csg Xponent

Xponent Web Services


Training Overview

Xponent Web Services – Learning Outcomes

- Introduction to RESTful Web Services
- Calling a REST web service end point from Xponent
- Use of different methods: GET, POST, PUT & DELETE
- Parameterization of web service calls
- Exposing Xponent logic as a web service
- Demonstrate Knowledge of Xponent Web Services
- Certification


Introduction to RESTful Web Services

What are RESTful Web Services

- REST stands for Representational State Transfer
- HTTP request format
  - <VERB> is one of the HTTP methods like GET, PUT, POST, DELETE, OPTIONS
  - <URI> is the URI of the resource on which the operation is going to be performed
  - <HTTP Version> is the version of HTTP, generally "HTTP v1.1" .
  - <Request Header> contains the metadata as a collection of key-value pairs of headers and their values. These settings contain information about the message and its sender like client type, the formats client supports, format type of the message body, cache settings for the response, and a lot more information.
  - <Request Body> is the actual message content. In a RESTful service, that's where the representations of resources sit in a message.

Representational State Transfer – RESTful Services

- Resource identification through Uniform Resource Identifiers:
  - http(s)://servername/resourcename/resourceval?param1=val1&param2=val2
- Operations
  - PUT creates a new resource, which can be then deleted by using DELETE.
  - GET retrieves the current state of a resource in some representation.
  - POST transfers a new state onto a resource
- Self-descriptive messages: Resources are decoupled from their representation so that their content can be accessed in a variety of formats, such as HTML, XML, plain text, PDF, JPEG, JSON, and others.
- Stateful interactions through hyperlinks: Every interaction with a resource is stateless

HTTP Response Codes

- 1xx – Informational
- 2xx – Success
  - 200 OK
  - 201 Created
- 3xx – Redirect
- 4xx – Client Error
  - 400 – Bad Request
  - 401 – Unauthorized
  - 404 – Not Found
  - 418 – I'm a teapot
- 5xx – Server Error
  - 500 – Internal Server Error

Useful Resources to test Web Services

- Postman – Chrome plugin or standalone app – very useful for test driven development

- curl – command line tool – for those who like command lines


Getting Started

In This Section

- Call a RESTful web service to get some information
- GET method
- POST method
- Using query parameters and parameterization
- Adding custom Headers
- DELETE method

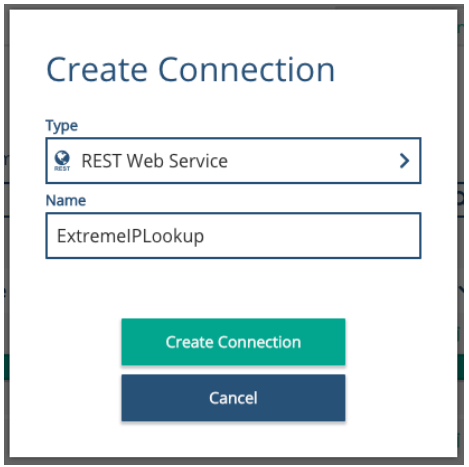
Extreme IP Lookup

- Returns the geographic location for a given IP address
- Will return XML or JSON response
- GET http://extreme-ip-lookup.com/json/68.70.164.200 returns:


```
{
  "businessName" : "MUTARE",
  "businessWebsite" : "",
  "city" : "Cascade",
  "continent" : "North America",
  "country" : "United States",
  "countryCode" : "US",
  "ipName" : "",
  "ipType" : "Business",
  "isp" : "NetSource Communications",
  "lat" : "42.2870",
  "lon" : "-91.0144",
  "org" : "MUTARE",
  "query" : "68.70.164.200",
  "region" : "Iowa",
  "status" : "success"
}
```

