服务框架 HSF 使用与配置

一江 更新时间: 2011-7-20

目录

HSF 常用 OPS 和开发工具介绍	1
HSF 的使用和配置	2
下载和安装 HSF	2
服务开发与部署	5
服务查询	10
服务调用	11
HSFUNIT 测试包的使用	13
HSF Jetty 插件的使用	15
路由规则和限流规则的介绍	21



HSF 常用 OPS 和开发工具介绍

HSF 常用链接入口: <u>http://hsf.taobao.net</u>,包括 HSF 服务查询 OPS,文档中心,下载中 心等链接。





HSF 的使用和配置

下载和安装 HSF

第一步:从 HSF 主页访问<u>HSF 软件下载中心</u>,下载 JBoss 4.2.2 和 HSF 1.4.8.4 压缩包。如 下图所示:

NSF‡	次件下载中心	1999 A	1+ 5 - 5 - 5 - 5 - 5 - 5 - 5 - 5 - 5 - 5	
(+ +	C 🔇 hs	f.taoba	ao.net/hsfversion/	☆ २
LIG	E Marajet	gFreetdarfloor gConcernerfloor		
	Contraction of the second	- 1	•载中心	
HSF各方	太本 卜			
软件名称	版本	下载地址	版本介绍	
HSF	1.4.3	下载	线上用的最多最稳定版本	
HSF	1.4.4	下载	Notify升级的版本,目前tc在用。	
HSF	1.4.5	下载	哈勃最新客户端1.0.7升级版本。	
HSF	1.4.7	下载	Notify客户端升级版本。	
HSF	1.4.7.8	下载	Notify升级到1.7.8版本	
HSF	1.4.8	下载	1.4.5基础上去掉statlog版本,强烈建议使用该版本。	
HSF	1.4.8.2	下载	HSF NDI(No Dependency Interface)版本,支持HSF调用端不需要依赖服	务接口类,就能够进行HSF服务调用。
HSF	1.4.8.4	下载		
HSF	1.6.1-core	下载	1.6.1版本核心包,只具备HSF基本功能,其他高级功能使用的时候,会自	动下载,增加线程池管理,扩展点等功能。 <u>推荐下载</u>
HSF	1.6.1-all	下载	1.6.1版本所有包,具备所有功能,比如webservice,pb等功能,不推荐T	下载该包,核心包运行的时候会自动下载。 <u>不推荐下载</u>
其他相	关软件下	载		
软件名称			版本	下载地址
JBoss			4.2.2	下载
Tomcat			6.0.20	下载
Hotswa	p Patch		0.2	下载
HSF-Sta	andalone 🏨	⊃手册	1.0.0	<u>下载</u>

图 2-1. HSF 下载中心

第二步: 解压 jboss-4.2.2.GA.zip 包到任意目录,如 D:\。这时 JBoss 应该位于 D:\jboss-4.2.2.GA 目录;解压 taobao-hsf.tgz 到%JBOSS_HOME%\server\default\deploy 目录。 至此,JBoss 和 HSF 安装完成。JBoss 服务器目录结构如图 2-2 所示。

本文的后续章节中将,

- 使用%JBOSS_HOME%指代 JBOSS 解压目录 D:\jboss-4.2.2.GA
- 使用%DEPLOY_DIR%指代%JBOSS_HOME%\server\default\deploy 目录





图 2-2. JBoss 和 HSF 安装后目录结构

第三步:执行%JBOSS_HOME%\bin\run.bat 启动 JBoss,这时访问 <u>http://localhost/</u>将能够 看到 JBoss 服务器默认首页,如图 2-3 所示。

Welcome to JBoss ^{ur}		I TO A PRO			X
🗲 🔮 C 🕓 localhost	:8080			☆	4
Boss					
JBoss Online Resources • JBoss Documentation • JBoss Wiki • JBoss JIRA • JBoss Forums					
JBoss Management • <u>Tomcat status (full) (XML)</u> • <u>JMX Console</u> • <u>JBoss Web Console</u>					
	JBo	ss™ Application Sei	rver		

