Calling REST Services

When calling a method on a REST resource, make sure to provide always an Accept header. If it is a request with content in the body (like PUT and POST), specify in the Content-Type header how the body is formatted.
This is best practice and so the Bridge will know, which content format has been send and which content format to provide as response. At the moment, the Bridge supports **application/json** and **text/xml** content types for responses (Accept header) and requests (Content-Type header).

- If the **Content-Type** header is set, the Bridge will assume that the request is of that type. All other request will be rejected (HTTP error code **406**).
- If no **Content-Type** header is set, but the **Accept** header is, the Bridge will deduce the request type from it. All other request will be rejected (HTTP error code **406**).
  To determine the format, the Bridge will take into account the quality factor and the order of the accept headers list.
- In absence of both headers, the Bridge will assume JSON. All other request will be rejected (HTTP error code **415**).

The full matching is done in a "best effort" manner. Given the type format `<type>/<subtype> [+<suffix>][; paramName=paramValue]*`, the Bridge first disregards the type and parameters. Then it checks, if the subtype is JSON or XML. If the subtype doesn't match with the supported types, it tries the suffix.

On this Page:

Related Pages:

- XML - UML Class Mapping

Examples

MIME String	Deduced Type
text/plain; charset=utf-8	-
application/json	JSON
application/json+xml	JSON
application/vnd.github+json	JSON
application/vnd.github.v3.raw+json	JSON
application/vnd.github.v3.raw+xml	XML
text/xml; charset=utf-8	XML
application/soap+xml	XML

REST Parameter Class Mapping

REST calls are mapped to UML classes as described on XML - UML Class Mapping.

Arrays

Having an array as output may lead to interoperability problems. There are also known vulnerabilities in browsers regarding top-level JSON arrays.
In case of an array as output parameter, the array will be wrapped in an enclosing XML element, if you choose **text/xml** for your response, nevertheless. The name of the element is deduced from the output parameter name.

```
<supportCases>
  <SupportCase id="000089c00002cb0d2e5ea" customerID="4711" customerName="
Wishes Unltd" [...]></SupportCase>
  <SupportCase id="0089c00000fccd3bae4d4" customerID="4711" customerName="
Wishes Unltd" [...]></SupportCase>
  <SupportCase id="170324000026a400000dd" customerID="4712" customerName="
Spring Corp." [...]></SupportCase>
  <SupportCase id="70fb1000026a400002e18" customerID="4712" customerName="
Spring Corp." [...]></SupportCase>
</supportCases>
```

HTTP Headers

Runtime 2019.9 Bridge xUML services read the following incoming HTTP headers containing correlation information:

- **X-Transaction-Id** or **xTransactionId** (in JMS context)
  This header identifies the transaction the call belongs to. You can set the transaction id manually with setTransactionID. If not set, the Runtime will generate one.
  This header will be passed through the callstack to identify all service calls that belong to a transaction.
- **X-Request-Id**
  This header should identify the unique request.
- **X-Sender-Host** and **X-Sender-Service**
  These headers should contain the sender host resp. the sender service.

These headers will be all logged to the transaction log. Having this information, you can use this for error analysis or usage metrics.

HTTP Status Codes

For HTTP success status codes (2xx), it may be doubtful which status code is return on a specific combination of REST operation and parameter. Find below a list that describes which status codes are returned under which conditions.

Order	Condition	Status Code Returned
1.	**no output parameter**	204 (= no content)
2.	**POST request**	201 (= created)
3.	**else**	200 (= OK)