

# Creating REST Clients

Oxford University  
Software Engineering Programme  
Dec 2013



© Paul Fremantle 2012. Portions © Jeremy Gibbons 2010, © WSO2 2005-2012 used with permission of the author(s).  
Licensed under the Creative Commons 3.0 BY-SA (Attribution-Sharealike) license.  
See <http://creativecommons.org/licenses/by-sa/3.0/>

# Just do it?!



© Paul Fremantle 2012. Portions © Jeremy Gibbons 2010, © WSO2 2005-2012 used with permission of the author(s).  
Licensed under the Creative Commons 3.0 BY-SA (Attribution-Sharealike) license.  
See <http://creativecommons.org/licenses/by-sa/3.0/>

# A good answer if you already know

- HTTP Client code
- XML
- JSON
- etc



# First try it out

- Chrome Advanced REST Client is a good start
- SOAPUI also provides test capabilities
- curl should be your friend too!



Advanced Rest Client Application

Offline Mail | Inbox (124,820) | WSO2 | WSO2, Inc. - Calendar | bitmark | Shorten with bit.ly | Gmail - Inbox (720)

# Advanced Rest Client

Request  
Socket  
Projects  
empty  
Saved  
History  
Settings  
About

Scroll to top

http://localhost:8080/OrderService-1.0-SNAPSHOT/hello/echo/blah

GET POST PUT PATCH DELETE HEAD OPTIONS Other

Raw Form Headers

Add new header

Authorization Bearer MtQqjITzrJ0pSSTpOsoFIS0NO3Ya

Clear Send

Status 200 OK Loading time: 309 ms

Request headers

Authorization: Bearer MtQqjITzrJ0pSSTpOsoFIS0NO3Ya  
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_8\_2) AppleWebKit/537.11 (KHTML, like Gecko) Chrome/23.0.1271.95 Safari/537.11  
Accept: \*/\*  
Accept-Encoding: gzip, deflate, sdch  
Accept-Language: en-US,en;q=0.8  
Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.3

Response headers

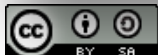
Server: Apache-Coyote/1.1  
Date: Wed, 05 Dec 2012 11:15:10 GMT  
Content-Type: text/plain  
Content-Length: 4

Raw Parsed Response

Open output in new window Copy to clipboard Save as file

blah

Code highlighting thanks to Code Mirror



# curl

```
curl -v http://localhost:8080/OrderService-1.0-SNAPSHOT/hello/echo/blah
* About to connect() to localhost port 8080 (#0)
*   Trying ::1...
* connected
* Connected to localhost (::1) port 8080 (#0)
> GET /OrderService-1.0-SNAPSHOT/hello/echo/blah HTTP/1.1
> User-Agent: curl/7.24.0 (x86_64-apple-darwin12.0) libcurl/7.24.0 OpenSSL/
0.9.8r zlib/1.2.5
> Host: localhost:8080
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Date: Wed, 05 Dec 2012 11:20:17 GMT
< Content-Type: text/plain
< Content-Length: 4
<
* Connection #0 to host localhost left intact
blah* Closing connection #0
```



© Paul Fremantle 2012. Portions © Jeremy Gibbons 2010, © WSO2 2005-2012 used with permission of the author(s).  
Licensed under the Creative Commons 3.0 BY-SA (Attribution-ShareAlike) license.  
See <http://creativecommons.org/licenses/by-sa/3.0/>

# Next choose your favourite HTTP library

- JS: XMLHttpRequest
- .NET: System.Net.Http.HttpClient
- Java: Many!
  - Built-in: java.net.URLConnection
  - CXF: org.apache.cxf.jaxrs.client.WebClient
  - Apache: org.apache.http.client.HttpClient
  - Google HTTP Client, etc, etc
- Ruby
  - [https://www.ruby-toolbox.com/categories/http\\_clients](https://www.ruby-toolbox.com/categories/http_clients) (too many to list)



# JAX-RS 2.0

- Includes support for clients
- Will be available in CXF 3.0.0
  - Currently on 2.7.7





# JAX-RS 2.0 Features

- Client API
- Client-side and Server-side Asynchronous
- Filters and Interceptors
- Improved Connection Negotiation
- Validation
- Hypermedia
- Alignment with JSR 330
- Model-View-Controller



# JAXRS 2.0 Client API

- Similar to CXF client
- Aiming to be much higher level than standard HTTP clients
- Not a bad idea, but don't give up on “loose coupling”
  - The client and the service are independent
  - Technology choice of one shouldn't influence the technology choice of the other



# Example HttpClient code

```
HttpClient client = new DefaultHttpClient();
{
    HttpGet request = new HttpGet(url);
    HttpResponse response = client.execute(request);
    System.out.println("Response status:" + response.getStatusLine());
    System.out.println("Response data:");
    HttpEntity entity = response.getEntity();
    if (entity != null) {

        BufferedReader br = new BufferedReader(new
InputStreamReader(
                (entity.getContent())));
        String output;
        while ((output = br.readLine()) != null) {
            System.out.println(output);
        }
    }
}
```



© Paul Fremantle 2012. Portions © Jeremy Gibbons 2010, © WSO2 2005-2012 used with permission of the author(s).  
Licensed under the Creative Commons 3.0 BY-SA (Attribution-Sharealike) license.  
See <http://creativecommons.org/licenses/by-sa/3.0/>

# Example CXF Client code

```
WebClient client = WebClient.create(url);  
Response r =  
client.accept(MediaType.APPLICATION_JSON).get();  
int status = r.getStatus();  
String jsonString = IOUtils.toString((InputStream)  
r.getEntity());
```



# Questions?



© Paul Fremantle 2012. Portions © Jeremy Gibbons 2010, © WSO2 2005-2012 used with permission of the author(s).  
Licensed under the Creative Commons 3.0 BY-SA (Attribution-Sharealike) license.  
See <http://creativecommons.org/licenses/by-sa/3.0/>