图 2-3. JBoss 服务器默认首页

如果启动时,遇到端口被占用异常: java.net.BindException: Address already in use: JVM_Bind:80,如图 2-4 所示。



java.net.BindException: Address already in use: JVM_Bind:80
at org.apache.tomcat.util.net.JIoEndpoint.init(JIoEndpoint.java:500)
at org.apache.tomcat.util.net.JIoEndpoint.start(JIoEndpoint.java:514)
at org.apache.coyote.http11.Http11Protocol.start(Http11Protocol.java:20
at org.apache.catalina.connector.Connector.start(Connector.java:1146)
at org.jboss.web.tomcat.service.JBossWeb.startConnectors <jbossweb.java< td=""></jbossweb.java<>
at org.jboss.web.tomcat.service.JBossWeb.handleNotification <jbossweb.j< td=""></jbossweb.j<>
at sun.reflect.GeneratedMethodAccessor4.invoke(Unknown Source)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcc
at java.lang.reflect.Method.invoke(Method.java:597)
at org.jboss.mx.notification.NotificationListenerProxy.invoke(Notifica
at \$Proxy44.handleNotification(Unknown Source)
at org.jboss.mx.util.JBossNotificationBroadcasterSupport.handleNotific.
java:127)
at org.jboss.mx.util.JBossNotificationBroadcasterSupport.sendNotificat
va:108)
at org.jboss.system.server.ServerImpl.sendNotification(ServerImpl.java
at org.jboss.system.server.ServerImpl.doStart(ServerImpl.java:497)
at org.jboss.system.server.ServerImpl.start(ServerImpl.java:362)
at org.jboss.Main.boot(Main.java:200)
at org.jboss.Main\$1.run(Main.java:508)
at java.lang.Thread.run(Thread.java:619)
15:31:45,098 WARN [JBossWeb] Failed to startConnectors
LifecycleException: service.getName(): "jboss.web"; Protocol handler start f
eady in use: JUM_Bind:80
at org.apache.catalina.connector.Connector.start(Connector.java:1153)
at org.jboss.web.tomcat.service.JBossWeb.startConn <u>ectors(JBossWeb.java</u>

图 2-4. JBoss 启动,端口占用异常

这说明 JBoss Web 默认使用的 80 端口被其他程序占用,这时你可以停掉占用 80 端口的 应用或修改 JBoss 端口配置。

修改 JBoss 端口的方法,打开%DEPLOY_DIR%\jboss-web.deployer\server.xml 文件,打到 图示的位置,修改端口即可。这里我使用的是 8080 端口。

13	白	<service name="jboss.web"></service>
14		
15	白	<pre><!-- A "Connector" represents an endpoint by which requests are received</pre--></pre>
16		and responses are returned. Documentation at :
17		Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
18		Java AJP Connector: /docs/config/ajp.html
19		APR (HTTP/AJP) Connector: /docs/apr.html
20		Define a non-SSL HTTP/1.1 Connector on port 8080
21	-	> 在这里修改jboss web连接端口
22		<connector <="" address="\${jboss.bind.address}" port="8080" td=""></connector>
23		maxThreads="250" maxHttpHeaderSize="8192"
24		emptySessionPath="true" protocol="HTTP/1.1"
25		enableLookups=" false " redirectPort="8443" acceptCount="100"
26		connectionTimeout="20000" disableUploadTimeout="true" />

图 2-5. 修改 JBoss Web 连接端口

到这里, Jboss 和 HSF 的安装就完成了。部署着 HSF 服务的 war 包类型的应用,可以直接部署到 Jboss 的%DEPLOY_DIR%。



服务开发与部署

本节开发和部署一个简单的 Hello World 服务,使用的开发工具是 <u>eclipse</u>,安装了 <u>m2clipse</u>插件。

第一步:使用 maven 命令或 eclipse 向导创建一个简单的 Web 工程 hsf-sample。 使用命令方式下,命令如下:

mvn archetype:create -DgroupId=com.taobao.hsf.test -DartifactId=hsf-sample -DpackageName=com.taobao.hsf.test -DarchetypeArtifactId=maven-archetype-webapp

使用 eclipse 方式下,过程如下:

1. 选择 File->New->Others (快捷键 Ctrl+N), 选择 Maven 中的 Maven Project, Next,

O Rew	_ 🗆 🗙
Select a wizard Create a Maven Project	
<u>W</u> izards:	
type filter text	
 Maven Checkout Maven Projects from SCM Maven Module Maven Project Plug-in Development Server SQL Development SVN User Assistance Web Web Web Web Services XML Examples 	
() (<u>Back</u> <u>Mext</u>) <u>Finish</u>	Cancel

图 2-6. 新建工程

2. 勾选 Create a simple project, Next,



💮 New Have	n Project	
New Maven Select projec	project ct name and location	M
✔Create a <u>s</u>	imple project (skip archetype selection)	
🗹 Use defaul	t <u>W</u> orkspace location	
Location:	D:\hsf=support\hsf=sample	Brows <u>e</u>
Add projec	t(s) to working set	
Wo <u>r</u> king set		Mor <u>e</u>
▶ Ad <u>v</u> anced		

图 2-7. 新建 Maven 工程

3. 进行配置项目属性配置,如下图所示。配置结束后,选择 Finish。

🔘 New Haver	n Project 📃 🗖 🗙
New Maven p Configure pro	ject M
Artifact	
Group Id:	com. taobao. hsf. sample 💌
Artifact Id:	hsf-sample 💌
Version:	1.0.0
Packaging:	war 💌
Name:	hsf-sample
Description:	hsf hello world sample
∠Parent Projec	et state of the st
Group Id:	
Artifact Id:	
Version:	Browse Clear
▶ Ad <u>v</u> anced	
?	<pre> Eack Mext > Finish Cancel </pre>

图 2-8. 项目属性配置

4. 创建完成的 eclipse 工程 hsf-sample,如图 2-9 所示。





提示:

- 使用命令方式创建的工程,同样可以导入到 eclipse 中。有两种方式:
 - File->Import->Existing Maven Projects,然后选择 Maven 工程目录,执行导入
 - 在 Maven 工程目录中执行: mvn eclipse:eclipse, 生成 eclipse 配置文件,通过 File->Import->Existing Projects into Workspace,然后选择目录执行导入
- 项目创建成功后,默认的使用的编译器为 JDK 1.4 版本,而实际的项目开发目前以
 1.6 为主。下一步中将会通过修改 pom.xml 修改该配置。

第二步: 在项目的 pom 配置中加入以下配置信息,用于<mark>指定编译和打包使用的 JDK 版</mark> 本为 1.6,如下所示。



修改完成后,右击工程,选择 Maven->Update Project Configuration, eclipse 将会刷新 Maven 配置。

第三步:在工程中添加两个两个基本依赖,添加如下内容到 pom.xml 中:

```
<dependencies>
<dependency>
        <groupId>org.apache.geronimo.specs</groupId>
        <artifactId>geronimo-servlet_2.5_spec</artifactId>
        <version>1.2</version>
        <type>jar</type>
        <scope>provided</scope>
</dependency>
```



```
<dependency>
    <groupId>org.springframework</groupId>
        <artifactId>spring</artifactId>
        <version>2.5.6</version>
        <type>jar</type>
        <scope>compile</scope>
        </dependency>
        </dependencies>
```

在这里,geronimo-servlet 用于为 Web 工程提供 servlet 支持,spring 用于为 servlet 提供 spring 应用上下文支持。 安装了 m2clipse 插件的 eclipse 将会自动刷新工程依赖,并下载、加入 jar 包。如果没有自动执行,可以使用 mvn eclipse:eclipse 命令或 m2clipse 选项中的 Update Dependencies (见图 2-10)完成该操作。完成这一步后,eclipse 中的工程将会引入图 2-11 所示的依赖包。



图 2-11. 项目依赖的 jar 包

```
第三步:在源代码中创建一个简单的 HelloWorld 服务,接口和实现如下:
```

```
package com.taobao.hsf.sample.service;

public interface HelloWorldService {

   String sayHello(final String name);

}

package com.taobao.hsf.test.service;

public class HelloWorldServiceImpl implements HelloWorldService {

   public String sayHello(final String name) {

      return "Hello, " + name + "!";

   }

}
```