Creating a REST Web Service Connection

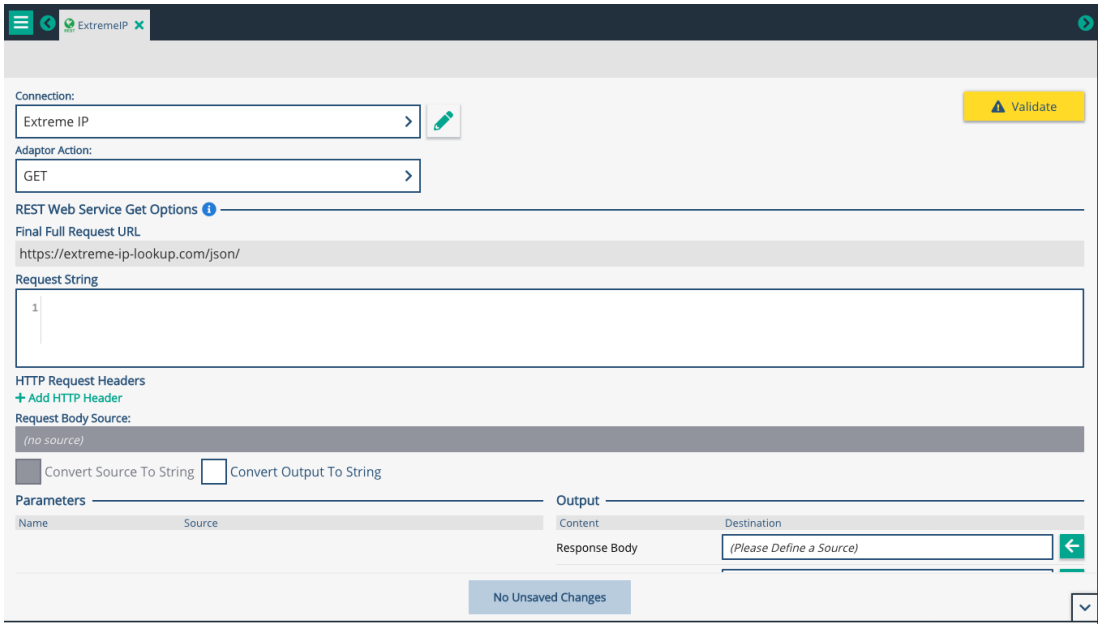
- Only an Owner has access to the Admin screen to create and test a connection
- The new connection will be created across all environments in that project.
- The connection can be set up differently in the Development environment and the Production environment
- Connections may require credentials or OAuth keys



Creating a Web Service GET Adaptor

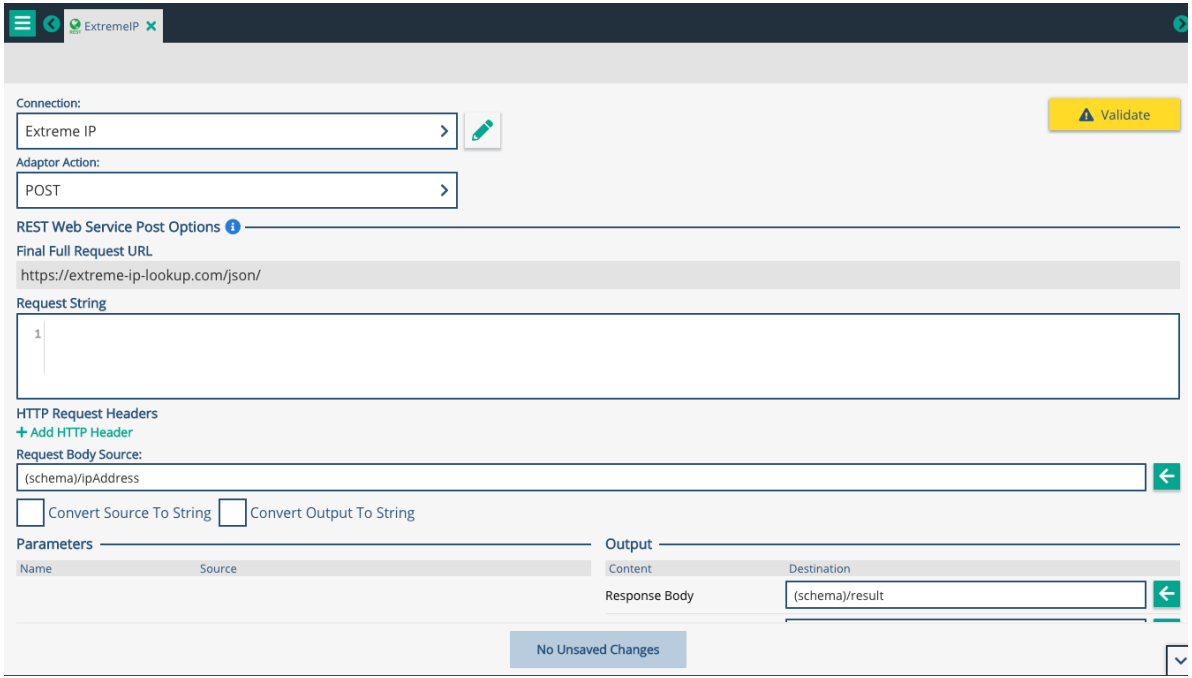
- Choose Connection
- Choose Method: GET
- Extend the request URL with:
  - Constant text
  - Query parameters
- Use parameters by using 2 '%' characters:

  %%paramName%%
- Remember to save and update parameters when making any changes



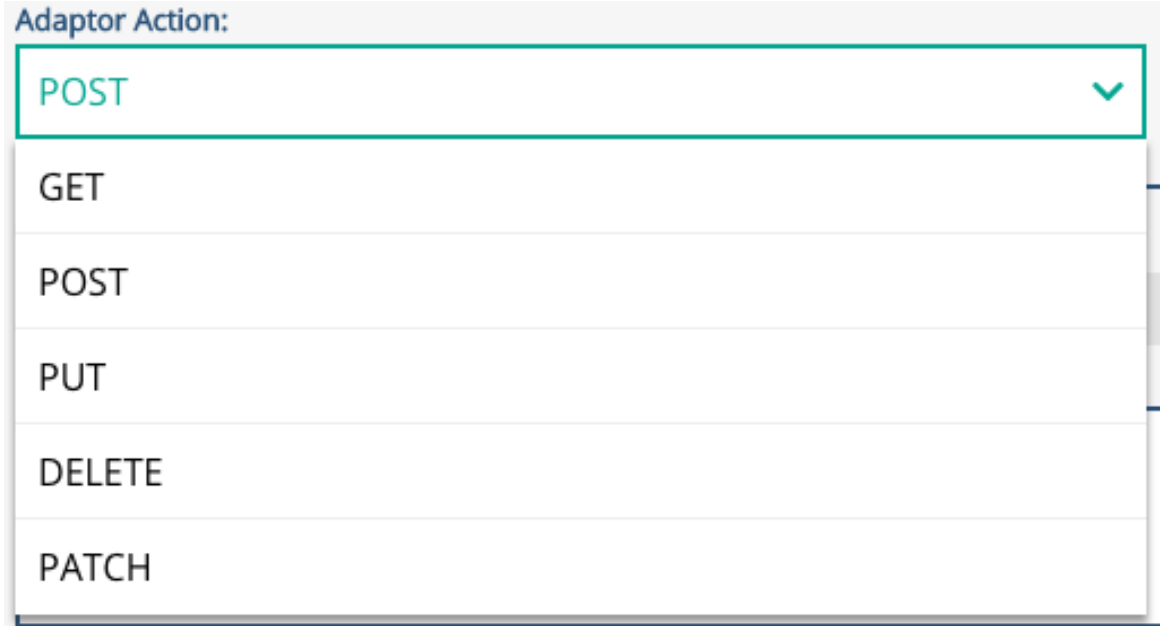
Creating a Web Service POST Adaptor

- Choose Connection and method: POST
- Indicate the Request Body Source in the schema
- Indicate where the result should go



