-8"?>
 2.
 3.
      <beans xmlns="http://www.springframework.org/schema/beans"</pre>
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 4.
 5.
           xmlns:jaxws="http://cxf.apache.org/jaxws"
 6.
           xsi:schemaLocation="
 7.
      http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-be
      ans.xsd
      http://cxf.apache.org/jaxws http://cxf.apache.org/schemas/jaxws.xsd">
 8.
 9.
10.
           <import resource="classpath:META-INF/cxf/cxf.xml" />
           <import resource="classpath:META-INF/cxf/cxf-extension-soap.xml" />
11.
           <import resource="classpath:META-INF/cxf/cxf-servlet.xml" />
12.
13.
14.
           <jaxws:endpoint id="helloWord"</pre>
15.
               implementor="com.demo.HelloWordImpl" address="/HelloWord" />
16.
17.
      </beans>
```



4 启动tomcat服务器,在浏览器里面输入http://localhost:8888/helloword后页面上会出现

{http://demo.com/}HelloWordImplPort,点击会看到生成的wsdl文件http://localhost:8888/helloword/HelloWord?wsdl

5 客户端Client.java和client-beans.xml

Java代码

```
1.
      package com.demo;
 2.
 3.
      import org.springframework.context.ApplicationContext;
 4.
      import org.springframework.context.support.ClassPathXmlApplicationContext;
 5.
      public final class Client ...{
 6.
 7.
           private Client() ...{
 8.
 9.
10.
          }
11.
12.
           public static void main(String args[]) throws Exception ...{
               // START SNIPPET: client
13.
14.
               ApplicationContext context = new ClassPathXmlApplicationContext(
                       new String[] ...{ "com/demo/client-beans.xml" });
15.
16.
               HelloWord client = (HelloWord) context.getBean("client");
17.
18.
19.
               String response = client.sayHi("+what????");
               System.out.println("Response: " + response);
20.
               System.exit(0);
21.
               // END SNIPPET: client
22.
23.
          }
24.
      }
```

Java代码

```
8.
      http://cxf.apache.org/jaxws http://cxf.apache.org/schema/jaxws.xsd">
 9.
10.
           <bean id="client" class="com.demo.HelloWord"</pre>
               factory-bean="clientFactory" factory-method="create" />
11.
12.
           <bean id="clientFactory"</pre>
13.
               class="org.apache.cxf.jaxws.JaxWsProxyFactoryBean">
14.
15.
               cproperty name="serviceClass" value="com.demo.HelloWord" />
               cproperty name="address"
16.
                   value="http://localhost:9002/HelloWord" />
17.
18.
           </bean>
19.
20.
      </beans>
      21.
22.
```

6 客户端运行时所需的服务器Server.java

Java代码

```
1.
      package com.demo;
 2.
 3.
      import org.mortbay.jetty.Connector;
      import org.mortbay.jetty.Handler;
 4.
      import org.mortbay.jetty.handler.DefaultHandler;
 5.
 6.
      import org.mortbay.jetty.handler.HandlerCollection;
 7.
      import org.mortbay.jetty.nio.SelectChannelConnector;
 8.
      import org.mortbay.jetty.webapp.WebAppContext;
 9.
10.
      public class Server ...{
11.
          protected Server() throws Exception ...{
12.
13.
               System.out.println("Starting Server");
14.
15.
               org.mortbay.jetty.Server server = new org.mortbay.jetty.Server();
16.
17.
               SelectChannelConnector connector = new SelectChannelConnector();
18.
               connector.setPort(9002);
               server.setConnectors(new Connector[] ...{connector});
19.
20.
21.
               WebAppContext webappcontext = new WebAppContext();
               webappcontext.setContextPath("/");
22.
23.
24.
               webappcontext.setWar("WebRoot");
25.
```

```
26.
               HandlerCollection handlers = new HandlerCollection();
27.
              handlers.setHandlers(new Handler[] ...{webappcontext, new DefaultHandler()});
28.
               server.setHandler(handlers);
29.
30.
               server.start();
               System.out.println("Server ready...");
31.
               server.join();
32.
33.
          }
34.
35.
          public static void main(String args[]) throws Exception ...{
               new Server();
36.
37.
38.
39.
```

7运行客户端Client.java的顺序,先启动tomcat,然后运行Server.java,最后运行Client.java