第四步: 在 src\main\resources 下添加服务的 Spring 配置: ApplicationContext.xml,内容 如下:

```
<?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"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">
    <bean id="helloWorldServiceImpl"</pre>
class="com.taobao.hsf.sample.service.HelloWorldServiceImpl" />
    <bean id="helloWorldServiceProvider"</pre>
class="com.taobao.hsf.app.spring.util.HSFSpringProviderBean"
         init-method="init">
         <property name="serviceInterface"></property name="serviceInterface">
             <value>com.taobao.hsf.sample.service.HelloWorldService</value>
         </property>
         <property name="target"></property name="target">
             <ref bean="helloWorldServiceImpl" />
         </property>
         <property name="serviceVersion"></property name="serviceVersion">
             <value>1.0.0.daily</value>
         </property>
    </bean>
</beans>
```

第五步: 定义 Web 工程的配置,创建 src\main\webapp\WEB-INF\web.xml,这里只加入 一个 Spring 监听器,用于在应用启动时让 Spring 完成服务初始化。配置如下:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    version="2.5">
```



<context-param></context-param>
<pre><param-name>contextConfigLocation</param-name></pre>
<param-value>classpath:ApplicationContext.xml</param-value>
tener>
<pre><listener-class>org.springframework.web.context.ContextLoaderListene</listener-class></pre>
r

第六步: 打包部署应用,在创建的项目根目录下执行, mvn clean package 进行打包。打完成功后, target 目录中将会生成名称为 hsf-sample.war 的 War 包。将该 War 包拷贝到 JBoss 的%DEPLOY_DIR%中,然后启动 JBoss,这就完成了 HSF 应用的发布。

在发布了一个服务后,如何查看该服务已经被成功发布,别人又如何能知道你这一服务 呢?这就要借助于 HSFOPS 平台了,登录 <u>http://hsf.taobao.net/</u>,进入线下查询地址链接,首 页就会显示当前机器所订阅和发布的所有服务,如图 2-12 所示。

本机(10.13.41.71)发布的服务

com.taobao.hsf.sample.service.HelloWorldService:1.0.0.daily

图 2-12. HSFOPS 服务查询

注意:

- 在这一应用的启动过程中,你可能会看到 log4j 相关的异常,这里缺失日志配置引起的,不会影响服务的发布和使用。
- 这里你可能会问,为什么工程没有依赖任何 HSF 的 jar 包,服务却能够发布成功。 原因是 JBoss 服务器中已经部署了 HSF 包, 国此,部署在 JBoss 中的所有应用都可 以透明地使用 HSF 所提供的功能。

服务查询

所有正常发布出去的 HSF 服务,都可以通过 HSFOPS 平台查询到其发布信息。HSFOPS 平台能够实时从配置中心查询服务的各种信息,包括:服务地址信息、服务的被调用信息,服务地址的实时推送情况等等。下表为 HSFOPS 查询语法的示例。

com.taobao.api.item.ltemService:1.0.0 根据 datald 精确检索服务



com.taobao.api.*	模糊匹配以 com.taobao.api.开头的服务
item	模糊匹配服务名称中包括 item 的服务
item type:consumer	匹配订阅了*item*的服务订阅者
uic group:HSF	检索 HSF 组内的*uic*服务
uic ip:172.23*	检索 172.23.网段发布的*uic*服务
uic ip:10.* type:consumer	检索 10.网段内订阅了*uic*服务的服务消费者

服务调用

下面介绍如何调用一个 HSF 服务。这里继续使用上一节中创建的 Web 工程 hsf-sample 完成这一实例。

第一步:添加服务调用配置信息,编辑 src\main\resources 中的 ApplicationContext.xml,添加以下内容:

配置文件中指定了要调用的接口名以及版本,这需要<mark>与前一节中的服务提供者的定义相</mark> 一致。

第二步:编写一个简单的 servlet,内容如下:

```
package com.taobao.hsf.sample.servlet;
import ...
@SuppressWarnings("serial")
public class HelloWorldServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
```