Supported Methods

- GET
- POST
- PUT
- DELETE
- PATCH




Xponent as a Web Service

Xponent Graph API

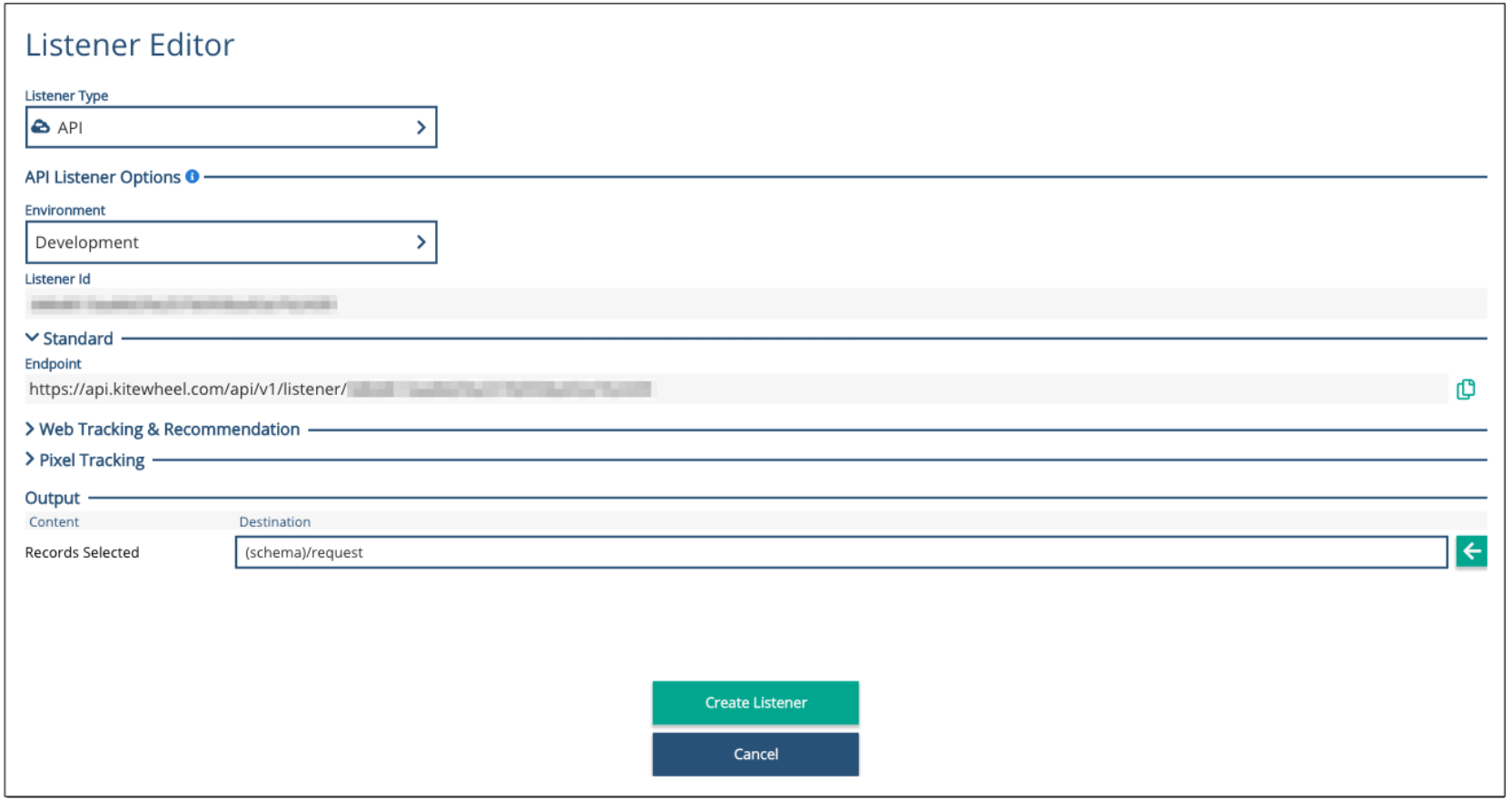
- Any graph in Xponent that is not a listener can be exposed as a web service endpoint by adding an API listener
  - Add an API listener using the dropdown in the top left of a graph
- The listener will generate a unique API end point for that graph by creating a unique Listener ID
- This ID is unique for the graph-environment-project combination. If any of these change, a new Listener ID will be created
- The URL format will be

  https://api[-region].Xponent.com/api/v1/listener/[listenerID]
- The API listener supports GET and POST requests only

Xponent Graph API (continued)

- It is designed for an RPC style interface
  - op=createCustomer&firstName=Neil&lastName=Skilling
  - op=deleteCustomer&id=1234
- The listener writes the request into a designated part of the schema
- This includes the “method” and other query parameters
- A “_kw” object is also provided that has extra information on the request

Xponent Graph API



Hands On: Create a Web Service

- Create a web service that offers the following methods

- Reject incorrect methods or resources with a suitable error

Method	URI	Operation
GET	op=getCustomer&id=2	Return customer with id 2
GET	op=createCustomer&firstName={fname}&lastName={lname}	Create customer returns id
GET	op=updateCustomer&id=5&age={age}	Update customer with id 5
GET	op=deleteCustomer&id=7	Delete customer with id 7

Next Steps

- Create Graph template to decode resource and method correctly
- Create input and output JSON objects
- Create database adaptors
- Create test cases in Postman or curl

Certification

Questions

- What kinds of Web Services does Xponent support?
- What does REST stand for?
- What are the four most common REST methods?
- How do you create a Web Service GET adaptor?
- What does a POST require that a GET does not?
- What is the difference between PUT and POST?
- What are the common HTTP error codes?
- Do I need to create my schema elements before calling a web service?
- How do I make a Xponent graph a web service?
- How do I know what the method is that I have been called with?
- Where are my query parameters?
- What objects can I return from my web service?

Thank You