## DWR推送（反向Ajax Server-Clients）

特别提醒：DWR会占用/dwr/\* 此路劲，项目其他模块请避免此路劲，包括webapp文件路劲

### 集成步骤

1. 添加jar包，maven依赖地址：

<dependency>

<groupId>org.directwebremoting</groupId>

<artifactId>dwr</artifactId>

<version>3.0.1-RELEASE</version>

</dependency>

1. 复制com.yinhai.plugin.dwr包代码到项目当中。
2. 修改web.xml，添加内容如下：

<!-- dwr Servlet -->

<servlet>

<display-name>DWR Servlet</display-name>

<servlet-name>dwr-invoker</servlet-name>

<servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>

<init-param>

<param-name>debug</param-name>

<param-value>true</param-value>

</init-param>

<!-- 开启反向Ajax -->

<init-param>

<param-name>activeReverseAjaxEnabled</param-name>

<param-value>true</param-value>

</init-param>

<!-- 自定义的scriptSession manager -->

<init-param>

<param-name>org.directwebremoting.extend.ScriptSessionManager</param-name>

<param-value>com.yinhai.plugin.dwr.DWRScriptSessionManager</param-value>

</init-param>

<!-- 开启 polling mode, 默认 poll rate 5000-->

<init-param>

<param-name>org.directwebremoting.extend.ServerLoadMonitor</param-name>

<param-value>org.directwebremoting.impl.PollingServerLoadMonitor</param-value>

</init-param>

<init-param>

<param-name>disconnectedTime</param-name>

<param-value>60000</param-value>

</init-param>

</servlet>

1. 于web.xml平级的位置添加dwr.xml，内容大致如下：

<!DOCTYPE dwr PUBLIC

"-//GetAhead Limited//DTD Direct Web Remoting 3.0//EN"

"http://getahead.org/dwr/dwr30.dtd">

<dwr>

<allow>

<create creator="new" javascript="MessagePush">

<param name="class" value="com.yinhai.plugin.dwr.demo.MessagePush"/>

</create>

<create creator="spring" javascript="CaptchaProducer">

<param name="beanName" value="captchaProducer"/>

<!-- <param name=location value="beans.xml"/> -->

</create>

</allow>

</dwr>

注意：默认是可以将ContextLoaderListener加载的spring bean加载到dwr当中。也可以通过location指定xml位置。

1. 接受和发送消息的jsp都需要引用的资源：

<script type='text/javascript' src='<%=basePath %>dwr/engine.js'></script>

<script type='text/javascript' src='<%=basePath %>dwr/util.js'></script>

<script type="text/javascript" src="<%=basePath %>dwr/interface/MessagePush.js"></script>

注意：这些资源都是默认就有的，直接引用即可。最后一个MessagePush.js。对应的dwr.xml当中的 <create creator="new" javascript="MessagePush">

1. demo当中一个receive jsp一个send jsp需要自己阅读，这里就不详述了。
2. server端发送消息代码请参照MessagePush.java，里面有给所有人，以及给指定Browser推送消息。

### 需要注意的地方

1. 按照以上配置，DWR与Controller或者Action没有关系，各管各的。
2. 按照以上配置，DWR采用轮询Polling模式进行推送，会降低服务端的压力。
3. ScriptSession并不是httpsession，每次刷新引用了engine的页面，ScriptSession都会重新创建（因此页面的异步化会更适合DWR的生存），我们可以通过ScriptSessionListener给ScriptSession加入我们需要的东西，页面初始化调用onPageLoad方法，也可以给ScriptSession加入我们需要的属性。