```
WebApplicationContext context =
WebApplicationContextUtils.getWebApplicationContext(getServletContext());
HelloWorldService helloWorldService = (HelloWorldService)
context.getBean("helloWorldService");
PrintWriter out = resp.getWriter();
out.println(helloWorldService.sayHello("yijiang"));
return;
}
```

第四步:在 web.xml 中添加 servlet,编辑 web.xml,添加以下内容:



第五步:重新编译,打包,并部署到 JBoss 中,使用与前一节相同的命令和部署方式。 部署成功后,启动 JBoss,尝试访问:<u>http://localhost:8080/hsf-sample/helloWorld</u>,如果服务 器端口经过修改,请将 URL 中的端口替换成修改后的端口。



图 2-13. HelloWorld 实例执行结果



注意:

- 该实例中,服务提供者和服务调用者位于同一个应用中。如果不在同一应用内, 服务调用者应用需要依赖相同服务定义接口包。实际项目中,应用的提供和使用 往往是在多个项目中的。
- 通过这个例子可以看到,HSF 主要是<mark>使用 HSFSpringProviderBean 和</mark> HSFSpringConsumerBean 两个 Spring Bean 工厂类完成了对服务的发布和调用。
- 这里的实例中的编程规范不一定符合线上项目开发规范,不作为开发参考。

HSFUNIT 测试包的使用

HSFUNIT 是目前推荐使用的 HSF 服务测试包,它提供了直接在 JUNIT 或类 JUNIT 测试场 景中调用 HSF 服务。下面介绍一下如何使用 HSFUNIT 测试包,HSF 服务以刚才我们发布的服 务为例。

第一步: 在项目的/src/test/resources 目录添加测试资源: ApplicationContext-Test.xml, 该文件中配置了一个 HSFSpringConsumerBean, 和上一节中的配置文件内容基本一致, 惟一的不同是增加了 target 属性,显式地指定了要调用的服务目标地址。

```
<?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"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">
    <bean id="helloWorldService"
class="com.taobao.hsf.app.spring.util.HSFSpringConsumerBean"
        init-method="init">
        <property name="interfaceName"></property name="interfaceName">
    <value>com.taobao.hsf.sample.service.HelloWorldService</value>
        </property>
        <property name="version"></property
            <value>1.0.0.daily</value>
        </property>
        <property name="target">
            <value>10.232.36.84:22200</value>
        </property>
    </hean>
</beans>
```



第二步:编写测试用例,测试用例中依赖了两个包 Junit 和 spring-test 包,可以添加以下依赖到 pom 配置中,

<dependency></dependency>
<groupid>log4j</groupid>
<artifactid>log4j</artifactid>
<version>1.2.15</version>
<type>jar</type>
<scope>compile</scope>
<dependency></dependency>
<groupid>junit</groupid>
<artifactid>junit</artifactid>
<pre><version>4.8.2</version></pre>
<type>jar</type>
<scope>test</scope>
<dependency></dependency>
<proupid>org.springframework</proupid>
<pre><artifactid>spring-test</artifactid></pre>
<pre><version>3.0.5.RELEASE</version></pre>
<type>jar</type>
<scope>test</scope>
<dependency></dependency>
<proupid>com.taobao.hsf</proupid>
<artifactid>hsfunit</artifactid>
<version>1.0.2</version>
<type>jar</type>
<scope>test</scope>

第三步:编写和执行测试用例。测试用例代码如下所示:



```
package com.taobao.hsf.sample.service;
import ...
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations =
{ "classpath:ApplicationContext-Test.xml" })
public class HelloWorldServiceTestCase {
   static {
      HSFEasyStarter.start("D:\\hsf-support\\hsf", "1.4.8.4");
   }
   @Autowired
   protected HelloWorldService helloWorldService;
   @Test
   public void testHelloWorld() {
      String result = helloWorldService.sayHello("yijiang");
      Assert.assertEquals("Hello, yijiang!", result);
   }
}
```

HSF Jetty 插件的使用

为了方便开发人员使用 HSF,提高开发效率,HSF 提供了支持 Jetty 服务器的 eclipse 插件。下面介绍如何安装和在开发中使用 Jetty 插件。这里采用的 eclipse 版本为 3.5.2。

第一部分:安装 Jetty 插件。在 eclipse 中选择 Help->Install New Software...,

🛇 Java - Eclipse			
<u>File Edit Source Refactor Navigate Search</u>	n <u>P</u> roject <u>R</u> un <u>W</u> indow	Help	
📬 • 🔜 🗁 😵 🈰 🛵 🏇 • 🕥 • 💁 •	👍 🖶 🞯 • 🗄 🍅 🖨 🖉	¥elcome	
ቹ Package Explorer 🛛 😓 Navigator 🗖 🗖		⑦ Help Contents	
		💖 Search	
		Dynamic Help	_
		Key Assist Ctrl+Shift+L	
		Tips and Tricks	
		🏽 Report Bug or Enhancement	
		<u>C</u> heat Sheets	
		성 Report Maven Issue	
	_	Check for Updates	
		Install New <u>S</u> oftware	
		About Eclipse	

图 3-1. Install New Software...

在弹出的 Install 对话框中, 输入 http://hsfops.taobao.net/hsf/hsf_jetty/update_site, 然后



选择 Add。在弹出的 Add Site 对话框中直接点击 OK。如图 3-1, 3-2 所示。

🛛 Install 📃 🗖 🗙
Available Software Select a site or enter the location of a site.
Work with: http://hsfops.taobao.net/hsf/hsf_jetty/update_site
Find more software by working with the <u>'Available Software Sites'</u> preferences.
type filter text
Name Version
U There is no site selected.
Details
✓ Show only the latest versions of available software ☐ Hide items that are already installed ✓ Group items by category What is already installed?
└└└└└ontact all update sites during install to find required software
() (<u>Back</u> Next > Einish Cancel

图 3-1. Install 对话框

🖸 Install	
Available Software Select a site or enter the location of a site.	
York with: http://hsfops.taobao.net/hsf/hsf_jetty/update_site	 erences.
type filte Name Location: http://hsfops.taobao.net/hsf/hsf_jetty/update_site Archive	
OK Cancel	
✓ Show only the latest versions of available software ☐ Hide items that are already instal. ✓ Group items by category What is already installed?	led
☑ <u>C</u> ontact all update sites during install to find required software	
() < Back Next > Einish Cancel	cel

图 3-2. Add Site 对话框

Eclipse 将搜索并显示出要安装的插件,选中插件,Next,如图 3-3 所示。



© Install	
Available Software Check the items that you wish to install.	
York with: http://hsfops.taobao.net/hsf/hsf_jetty/update_site Find more software by working with the <u>'Available Software Sites</u>	<u>A</u> dd s' preferences.
type filter text	
Name Version	
□ ♥ WW HSF_Jetty和符 ▼ 禄 Feature 1.0.7	
 ✓ Show only the latest versions of available software ☐ Hide items that are already ✓ Group items by category ✓ Contact all update sites during install to find required software 	installed
(<u>Back</u> <u>Next</u> >	Cancel

图 3-3. Install 对话框

然后将出现安装描述信息对话框,选中 Feature, Next,如图 3-4 所示。

🖸 Install				
Install Details Review the items to be installed.				
Name	Version	Id		
🌆 Feature	1.0.7	com.taobao.hsf	. jetty. feature. f	eatur
Size: Unknown				
Details 1.0.2 修改资源路径找不到资源的bug。				
L				More
0	< <u>B</u> ack	<u>N</u> ext >	<u>F</u> inish	Cancel

图 3-4. Install Details 对话框

