

# SOAP Call and HTTP Headers

## Providing HTTP Headers on a SOAP Call

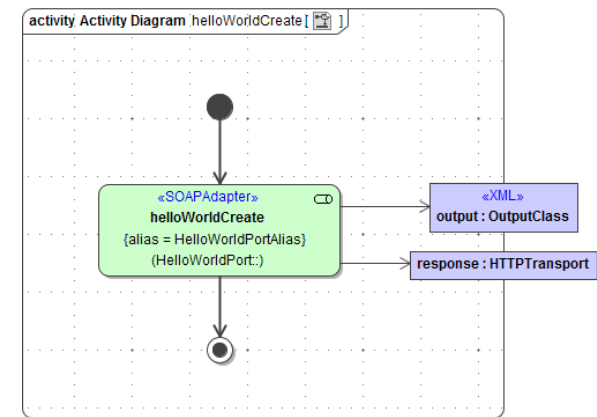
Builder 7.12.0 Runtime 2020.12 In the context of HTTP based adapters like the SOAP adapter, you can enable automatic header generation for dedicated headers. To do this, specify a list of header generation rules in tag `requestHttpHeaderRoles` on the SOAP alias.

`requestHttpHeaderRoles` can hold a list of definitions in format `<http_header name>:<role>`. The listed headers will automatically be generated with the specified role for each adapter call on this alias. These definitions overwrite the default behavior, and **X-Transaction-Id**, **X-Request-Id**, **X-Sender-Host** and/or **X-Sender-Service** will be substituted by this definition. Refer to [HTTP Header Support > Overwriting the Standard HTTP Headers](#) for more details on header roles, and to [SOAP Adapter Reference](#) for the list of allowed values.

## Accessing HTTP Headers after a SOAP Call

Each SOAP request is transported via HTTP. The contents of the HTTP header can be accessed after the SOAP call, e.g. to check the HTTP status code.

The following example shows a SOAP adapter call with getting the HTTP header.



Parameter `response` contains the HTTP response of the SOAP call. You can also read dedicated headers using the [getServiceContextValue\(\) Function](#).

Runtime 2019.9 With xUML service adapter calls, the xUML Runtime adds the following outgoing HTTP headers containing correlation information to the request:

- **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 identifies the unique request. The Runtime generates a unique number for each adapter call.
- **X-Sender-Host** and **X-Sender-Service**  
These headers contain the sender host resp. the sender service. They are set by the Runtime automatically.

Transaction id and request id will be [logged to the transaction log](#) on the adapter call. Having this information, you can use this for error analysis or usage metrics.

## Accessing the Headers

Class	Attribute	Type	Description
HTTPTransport	headers	Array of <a href="#">HeaderField</a>	<div>DeprecatedThis attribute is deprecated as of Runtime 2020.11. Please use <code>httpHeaderMap</code> (see below) for new implementations as its implementation complies to the HTTP specification.</div> <div>Header information as array of name/value pairs.</div>

**On this Page:**

- [Providing HTTP Headers on a SOAP Call](#)
- [Accessing HTTP Headers after a SOAP Call](#)
  - [Accessing the Headers](#)

**Related Pages:**

- [SOAP Adapter Reference](#)
- [getServiceContextValue\(\) Function](#)

	<b>statusCode</b>	Integer	HTTP status code of the SOAP request.
	<b>httpHeaderMap</b>	Map of <a href="#">Entry</a>	<p>Runtime 2020.11 Header information as a map. The map contains arrays of header value strings whereas the header name is the key of the map.</p> <ul style="list-style-type: none"> <li>Header names are lowercase and treated case insensitive.</li> <li>Multiple headers with the same name are treated as arrays.</li> </ul> <p>Refer to <a href="#">HTTP Header Support</a> for more information on the standard xUML HTTP headers.</p>

Class	Attribute	Type
<b>Entry</b>	key	String
	value	Array of Any. The dynamic type for <b>httpHeaderMap</b> is <b>String</b> .
<b>HeaderField</b>	name	
	value	