JSON-RPC 轻量级远程调用协议介绍及使用

作者：hiquanta
原文链接：https://ld246.com/article/1506734383513
来源网站：链滴

原文
<h2 id="技术简介">技术简介</h2>
json-rpc 是基于 json 的跨语言远程调用协议，比 xml-rpc、webservice 等基于文本的协议传数据格小；相对 hessian、java-rpc 等二进制协议便于调试、实现、扩展，是非常优秀的一种远程调协议。目前主流语言都已有 json-rpc 的实现框架，java 语言中较好的 json-rpc 实现框架有 jsonrpc4、jpoxy、json-rpc。三者之中 jsonrpc4j 既可独立使用，又可与 spring 无缝集合，比较适合于基于pring 的项目开发。
<h2 id="一-JSON-RPC协议描述">一、JSON-RPC 协议描述</h2>
json-rpc 协议非常简单，发起远程调用时向服务端传输数据格式如下：
{ "method": "sayHello", "params": ["Hello JSON-RPC"], "id": 1}
参数说明：
method： 调用的方法名
params： 方法传入的参数，若无参数则传入 []
id ： 调用标识符，用于标示一次远程调用过程
服务器其收到调用请求，处理方法调用，将方法效用结果效应给调用方；返回数据格式：
<pre><code class="highlight-chroma"> {
</code></pre>
<pre><code class="highlight-chroma"> "result": "Hello JSON-RPC",
</code></pre>
<pre><code class="highlight-chroma"> "error": null,
</code></pre>
<pre><code class="highlight-chroma"> "id": 1
</code></pre>
<pre><code class="highlight-chroma">}
</code></pre>
参数说明:
result: 方法返回值，若无返回值，则返回 null。若调用错误，返回 null。
error ： 调用时错误，无错误返回 null。
id : 调用标识符，与调用方传入的标识符一致。
以上就是 json-rpc 协议规范，非常简单，小巧，便于各种语言实现。
<h2 id="二-JSON-RPC简单示例">二、JSON-RPC 简单示例</h2>
<h2 id="2-1-服务器端Java调用示例">2.1、服务器端 Java 调用示例</h2>
jsonrpc4j 服务器端 java 示例：
public class HelloWorldServlet extends HttpServlet {
<pre><code class="highlight-chroma">**private** **static** **final** **long** _serialVersionUID_ = 3638336826344504848L;

private JsonRcServer rpcService = **null**;

@Override

public **void** init(ServletConfig config) **throws** ServletException {

    super.init(config);</code></pre>

```
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    rpcService = **
ew** JsonRpcServer(**new** HelloWorldService(), HelloWorldService.**class**);
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">}
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">@Override
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">**protected** **vo
d** service(HttpServletRequest req, HttpServletResponse resp)
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">        **throws** Se
vletException, IOException {
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    rpcService.hand
e(req, resp);
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">}
</span></span></code></pre>
<p>}</p>
<h2 id="2-2-Java客户端调用示例">2.2、Java 客户端调用示例</h2>
<p>jsonrpc4j 的 Java 客户端调用示例：</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">    JsonRpcHttpClient client = **new** JsonRpcHttpClient(
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">        **new** URL
"http://127.0.0.1:8080/index.json"));
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    Map headers =
*new** HashMap();
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    headers.put("n
me", "剑白");
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">client.setHeaders
headers);
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    String propertie
 = client.invoke("getSystemProperties", **null**, String.**class**);
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    System._out_.pr
ntln(properties);
</span></span></code></pre>
<h2 id="2-3-PHP客户端调用示例">2.3、PHP 客户端调用示例</h2>
<p>基于 json-rpc-php 的 PHP 客户端调用示例：</p>
<p>include(dirname(<strong>FILE</strong>)."/lib/client/JsonRpcClient.php");</p>
<p>$client = new JsonRpcClient("<a href="https://ld246.com/forward?goto=http%3A%2F%2
10.13.49.234%3A8080" target="_blank" rel="nofollow ugc">http://10.13.49.234</a>/index.js
n");</p>
<p><span class="language-math">response = </span>client-&gt;getSystemProperties();</
>
<p>echo $response-&gt;result;</p>
<p>?&gt;</p>
<h2 id="2-3-JavaScript客户端调用示例">2.3、JavaScript 客户端调用示例</h2>
```

基于 jsonrpcjs 的 JavaScript 客户端调用示例：
var rpc = new jsonrpc.JsonRpc('http://127.0.0.1/index.json');
rpc.call('getSystemProperties', function(result){
alert(result);
});
<h2 id="2-4-直接GET请求进行调用">2.4、直接 GET 请求进行调用</h2>
无需任何客户端，只需手工拼接参数进行远程调用，请求 URL 如下：
http://127.0.0.1:8080/index.json?method=getSysteProperties&id=3325235235235¶ms=JTViJTVk
参数说明:
method : 方法名
params： 调用参数，json 的数组格式[], 将参数需先进行 url 编码，再进行 base64 编码
id : 调用标识符，任意值。
<h2 id="三-JSON-RPC总结">三、JSON-RPC 总结</h2>
json-rpc 是一种非常轻量级的跨语言远程调用协议，实现及使用简单。仅需几十行代码，即可实一个远程调用的客户端，方便语言扩展客户端的实现。服务器端有 php、java、python、ruby、.net 等语言实现，是非常不错的及轻量级的远程调用协议。
<h2 id="参考文档">参考文档</h2>
http://code.google.com/p/jsonrpc4j/
<a href="https://ld246.com/forward?goto=http%3A%2F%2Fjson-rpc.org%2Fwiki%2Fimplementations" target="_blank" rel="nofollow ugc">http://json-rpc.org/wiki/implementations<a>
http://en.wikipedia.org/wiki/JSON-RPC
https://github.com/gimmi/jsonrpcjs
http://bitbucket.org/jbg/php-json-rpc</>
https://github.com/Pozo/json-rpc-php
https://github.com/subutux/json-rpc2ph