接受授权, Next。如图 3-5 所示。



O Install	
Review Licenses Licenses must be reviewed and accepted before the software can be installed.	
License text:	
 0.2 修改资源路径找不到资源的bug。 1.0.3 修改classpath路径过长号致后动不了的bug。 1.0.4 jetty cache文件的问题,修改file cache为false。 1.0.5 l.修改webx3里面getResourcePaths找错贷源路径问题. 2.可以不需要指定相比sfl要处过运行. 3.新建启动项JVM参数默认为:-Xms1024m -XX:PermSize=512m -XX:MaxPe: 4.增加默认启动项。 1.0.6 修改autocorfig,支持jar包config。 1.0.7 测试类及资源文件不加载和HSF单元测试包不加载的bug。 powered by 玄肖 	rmSize=512m
? (Back Next > Finish	Cancel

图 3-5. Review License 对话框

接下来, eclispe 将下载并执行安装, 中间会有一次弹出安全警告对话框, 选择 Ok 确定 继续安装。如图 3-6 所示。

	🛛 Security Warning 📃 🗖 🗙	
	Warning; You are installing software that contains unsigned content. The authenticity or validity of this software cannot be established. Do you want to continue with the installation?	
	OK Cancel Details >>	
🛃 Problems	🛿 Javadoc 😟 Declaration 📮 Console 🖾 Progress 🕺 🛛 Ju JUnit	×
🛞 Install		

图 3-6. 安全警告对话框

安装完成后 eclipse 会提示重新启动以启用该插件。重启后, Jetty 插件就安装完成了。

第二部分:使用 Jetty 插件。

在 eclipse 中选择 Run->Run Configuration...,如图 3-7 所示。



<u>F</u> ile <u>E</u> dit <u>Source</u> Refac <u>tor</u> <u>Navigate</u> Se <u>a</u> rch <u>P</u> roject <u>Run</u> <u>W</u> indow <u>H</u> elp
╡ 📬 ▾ 🔜 🚔 ╡ ॐ ▾ 💽 ▾ ¥ 😢 😫 📽 🮯 ▾ ╡ 🕭 🖋 ▾ ╡ 🧏 ▾ 🖗 ▾ 🖓 ▾ ↔ ↔ ▾ ↔ ▾
🛱 Package Explo 🔀 JU <u>1</u> HelloWorldServiceTestCase
<u>R</u> un As
Run Configurations
🖨 🥮 src/main/java Organize Fayorites
😟 🌐 com. taobao. hsf. sample. service
🗄 🖶 com. taobao. hsf. sample. servlet
🚊 🥮 src/main/resources
ApplicationContext.xml
🚊 🥮 src/test/java
🚊 🖶 com. taobao. hsf. sample. service
🗄 🕖 HelloWorldServiceTestCase. j
i i i i i i i i i i i i i i i i i i i

图 3-7. Run Configuration...

按照图 3-8 所示逐一配置各个参数, 就完成了配置过程, 点击 Run, 就可以启动应用了。

💭 Run Configurations		×
Create, manage, and run co Create a configuration to law	nfigurations mch a web application with Jetty.	\bigcirc
 type filter text Apache Tomcat Eclipse Application Eclipse Data Tools Generic Server Generic Server (Extern HTTP Preview JZEE Preview Java Applet Java Applet Jetty Webapp Enframple JU Junit JU HelloWorldService' JU Junit Plug-in Test OSGi Framework XSL 	Name: hsf-sample E Jetty @= Arguments 副 JRE @ Classpath Source E Environm Project hsf-sample Ports HTTP 8888 HTTP 8888 HTTP 8888 HTTP Bassword E Evy Password Web Application Context / WebApp dir Src/main/webapp HSF Dir D:\taobao-hsf. sar HSF包路径 配置文件 配置文件 配置文件目录 代码热替换? (使用代码热替换	ent Common Browse Browse Browse Browse
Filter matched 17 of 17 item	Apply	Revert
?	<u><u>R</u>un</u>	Close

图 3-8. 配置 Jetty 插件运行参数



注意:

启动肘如果遇到错误:

Error occurred during initialization of VM Could not reserve enough space for object heap

可以在配置对话框的Arguments中将启动参数适当调小。如图3-9所示。

🔘 Run Configurations	
Create, manage, and run co	onfigurations
Image: Second	Mane: hsf=sample Image: Source ima
Filter matched 17 of 17 item	Appl <u>y</u> Reyert
?	<u>Kun</u> Close

图 3-9. 配置 Jetty 插件启动参数

成功启动 Jetty 插件后,尝试访问 <u>http://localhost:8888/helloWorld</u>,应该可以看到与图 2-11 一样的返回结果。

Jetty 插件还提供了数据源配置功能,这里不详细介绍,有兴趣的可以参考 HSF Jetty 插 件使用指南。



路由规则和限流规则的介绍

HSF 路由规则支持接口路由,方法路由,参数路由,并且在路由规则的情况下支持权重规则, HSF 采用 Groovy 脚本作为路由规则设置内容。详细介绍见 confluence: http://confluence.taobao.ali.com:8080/pages/viewpage.action?pageId=100270328

HSF 限流规则提供了对某应用或服务的访问流量进行控制的功能。详细介绍见 confluence: http://confluence.taobao.ali.com:8080/pages/viewpage.action?pageId=192188098