文章
Louis Lu · 一月 7, 2021 阅读大约需 4 分钟

创建“虚拟”的SOAP Web 服务

在 Caché 中处理 SOAP 请求时，有时需要通过直接访问（有时是编辑）所发送的 XML（即 SOAP 请求和随后的 SOAP 响应）来调试错误。 如果要调试 Caché Web 服务，使用 SoapUI (https://www.soapui.org/) 之类的工具手动创建和控制 SOAP 请求通常很有用，这样可以很容易地在 Caché Web 服务上看到调整的效果。

但是如果已经有 Web 服务（可能不是 Caché），并且想要调试相关的 Caché Web 客户端该怎么办？ 您可能已将 SOAP 响应 XML 保存在文件中（例如 Caché SOAP 日志），您需要一个“虚拟”Web 服务将其发送到 Caché Web 客户端，就像实际的 Web 服务一样操作。

由于我经常在技术支持的过程中需要调试客户的 Caché Web 客户端，我创建了这样一个“虚拟”的Web 服务 – 见下文：

```
Class JSUtil.DummyWebService Extends %CSP.Page
{

/// Mimic a SOAP Web Service by sending the specified SOAP Response XML.
/// Typically this XML will be copied-and-pasted from a SOAP log.
Parameter CONTENTTYPE = "text/xml";

/// File containing the SOAP Response XML:
Parameter XMLFILENAME = "C:\data\soapresponse.txt";

ClassMethod OnPage() As %Status
   {
                 set XML=""
                 set stream = ##class(%Stream.FileCharacter).%New()
                 set sc = stream.LinkToFile(..#XMLFILENAME)

                 while 'stream.AtEnd {
                  set XML = XML_stream.Read()
                 }

    write XML
   quit $$$OK
   }

}
```

要使用 JSUtil.DummyWebService 类：

1.将参数 XMLFILENAME 的值更改为包含来自 Web 服务的 SOAP 响应的 XML 的位置。
通常，此文件可内容可通过手动剪切和粘贴 Caché SOAP 日志中的响应 XML 消息来创建。
2.使用 Studio 的“View Web Page”获取此 CSP 页面的 URL，它应显示响应的XML消息内容。
3.将上面的 URL 粘贴到 Caché Web 客户端的 LOCATION 参数中。
现在，当调用 Caché Web 客户端时，它将收到参数 XMLFILENAME 指向的 SOAP 请求 XML。

我已经多次使用这种方法来帮助调试 Caché Web 客户端。 参考以下示例：

SOAP 日志的“ Web 客户端的输入 ”中包含以下错误：

ERROR #6203: Unexpected Element

（完整的 SOAP 日志可在此处找到：*https://github.com/ISC-schulman/InterSystems/raw/master/soaplog.txt*）

我们还有 Web 服务的 WSDL (*https://github.com/ISC-schulman/InterSystems/raw/master/example.wsdl*)，但无法访问 Web 服务本身。
（虽然这对于 InterSystems 支持来说很常见，但不可否认，对于客户可能并不常见 – 这只是一个简单的示例，说明如何使用 DummyWebService。）

首先，使用 DummyWebService 重现错误：

1.使用 SOAP 向导从提供的 WSDL 生成 Web 客户端。 所有参数均使用默认值（尽管您可以指定包名称。）
2.查看 SOAP 日志，然后将 Web 服务响应的XML（“ Input to Web client ”）剪切并粘贴到文件中：

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:req
="http://test.systelab.com/">
<soapenv:Header/>
<soapenv:Body><req:createAsynchronuosRequestResponseElement>OK</req:createAsynchronuo
sRequestResponseElement>
</soapenv:Body>
</soapenv:Envelope>
```

3.通过 JSUtil.DummyWebService 中的参数 XMLFILENAME 指向此文件。
4.使用 Studio 的“ Display Web Page ”查看 DummyWebService – 它应显示步骤 2 中创建的文件内容。
5.将步骤 4 中的 URL 剪切并粘贴到步骤 1 中生成的 Web 客户端的 LOCATION 参数中。
6.调用 Web 客户端，例如

```
set client = ##class(MyWebService.RequestWSSoapHttpPort).%New()
do client.createAsynchronuosRequest("x")
quit
```

7. 验证（例如通过 SOAP 日志）是否发生相同错误，即“ ERROR #6203: Unexpected Element ”。

接下来我们解决这个错误。 这里会经历一些过程或反复试验，但同样只是为了说明如何使用 DummyWebService。

1.像之前一样使用 SOAP 向导和提供的 WSDL 生成 Web 客户端 – 指定不同的包名称以防止覆盖第一个 Web 客户端。 另外，参数全部采用默认值，除了一处：
在 SOAP 向导的步骤 3 中选择“ 对于文档样式的 Web 方法使用未包装的消息格式(Use unwrapped message format for document style web methods) ”
2.重复上面的步骤 5 和 6。
3.验证（例如通过 SOAP 日志）Web 客户端错误是否已修正。
（注意：使用“ 未包装的消息格式(unwrapped message format) ”是解决 Web 客户端问题的常见解决方案 – 有关详细信息，请参见我们的文档中的“ 使用 SOAP 向导 。）

总结

可以使用 DummyWebService 类将指定的 SOAP 响应（例如，从 SOAP 日志）发送到 Caché Web 客户端，以模拟 Web 服务的响应。

#SOAP #Caché

源URL:
https://cn.community.intersystems.com/post/%E5%88%9B%E5%BB%BA%E2%80%9C%E8%99%9A%E6%8B%9F%E2%80%9D%E7%9A%84soap-web-%E6%9C%8D%E5%8A%A1