```
二不单独编写Server.java文件,让其和tomcat一同启动.需要改动的地方
```

1 web.xml

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
     WEB-INF/applicationContext.xml
     WEB-INF/beans.xml
    </param-value>
  </context-param>
  stener>
    listener-class>
      org.spring framework.web.context. Context Loader Listener\\
    </listener-class>
  </listener>
  <servlet>
    <servlet-name>CXFServlet</servlet-name>
    <display-name>CXF Servlet</display-name>
    <servlet-class>
      org.apache.cxf.transport.servlet.CXFServlet
    </servlet-class>
    <load-on-startup>2</load-on-startup>
  </servlet>
```

```
<servlet-mapping>
    <servlet-name>CXFServlet</servlet-name>
    <url-pattern>/webservice/*</url-pattern>
  </servlet-mapping>
2 applicationContext.xml 配置文件中com.demo.HelloWordFind, HelloWordFind.java可以写一个查询表HELLO_WORD中所有内容的代
码。相应的HelloWord.java和HelloWordImpl.java也需要修改
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.org/dtd/spring-beans.dtd">
<beans default-autowire="byName">
 <br/> <bean id="propertyConfigurer" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
    property name="locations">
      t>
        <value>WEB-INF/jdbc.properties</value>
      </list>
    </property>
  </bean>
  <br/> <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
    cproperty name="driverClassName" value="${jdbc.driverClassName}"/>
    cproperty name="url" value="${jdbc.url}"/>
    cproperty name="username" value="${jdbc.username}"/>
    cproperty name="password" value="${jdbc.password}"/>
  </bean>
  <!-- Hibernate SessionFactory -->
  <br/><bean id="sessionFactory" class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
    cproperty name="dataSource" ref="dataSource"/>
    property name="mappingResources">
      t>
        <value>mapping/HelloWord.hbm.xml</value>
      </list>
    </property>
    property name="hibernateProperties">
      ops>
        prop key="hibernate.show_sql">true
      </props>
    property name="eventListeners">
      <map>
```

```
<entry key="merge">
         <bean class="org.springframework.orm.hibernate3.support.ldTransferringMergeEventListener"/>
      </map>
    </property>
  </bean>
  <bean id="hellowordfind" class="com.demo.HelloWordFind">
    cproperty name="dataSource" ref="dataSource"/>
  </bean>
</beans>
3 beans.xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"</pre>
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:jaxws="http://cxf.apache.org/jaxws"
  xsi:schemaLocation="
http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans.xsd
http://cxf.apache.org/jaxws http://cxf.apache.org/schemas/jaxws.xsd">
  <import resource="classpath:META-INF/cxf/cxf.xml" />
  <import resource="classpath:META-INF/cxf/cxf-extension-soap.xml" />
  <import resource="classpath:META-INF/cxf/cxf-servlet.xml" />
  <bean id="hellowordimpl" class="com.demo.HelloWordImpl">
   cproperty name="sessionFactory" ref="sessionFactory"/>
   cproperty name="dataSource" ref="dataSource"/>
   cproperty name="hellowordfind" ref="hellowordfind" />
  </bean>
  <jaxws:endpoint id="helloword" implementor="#hellowordimpl" address="/hellword" />
</beans>
4 主要需要改动的已经完成。客户端代码变化不大。
学了Java,还找不到工作?
为何学苹果手机开发工资高一倍? 名企委托培养, 本月报名学费75折...
iphone.peixun.it
                                                           Google 提供的广告
                                                                         spring执行定时任务 | XFire与Spring结合
```

					
表情图标	字体颜色: □□ 字体大小: □□ 对齐: □□				
**************************************	提示: 选择您需要装饰的文字, 按上列按钮即可添加上相应的标签				
80000					
⋄ ⋄ ⋄ ⋄					
	您还没有登录,请 <u>登录</u> 后发表评论(快捷键 Alt+S / Ctrl+Enter)				