分享经验

jxh118

浏览: 33500 次

性别:

来自: 广州

我现在离线

详细资料 留言簿

搜索本博客

搜索

最近访客 >>更多访客

pn2008 rendajuan

taizi982233 jiming

博客分类

spring执行定时任务 | XFire与Spring结合

2008-09-08

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>
```

- 全部博客 (55)
- Seam (1)
- JSF (11)
- Struts2 (1)
- Richfaces (5)
- Hibernate (4)
- Spring (10)
- WebServices (7)
- XML (3)
- Ajax (3)
- Compass (1)
- JMS (3)

我的留言簿 >>更多留言

- 交个朋友,我做搜索的

  www.geshuai.com

  www.lucene.c ...

  -- by sonx2

其他分类

- 我的收藏 (52)
- 我的论坛主题贴 (55)
- 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


```
        <param-name>contextConfigLocation</param-name>
        <param-value>WEB-INF/beans.xml</param-value>
    </context-param>

    <listener>
        <listener-class>
            org.springframework.web.context.ContextLoaderListener
        </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>1</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>CXFServlet</servlet-name>
        <url-pattern>/*</url-pattern>
    </servlet-mapping>

</web-app>
```

2 接口HelloWord.java和接口的实现HelloWordImpl.java

Java代码

```
package com.demo;

import javax.jws.WebService;

/** *//**
* service interface
*
* @author
*
*/
@WebService
public interface HelloWord ...{
    String sayHi(String text);
}
```

- struts2

存档

- 2008-09 (7)
- 2008-08 (7)
- 2008-07 (4)
- 更多存档...


```
16.
17.
18.
19.  package com.demo;
20.
21.  import javax.jws.WebService;
22.
23.  /** *//**
24.  * implementation
25.  *
26.  * @author
27.  *
28.  */
29.  @WebService
30.  public class HelloWordImpl implements HelloWord ...{
31.
32.      public String sayHi(String text) ...{
33.          return "Hi" + text;
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"
4.      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
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
8.  http://cxf.apache.org/jaxws http://cxf.apache.org/schemas/jaxws.xsd">
9.
10.     <import resource="classpath:META-INF/cxf/cxf.xml" />
11.     <import resource="classpath:META-INF/cxf/cxf-extension-soap.xml" />
12.     <import resource="classpath:META-INF/cxf/cxf-servlet.xml" />
13.
14.     <jaxws:endpoint id="helloWord"
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

```java
package com.demo;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public final class Client ...{

    private Client() ...{

    }

    public static void main(String args[]) throws Exception ...{
        // START SNIPPET: client
        ApplicationContext  context = new ClassPathXmlApplicationContext(
                new String[] ...{ "com/demo/client-beans.xml" });

        HelloWord client = (HelloWord) context.getBean("client");

        String response = client.sayHi("+what????");
        System.out.println("Response: " + response);
        System.exit(0);
        // END SNIPPET: client
    }
}
```

Java代码

```java
<pre name="code" class="xml"><?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    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/spring-beans-2.0.xsd
```

```
http://cxf.apache.org/jaxws http://cxf.apache.org/schema/jaxws.xsd">

    <bean id="client" class="com.demo.HelloWord"
        factory-bean="clientFactory" factory-method="create" />

    <bean id="clientFactory"
        class="org.apache.cxf.jaxws.JaxWsProxyFactoryBean">
        <property name="serviceClass" value="com.demo.HelloWord" />
        <property name="address"
            value="http://localhost:9002/HelloWord" />
    </bean>

</beans>
</pre>

```

6 客户端运行时所需的服务器Server.java

Java代码

```
package com.demo;

import org.mortbay.jetty.Connector;
import org.mortbay.jetty.Handler;
import org.mortbay.jetty.handler.DefaultHandler;
import org.mortbay.jetty.handler.HandlerCollection;
import org.mortbay.jetty.nio.SelectChannelConnector;
import org.mortbay.jetty.webapp.WebAppContext;

public class Server ...{

    protected Server() throws Exception ...{
        System.out.println("Starting Server");

        org.mortbay.jetty.Server server = new org.mortbay.jetty.Server();

        SelectChannelConnector connector = new SelectChannelConnector();
        connector.setPort(9002);
        server.setConnectors(new Connector[] ...{connector});

        WebAppContext webappcontext = new WebAppContext();
        webappcontext.setContextPath("/");

        webappcontext.setWar("WebRoot");

```

```
            HandlerCollection handlers = new HandlerCollection();
            handlers.setHandlers(new Handler[] ...{webappcontext, new DefaultHandler()});

            server.setHandler(handlers);
            server.start();
            System.out.println("Server ready...");
            server.join();
        }

        public static void main(String args[]) throws Exception ...{
            new Server();
        }

    }
```

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>

  <listener>
     <listener-class>
        org.springframework.web.context.ContextLoaderListener
     </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">

  <bean id="propertyConfigurer" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
    <property name="locations">
      <list>
        <value>WEB-INF/jdbc.properties</value>
      </list>
    </property>
  </bean>
  <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
    <property name="driverClassName" value="${jdbc.driverClassName}"/>
    <property name="url" value="${jdbc.url}"/>
    <property name="username" value="${jdbc.username}"/>
    <property name="password" value="${jdbc.password}"/>
  </bean>
  <!-- Hibernate SessionFactory -->
  <bean id="sessionFactory" class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
    <property name="dataSource" ref="dataSource"/>
    <property name="mappingResources">
      <list>
        <value>mapping/HelloWord.hbm.xml</value>
      </list>
    </property>
    <property name="hibernateProperties">
      <props>
        <prop key="hibernate.dialect">org.hibernate.dialect.DB2Dialect</prop>
        <prop key="hibernate.show_sql">true</prop>
      </props>
    </property>
    <property name="eventListeners">
      <map>
```

```
            <entry key="merge">
               <bean class="org.springframework.orm.hibernate3.support.IdTransferringMergeEventListener"/>
            </entry>
          </map>
      </property>
   </bean>

   <bean id="hellowordfind" class="com.demo.HelloWordFind">
      <property name="sessionFactory" ref="sessionFactory"/>
      <property name="dataSource" ref="dataSource"/>
   </bean>
</beans>
```

3 beans.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
   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/spring-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">
     <property name="sessionFactory" ref="sessionFactory"/>
     <property name="dataSource" ref="dataSource"/>
     <property name="hellowordfind" ref="hellowordfind" />
   </bean>
   <jaxws:endpoint id="helloword"  implementor="#hellowordimpl" address="/hellword" />
</beans>
```

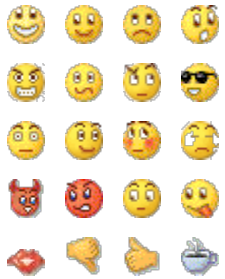
4 主要需要改动的已经完成。客户端代码变化不大。

10:48 | 浏览 (594) | 评论 (0) | 分类: WebServices | 相关推荐

评论

发表评论

表情图标



B I U Quote Code List Img URL Flash Table

字体颜色: 标准 字体大小: 标准 对齐: 标准

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

您还没有登录，请登录后发表评论(快捷键 Alt+S / Ctrl+Enter) 提交